

glucat

0.8.2

Generated by Doxygen 1.8.14



# Contents

<b>1</b>	<b>Namespace Index</b>	<b>1</b>
1.1	Namespace List . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List . . . . .	7
<b>5</b>	<b>Namespace Documentation</b>	<b>9</b>
5.1	cga3 Namespace Reference . . . . .	9
5.1.1	Detailed Description . . . . .	9
5.1.2	Function Documentation . . . . .	9
5.1.2.1	agc3() . . . . .	9
5.1.2.2	cga3() . . . . .	10
5.1.2.3	cga3std() . . . . .	10
5.2	glucat Namespace Reference . . . . .	10
5.2.1	Typedef Documentation . . . . .	21
5.2.1.1	index_t . . . . .	21
5.2.1.2	intfn . . . . .	22
5.2.1.3	intintfn . . . . .	22
5.2.1.4	set_value_t . . . . .	22

5.2.2	Enumeration Type Documentation	22
5.2.2.1	precision_t	22
5.2.3	Function Documentation	23
5.2.3.1	_GLUCAT_CTAssert() [1/3]	23
5.2.3.2	_GLUCAT_CTAssert() [2/3]	23
5.2.3.3	_GLUCAT_CTAssert() [3/3]	23
5.2.3.4	abs()	23
5.2.3.5	acos() [1/2]	24
5.2.3.6	acos() [2/2]	24
5.2.3.7	acosh() [1/2]	24
5.2.3.8	acosh() [2/2]	25
5.2.3.9	asin() [1/2]	25
5.2.3.10	asin() [2/2]	25
5.2.3.11	asinh() [1/2]	26
5.2.3.12	asinh() [2/2]	26
5.2.3.13	atan() [1/2]	26
5.2.3.14	atan() [2/2]	27
5.2.3.15	atanh() [1/2]	27
5.2.3.16	atanh() [2/2]	27
5.2.3.17	cascade_log()	28
5.2.3.18	check_complex()	28
5.2.3.19	clifford_exp()	28
5.2.3.20	compare()	29
5.2.3.21	complexifier()	29
5.2.3.22	conj()	29
5.2.3.23	cos() [1/2]	30
5.2.3.24	cos() [2/2]	30
5.2.3.25	cosh()	30
5.2.3.26	crd_of_mult() [1/2]	31
5.2.3.27	crd_of_mult() [2/2]	31

5.2.3.28	<code>db_sqrt()</code>	31
5.2.3.29	<code>db_step()</code>	31
5.2.3.30	<code>elliptic()</code>	32
5.2.3.31	<code>even()</code>	32
5.2.3.32	<code>exp()</code> $[1/2]$	32
5.2.3.33	<code>exp()</code> $[2/2]$	33
5.2.3.34	<code>fast()</code>	33
5.2.3.35	<code>folded_dim()</code>	33
5.2.3.36	<code>imag()</code>	34
5.2.3.37	<code>inv()</code>	34
5.2.3.38	<code>inverse_gray()</code>	34
5.2.3.39	<code>inverse_reversed_gray()</code>	34
5.2.3.40	<code>involute()</code>	35
5.2.3.41	<code>log()</code> $[1/4]$	35
5.2.3.42	<code>log()</code> $[2/4]$	35
5.2.3.43	<code>log()</code> $[3/4]$	36
5.2.3.44	<code>log()</code> $[4/4]$	36
5.2.3.45	<code>log2()</code>	36
5.2.3.46	<code>matrix_log()</code>	37
5.2.3.47	<code>matrix_sqrt()</code>	37
5.2.3.48	<code>max_abs()</code>	37
5.2.3.49	<code>max_pos()</code>	38
5.2.3.50	<code>min_neg()</code>	38
5.2.3.51	<code>norm()</code>	38
5.2.3.52	<code>odd()</code>	38
5.2.3.53	<code>offset_level()</code>	39
5.2.3.54	<code>operator &amp;()</code> $[1/8]$	39
5.2.3.55	<code>operator &amp;()</code> $[2/8]$	39
5.2.3.56	<code>operator &amp;()</code> $[3/8]$	39
5.2.3.57	<code>operator &amp;()</code> $[4/8]$	40

5.2.3.58	operator &() [5/8]	40
5.2.3.59	operator &() [6/8]	40
5.2.3.60	operator &() [7/8]	40
5.2.3.61	operator &() [8/8]	41
5.2.3.62	operator !=() [1/3]	41
5.2.3.63	operator !=() [2/3]	41
5.2.3.64	operator !=() [3/3]	41
5.2.3.65	operator %() [1/3]	42
5.2.3.66	operator %() [2/3]	42
5.2.3.67	operator %() [3/3]	42
5.2.3.68	operator *() [1/6]	43
5.2.3.69	operator *() [2/6]	43
5.2.3.70	operator *() [3/6]	43
5.2.3.71	operator *() [4/6]	44
5.2.3.72	operator *() [5/6]	44
5.2.3.73	operator *() [6/6]	44
5.2.3.74	operator +() [1/3]	44
5.2.3.75	operator +() [2/3]	45
5.2.3.76	operator +() [3/3]	45
5.2.3.77	operator -() [1/3]	45
5.2.3.78	operator -() [2/3]	45
5.2.3.79	operator -() [3/3]	46
5.2.3.80	operator /() [1/5]	46
5.2.3.81	operator /() [2/5]	46
5.2.3.82	operator /() [3/5]	47
5.2.3.83	operator /() [4/5]	47
5.2.3.84	operator /() [5/5]	47
5.2.3.85	operator <<() [1/4]	47
5.2.3.86	operator <<() [2/4]	48
5.2.3.87	operator <<() [3/4]	48

5.2.3.88	<code>operator&lt;&lt;()</code>	[ 4/4 ]	48
5.2.3.89	<code>operator&gt;&gt;()</code>	[ 1/3 ]	48
5.2.3.90	<code>operator&gt;&gt;()</code>	[ 2/3 ]	49
5.2.3.91	<code>operator&gt;&gt;()</code>	[ 3/3 ]	49
5.2.3.92	<code>operator^()</code>	[ 1/4 ]	49
5.2.3.93	<code>operator^()</code>	[ 2/4 ]	49
5.2.3.94	<code>operator^()</code>	[ 3/4 ]	50
5.2.3.95	<code>operator^()</code>	[ 4/4 ]	50
5.2.3.96	<code>operator"   ()</code>	[ 1/4 ]	50
5.2.3.97	<code>operator"   ()</code>	[ 2/4 ]	50
5.2.3.98	<code>operator"   ()</code>	[ 3/4 ]	51
5.2.3.99	<code>operator"   ()</code>	[ 4/4 ]	51
5.2.3.100	<code>outer_pow()</code>		51
5.2.3.101	<code>pade_approx()</code>		51
5.2.3.102	<code>pade_log()</code>		52
5.2.3.103	<code>pos_mod()</code>		52
5.2.3.104	<code>pow()</code>	[ 1/2 ]	52
5.2.3.105	<code>pow()</code>	[ 2/2 ]	53
5.2.3.106	<code>pure()</code>		53
5.2.3.107	<code>quad()</code>		53
5.2.3.108	<code>real()</code>		53
5.2.3.109	<code>reframe()</code>		54
5.2.3.110	<code>reverse()</code>		54
5.2.3.111	<code>scalar()</code>		54
5.2.3.112	<code>sign_of_square()</code>		55
5.2.3.113	<code>sin()</code>	[ 1/2 ]	55
5.2.3.114	<code>sin()</code>	[ 2/2 ]	55
5.2.3.115	<code>sinh()</code>		56
5.2.3.116	<code>sqrt()</code>	[ 1/4 ]	56
5.2.3.117	<code>sqrt()</code>	[ 2/4 ]	56

5.2.3.118	<a href="#">sqrt()</a> [3/4]	57
5.2.3.119	<a href="#">sqrt()</a> [4/4]	57
5.2.3.120	<a href="#">star()</a> [1/3]	57
5.2.3.121	<a href="#">star()</a> [2/3]	57
5.2.3.122	<a href="#">star()</a> [3/3]	58
5.2.3.123	<a href="#">tan()</a> [1/2]	58
5.2.3.124	<a href="#">tan()</a> [2/2]	58
5.2.3.125	<a href="#">tanh()</a>	59
5.2.3.126	<a href="#">to_demote()</a>	59
5.2.3.127	<a href="#">to_promote()</a>	59
5.2.3.128	<a href="#">try_catch()</a> [1/2]	59
5.2.3.129	<a href="#">try_catch()</a> [2/2]	60
5.2.3.130	<a href="#">vector_part()</a>	60
5.2.4	Variable Documentation	60
5.2.4.1	<a href="#">BITS_PER_SET_VALUE</a>	60
5.2.4.2	<a href="#">DEFAULT_Basis_Max_Count</a>	60
5.2.4.3	<a href="#">DEFAULT_Div_Max_Steps</a>	61
5.2.4.4	<a href="#">DEFAULT_Fast_Size_Threshold</a>	61
5.2.4.5	<a href="#">DEFAULT_Function_Precision</a>	61
5.2.4.6	<a href="#">DEFAULT_HI</a>	61
5.2.4.7	<a href="#">DEFAULT_Inv_Fast_Dim_Threshold</a>	61
5.2.4.8	<a href="#">DEFAULT_Log_Max_Inner_Steps</a>	61
5.2.4.9	<a href="#">DEFAULT_Log_Max_Outer_Steps</a>	62
5.2.4.10	<a href="#">DEFAULT_Mult_Matrix_Threshold</a>	62
5.2.4.11	<a href="#">DEFAULT_Products_Size_Threshold</a>	62
5.2.4.12	<a href="#">DEFAULT_Sqrt_Max_Steps</a>	62
5.2.4.13	<a href="#">DEFAULT_TRUNCATION</a>	62
5.2.4.14	<a href="#">I_ln2</a>	62
5.2.4.15	<a href="#">I_pi</a>	63
5.2.4.16	<a href="#">MS_PER_S</a>	63



5.3	<a href="#">glucat::gen Namespace Reference</a>	63
5.3.1	<a href="#">Typedef Documentation</a>	63
5.3.1.1	<a href="#">signature_t</a>	63
5.3.2	<a href="#">Variable Documentation</a>	64
5.3.2.1	<a href="#">offset_to_super</a>	64
5.4	<a href="#">glucat::matrix Namespace Reference</a>	64
5.4.1	<a href="#">Enumeration Type Documentation</a>	65
5.4.1.1	<a href="#">eig_case_t</a>	65
5.4.2	<a href="#">Function Documentation</a>	66
5.4.2.1	<a href="#">classify_eigenvalues()</a>	66
5.4.2.2	<a href="#">eigenvalues()</a>	66
5.4.2.3	<a href="#">inner()</a>	66
5.4.2.4	<a href="#">isnan()</a>	67
5.4.2.5	<a href="#">kron()</a>	67
5.4.2.6	<a href="#">mono_kron()</a>	67
5.4.2.7	<a href="#">mono_prod()</a>	67
5.4.2.8	<a href="#">nnz()</a>	68
5.4.2.9	<a href="#">nork()</a>	68
5.4.2.10	<a href="#">nork_range()</a>	68
5.4.2.11	<a href="#">norm_frob2()</a>	69
5.4.2.12	<a href="#">prod()</a>	69
5.4.2.13	<a href="#">signed_perm_nork()</a>	69
5.4.2.14	<a href="#">sparse_prod()</a>	69
5.4.2.15	<a href="#">to_lapack()</a>	70
5.4.2.16	<a href="#">trace()</a>	70
5.4.2.17	<a href="#">unit()</a>	70
5.5	<a href="#">glucat::timing Namespace Reference</a>	70
5.5.1	<a href="#">Function Documentation</a>	71
5.5.1.1	<a href="#">elapsed()</a>	71
5.5.2	<a href="#">Variable Documentation</a>	71

5.5.2.1	EXTRA_TRIALS	71
5.5.2.2	MS_PER_CLOCK	71
5.5.2.3	MS_PER_SEC	71
5.6	PyClical Namespace Reference	72
5.6.1	Function Documentation	72
5.6.1.1	_test()	72
5.6.1.2	clifford_hidden_doctests()	73
5.6.1.3	e()	74
5.6.1.4	index_set_hidden_doctests()	74
5.6.1.5	istpq()	76
5.6.2	Variable Documentation	76
5.6.2.1	__version__	76
5.6.2.2	cl	76
5.6.2.3	fill	76
5.6.2.4	i	77
5.6.2.5	ist	77
5.6.2.6	ixt	77
5.6.2.7	nbar3	77
5.6.2.8	ninf3	77
5.6.2.9	obj	78
5.6.2.10	pi	78
5.6.2.11	tau	78
5.7	std Namespace Reference	78

<b>6 Class Documentation</b>	<b>79</b>
6.1 <code>glucat::basis_table&lt; Scalar_T, LO, HI, Matrix_T &gt;</code> Class Template Reference	79
6.1.1 Detailed Description	80
6.1.2 Constructor & Destructor Documentation	80
6.1.2.1 <code>basis_table()</code> [1/2]	80
6.1.2.2 <code>~basis_table()</code>	80
6.1.2.3 <code>basis_table()</code> [2/2]	81
6.1.3 Member Function Documentation	81
6.1.3.1 <code>basis()</code>	81
6.1.3.2 <code>operator=()</code>	81
6.1.4 Friends And Related Function Documentation	81
6.1.4.1 <code>friend_for_private_destructor</code>	81
6.2 <code>glucat::bool_to_type&lt; truth_value &gt;</code> Class Template Reference	82
6.2.1 Detailed Description	82
6.2.2 Member Enumeration Documentation	82
6.2.2.1 <code>anonymous enum</code>	82
6.3 <code>PyClical.clifford</code> Class Reference	83
6.3.1 Detailed Description	84
6.3.2 Member Function Documentation	84
6.3.2.1 <code>__add__()</code>	85
6.3.2.2 <code>__and__()</code>	85
6.3.2.3 <code>__call__()</code>	85
6.3.2.4 <code>__cinit__()</code>	86
6.3.2.5 <code>__contains__()</code>	86
6.3.2.6 <code>__dealloc__()</code>	86
6.3.2.7 <code>__div__()</code>	87
6.3.2.8 <code>__getitem__()</code>	87
6.3.2.9 <code>__iadd__()</code>	87
6.3.2.10 <code>__iand__()</code>	88
6.3.2.11 <code>__idiv__()</code>	88

6.3.2.12	<code>__imod__()</code>	88
6.3.2.13	<code>__imul__()</code>	89
6.3.2.14	<code>__ior__()</code>	89
6.3.2.15	<code>__isub__()</code>	89
6.3.2.16	<code>__iter__()</code>	90
6.3.2.17	<code>__ixor__()</code>	90
6.3.2.18	<code>__mod__()</code>	90
6.3.2.19	<code>__mul__()</code>	91
6.3.2.20	<code>__neg__()</code>	91
6.3.2.21	<code>__or__()</code>	91
6.3.2.22	<code>__pos__()</code>	92
6.3.2.23	<code>__pow__()</code>	92
6.3.2.24	<code>__repr__()</code>	92
6.3.2.25	<code>__richcmp__()</code>	93
6.3.2.26	<code>__str__()</code>	93
6.3.2.27	<code>__sub__()</code>	93
6.3.2.28	<code>__xor__()</code>	94
6.3.2.29	<code>abs()</code>	94
6.3.2.30	<code>conj()</code>	94
6.3.2.31	<code>even()</code>	95
6.3.2.32	<code>frame()</code>	95
6.3.2.33	<code>inv()</code>	95
6.3.2.34	<code>involute()</code>	96
6.3.2.35	<code>isnan()</code>	96
6.3.2.36	<code>max_abs()</code>	96
6.3.2.37	<code>norm()</code>	97
6.3.2.38	<code>odd()</code>	97
6.3.2.39	<code>outer_pow()</code>	97
6.3.2.40	<code>pow()</code>	98
6.3.2.41	<code>pure()</code>	98

6.3.2.42	<a href="#">quad()</a>	98
6.3.2.43	<a href="#">reframe()</a>	99
6.3.2.44	<a href="#">reverse()</a>	99
6.3.2.45	<a href="#">scalar()</a>	99
6.3.2.46	<a href="#">truncated()</a>	100
6.3.2.47	<a href="#">vector_part()</a>	100
6.3.3	<a href="#">Member Data Documentation</a>	100
6.3.3.1	<a href="#">instance</a>	100
6.4	<a href="#">glucat::clifford_algebra&lt; Scalar_T, Index_Set_T, Multivector_T &gt; Class Template Reference</a>	101
6.4.1	<a href="#">Detailed Description</a>	103
6.4.2	<a href="#">Member Typedef Documentation</a>	103
6.4.2.1	<a href="#">index_set_t</a>	103
6.4.2.2	<a href="#">multivector_t</a>	103
6.4.2.3	<a href="#">pair_t</a>	103
6.4.2.4	<a href="#">scalar_t</a>	103
6.4.2.5	<a href="#">vector_t</a>	104
6.4.3	<a href="#">Constructor &amp; Destructor Documentation</a>	104
6.4.3.1	<a href="#">~clifford_algebra()</a>	104
6.4.4	<a href="#">Member Function Documentation</a>	104
6.4.4.1	<a href="#">classname()</a>	104
6.4.4.2	<a href="#">conj()</a>	104
6.4.4.3	<a href="#">even()</a>	105
6.4.4.4	<a href="#">frame()</a>	105
6.4.4.5	<a href="#">grade()</a>	105
6.4.4.6	<a href="#">inv()</a>	105
6.4.4.7	<a href="#">involute()</a>	105
6.4.4.8	<a href="#">isnan()</a>	106
6.4.4.9	<a href="#">max_abs()</a>	106
6.4.4.10	<a href="#">norm()</a>	106
6.4.4.11	<a href="#">odd()</a>	106

6.4.4.12	<code>operator &amp;=()</code>	106
6.4.4.13	<code>operator %=()</code>	107
6.4.4.14	<code>operator()()</code>	107
6.4.4.15	<code>operator*=( ) [1/2]</code>	107
6.4.4.16	<code>operator*=( ) [2/2]</code>	107
6.4.4.17	<code>operator+=( ) [1/2]</code>	107
6.4.4.18	<code>operator+=( ) [2/2]</code>	108
6.4.4.19	<code>operator-()</code>	108
6.4.4.20	<code>operator-=( )</code>	108
6.4.4.21	<code>operator/=( ) [1/2]</code>	108
6.4.4.22	<code>operator/=( ) [2/2]</code>	108
6.4.4.23	<code>operator==( ) [1/2]</code>	109
6.4.4.24	<code>operator==( ) [2/2]</code>	109
6.4.4.25	<code>operator[]()</code>	109
6.4.4.26	<code>operator^=( )</code>	109
6.4.4.27	<code>operator"  =()</code>	109
6.4.4.28	<code>outer_pow()</code>	110
6.4.4.29	<code>pow()</code>	110
6.4.4.30	<code>pure()</code>	110
6.4.4.31	<code>quad()</code>	110
6.4.4.32	<code>reverse()</code>	110
6.4.4.33	<code>scalar()</code>	111
6.4.4.34	<code>truncated()</code>	111
6.4.4.35	<code>vector_part() [1/2]</code>	111
6.4.4.36	<code>vector_part() [2/2]</code>	111
6.4.4.37	<code>write() [1/2]</code>	111
6.4.4.38	<code>write() [2/2]</code>	112
6.5	<code>glucat::compare_types&lt; LHS_T, RHS_T &gt; Class Template Reference</code>	112
6.5.1	Detailed Description	112
6.5.2	Member Enumeration Documentation	112

6.5.2.1	anonymous enum	112
6.6	glucat::compare_types< T, T > Class Template Reference	113
6.6.1	Detailed Description	113
6.6.2	Member Enumeration Documentation	113
6.6.2.1	anonymous enum	113
6.7	glucat::control_t Class Reference	113
6.7.1	Detailed Description	114
6.7.2	Constructor & Destructor Documentation	115
6.7.2.1	control_t() [1/3]	115
6.7.2.2	control_t() [2/3]	115
6.7.2.3	~control_t()	115
6.7.2.4	control_t() [3/3]	115
6.7.3	Member Function Documentation	115
6.7.3.1	call() [1/2]	116
6.7.3.2	call() [2/2]	116
6.7.3.3	catch_exceptions()	116
6.7.3.4	control()	116
6.7.3.5	operator=()	117
6.7.3.6	valid()	117
6.7.3.7	verbose()	117
6.7.4	Friends And Related Function Documentation	117
6.7.4.1	friend_for_private_destructor	117
6.7.5	Member Data Documentation	117
6.7.5.1	m_catch_exceptions	118
6.7.5.2	m_valid	118
6.7.5.3	m_verbose_output	118
6.8	glucat::CTAssertion< bool > Struct Template Reference	118
6.8.1	Detailed Description	119
6.9	glucat::CTAssertion< true > Struct Template Reference	119
6.9.1	Detailed Description	119

6.10	<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;::demoted&lt;&gt; Struct Template Reference</a>	119
6.10.1	<a href="#">Detailed Description</a>	119
6.10.2	<a href="#">Member Typedef Documentation</a>	120
6.10.2.1	<a href="#">type [1/2]</a>	120
6.10.2.2	<a href="#">type [2/2]</a>	120
6.11	<a href="#">glucat::matrix::eig_genus&lt; Matrix_T &gt; Struct Template Reference</a>	120
6.11.1	<a href="#">Detailed Description</a>	121
6.11.2	<a href="#">Member Typedef Documentation</a>	121
6.11.2.1	<a href="#">Scalar_T</a>	121
6.11.3	<a href="#">Member Data Documentation</a>	121
6.11.3.1	<a href="#">m_eig_case</a>	121
6.11.3.2	<a href="#">m_safe_arg</a>	121
6.12	<a href="#">glucat::error&lt; Class_T &gt; Class Template Reference</a>	122
6.12.1	<a href="#">Detailed Description</a>	123
6.12.2	<a href="#">Constructor &amp; Destructor Documentation</a>	123
6.12.2.1	<a href="#">error() [1/2]</a>	123
6.12.2.2	<a href="#">error() [2/2]</a>	123
6.12.3	<a href="#">Member Function Documentation</a>	123
6.12.3.1	<a href="#">classname()</a>	123
6.12.3.2	<a href="#">heading()</a>	124
6.12.3.3	<a href="#">print_error_msg()</a>	124
6.13	<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt; Class Template Reference</a>	124
6.13.1	<a href="#">Detailed Description</a>	127
6.13.2	<a href="#">Member Typedef Documentation</a>	127
6.13.2.1	<a href="#">const_iterator</a>	128
6.13.2.2	<a href="#">error_t</a>	128
6.13.2.3	<a href="#">framed_multi_t</a>	128
6.13.2.4	<a href="#">framed_pair_t</a>	128
6.13.2.5	<a href="#">index_set_t</a>	128
6.13.2.6	<a href="#">iterator</a>	129



6.13.2.7	<code>map_t</code>	129
6.13.2.8	<code>matrix_multi_t</code>	129
6.13.2.9	<code>matrix_t</code>	129
6.13.2.10	<code>multivector_t</code>	129
6.13.2.11	<code>scalar_t</code>	130
6.13.2.12	<code>size_type</code>	130
6.13.2.13	<code>sorted_map_t</code>	130
6.13.2.14	<code>term_t</code>	130
6.13.2.15	<code>var_term_t</code>	130
6.13.2.16	<code>vector_t</code>	131
6.13.3	Constructor & Destructor Documentation	131
6.13.3.1	<code>~framed_multi()</code>	131
6.13.3.2	<code>framed_multi()</code> [1/15]	131
6.13.3.3	<code>framed_multi()</code> [2/15]	131
6.13.3.4	<code>framed_multi()</code> [3/15]	132
6.13.3.5	<code>framed_multi()</code> [4/15]	132
6.13.3.6	<code>framed_multi()</code> [5/15]	132
6.13.3.7	<code>framed_multi()</code> [6/15]	132
6.13.3.8	<code>framed_multi()</code> [7/15]	133
6.13.3.9	<code>framed_multi()</code> [8/15]	133
6.13.3.10	<code>framed_multi()</code> [9/15]	133
6.13.3.11	<code>framed_multi()</code> [10/15]	133
6.13.3.12	<code>framed_multi()</code> [11/15]	134
6.13.3.13	<code>framed_multi()</code> [12/15]	134
6.13.3.14	<code>framed_multi()</code> [13/15]	134
6.13.3.15	<code>framed_multi()</code> [14/15]	134
6.13.3.16	<code>framed_multi()</code> [15/15]	135
6.13.4	Member Function Documentation	135
6.13.4.1	<code>centre_pm4_qp4()</code>	135
6.13.4.2	<code>centre_pp4_qm4()</code>	135

6.13.4.3	<code>centre_qp1_pm1()</code>	136
6.13.4.4	<code>classname()</code>	136
6.13.4.5	<code>divide()</code>	136
6.13.4.6	<code>fast()</code>	137
6.13.4.7	<code>fast_framed_multi()</code>	137
6.13.4.8	<code>fast_matrix_multi()</code>	137
6.13.4.9	<code>fold()</code>	138
6.13.4.10	<code>nbr_terms()</code>	138
6.13.4.11	<code>operator+=()</code>	138
6.13.4.12	<code>random()</code>	138
6.13.4.13	<code>unfold()</code>	139
6.13.5	Friends And Related Function Documentation	139
6.13.5.1	<code>exp</code>	139
6.13.5.2	<code>framed_multi</code>	139
6.13.5.3	<code>matrix_multi</code>	139
6.13.5.4	<code>operator &amp;</code>	140
6.13.5.5	<code>operator%</code>	140
6.13.5.6	<code>operator*</code>	140
6.13.5.7	<code>operator/</code>	140
6.13.5.8	<code>operator&lt;&lt; [1/2]</code>	140
6.13.5.9	<code>operator&lt;&lt; [2/2]</code>	141
6.13.5.10	<code>operator&gt;&gt;</code>	141
6.13.5.11	<code>operator^</code>	141
6.13.5.12	<code>operator"  </code>	141
6.13.5.13	<code>star</code>	141
6.14	<code>glucat::gen::generator_table&lt; Matrix_T &gt;</code> Class Template Reference	142
6.14.1	Detailed Description	143
6.14.2	Constructor & Destructor Documentation	143
6.14.2.1	<code>generator_table() [1/2]</code>	143
6.14.2.2	<code>~generator_table()</code>	143

6.14.2.3	<a href="#">generator_table()</a> [2/2]	144
6.14.3	<a href="#">Member Function Documentation</a>	144
6.14.3.1	<a href="#">gen_from_pm1_qm1()</a>	144
6.14.3.2	<a href="#">gen_from_pm4_qp4()</a>	144
6.14.3.3	<a href="#">gen_from_pp4_qm4()</a>	144
6.14.3.4	<a href="#">gen_from_qp1_pm1()</a>	145
6.14.3.5	<a href="#">gen_vector()</a>	145
6.14.3.6	<a href="#">generator()</a>	145
6.14.3.7	<a href="#">operator()()</a>	145
6.14.3.8	<a href="#">operator=()</a>	146
6.14.4	<a href="#">Friends And Related Function Documentation</a>	146
6.14.4.1	<a href="#">friend_for_private_destructor</a>	146
6.15	<a href="#">glucat::glucat_error Class Reference</a>	146
6.15.1	<a href="#">Detailed Description</a>	147
6.15.2	<a href="#">Constructor &amp; Destructor Documentation</a>	147
6.15.2.1	<a href="#">glucat_error()</a>	147
6.15.2.2	<a href="#">~glucat_error()</a>	148
6.15.3	<a href="#">Member Function Documentation</a>	148
6.15.3.1	<a href="#">classname()</a>	148
6.15.3.2	<a href="#">heading()</a>	148
6.15.3.3	<a href="#">print_error_msg()</a>	148
6.15.4	<a href="#">Member Data Documentation</a>	148
6.15.4.1	<a href="#">name</a>	148
6.16	<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;::hash_size_t Class Reference</a>	149
6.16.1	<a href="#">Detailed Description</a>	149
6.16.2	<a href="#">Constructor &amp; Destructor Documentation</a>	149
6.16.2.1	<a href="#">hash_size_t()</a>	149
6.16.3	<a href="#">Member Function Documentation</a>	149
6.16.3.1	<a href="#">operator()()</a>	149
6.16.4	<a href="#">Member Data Documentation</a>	150

6.16.4.1	n	150
6.17	glucat::index_set< LO, HI > Class Template Reference	150
6.17.1	Detailed Description	153
6.17.2	Member Typedef Documentation	153
6.17.2.1	bitset_t	153
6.17.2.2	error_t	153
6.17.2.3	index_pair_t	154
6.17.2.4	index_set_t	154
6.17.3	Constructor & Destructor Documentation	154
6.17.3.1	index_set() [1/6]	154
6.17.3.2	index_set() [2/6]	154
6.17.3.3	index_set() [3/6]	155
6.17.3.4	index_set() [4/6]	155
6.17.3.5	index_set() [5/6]	155
6.17.3.6	index_set() [6/6]	155
6.17.4	Member Function Documentation	156
6.17.4.1	BOOST_STATIC_ASSERT()	156
6.17.4.2	classname()	156
6.17.4.3	count()	156
6.17.4.4	count_neg()	156
6.17.4.5	count_pos()	157
6.17.4.6	flip() [1/2]	157
6.17.4.7	flip() [2/2]	157
6.17.4.8	fold() [1/2]	157
6.17.4.9	fold() [2/2]	158
6.17.4.10	hash_fn()	158
6.17.4.11	is_contiguous()	158
6.17.4.12	lex_less_than()	158
6.17.4.13	max()	159
6.17.4.14	min()	159

6.17.4.15 operator &=()	159
6.17.4.16 operator !=()	159
6.17.4.17 operator <()	160
6.17.4.18 operator ==()	160
6.17.4.19 operator []() [1/2]	160
6.17.4.20 operator []() [2/2]	160
6.17.4.21 operator ^=()	161
6.17.4.22 operator "   =()	161
6.17.4.23 operator ~()	161
6.17.4.24 reset() [1/2]	161
6.17.4.25 reset() [2/2]	162
6.17.4.26 set() [1/3]	162
6.17.4.27 set() [2/3]	162
6.17.4.28 set() [3/3]	162
6.17.4.29 sign_of_mult()	163
6.17.4.30 sign_of_square()	163
6.17.4.31 test()	163
6.17.4.32 unfold()	163
6.17.4.33 value_of_fold()	164
6.17.5 Friends And Related Function Documentation	164
6.17.5.1 compare	164
6.17.5.2 operator &	164
6.17.5.3 operator ^	164
6.17.5.4 operator "	164
6.17.5.5 reference	165
6.17.6 Member Data Documentation	165
6.17.6.1 v_hi	165
6.17.6.2 v_lo	165
6.18 PyClical.index_set Class Reference	165
6.18.1 Detailed Description	166

6.18.2	Member Function Documentation	167
6.18.2.1	<code>__and__()</code>	167
6.18.2.2	<code>__cinit__()</code>	167
6.18.2.3	<code>__contains__()</code>	168
6.18.2.4	<code>__dealloc__()</code>	168
6.18.2.5	<code>__getitem__()</code>	168
6.18.2.6	<code>__iand__()</code>	169
6.18.2.7	<code>__invert__()</code>	169
6.18.2.8	<code>__ior__()</code>	169
6.18.2.9	<code>__iter__()</code>	170
6.18.2.10	<code>__ixor__()</code>	170
6.18.2.11	<code>__or__()</code>	170
6.18.2.12	<code>__repr__()</code>	171
6.18.2.13	<code>__richcmp__()</code>	171
6.18.2.14	<code>__setitem__()</code>	171
6.18.2.15	<code>__str__()</code>	172
6.18.2.16	<code>__xor__()</code>	172
6.18.2.17	<code>count()</code>	172
6.18.2.18	<code>count_neg()</code>	173
6.18.2.19	<code>count_pos()</code>	173
6.18.2.20	<code>hash_fn()</code>	173
6.18.2.21	<code>max()</code>	174
6.18.2.22	<code>min()</code>	174
6.18.2.23	<code>sign_of_mult()</code>	174
6.18.2.24	<code>sign_of_square()</code>	175
6.18.3	Member Data Documentation	175
6.18.3.1	<code>instance</code>	175
6.19	<code>glucat::index_set_hash&lt; LO, HI &gt;</code> Class Template Reference	175
6.19.1	Detailed Description	176
6.19.2	Member Typedef Documentation	176

6.19.2.1	<code>index_set_t</code>	176
6.19.3	Member Function Documentation	176
6.19.3.1	<code>operator()</code>	176
6.20	<code>glucat::matrix_multi&lt; Scalar_T, LO, HI &gt;</code> Class Template Reference	177
6.20.1	Detailed Description	180
6.20.2	Member Typedef Documentation	180
6.20.2.1	<code>basis_matrix_t</code>	180
6.20.2.2	<code>error_t</code>	180
6.20.2.3	<code>framed_multi_t</code>	181
6.20.2.4	<code>index_set_t</code>	181
6.20.2.5	<code>matrix_index_t</code>	181
6.20.2.6	<code>matrix_multi_t</code>	181
6.20.2.7	<code>matrix_t</code>	181
6.20.2.8	<code>multivector_t</code>	182
6.20.2.9	<code>orientation_t</code>	182
6.20.2.10	<code>scalar_t</code>	182
6.20.2.11	<code>term_t</code>	182
6.20.2.12	<code>vector_t</code>	182
6.20.3	Constructor & Destructor Documentation	183
6.20.3.1	<code>~matrix_multi()</code>	183
6.20.3.2	<code>matrix_multi()</code> [1/17]	183
6.20.3.3	<code>matrix_multi()</code> [2/17]	183
6.20.3.4	<code>matrix_multi()</code> [3/17]	184
6.20.3.5	<code>matrix_multi()</code> [4/17]	184
6.20.3.6	<code>matrix_multi()</code> [5/17]	184
6.20.3.7	<code>matrix_multi()</code> [6/17]	185
6.20.3.8	<code>matrix_multi()</code> [7/17]	185
6.20.3.9	<code>matrix_multi()</code> [8/17]	185
6.20.3.10	<code>matrix_multi()</code> [9/17]	185
6.20.3.11	<code>matrix_multi()</code> [10/17]	186

6.20.3.12	<a href="#">matrix_multi()</a> [11/17]	186
6.20.3.13	<a href="#">matrix_multi()</a> [12/17]	186
6.20.3.14	<a href="#">matrix_multi()</a> [13/17]	186
6.20.3.15	<a href="#">matrix_multi()</a> [14/17]	187
6.20.3.16	<a href="#">matrix_multi()</a> [15/17]	187
6.20.3.17	<a href="#">matrix_multi()</a> [16/17]	187
6.20.3.18	<a href="#">matrix_multi()</a> [17/17]	188
6.20.4	<a href="#">Member Function Documentation</a>	188
6.20.4.1	<a href="#">basis_element()</a>	188
6.20.4.2	<a href="#">classname()</a>	188
6.20.4.3	<a href="#">fast_framed_multi()</a>	189
6.20.4.4	<a href="#">fast_matrix_multi()</a>	189
6.20.4.5	<a href="#">operator+=()</a>	189
6.20.4.6	<a href="#">operator=()</a>	189
6.20.4.7	<a href="#">random()</a>	190
6.20.5	<a href="#">Friends And Related Function Documentation</a>	190
6.20.5.1	<a href="#">framed_multi</a>	190
6.20.5.2	<a href="#">matrix_log</a>	190
6.20.5.3	<a href="#">matrix_multi</a>	190
6.20.5.4	<a href="#">matrix_sqrt</a>	191
6.20.5.5	<a href="#">operator &amp;</a>	191
6.20.5.6	<a href="#">operator%</a>	191
6.20.5.7	<a href="#">operator*</a>	191
6.20.5.8	<a href="#">operator/</a>	191
6.20.5.9	<a href="#">operator&lt;&lt;</a> [1/2]	192
6.20.5.10	<a href="#">operator&lt;&lt;</a> [2/2]	192
6.20.5.11	<a href="#">operator&gt;&gt;</a>	192
6.20.5.12	<a href="#">operator^</a>	192
6.20.5.13	<a href="#">operator"  </a>	192
6.20.5.14	<a href="#">reframe</a>	193



6.20.5.15	star	193
6.20.6	Member Data Documentation	193
6.20.6.1	m_frame	193
6.20.6.2	m_matrix	193
6.21	std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > > Struct Template Reference	194
6.21.1	Detailed Description	194
6.22	std::numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > > Struct Template Reference	195
6.22.1	Detailed Description	195
6.23	glucat::numeric_traits< Scalar_T > Class Template Reference	196
6.23.1	Detailed Description	198
6.23.2	Member Function Documentation	198
6.23.2.1	abs()	198
6.23.2.2	acos()	199
6.23.2.3	asin()	199
6.23.2.4	atan()	199
6.23.2.5	conj()	199
6.23.2.6	cos()	200
6.23.2.7	cosh()	200
6.23.2.8	exp()	200
6.23.2.9	fmod()	200
6.23.2.10	imag()	201
6.23.2.11	isInf() [1/3]	201
6.23.2.12	isInf() [2/3]	201
6.23.2.13	isInf() [3/3]	201
6.23.2.14	isNaN() [1/3]	202
6.23.2.15	isNaN() [2/3]	202
6.23.2.16	isNaN() [3/3]	202
6.23.2.17	isNaN_or_isInf()	202
6.23.2.18	ln_2() [1/2]	203
6.23.2.19	ln_2() [2/2]	203

6.23.2.20 <code>log()</code>	203
6.23.2.21 <code>log2()</code>	203
6.23.2.22 <code>NaN()</code>	204
6.23.2.23 <code>pi()</code> [1/2]	204
6.23.2.24 <code>pi()</code> [2/2]	204
6.23.2.25 <code>pow()</code>	204
6.23.2.26 <code>real()</code>	205
6.23.2.27 <code>sin()</code>	205
6.23.2.28 <code>sinh()</code>	205
6.23.2.29 <code>sqrt()</code>	205
6.23.2.30 <code>tan()</code>	206
6.23.2.31 <code>tanh()</code>	206
6.23.2.32 <code>to_double()</code>	206
6.23.2.33 <code>to_int()</code>	206
6.23.2.34 <code>to_scalar_t()</code> [1/9]	207
6.23.2.35 <code>to_scalar_t()</code> [2/9]	207
6.23.2.36 <code>to_scalar_t()</code> [3/9]	207
6.23.2.37 <code>to_scalar_t()</code> [4/9]	207
6.23.2.38 <code>to_scalar_t()</code> [5/9]	208
6.23.2.39 <code>to_scalar_t()</code> [6/9]	208
6.23.2.40 <code>to_scalar_t()</code> [7/9]	208
6.23.2.41 <code>to_scalar_t()</code> [8/9]	208
6.23.2.42 <code>to_scalar_t()</code> [9/9]	209
6.24 <code>glucat::numeric_traits&lt; Scalar_T &gt;::promoted Struct Reference</code>	209
6.24.1 Detailed Description	209
6.24.2 Member Typedef Documentation	209
6.24.2.1 <code>type</code>	209
6.25 <code>glucat::random_generator&lt; Scalar_T &gt; Class Template Reference</code>	210
6.25.1 Detailed Description	210
6.25.2 Constructor & Destructor Documentation	211

6.25.2.1	<a href="#">random_generator()</a> [1/2]	211
6.25.2.2	<a href="#">random_generator()</a> [2/2]	211
6.25.2.3	<a href="#">~random_generator()</a>	211
6.25.3	<a href="#">Member Function Documentation</a>	211
6.25.3.1	<a href="#">generator()</a>	211
6.25.3.2	<a href="#">normal()</a>	212
6.25.3.3	<a href="#">operator=()</a>	212
6.25.3.4	<a href="#">uniform()</a>	212
6.25.4	<a href="#">Friends And Related Function Documentation</a>	212
6.25.4.1	<a href="#">friend_for_private_destructor</a>	212
6.25.5	<a href="#">Member Data Documentation</a>	212
6.25.5.1	<a href="#">normal_dist</a>	213
6.25.5.2	<a href="#">seed</a>	213
6.25.5.3	<a href="#">uint_gen</a>	213
6.25.5.4	<a href="#">uniform_dist</a>	213
6.26	<a href="#">glucat::index_set&lt; LO, HI &gt;::reference Class Reference</a>	214
6.26.1	<a href="#">Detailed Description</a>	215
6.26.2	<a href="#">Constructor &amp; Destructor Documentation</a>	215
6.26.2.1	<a href="#">reference()</a> [1/2]	215
6.26.2.2	<a href="#">reference()</a> [2/2]	215
6.26.2.3	<a href="#">~reference()</a>	216
6.26.3	<a href="#">Member Function Documentation</a>	216
6.26.3.1	<a href="#">flip()</a>	216
6.26.3.2	<a href="#">operator bool()</a>	216
6.26.3.3	<a href="#">operator=()</a> [1/2]	216
6.26.3.4	<a href="#">operator=()</a> [2/2]	217
6.26.3.5	<a href="#">operator~()</a>	217
6.26.4	<a href="#">Friends And Related Function Documentation</a>	217
6.26.4.1	<a href="#">index_set</a>	217
6.26.5	<a href="#">Member Data Documentation</a>	217

6.26.5.1	<a href="#">m_idx</a>	217
6.26.5.2	<a href="#">m_pst</a>	218
6.27	<a href="#">glucat::sorted_range&lt; Map_T, Sorted_Map_T &gt; Class Template Reference</a>	218
6.27.1	<a href="#">Detailed Description</a>	218
6.27.2	<a href="#">Member Typedef Documentation</a>	218
6.27.2.1	<a href="#">map_t</a>	219
6.27.2.2	<a href="#">sorted_iterator</a>	219
6.27.2.3	<a href="#">sorted_map_t</a>	219
6.27.3	<a href="#">Constructor &amp; Destructor Documentation</a>	219
6.27.3.1	<a href="#">sorted_range()</a>	219
6.27.4	<a href="#">Member Data Documentation</a>	219
6.27.4.1	<a href="#">sorted_begin</a>	220
6.27.4.2	<a href="#">sorted_end</a>	220
6.28	<a href="#">glucat::sorted_range&lt; Sorted_Map_T, Sorted_Map_T &gt; Class Template Reference</a>	220
6.28.1	<a href="#">Detailed Description</a>	220
6.28.2	<a href="#">Member Typedef Documentation</a>	221
6.28.2.1	<a href="#">map_t</a>	221
6.28.2.2	<a href="#">sorted_iterator</a>	221
6.28.2.3	<a href="#">sorted_map_t</a>	221
6.28.3	<a href="#">Constructor &amp; Destructor Documentation</a>	221
6.28.3.1	<a href="#">sorted_range()</a>	221
6.28.4	<a href="#">Member Data Documentation</a>	221
6.28.4.1	<a href="#">sorted_begin</a>	222
6.28.4.2	<a href="#">sorted_end</a>	222
6.29	<a href="#">glucat::tuning&lt; Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_↵ Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision &gt; Struct Template Reference</a>	222
6.29.1	<a href="#">Detailed Description</a>	223
6.29.2	<a href="#">Member Enumeration Documentation</a>	223
6.29.2.1	<a href="#">anonymous enum</a>	223
6.29.2.2	<a href="#">anonymous enum</a>	223

6.29.2.3	<a href="#">anonymous enum</a>	224
6.29.2.4	<a href="#">anonymous enum</a>	224
6.29.2.5	<a href="#">anonymous enum</a>	225
6.29.2.6	<a href="#">anonymous enum</a>	225
6.29.2.7	<a href="#">anonymous enum</a>	226
6.29.2.8	<a href="#">anonymous enum</a>	226
6.29.2.9	<a href="#">anonymous enum</a>	226
6.29.3	<a href="#">Member Data Documentation</a>	227
6.29.3.1	<a href="#">function_precision</a>	227
6.30	<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;::var_term Class Reference</a>	227
6.30.1	<a href="#">Detailed Description</a>	228
6.30.2	<a href="#">Member Typedef Documentation</a>	229
6.30.2.1	<a href="#">var_pair_t</a>	229
6.30.3	<a href="#">Constructor &amp; Destructor Documentation</a>	229
6.30.3.1	<a href="#">~var_term()</a>	229
6.30.3.2	<a href="#">var_term() [1/2]</a>	229
6.30.3.3	<a href="#">var_term() [2/2]</a>	229
6.30.4	<a href="#">Member Function Documentation</a>	230
6.30.4.1	<a href="#">classname()</a>	230
6.30.4.2	<a href="#">operator*=( )</a>	230

<b>7 File Documentation</b>	<b>231</b>
7.1 glucat/clifford_algebra.h File Reference	231
7.1.1 Macro Definition Documentation	238
7.1.1.1 _GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS	238
7.2 glucat/clifford_algebra_imp.h File Reference	238
7.3 glucat/errors.h File Reference	245
7.4 glucat/errors_imp.h File Reference	246
7.5 glucat/framed_multi.h File Reference	246
7.5.1 Macro Definition Documentation	249
7.5.1.1 _GLUCAT_MAP_IS_HASH	249
7.6 glucat/framed_multi_imp.h File Reference	249
7.6.1 Macro Definition Documentation	251
7.6.1.1 _GLUCAT_HASH_N	252
7.6.1.2 _GLUCAT_HASH_SIZE_T	252
7.7 glucat/generation.h File Reference	252
7.8 glucat/generation_imp.h File Reference	254
7.9 glucat/global.h File Reference	254
7.9.1 Macro Definition Documentation	257
7.9.1.1 _GLUCAT_CTAssert	257
7.10 glucat/glucat.h File Reference	257
7.11 glucat/glucat_config.h File Reference	258
7.11.1 Macro Definition Documentation	259
7.11.1.1 GLUCAT_HAVE_INTTYPES_H	259
7.11.1.2 GLUCAT_HAVE_MEMORY_H	259
7.11.1.3 GLUCAT_HAVE_STDINT_H	259
7.11.1.4 GLUCAT_HAVE_STDLIB_H	260
7.11.1.5 GLUCAT_HAVE_STRING_H	260
7.11.1.6 GLUCAT_HAVE_STRINGS_H	260
7.11.1.7 GLUCAT_HAVE_SYS_STAT_H	260
7.11.1.8 GLUCAT_HAVE_SYS_TYPES_H	260

7.11.1.9	GLUCAT_HAVE_UNISTD_H	260
7.11.1.10	GLUCAT_PACKAGE	261
7.11.1.11	GLUCAT_PACKAGE_BUGREPORT	261
7.11.1.12	GLUCAT_PACKAGE_NAME	261
7.11.1.13	GLUCAT_PACKAGE_STRING	261
7.11.1.14	GLUCAT_PACKAGE_TARNAME	261
7.11.1.15	GLUCAT_PACKAGE_URL	261
7.11.1.16	GLUCAT_PACKAGE_VERSION	262
7.11.1.17	GLUCAT_STDC_HEADERS	262
7.11.1.18	GLUCAT_VERSION	262
7.12	glucat/glucat_imp.h File Reference	262
7.13	glucat/index_set.h File Reference	263
7.14	glucat/index_set_imp.h File Reference	265
7.15	glucat/long_double.h File Reference	266
7.16	glucat/matrix.h File Reference	267
7.17	glucat/matrix_imp.h File Reference	269
7.18	glucat/matrix_multi.h File Reference	271
7.19	glucat/matrix_multi_imp.h File Reference	274
7.20	glucat/portability.h File Reference	276
7.20.1	Macro Definition Documentation	277
7.20.1.1	_GLUCAT_ISINF	277
7.20.1.2	_GLUCAT_ISNAN	277
7.20.1.3	UBLAS_ABS	278
7.20.1.4	UBLAS_SQRT	278
7.21	glucat/qd.h File Reference	278
7.22	glucat/random.h File Reference	279
7.23	glucat/scalar.h File Reference	280
7.24	glucat/scalar_imp.h File Reference	282
7.25	pyclical/glucat.pxd File Reference	283
7.26	pyclical/PyClical.cpp File Reference	283

7.26.1	Macro Definition Documentation . . . . .	283
7.26.1.1	PY_SSIZE_T_CLEAN . . . . .	284
7.27	pyclical/PyClical.h File Reference . . . . .	284
7.27.1	Typedef Documentation . . . . .	285
7.27.1.1	Clifford . . . . .	285
7.27.1.2	IndexSet . . . . .	285
7.27.1.3	scalar_t . . . . .	285
7.27.1.4	String . . . . .	286
7.27.1.5	Tune_P . . . . .	286
7.27.2	Function Documentation . . . . .	286
7.27.2.1	clifford_to_repr() . . . . .	286
7.27.2.2	clifford_to_str() . . . . .	286
7.27.2.3	index_set_to_repr() . . . . .	287
7.27.2.4	index_set_to_str() . . . . .	287
7.27.2.5	PyFloat_FromDouble() . . . . .	287
7.27.3	Variable Documentation . . . . .	287
7.27.3.1	hi_ndx . . . . .	287
7.27.3.2	lo_ndx . . . . .	288
7.28	pyclical/PyClical.pxd File Reference . . . . .	288
7.29	pyclical/PyClical.pyx File Reference . . . . .	288
7.30	pyclical/PyClical_nocython.cpp File Reference . . . . .	289
7.30.1	Macro Definition Documentation . . . . .	289
7.30.1.1	PY_SSIZE_T_CLEAN . . . . .	289
7.31	test/control.h File Reference . . . . .	289
7.32	test/driver.h File Reference . . . . .	290
7.33	test/timing.h File Reference . . . . .	290
7.34	test/try_catch.h File Reference . . . . .	291
7.35	test/tuning.h File Reference . . . . .	292
7.35.1	Macro Definition Documentation . . . . .	293
7.35.1.1	__TEST_TUNING_DEFAULT_CONSTANT . . . . .	293



7.35.2	Typedef Documentation . . . . .	293
7.35.2.1	precision_t . . . . .	293
7.35.2.2	Tune_P . . . . .	293
7.35.3	Function Documentation . . . . .	293
7.35.3.1	__TEST_TUNING_DEFAULT_CONSTANT() [1/9] . . . . .	294
7.35.3.2	__TEST_TUNING_DEFAULT_CONSTANT() [2/9] . . . . .	294
7.35.3.3	__TEST_TUNING_DEFAULT_CONSTANT() [3/9] . . . . .	294
7.35.3.4	__TEST_TUNING_DEFAULT_CONSTANT() [4/9] . . . . .	294
7.35.3.5	__TEST_TUNING_DEFAULT_CONSTANT() [5/9] . . . . .	294
7.35.3.6	__TEST_TUNING_DEFAULT_CONSTANT() [6/9] . . . . .	294
7.35.3.7	__TEST_TUNING_DEFAULT_CONSTANT() [7/9] . . . . .	294
7.35.3.8	__TEST_TUNING_DEFAULT_CONSTANT() [8/9] . . . . .	295
7.35.3.9	__TEST_TUNING_DEFAULT_CONSTANT() [9/9] . . . . .	295
7.35.3.10	_GLUCAT_CTAssert() . . . . .	295
7.35.4	Variable Documentation . . . . .	295
7.35.4.1	Test_Tuning_Function_Precision . . . . .	295
7.35.4.2	Test_Tuning_Max_Threshold . . . . .	295
7.36	test/undefine.h File Reference . . . . .	295
<b>Index</b>		<b>297</b>



# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">cga3</a>	Definitions for 3D Conformal Geometric Algebra [DL]	9
<a href="#">glucat</a>		10
<a href="#">glucat::gen</a>		63
<a href="#">glucat::matrix</a>		64
<a href="#">glucat::timing</a>		70
<a href="#">PyClical</a>		72
<a href="#">std</a>		78



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

bitset	
glucat::index_set< LO, HI > . . . . .	150
glucat::bool_to_type< truth_value > . . . . .	82
cdef	
PyClical.clifford . . . . .	83
PyClical.index_set . . . . .	165
Clifford	
PyClical.clifford . . . . .	83
glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T > . . . . .	101
glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > > . . . . .	101
glucat::framed_multi< Scalar_T, LO, HI > . . . . .	124
glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > > . . . . .	101
glucat::matrix_multi< Scalar_T, LO, HI > . . . . .	177
glucat::compare_types< LHS_T, RHS_T > . . . . .	112
glucat::compare_types< T, T > . . . . .	113
glucat::control_t . . . . .	113
glucat::CTAssertion< bool > . . . . .	118
glucat::CTAssertion< true > . . . . .	119
glucat::numeric_traits< Scalar_T >::demoted<> . . . . .	119
glucat::matrix::eig_genus< Matrix_T > . . . . .	120
glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t . . . . .	149
glucat::index_set_hash< LO, HI > . . . . .	175
IndexSet	
PyClical.index_set . . . . .	165
inline	
PyClical.clifford . . . . .	83
PyClical.index_set . . . . .	165
logic_error	
glucat::glucat_error . . . . .	146
glucat::error< Class_T > . . . . .	122
map	
glucat::basis_table< Scalar_T, LO, HI, Matrix_T > . . . . .	79
glucat::gen::generator_table< Matrix_T > . . . . .	142
numeric_limits	
std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > > . . . . .	194

std::numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > > . . . . .	195
glucat::numeric_traits< Scalar_T > . . . . .	196
obj	
PyClical.clifford . . . . .	83
PyClical.index_set . . . . .	165
pair	
glucat::framed_multi< Scalar_T, LO, HI >::var_term . . . . .	227
glucat::numeric_traits< Scalar_T >::promoted . . . . .	209
glucat::random_generator< Scalar_T > . . . . .	210
glucat::index_set< LO, HI >::reference . . . . .	214
glucat::sorted_range< Map_T, Sorted_Map_T > . . . . .	218
glucat::sorted_range< Sorted_Map_T, Sorted_Map_T > . . . . .	220
toClifford	
PyClical.clifford . . . . .	83
toIndexSet	
PyClical.index_set . . . . .	165
glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision > . . . . .	222
unordered_map	
glucat::framed_multi< Scalar_T, LO, HI > . . . . .	124

## Chapter 3

# Class Index

### 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">glucat::basis_table&lt; Scalar_T, LO, HI, Matrix_T &gt;</a>	79
Table of basis elements used as a cache by basis_element()	
<a href="#">glucat::bool_to_type&lt; truth_value &gt;</a>	82
Bool to type	
<a href="#">PyClical.clifford</a>	83
<a href="#">glucat::clifford_algebra&lt; Scalar_T, Index_Set_T, Multivector_T &gt;</a>	
Clifford_algebra<> declares the operations of a Clifford algebra	101
<a href="#">glucat::compare_types&lt; LHS_T, RHS_T &gt;</a>	
Type comparison	112
<a href="#">glucat::compare_types&lt; T, T &gt;</a>	113
<a href="#">glucat::control_t</a>	
Parameters to control tests	113
<a href="#">glucat::CTAssertion&lt; bool &gt;</a>	
Compile time assertion	118
<a href="#">glucat::CTAssertion&lt; true &gt;</a>	119
<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;::demoted&lt;&gt;</a>	
Demoted type for long double	119
<a href="#">glucat::matrix::eig_genus&lt; Matrix_T &gt;</a>	
Structure containing classification of eigenvalues	120
<a href="#">glucat::error&lt; Class_T &gt;</a>	
Specific exception class	122
<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;</a>	
A framed_multi<Scalar_T,LO,HI> is a framed approximation to a multivector	124
<a href="#">glucat::gen::generator_table&lt; Matrix_T &gt;</a>	
Table of generators for specific signatures	142
<a href="#">glucat::glucat_error</a>	
Abstract exception class	146
<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;::hash_size_t</a>	149
<a href="#">glucat::index_set&lt; LO, HI &gt;</a>	
Index set class based on std::bitset<> in Gnu standard C++ library	150
<a href="#">PyClical.index_set</a>	165
<a href="#">glucat::index_set_hash&lt; LO, HI &gt;</a>	175
<a href="#">glucat::matrix_multi&lt; Scalar_T, LO, HI &gt;</a>	
A matrix_multi<Scalar_T,LO,HI> is a matrix approximation to a multivector	177
<a href="#">std::numeric_limits&lt; glucat::framed_multi&lt; Scalar_T, LO, HI &gt; &gt;</a>	
Numeric limits for framed_multi inherit limits for the corresponding scalar type	194

<a href="#">std::numeric_limits&lt; glucat::matrix_multi&lt; Scalar_T, LO, HI &gt; &gt;</a>	
Numeric limits for matrix_multi inherit limits for the corresponding scalar type . . . . .	195
<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;</a>	
Extra traits which extend numeric limits . . . . .	196
<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;::promoted</a>	
Promoted type . . . . .	209
<a href="#">glucat::random_generator&lt; Scalar_T &gt;</a>	
Random number generator with single instance per Scalar_T . . . . .	210
<a href="#">glucat::index_set&lt; LO, HI &gt;::reference</a>	
Index set member reference . . . . .	214
<a href="#">glucat::sorted_range&lt; Map_T, Sorted_Map_T &gt;</a>	
Sorted range for use with output . . . . .	218
<a href="#">glucat::sorted_range&lt; Sorted_Map_T, Sorted_Map_T &gt;</a>	220
<a href="#">glucat::tuning&lt; Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis</a>	
Tuning policy . . . . .	222
<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;::var_term</a>	
Variable term . . . . .	227



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

glucat/clifford_algebra.h	231
glucat/clifford_algebra_imp.h	238
glucat/errors.h	245
glucat/errors_imp.h	246
glucat/framed_multi.h	246
glucat/framed_multi_imp.h	249
glucat/generation.h	252
glucat/generation_imp.h	254
glucat/global.h	254
glucat/glucat.h	257
glucat/glucat_config.h	258
glucat/glucat_imp.h	262
glucat/index_set.h	263
glucat/index_set_imp.h	265
glucat/long_double.h	266
glucat/matrix.h	267
glucat/matrix_imp.h	269
glucat/matrix_multi.h	271
glucat/matrix_multi_imp.h	274
glucat/portability.h	276
glucat/qd.h	278
glucat/random.h	279
glucat/scalar.h	280
glucat/scalar_imp.h	282
pyclical/glucat.pxd	283
pyclical/PyClical.cpp	283
pyclical/PyClical.h	284
pyclical/PyClical.pxd	288
pyclical/PyClical.pyx	288
pyclical/PyClical_nocython.cpp	289
test/control.h	289
test/driver.h	290
test/timing.h	290
test/try_catch.h	291
test/tuning.h	292
test/undefine.h	295



## Chapter 5

# Namespace Documentation

### 5.1 cga3 Namespace Reference

Definitions for 3D Conformal Geometric Algebra [DL].

#### Functions

- `template<typename Multivector_T >`  
`Multivector_T cga3 (const Multivector_T &x)`  
*Convert Euclidean 3D vector to Conformal Geometric Algebra null vector [DL (10.50)].*
- `template<typename Multivector_T >`  
`Multivector_T cga3std (const Multivector_T &X)`  
*Convert CGA3 null vector to standard Conformal Geometric Algebra null vector [DL (10.52)].*
- `template<typename Multivector_T >`  
`Multivector_T agc3 (const Multivector_T &X)`  
*Convert CGA3 null vector to Euclidean 3D vector [DL (10.50)].*

#### 5.1.1 Detailed Description

Definitions for 3D Conformal Geometric Algebra [DL].

#### 5.1.2 Function Documentation

##### 5.1.2.1 agc3()

```
template<typename Multivector_T >
Multivector_T cga3::agc3 (
    const Multivector_T & X ) [inline]
```

Convert CGA3 null vector to Euclidean 3D vector [DL (10.50)].

Definition at line 138 of file PyClical.h.

References `cga3std()`, `PyClical::cl`, and `PyClical::ist`.

### 5.1.2.2 cga3()

```
template<typename Multivector_T >
Multivector_T cga3::cga3 (
    const Multivector_T & x ) [inline]
```

Convert Euclidean 3D vector to Conformal Geometric Algebra null vector [DL (10.50)].

Definition at line 115 of file PyClical.h.

References PyClical::cl, PyClical::ist, and PyClical::ninf3.

### 5.1.2.3 cga3std()

```
template<typename Multivector_T >
Multivector_T cga3::cga3std (
    const Multivector_T & X ) [inline]
```

Convert CGA3 null vector to standard Conformal Geometric Algebra null vector [DL (10.52)].

Definition at line 126 of file PyClical.h.

References PyClical::cl, PyClical::ist, and PyClical::ninf3.

Referenced by agc3().

## 5.2 glucat Namespace Reference

### Namespaces

- [gen](#)
- [matrix](#)
- [timing](#)

### Classes

- class [basis\\_table](#)  
*Table of basis elements used as a cache by basis\_element()*
- class [bool\\_to\\_type](#)  
*Bool to type.*
- class [clifford\\_algebra](#)  
*clifford\_algebra<> declares the operations of a Clifford algebra*
- class [compare\\_types](#)  
*Type comparison.*
- class [compare\\_types< T, T >](#)
- class [control\\_t](#)  
*Parameters to control tests.*
- struct [CTAssertion](#)

- *Compile time assertion.*
- struct [CTAssertion< true >](#)
- class [error](#)
- *Specific exception class.*
- class [framed\\_multi](#)
- *A framed\_multi<Scalar\_T,LO,HI> is a framed approximation to a multivector.*
- class [glucat\\_error](#)
- *Abstract exception class.*
- class [index\\_set](#)
- *Index set class based on std::bitset<> in Gnu standard C++ library.*
- class [index\\_set\\_hash](#)
- class [matrix\\_multi](#)
- *A matrix\_multi<Scalar\_T,LO,HI> is a matrix approximation to a multivector.*
- class [numeric\\_traits](#)
- *Extra traits which extend numeric limits.*
- class [random\\_generator](#)
- *Random number generator with single instance per Scalar\_T.*
- class [sorted\\_range](#)
- *Sorted range for use with output.*
- class [sorted\\_range< Sorted\\_Map\\_T, Sorted\\_Map\\_T >](#)
- struct [tuning](#)
- *Tuning policy.*

## Typedefs

- typedef int [index\\_t](#)
- *Size of index\_t should be enough to represent LO, HI.*
- typedef unsigned long [set\\_value\\_t](#)
- *Size of set\_value\_t should be enough to contain index\_set<LO,HI>*
- typedef int(\* [intfn](#)) ()
- *For exception catching: pointer to function returning int.*
- typedef int(\* [intintfn](#)) (int)
- *For exception catching: pointer to function of int returning int.*

## Enumerations

- enum [precision\\_t](#) { [precision\\_demoted](#), [precision\\_same](#), [precision\\_promoted](#) }
- *Precision policy.*

## Functions

- template<template< typename, const index\_t, const index\_t > class Multivector, template< typename, const index\_t, const index\_t > class RHS, typename Scalar\_T , const index\_t LO, const index\_t HI>  
bool [operator!=](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const RHS< Scalar\_T, LO, HI > &rhs)
- *Test for inequality of multivectors.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
bool [operator!=](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const Scalar\_T &scr)
- *Test for inequality of multivector and scalar.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`bool operator!= (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Test for inequality of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator+ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`  
*Geometric sum of multivector and scalar.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator+ (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Geometric sum of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator+ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Geometric sum.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator- (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`  
*Geometric difference of multivector and scalar.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator- (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Geometric difference of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator- (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Geometric difference.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator* (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`  
*Product of multivector and scalar.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator* (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Product of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator* (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Geometric product.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator^ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Outer product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator & (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Inner product.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator% (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Left contraction.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T star (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Hestenes scalar product.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`  
*Quotient of multivector and scalar.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator/ (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Quotient of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Geometric quotient.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > operator| (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Transformation via twisted adjoint action.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > inv (const Multivector< Scalar_T, LO, HI > &val)`  
*Geometric multiplicative inverse.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)`  
*Integer power of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > pow (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Multivector power of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > outer\_pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)`  
*Outer product power of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T scalar (const Multivector< Scalar_T, LO, HI > &val)`  
*Scalar part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T real (const Multivector< Scalar_T, LO, HI > &val)`  
*Real part: synonym for scalar part.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T imag (const Multivector< Scalar_T, LO, HI > &val)`  
*Imaginary part: deprecated (always 0)*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > pure (const Multivector< Scalar_T, LO, HI > &val)`  
*Pure part.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > even (const Multivector< Scalar_T, LO, HI > &val)`  
*Even part.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > odd (const Multivector< Scalar_T, LO, HI > &val)`  
*Odd part.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const std::vector< Scalar_T > vector_part (const Multivector< Scalar_T, LO, HI > &val)`  
*Vector part of multivector, as a vector\_t with respect to frame()*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > involute (const Multivector< Scalar_T, LO, HI > &val)`  
*Main involution, each {i} is replaced by -{i} in each term, eg. {1}\*{2} -> (-{2})\*(-{1})*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > reverse (const Multivector< Scalar_T, LO, HI > &val)`  
*Reversion, eg. {1}\*{2} -> {2}\*{1}.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > conj (const Multivector< Scalar_T, LO, HI > &val)`  
*Conjugation, rev o invo == invo o rev.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T quad (const Multivector< Scalar_T, LO, HI > &val)`  
*Scalar\_T quadratic form == (rev(x)\*x)(0)*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T norm (const Multivector< Scalar_T, LO, HI > &val)`  
*Scalar\_T norm == sum of norm of coordinates.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T abs (const Multivector< Scalar_T, LO, HI > &val)`  
*Absolute value == sqrt(norm)*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T max_abs (const Multivector< Scalar_T, LO, HI > &val)`  
*Maximum of absolute values of components of multivector: multivector infinity norm.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > complexifier (const Multivector< Scalar_T, LO, HI > &val)`



*Square root of -1 which commutes with all members of the frame of the given multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > elliptic (const Multivector< Scalar_T, LO, HI > &val)`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > sqrt (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`

*Square root of multivector with specified complexifier.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > sqrt (const Multivector< Scalar_T, LO, HI > &val)`

*Square root of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > clifford\_exp (const Multivector< Scalar_T, LO, HI > &val)`

*Exponential of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > log (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`

*Natural logarithm of multivector with specified complexifier.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > log (const Multivector< Scalar_T, LO, HI > &val)`

*Natural logarithm of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > cos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`

*Cosine of multivector with specified complexifier.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > cos (const Multivector< Scalar_T, LO, HI > &val)`

*Cosine of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > acos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`

*Inverse cosine of multivector with specified complexifier.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > acos (const Multivector< Scalar_T, LO, HI > &val)`

*Inverse cosine of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > cosh (const Multivector< Scalar_T, LO, HI > &val)`

*Hyperbolic cosine of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > acosh (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`

*Inverse hyperbolic cosine of multivector with specified complexifier.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > acosh (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse hyperbolic cosine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > sin (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Sine of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > sin (const Multivector< Scalar_T, LO, HI > &val)`  
*Sine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > asin (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Inverse sine of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > asin (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse sine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > sinh (const Multivector< Scalar_T, LO, HI > &val)`  
*Hyperbolic sine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > asinh (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Inverse hyperbolic sine of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > asinh (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse hyperbolic sine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > tan (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Tangent of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > tan (const Multivector< Scalar_T, LO, HI > &val)`  
*Tangent of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > atan (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Inverse tangent of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > atan (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse tangent of multivector.*

- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [tanh](#) (const Multivector< Scalar\_T, LO, HI > &val)  
*Hyperbolic tangent of multivector.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [atanh](#) (const Multivector< Scalar\_T, LO, HI > &val, const Multivector< Scalar\_T, LO, HI > &i, const bool prechecked=false)  
*Inverse hyperbolic tangent of multivector with specified complexifier.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [atanh](#) (const Multivector< Scalar\_T, LO, HI > &val)  
*Inverse hyperbolic tangent of multivector.*
- template<template< typename, const index\_t, const index\_t > class Multivector, template< typename, const index\_t, const index\_t > class RHS, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [operator &](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const RHS< Scalar\_T, LO, HI > &rhs)  
*Inner product.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 static void [check\\_complex](#) (const Multivector< Scalar\_T, LO, HI > &val, const Multivector< Scalar\_T, LO, HI > &i, const bool prechecked=false)  
*Check that i is a valid complexifier for val.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const [framed\\_multi](#)< Scalar\_T, LO, HI > [operator\\*](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Geometric product.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const [framed\\_multi](#)< Scalar\_T, LO, HI > [operator^](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Outer product.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const [framed\\_multi](#)< Scalar\_T, LO, HI > [operator &](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Inner product.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const [framed\\_multi](#)< Scalar\_T, LO, HI > [operator%](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Left contraction.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 Scalar\_T [star](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Hestenes scalar product.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const [framed\\_multi](#)< Scalar\_T, LO, HI > [operator/](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Geometric quotient.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const [framed\\_multi](#)< Scalar\_T, LO, HI > [operator|](#) (const [framed\\_multi](#)< Scalar\_T, LO, HI > &lhs, const [framed\\_multi](#)< Scalar\_T, LO, HI > &rhs)  
*Transformation via twisted adjoint action.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 std::istream & [operator>>](#) (std::istream &s, [framed\\_multi](#)< Scalar\_T, LO, HI > &val)  
*Read multivector from input.*
- template<typename Scalar\_T , const index\_t LO, const index\_t HI>  
 std::ostream & [operator<<](#) (std::ostream &os, const [framed\\_multi](#)< Scalar\_T, LO, HI > &val)

*Write multivector to output.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
std::ostream & operator<< (std::ostream &os, const std::pair< const index_set< LO, HI >, Scalar_T >  
&term)`

*Write term to output.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
const framed_multi< Scalar_T, LO, HI > exp (const framed_multi< Scalar_T, LO, HI > &val)`

*Exponential of multivector.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
static Scalar_T crd_of_mult (const std::pair< const index_set< LO, HI >, Scalar_T > &lhs, const std::pair<  
const index_set< LO, HI >, Scalar_T > &rhs)`

*Coordinate of product of terms.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
const std::pair< const index_set< LO, HI >, Scalar_T > operator* (const std::pair< const index_set< LO,  
HI >, Scalar_T > &lhs, const std::pair< const index_set< LO, HI >, Scalar_T > &rhs)`

*Product of terms.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
const framed_multi< Scalar_T, LO, HI > sqrt (const framed_multi< Scalar_T, LO, HI > &val, const  
framed_multi< Scalar_T, LO, HI > &i, bool prechecked)`

*Square root of multivector with specified complexifier.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
const framed_multi< Scalar_T, LO, HI > log (const framed_multi< Scalar_T, LO, HI > &val, const  
framed_multi< Scalar_T, LO, HI > &i, bool prechecked)`

*Natural logarithm of multivector with specified complexifier.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
const framed_multi< Scalar_T, LO, HI > operator & (const framed_multi< Scalar_T, LO, HI > &lhs, const  
framed_multi< Scalar_T, LO, HI > &rhs)`

*Inner product.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>  
static Scalar_T crd_of_mult (const std::pair< const index_set< LO, HI >, Scalar_T > &lhs, const std::pair<  
const index_set< LO, HI >, Scalar_T > &rhs)`

*Coordinate of product of terms.*

- `_GLUCAT_CTAssert (std::numeric_limits< unsigned char >::radix==2, CannotDetermineBitsPerChar) const  
index_t BITS_PER_CHAR`

*If radix of unsigned char is not 2, we can't easily determine number of bits from sizeof.*

- `_GLUCAT_CTAssert ( _GLUCAT_BITS_PER_ULONG==BITS_PER_SET_VALUE, BitsPerULongDoesNot  
MatchSetValueT) const index_t DEFAULT_LO`

*Default lowest index in an index set.*

- `template<typename LHS_T , typename RHS_T >  
LHS_T pos_mod (LHS_T lhs, RHS_T rhs)`

*Modulo function which works reliably for lhs < 0.*

- `template<const index_t LO, const index_t HI>  
const index_set< LO, HI > operator^ (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`

*Symmetric set difference: exclusive or.*

- `template<const index_t LO, const index_t HI>  
const index_set< LO, HI > operator & (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`

*Set intersection: and.*

- `template<const index_t LO, const index_t HI>  
const index_set< LO, HI > operator| (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`

*Set union: or.*

- `template<const index_t LO, const index_t HI>  
int compare (const index_set< LO, HI > &a, const index_set< LO, HI > &b)`

*"lexicographic compare" eg. {3,4,5} is less than {3,7,8}*

- `_GLUCAT_CTAssert` (`sizeof(set_value_t) >= sizeof(std::bitset< DEFAULT_HI-DEFAULT_LO >)`), Default↔  
`_index_set_too_big_for_value` template< const `index_t` LO  
*Size of set\_value\_t should be enough to contain bitset<DEFAULT\_HI-DEFAULT\_LO>*
- const `index_t` HI `std::ostream & operator<<` (`std::ostream &os`, const `index_set`< LO, HI > &ist)  
*Write out index set.*
- template<const `index_t` LO, const `index_t` HI>  
`std::istream & operator>>` (`std::istream &s`, `index_set`< LO, HI > &ist)  
*Read in index set.*
- int `sign_of_square` (`index_t` j)  
*Square of generator {j}.*
- template<const `index_t` LO, const `index_t` HI>  
`index_t min_neg` (const `index_set`< LO, HI > &ist)  
*Minimum negative index, or 0 if none.*
- template<const `index_t` LO, const `index_t` HI>  
`index_t max_pos` (const `index_set`< LO, HI > &ist)  
*Maximum positive index, or 0 if none.*
- template<const `index_t` LO, const `index_t` HI>  
const `index_set`< LO, HI > `operator &` (const `index_set`< LO, HI > &lhs, const `index_set`< LO, HI > &rhs)  
*Set intersection: and.*
- static unsigned long `inverse_reversed_gray` (unsigned long x)  
*Inverse reversed Gray code.*
- static unsigned long `inverse_gray` (unsigned long x)  
*Inverse Gray code.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
const `matrix_multi`< Scalar\_T, LO, HI > `operator*` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const  
`matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Geometric product.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
const `matrix_multi`< Scalar\_T, LO, HI > `operator^` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const  
`matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Outer product.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
const `matrix_multi`< Scalar\_T, LO, HI > `operator &` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const  
`matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Inner product.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
const `matrix_multi`< Scalar\_T, LO, HI > `operator%` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const  
`matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Left contraction.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
Scalar\_T `star` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const `matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Hestenes scalar product.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
const `matrix_multi`< Scalar\_T, LO, HI > `operator/` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const  
`matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Geometric quotient.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
const `matrix_multi`< Scalar\_T, LO, HI > `operator|` (const `matrix_multi`< Scalar\_T, LO, HI > &lhs, const  
`matrix_multi`< Scalar\_T, LO, HI > &rhs)  
*Transformation via twisted adjoint action.*
- template<typename Scalar\_T , const `index_t` LO, const `index_t` HI>  
`std::istream & operator>>` (`std::istream &s`, `matrix_multi`< Scalar\_T, LO, HI > &val)  
*Read multivector from input.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::ostream & operator<< (std::ostream &os, const matrix\_multi< Scalar_T, LO, HI > &val)`  
*Write multivector to output.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const index\_set< LO, HI > reframe (const matrix\_multi< Scalar_T, LO, HI > &lhs, const matrix\_multi< Scalar_T, LO, HI > &rhs, matrix\_multi< Scalar_T, LO, HI > &lhs_reframed, matrix\_multi< Scalar_T, LO, HI > &rhs_reframed)`  
*Find a common frame for operands of a binary operator.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix\_multi< Scalar_T, LO, HI > sqrt (const matrix\_multi< Scalar_T, LO, HI > &val, const matrix\_multi< Scalar_T, LO, HI > &i, bool prechecked)`  
*Square root of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix\_multi< Scalar_T, LO, HI > matrix\_sqrt (const matrix\_multi< Scalar_T, LO, HI > &val, const matrix\_multi< Scalar_T, LO, HI > &i)`  
*Square root of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix\_multi< Scalar_T, LO, HI > log (const matrix\_multi< Scalar_T, LO, HI > &val, const matrix\_multi< Scalar_T, LO, HI > &i, bool prechecked)`  
*Natural logarithm of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix\_multi< Scalar_T, LO, HI > matrix\_log (const matrix\_multi< Scalar_T, LO, HI > &val, const matrix\_multi< Scalar_T, LO, HI > &i)`  
*Natural logarithm of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix\_multi< Scalar_T, LO, HI > exp (const matrix\_multi< Scalar_T, LO, HI > &val)`  
*Exponential of multivector.*
- `index\_t offset\_level (const index\_t p, const index\_t q)`  
*Determine the log2 dim corresponding to signature p, q.*
- `template<typename Matrix_Index_T , const index_t LO, const index_t HI>`  
`static Matrix_Index_T folded\_dim (const index\_set< LO, HI > &sub)`  
*Determine the matrix dimension of the fold of a subalegebra.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix\_multi< Scalar_T, LO, HI > operator & (const matrix\_multi< Scalar_T, LO, HI > &lhs, const matrix\_multi< Scalar_T, LO, HI > &rhs)`  
*Inner product.*
- `template<typename Multivector_T , typename Matrix_T , typename Basis_Matrix_T >`  
`static Multivector_T fast (const Matrix_T &X, index\_t level)`  
*Inverse generalized Fast Fourier Transform.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix\_multi< Scalar_T, LO, HI > pade\_approx (const int array_size, const Scalar_T a[], const Scalar_T b[], const matrix\_multi< Scalar_T, LO, HI > &X)`  
*Pade' approximation.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static void db\_step (matrix\_multi< Scalar_T, LO, HI > &M, matrix\_multi< Scalar_T, LO, HI > &Y)`  
*Single step of product form of Denman-Beavers square root iteration.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix\_multi< Scalar_T, LO, HI > db\_sqrt (const matrix\_multi< Scalar_T, LO, HI > &val)`  
*Product form of Denman-Beavers square root iteration.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix\_multi< Scalar_T, LO, HI > pade\_log (const matrix\_multi< Scalar_T, LO, HI > &val)`  
*Pade' approximation of log.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix\_multi< Scalar_T, LO, HI > cascade\_log (const matrix\_multi< Scalar_T, LO, HI > &val)`

- Incomplete square root cascade and Pade' approximation of log.*
  - template<typename Scalar\_T >  
 Scalar\_T [log2](#) (const Scalar\_T &x)  
*Log base 2 of scalar.*
- template<typename Scalar\_T >  
[numeric\\_traits](#)< Scalar\_T >::promoted::type [to\\_promote](#) (const Scalar\_T &val)  
*Cast to promote.*
- template<typename Scalar\_T >  
[numeric\\_traits](#)< Scalar\_T >::demoted::type [to\\_demote](#) (const Scalar\_T &val)  
*Cast to demote.*
- int [try\\_catch](#) (intfn f)  
*Exception catching for functions returning int.*
- int [try\\_catch](#) (intintfn f, int arg)  
*Exception catching for functions of int returning int.*

## Variables

- const double [MS\\_PER\\_S](#) = 1000.0  
*Timing constant: deprecated here - moved to [test/timing.h](#).*
- const [index\\_t](#) [BITS\\_PER\\_SET\\_VALUE](#) = std::numeric\_limits<[set\\_value\\_t](#)>::digits  
*Number of bits in [set\\_value\\_t](#).*
- const [index\\_t](#) [DEFAULT\\_HI](#) = [index\\_t](#)([BITS\\_PER\\_SET\\_VALUE](#) / 2)  
*Default highest index in an index set.*
- const double [DEFAULT\\_TRUNCATION](#) = std::numeric\_limits<float>::epsilon()  
*Default for truncation.*
- const unsigned int [DEFAULT\\_Mult\\_Matrix\\_Threshold](#) = 8
- const unsigned int [DEFAULT\\_Div\\_Max\\_Steps](#) = 4
- const unsigned int [DEFAULT\\_Sqrt\\_Max\\_Steps](#) = 256
- const unsigned int [DEFAULT\\_Log\\_Max\\_Outer\\_Steps](#) = 256
- const unsigned int [DEFAULT\\_Log\\_Max\\_Inner\\_Steps](#) = 32
- const unsigned int [DEFAULT\\_Basis\\_Max\\_Count](#) = 12
- const unsigned int [DEFAULT\\_Fast\\_Size\\_Threshold](#) = 1 << 6
- const unsigned int [DEFAULT\\_Inv\\_Fast\\_Dim\\_Threshold](#) = 1 << 3
- const unsigned int [DEFAULT\\_Products\\_Size\\_Threshold](#) = 1 << 22
- const [precision\\_t](#) [DEFAULT\\_Function\\_Precision](#) = [precision\\_same](#)
- static const long double [I\\_pi](#) = 3.1415926535897932384626433832795029L
- static const long double [I\\_ln2](#) = 0.6931471805599453094172321214581766L

### 5.2.1 Typedef Documentation

#### 5.2.1.1 [index\\_t](#)

```
typedef int glucat::index\_t
```

Size of [index\\_t](#) should be enough to represent LO, HI.

Definition at line 77 of file [global.h](#).

### 5.2.1.2 intfn

```
typedef int(* glucat::intfn) ()
```

For exception catching: pointer to function returning int.

Definition at line 37 of file try\_catch.h.

### 5.2.1.3 intintfn

```
typedef int(* glucat::intintfn) (int)
```

For exception catching: pointer to function of int returning int.

Definition at line 40 of file try\_catch.h.

### 5.2.1.4 set\_value\_t

```
typedef unsigned long glucat::set_value_t
```

Size of set\_value\_t should be enough to contain index\_set<LO,HI>

Definition at line 79 of file global.h.

## 5.2.2 Enumeration Type Documentation

### 5.2.2.1 precision\_t

```
enum glucat::precision_t
```

Precision policy.

Enumerator

precision_demoted	
precision_same	
precision_promoted	

Definition at line 117 of file global.h.



### 5.2.3 Function Documentation

#### 5.2.3.1 `_GLUCAT_CTAssert()` [1/3]

```
glucat::_GLUCAT_CTAssert (
    std::numeric_limits< unsigned char >::radix  == 2,
    CannotDetermineBitsPerChar ) const
```

If radix of unsigned char is not 2, we can't easily determine number of bits from sizeof.

Number of bits per char is used to determine number of bits in `set_value_t`

#### 5.2.3.2 `_GLUCAT_CTAssert()` [2/3]

```
glucat::_GLUCAT_CTAssert (
    _GLUCAT_BITS_PER_ULONG  == BITS_PER_SET_VALUE,
    BitsPerUlongDoesNotMatchSetValueT ) const
```

Default lowest index in an index set.

#### 5.2.3.3 `_GLUCAT_CTAssert()` [3/3]

```
glucat::_GLUCAT_CTAssert (
    sizeof(set_value_t) >= sizeof(std::bitset< DEFAULT_HI-DEFAULT_LO > ) ,
    Default_index_set_too_big_for_value ) const
```

Size of `set_value_t` should be enough to contain `bitset<DEFAULT_HI-DEFAULT_LO>`

Write out index set

#### 5.2.3.4 `abs()`

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↔
_T , const index_t LO, const index_t HI>
Scalar_T glucat::abs (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Absolute value == `sqrt(norm)`

Definition at line 491 of file `clifford_algebra_imp.h`.

References `glucat::numeric_traits< Scalar_T >::sqrt()`.

Referenced by `PyClical.clifford::abs()`, `acos()`, `asin()`, `glucat::matrix::classify_eigenvalues()`, `clifford_to_str()`, `glucat::framed_multi< Scalar_T, LO, HI >::framed_multi()`, `matrix_log()`, and `matrix_sqrt()`.

**5.2.3.5** `acos()` [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::acos (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse cosine of multivector with specified complexifier.

Definition at line 798 of file `clifford_algebra_imp.h`.

References `abs()`, `acosh()`, `check_complex()`, and `PyClical::i`.

Referenced by `glucat::numeric_traits< Scalar_T >::acos()`, and `acos()`.

**5.2.3.6** `acos()` [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::acos (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Inverse cosine of multivector.

Definition at line 818 of file `clifford_algebra_imp.h`.

References `acos()`, and `complexifier()`.

**5.2.3.7** `acosh()` [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::acosh (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse hyperbolic cosine of multivector with specified complexifier.

Definition at line 738 of file `clifford_algebra_imp.h`.

References `check_complex()`, `PyClical::i`, `log()`, `norm()`, and `sqrt()`.

Referenced by `acos()`, and `acosh()`.

**5.2.3.8 acosh()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::acosh (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Inverse hyperbolic cosine of multivector.

Definition at line 758 of file clifford\_algebra\_imp.h.

References `acosh()`, and `complexifier()`.

**5.2.3.9 asin()** [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::asin (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse sine of multivector with specified complexifier.

Definition at line 905 of file clifford\_algebra\_imp.h.

References `abs()`, `asinh()`, `check_complex()`, and `PyClical::i`.

Referenced by `glucat::numeric_traits< Scalar_T >::asin()`, and `asin()`.

**5.2.3.10 asin()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::asin (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Inverse sine of multivector.

Definition at line 925 of file clifford\_algebra\_imp.h.

References `asin()`, and `complexifier()`.

**5.2.3.11 asinh()** [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::asinh (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse hyperbolic sine of multivector with specified complexifier.

Definition at line 845 of file `clifford_algebra_imp.h`.

References `check_complex()`, `PyClical::i`, `log()`, `norm()`, and `sqrt()`.

Referenced by `asin()`, and `asinh()`.

**5.2.3.12 asinh()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::asinh (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Inverse hyperbolic sine of multivector.

Definition at line 865 of file `clifford_algebra_imp.h`.

References `asinh()`, and `complexifier()`.

**5.2.3.13 atan()** [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::atan (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse tangent of multivector with specified complexifier.

Definition at line 1005 of file `clifford_algebra_imp.h`.

References `atanh()`, `check_complex()`, `PyClical::i`, and `scalar()`.

Referenced by `glucat::numeric_traits< Scalar_T >::atan()`, and `atan()`.

**5.2.3.14 atan()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::atan (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Inverse tangent of multivector.

Definition at line 1025 of file clifford\_algebra\_imp.h.

References atan(), and complexifier().

**5.2.3.15 atanh()** [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::atanh (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse hyperbolic tangent of multivector with specified complexifier.

Definition at line 952 of file clifford\_algebra\_imp.h.

References check\_complex(), PyClical::i, log(), and norm().

Referenced by atan(), and atanh().

**5.2.3.16 atanh()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::atanh (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Inverse hyperbolic tangent of multivector.

Definition at line 969 of file clifford\_algebra\_imp.h.

References atanh(), and complexifier().

### 5.2.3.17 cascade\_log()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix_multi<Scalar_T,LO,HI> glucat::cascade_log (
    const matrix_multi< Scalar_T, LO, HI > & val ) [static]
```

Incomplete square root cascade and Pade' approximation of log.

Definition at line 1976 of file matrix\_multi\_imp.h.

References db\_step(), glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::isnan(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::log\_max\_inner\_steps, glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::log\_max\_outer\_steps, norm(), pade\_log(), and pow().

Referenced by matrix\_log().

### 5.2.3.18 check\_complex()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
static void glucat::check_complex (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline], [static]
```

Check that i is a valid complexifier for val.

Definition at line 566 of file clifford\_algebra\_imp.h.

References complexifier(), and PyClical::i.

Referenced by acos(), acosh(), asin(), asinh(), atan(), atanh(), cos(), log(), sin(), sqrt(), and tan().

### 5.2.3.19 clifford\_exp()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::clifford_exp (
    const Multivector< Scalar_T, LO, HI > & val )
```

Exponential of multivector.

Definition at line 604 of file clifford\_algebra\_imp.h.

References exp(), log2(), pow(), and scalar().

Referenced by exp().

### 5.2.3.20 compare()

```
template<const index_t LO, const index_t HI>
int glucat::compare (
    const index_set< LO, HI > & a,
    const index_set< LO, HI > & b ) [inline]
```

"lexicographic compare" eg. {3,4,5} is less than {3,7,8}

Lexicographic ordering of two sets: -1 if  $a < b$ , +1 if  $a > b$ , 0 if  $a == b$ .

Definition at line 573 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::lex\_less\_than().

### 5.2.3.21 complexifier()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::complexifier (
    const Multivector< Scalar_T, LO, HI > & val )
```

Square root of -1 which commutes with all members of the frame of the given multivector.

Definition at line 506 of file clifford\_algebra\_imp.h.

References pos\_mod().

Referenced by acos(), acosh(), asin(), asinh(), atan(), atanh(), check\_complex(), cos(), elliptic(), log(), sin(), sqrt(), and tan().

### 5.2.3.22 conj()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::conj (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Conjugation,  $rev \circ inv == inv \circ rev$ .

Definition at line 467 of file clifford\_algebra\_imp.h.

**5.2.3.23 cos()** [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::cos (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false )
```

Cosine of multivector with specified complexifier.

Definition at line 765 of file `clifford_algebra_imp.h`.

References `check_complex()`, `exp()`, `PyClical::i`, `PyClical::pi`, and `scalar()`.

Referenced by `glucat::numeric_traits< Scalar_T >::cos()`, `cos()`, and `tan()`.

**5.2.3.24 cos()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::cos (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Cosine of multivector.

Definition at line 789 of file `clifford_algebra_imp.h`.

References `complexifier()`, and `cos()`.

**5.2.3.25 cosh()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::cosh (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Hyperbolic cosine of multivector.

Definition at line 720 of file `clifford_algebra_imp.h`.

References `exp()`, and `scalar()`.

Referenced by `glucat::numeric_traits< Scalar_T >::cosh()`, and `tanh()`.



**5.2.3.26** `crd_of_mult()` [1/2]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static Scalar_T glucat::crd_of_mult (
    const std::pair< const index\_set< LO, HI >, Scalar_T > & lhs,
    const std::pair< const index\_set< LO, HI >, Scalar_T > & rhs ) [inline], [static]
```

Coordinate of product of terms.

Referenced by operator &(), operator%(), operator\*(), and operator^().

**5.2.3.27** `crd_of_mult()` [2/2]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static Scalar_T glucat::crd_of_mult (
    const std::pair< const index\_set< LO, HI >, Scalar_T > & lhs,
    const std::pair< const index\_set< LO, HI >, Scalar_T > & rhs ) [inline], [static]
```

Coordinate of product of terms.

Definition at line 1906 of file `framed_multi_imp.h`.

**5.2.3.28** `db_sqrt()`

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix\_multi<Scalar_T,LO,HI> glucat::db_sqrt (
    const matrix\_multi< Scalar_T, LO, HI > & val ) [static]
```

Product form of Denman-Beavers square root iteration.

Definition at line 1375 of file `matrix_multi_imp.h`.

References `db_step()`, `norm()`, `pow()`, and `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::sqrt_max_steps`.

Referenced by `matrix_sqrt()`.

**5.2.3.29** `db_step()`

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static void glucat::db_step (
    matrix\_multi< Scalar_T, LO, HI > & M,
    matrix\_multi< Scalar_T, LO, HI > & Y ) [inline], [static]
```

Single step of product form of Denman-Beavers square root iteration.

Definition at line 1362 of file `matrix_multi_imp.h`.

References `inv()`.

Referenced by `cascade_log()`, and `db_sqrt()`.

### 5.2.3.30 elliptic()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::elliptic (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Square root of -1 which commutes with all members of the frame of the given multivector The name "elliptic" is now deprecated: use "complexifier" instead.

Definition at line 557 of file clifford\_algebra\_imp.h.

References complexifier().

### 5.2.3.31 even()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::even (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Even part.

Definition at line 427 of file clifford\_algebra\_imp.h.

### 5.2.3.32 exp() [1/2]

```
template<typename Scalar_T, const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::exp (
    const framed_multi< Scalar_T, LO, HI > & val )
```

Exponential of multivector.

Definition at line 1947 of file framed\_multi\_imp.h.

References clifford\_exp(), glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::frame(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::function\_precision, glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::isnan(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::mult\_matrix\_threshold, precision\_demoted, precision\_promoted, and scalar().

Referenced by clifford\_exp(), cos(), cosh(), glucat::numeric\_traits< Scalar\_T >::exp(), exp(), matrix\_log(), matrix\_sqrt(), pow(), PyClical.clifford::pow(), sin(), and sinh().

**5.2.3.33 exp()** [2/2]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::exp (
    const matrix_multi< Scalar_T, LO, HI > & val )
```

Exponential of multivector.

Definition at line 2140 of file matrix\_multi\_imp.h.

References `clifford_exp()`, `exp()`, `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::function_precision`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::isnan()`, `precision_demoted`, `precision_promoted`, and `scalar()`.

**5.2.3.34 fast()**

```
template<typename Multivector_T , typename Matrix_T , typename Basis_Matrix_T >
static Multivector_T glucat::fast (
    const Matrix_T & X,
    index_t level ) [static]
```

Inverse generalized Fast Fourier Transform.

Definition at line 1083 of file matrix\_multi\_imp.h.

References `glucat::matrix::signed_perm_nork()`.

**5.2.3.35 folded\_dim()**

```
template<typename Matrix_Index_T , const index_t LO, const index_t HI>
static Matrix_Index_T glucat::folded_dim (
    const index_set< LO, HI > & sub ) [inline], [static]
```

Determine the matrix dimension of the fold of a subalgebra.

Definition at line 91 of file matrix\_multi\_imp.h.

References `glucat::index_set< LO, HI >::count_neg()`, `glucat::index_set< LO, HI >::count_pos()`, and `offset_level()`.

**5.2.3.36 imag()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar←
_T , const index_t LO, const index_t HI>
Scalar_T glucat::imag (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Imaginary part: deprecated (always 0)

Definition at line 411 of file clifford\_algebra\_imp.h.

Referenced by glucat::matrix::classify\_eigenvalues().

**5.2.3.37 inv()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar←
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::inv (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Geometric multiplicative inverse.

Definition at line 321 of file clifford\_algebra\_imp.h.

Referenced by db\_step(), matrix\_log(), and matrix\_sqrt().

**5.2.3.38 inverse\_gray()**

```
static unsigned long glucat::inverse_gray (
    unsigned long x ) [inline], [static]
```

Inverse Gray code.

Definition at line 861 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::sign\_of\_mult().

**5.2.3.39 inverse\_reversed\_gray()**

```
static unsigned long glucat::inverse_reversed_gray (
    unsigned long x ) [inline], [static]
```

Inverse reversed Gray code.

Definition at line 844 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::sign\_of\_mult().

## 5.2.3.40 involute()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::involute (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Main involution, each {i} is replaced by -{i} in each term, eg. {1}\*{2} -> (-{2})\*(-{1})

Main involution, each {i} is replaced by -{i} in each term, eg. {1} -> -{1}.

Definition at line 451 of file clifford\_algebra\_imp.h.

## 5.2.3.41 log() [1/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::log (
    const matrix_multi< Scalar_T, LO, HI > & val,
    const matrix_multi< Scalar_T, LO, HI > & i,
    bool prechecked )
```

Natural logarithm of multivector with specified complexifier.

Definition at line 2023 of file matrix\_multi\_imp.h.

References `check_complex()`, `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::function_precision`, `PyClical::i`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::isnan()`, `matrix_log()`, `precision_demoted`, and `precision_promoted`.

## 5.2.3.42 log() [2/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::log (
    const framed_multi< Scalar_T, LO, HI > & val,
    const framed_multi< Scalar_T, LO, HI > & i,
    bool prechecked )
```

Natural logarithm of multivector with specified complexifier.

Definition at line 1997 of file framed\_multi\_imp.h.

References `check_complex()`, `PyClical::i`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::isnan()`, `log()`, `PyClical::pi`, and `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::scalar()`.

**5.2.3.43** `log()` [3/4]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::log (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Natural logarithm of multivector with specified complexifier.

Definition at line 704 of file `clifford_algebra_imp.h`.

References `PyClical::i`.

Referenced by `acosh()`, `asinh()`, `atanh()`, `glucat::numeric_traits< Scalar_T >::log()`, `log()`, `matrix_log()`, `glucat↵
::numeric_traits< Scalar_T >::NaN()`, `pow()`, and `PyClical.clifford::pow()`.

**5.2.3.44** `log()` [4/4]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::log (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Natural logarithm of multivector.

Definition at line 712 of file `clifford_algebra_imp.h`.

References `complexifier()`, and `log()`.

**5.2.3.45** `log2()`

```
template<typename Scalar_T >
Scalar_T glucat::log2 (
    const Scalar_T & x ) [inline]
```

Log base 2 of scalar.

Definition at line 302 of file `scalar.h`.

References `glucat::numeric_traits< Scalar_T >::log2()`.

Referenced by `clifford_exp()`.

## 5.2.3.46 matrix\_log()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::matrix_log (
    const matrix_multi< Scalar_T, LO, HI > & val,
    const matrix_multi< Scalar_T, LO, HI > & i )
```

Natural logarithm of multivector with specified complexifier.

Definition at line 2064 of file matrix\_multi\_imp.h.

References `abs()`, `glucat::matrix::both_eig_case`, `cascade_log()`, `glucat::matrix::classify_eigenvalues()`, `exp()`, `PyClicl::i`, `inv()`, `glucat::matrix::isnan()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::isnan()`, `log()`, `glucat::matrix::eig_genus< Matrix_T >::m_eig_case`, `glucat::matrix::eig_genus< Matrix_T >::m_safe_arg`, `glucat::matrix::negative_eig_case`, `norm()`, `PyClicl::pi`, and `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::scalar()`.

Referenced by `log()`.

## 5.2.3.47 matrix\_sqrt()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::matrix_sqrt (
    const matrix_multi< Scalar_T, LO, HI > & val,
    const matrix_multi< Scalar_T, LO, HI > & i )
```

Square root of multivector with specified complexifier.

Definition at line 1645 of file matrix\_multi\_imp.h.

References `abs()`, `glucat::matrix::both_eig_case`, `glucat::matrix::classify_eigenvalues()`, `db_sqrt()`, `exp()`, `PyClicl::i`, `inv()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::isnan()`, `glucat::matrix::eig_genus< Matrix_T >::m_eig_case`, `glucat::matrix::eig_genus< Matrix_T >::m_safe_arg`, `glucat::matrix::negative_eig_case`, `norm()`, `pade_approx()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::scalar()`, and `sqrt()`.

Referenced by `sqrt()`.

## 5.2.3.48 max\_abs()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::max_abs (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Maximum of absolute values of components of multivector: multivector infinity norm.

Definition at line 499 of file clifford\_algebra\_imp.h.

**5.2.3.49 max\_pos()**

```
template<const index_t LO, const index_t HI>
index_t glucat::max_pos (
    const index_set< LO, HI > & ist ) [inline]
```

Maximum positive index, or 0 if none.

Definition at line 974 of file index\_set\_imp.h.

References PyClical::ist.

**5.2.3.50 min\_neg()**

```
template<const index_t LO, const index_t HI>
index_t glucat::min_neg (
    const index_set< LO, HI > & ist ) [inline]
```

Minimum negative index, or 0 if none.

Definition at line 967 of file index\_set\_imp.h.

References PyClical::ist.

**5.2.3.51 norm()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
Scalar_T glucat::norm (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Scalar\_T norm == sum of norm of coordinates.

Definition at line 483 of file clifford\_algebra\_imp.h.

Referenced by acosh(), asinh(), atanh(), cascade\_log(), glucat::matrix::classify\_eigenvalues(), db\_sqrt(), matrix\_↵log(), and matrix\_sqrt().

**5.2.3.52 odd()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::odd (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Odd part.

Definition at line 435 of file clifford\_algebra\_imp.h.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast().



**5.2.3.53 offset\_level()**

```
index_t glucat::offset_level (
    const index_t p,
    const index_t q ) [inline]
```

Determine the log2 dim corresponding to signature p, q.

Definition at line 76 of file matrix\_multi\_imp.h.

References pos\_mod().

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element(), and folded\_dim().

**5.2.3.54 operator &() [1/8]**

```
template<const index_t LO, const index_t HI>
const index_set<LO,HI> glucat::operator& (
    const index_set< LO, HI > & lhs,
    const index_set< LO, HI > & rhs ) [inline]
```

Set intersection: and.

Definition at line 186 of file index\_set\_imp.h.

**5.2.3.55 operator &() [2/8]**

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi<Scalar_T,LO,HI> glucat::operator& (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Inner product.

Definition at line 616 of file matrix\_multi\_imp.h.

**5.2.3.56 operator &() [3/8]**

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi<Scalar_T,LO,HI> glucat::operator& (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs )
```

Inner product.

Definition at line 601 of file framed\_multi\_imp.h.

References \_GLUCAT\_HASH\_SIZE\_T, crd\_of\_mult(), glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::frame(), and glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::products\_size\_threshold.

**5.2.3.57 operator &()** [4/8]

```
template<const index_t LO, const index_t HI>
const index_set<LO,HI> glucat::operator& (
    const index_set< LO, HI > & lhs,
    const index_set< LO, HI > & rhs ) [inline]
```

Set intersection: and.

Definition at line 186 of file index\_set\_imp.h.

**5.2.3.58 operator &()** [5/8]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector<Scalar_T,LO,HI> glucat::operator& (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Inner product.

Definition at line 228 of file clifford\_algebra\_imp.h.

**5.2.3.59 operator &()** [6/8]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector<Scalar_T,LO,HI> glucat::operator& (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Inner product.

Definition at line 228 of file clifford\_algebra\_imp.h.

**5.2.3.60 operator &()** [7/8]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi<Scalar_T,LO,HI> glucat::operator& (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs )
```

Inner product.

Definition at line 601 of file framed\_multi\_imp.h.

References `_GLUCAT_HASH_SIZE_T`, `crd_of_mult()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::frame()`, and `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::products_size_threshold`.

**5.2.3.61 operator &()** [8/8]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi<Scalar_T,LO,HI> glucat::operator& (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Inner product.

Definition at line 616 of file matrix\_multi\_imp.h.

**5.2.3.62 operator!=(())** [1/3]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
bool glucat::operator!=( (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Test for inequality of multivectors.

Definition at line 78 of file clifford\_algebra\_imp.h.

**5.2.3.63 operator!=(())** [2/3]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
bool glucat::operator!=( (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Test for inequality of multivector and scalar.

Definition at line 86 of file clifford\_algebra\_imp.h.

**5.2.3.64 operator!=(())** [3/3]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
bool glucat::operator!=( (
    const Scalar_T & scr,
    const Multivector< Scalar_T, LO, HI > & rhs ) [inline]
```

Test for inequality of scalar and multivector.

Definition at line 94 of file clifford\_algebra\_imp.h.

**5.2.3.65 operator%()** [1/3]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator% (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Left contraction.

Definition at line 635 of file matrix\_multi\_imp.h.

**5.2.3.66 operator%()** [2/3]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::operator% (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs )
```

Left contraction.

Definition at line 719 of file framed\_multi\_imp.h.

References `_GLUCAT_HASH_SIZE_T`, `crd_of_mult()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::frame()`, and `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt↵_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_↵Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::products_size_threshold`.

**5.2.3.67 operator%()** [3/3]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator% (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Left contraction.

Definition at line 243 of file clifford\_algebra\_imp.h.

**5.2.3.68 operator\*()** [1/6]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator* (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric product.

Definition at line 547 of file matrix\_multi\_imp.h.

References glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::isnan(), glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix, and reframe().

**5.2.3.69 operator\*()** [2/6]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::operator* (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs )
```

Geometric product.

Definition at line 400 of file framed\_multi\_imp.h.

References \_GLUCAT\_HASH\_SIZE\_T, glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::frame(), glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::isnan(), and glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::mult\_matrix\_threshold.

**5.2.3.70 operator\*()** [3/6]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator* (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Product of multivector and scalar.

Definition at line 172 of file clifford\_algebra\_imp.h.

**5.2.3.71 operator\*()** [4/6]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator* (
    const Scalar_T & scr,
    const Multivector< Scalar_T, LO, HI > & rhs ) [inline]
```

Product of scalar and multivector.

Definition at line 183 of file clifford\_algebra\_imp.h.

**5.2.3.72 operator\*()** [5/6]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T, const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator* (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric product.

Definition at line 198 of file clifford\_algebra\_imp.h.

**5.2.3.73 operator\*()** [6/6]

```
template<typename Scalar_T, const index_t LO, const index_t HI>
const std::pair< const index_set< LO, HI >, Scalar_T > glucat::operator* (
    const std::pair< const index_set< LO, HI >, Scalar_T > & lhs,
    const std::pair< const index_set< LO, HI >, Scalar_T > & rhs ) [inline]
```

Product of terms.

Definition at line 1914 of file framed\_multi\_imp.h.

References `crd_of_mult()`.

**5.2.3.74 operator+()** [1/3]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator+ (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Geometric sum of multivector and scalar.

Definition at line 102 of file clifford\_algebra\_imp.h.

**5.2.3.75 operator+()** [2/3]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator+ (
    const Scalar_T & scr,
    const Multivector< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric sum of scalar and multivector.

Definition at line 113 of file clifford\_algebra\_imp.h.

**5.2.3.76 operator+()** [3/3]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator+ (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric sum.

Definition at line 127 of file clifford\_algebra\_imp.h.

**5.2.3.77 operator-()** [1/3]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator- (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Geometric difference of multivector and scalar.

Definition at line 138 of file clifford\_algebra\_imp.h.

**5.2.3.78 operator-()** [2/3]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator- (
    const Scalar_T & scr,
    const Multivector< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric difference of scalar and multivector.

Definition at line 149 of file clifford\_algebra\_imp.h.

**5.2.3.79 operator-()** [3/3]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator- (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric difference.

Definition at line 161 of file clifford\_algebra\_imp.h.

**5.2.3.80 operator/()** [1/5]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator/ (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs )
```

Geometric quotient.

Definition at line 668 of file matrix\_multi\_imp.h.

References glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::div\_max\_steps, glucat::matrix::isnan(), glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::isnan(), glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, and reframe().

**5.2.3.81 operator/()** [2/5]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::operator/ (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric quotient.

Definition at line 914 of file framed\_multi\_imp.h.

References glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::frame().



**5.2.3.82 operator/()** [3/5]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator/ (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Quotient of multivector and scalar.

Definition at line 269 of file clifford\_algebra\_imp.h.

**5.2.3.83 operator/()** [4/5]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator/ (
    const Scalar_T & scr,
    const Multivector< Scalar_T, LO, HI > & rhs ) [inline]
```

Quotient of scalar and multivector.

Definition at line 280 of file clifford\_algebra\_imp.h.

**5.2.3.84 operator/()** [5/5]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator/ (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Geometric quotient.

Definition at line 295 of file clifford\_algebra\_imp.h.

**5.2.3.85 operator<<()** [1/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
std::ostream & glucat::operator<< (
    std::ostream & os,
    const matrix\_multi< Scalar_T, LO, HI > & val ) [inline]
```

Write multivector to output.

Definition at line 1025 of file matrix\_multi\_imp.h.

**5.2.3.86 operator<<()** [2/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
std::ostream & glucat::operator<< (
    std::ostream & os,
    const framed_multi< Scalar_T, LO, HI > & val )
```

Write multivector to output.

Definition at line 1366 of file framed\_multi\_imp.h.

References glucat::sorted\_range< Map\_T, Sorted\_Map\_T >::sorted\_begin, and glucat::sorted\_range< Map\_T, Sorted\_Map\_T >::sorted\_end.

**5.2.3.87 operator<<()** [3/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
std::ostream & glucat::operator<< (
    std::ostream & os,
    const std::pair< const index_set< LO, HI >, Scalar_T > & term )
```

Write term to output.

Definition at line 1398 of file framed\_multi\_imp.h.

References pow(), and glucat::numeric\_traits< Scalar\_T >::to\_double().

**5.2.3.88 operator<<()** [4/4]

```
std::ostream & glucat::operator<< (
    std::ostream & os,
    const index_set< LO, HI > & ist )
```

Write out index set.

Definition at line 611 of file index\_set\_imp.h.

References PyClical::i, and PyClical::ist.

**5.2.3.89 operator>>()** [1/3]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
std::istream & glucat::operator>> (
    std::istream & s,
    matrix_multi< Scalar_T, LO, HI > & val ) [inline]
```

Read multivector from input.

Definition at line 1035 of file matrix\_multi\_imp.h.

**5.2.3.90 operator>>()** [2/3]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
std::istream & glucat::operator>> (
    std::istream & s,
    framed_multi< Scalar_T, LO, HI > & val )
```

Read multivector from input.

Definition at line 1436 of file framed\_multi\_imp.h.

References PyClical::ist.

**5.2.3.91 operator>>()** [3/3]

```
template<const index_t LO, const index_t HI>
std::istream & glucat::operator>> (
    std::istream & s,
    index_set< LO, HI > & ist )
```

Read in index set.

Definition at line 633 of file index\_set\_imp.h.

References PyClical::i, PyClical::ist, and glucat::index\_set< LO, HI >::set().

**5.2.3.92 operator^()** [1/4]

```
template<const index_t LO, const index_t HI>
const index_set< LO, HI > glucat::operator^ (
    const index_set< LO, HI > & lhs,
    const index_set< LO, HI > & rhs ) [inline]
```

Symmetric set difference: exclusive or.

Definition at line 161 of file index\_set\_imp.h.

**5.2.3.93 operator^()** [2/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator^ (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Outer product.

Definition at line 597 of file matrix\_multi\_imp.h.

**5.2.3.94** `operator^()` [3/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::operator^ (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs )
```

Outer product.

Definition at line 501 of file framed\_multi\_imp.h.

References `_GLUCAT_HASH_SIZE_T`, `crd_of_mult()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::frame()`, and `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_↵_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_↵Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::products_size_threshold`.

**5.2.3.95** `operator^()` [4/4]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator^ (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Outer product.

Definition at line 213 of file clifford\_algebra\_imp.h.

**5.2.3.96** `operator" |()` [1/4]

```
template<const index_t LO, const index_t HI>
const index_set< LO, HI > glucat::operator| (
    const index_set< LO, HI > & lhs,
    const index_set< LO, HI > & rhs ) [inline]
```

Set union: or.

Definition at line 211 of file index\_set\_imp.h.

**5.2.3.97** `operator" |()` [2/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator| (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Transformation via twisted adjoint action.

Definition at line 777 of file matrix\_multi\_imp.h.

References `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >↵::involute()`.

**5.2.3.98 operator" | () [3/4]**

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::operator| (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Transformation via twisted adjoint action.

Definition at line 940 of file framed\_multi\_imp.h.

References glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::involute().

**5.2.3.99 operator" | () [4/4]**

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::operator| (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Transformation via twisted adjoint action.

Definition at line 310 of file clifford\_algebra\_imp.h.

**5.2.3.100 outer\_pow()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::outer_pow (
    const Multivector< Scalar_T, LO, HI > & lhs,
    int rhs )
```

Outer product power of multivector.

Definition at line 384 of file clifford\_algebra\_imp.h.

**5.2.3.101 pade\_approx()**

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix_multi<Scalar_T,LO,HI> glucat::pade_approx (
    const int array_size,
    const Scalar_T a[],
    const Scalar_T b[],
    const matrix_multi< Scalar_T, LO, HI > & X ) [inline], [static]
```

Pade' approximation.

Definition at line 1302 of file matrix\_multi\_imp.h.

References glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::isnan().

Referenced by matrix\_sqrt(), and pade\_log().

### 5.2.3.102 pade\_log()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix_multi<Scalar_T,LO,HI> glucat::pade_log (
    const matrix_multi< Scalar_T, LO, HI > & val ) [static]
```

Pade' approximation of log.

Definition at line 1955 of file matrix\_multi\_imp.h.

References glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::isnan(), and pade\_approx().

Referenced by cascade\_log().

### 5.2.3.103 pos\_mod()

```
template<typename LHS_T , typename RHS_T >
LHS_T glucat::pos_mod (
    LHS_T lhs,
    RHS_T rhs ) [inline]
```

Modulo function which works reliably for lhs < 0.

Definition at line 187 of file global.h.

Referenced by complexifier(), glucat::matrix\_multi< Scalar\_T, LO, HI >::fast\_framed\_multi(), glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi(), glucat::gen::generator\_table< Matrix\_T >::gen\_vector(), offset\_level(), and glucat::gen::generator\_table< Matrix\_T >::operator().

### 5.2.3.104 pow() [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::pow (
    const Multivector< Scalar_T, LO, HI > & lhs,
    int rhs )
```

Integer power of multivector.

Definition at line 328 of file clifford\_algebra\_imp.h.

Referenced by cascade\_log(), clifford\_exp(), db\_sqrt(), operator<<(), and glucat::numeric\_traits< Scalar\_T >::pow().

**5.2.3.105 pow()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::pow (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Multivector power of multivector.

Definition at line 361 of file clifford\_algebra\_imp.h.

References `exp()`, and `log()`.

**5.2.3.106 pure()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::pure (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Pure part.

Definition at line 419 of file clifford\_algebra\_imp.h.

**5.2.3.107 quad()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
Scalar_T glucat::quad (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Scalar\_T quadratic form == (rev(x)\*x)(0)

Definition at line 475 of file clifford\_algebra\_imp.h.

**5.2.3.108 real()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
Scalar_T glucat::real (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Real part: synonym for scalar part.

Definition at line 400 of file clifford\_algebra\_imp.h.

Referenced by `glucat::matrix::classify_eigenvalues()`.

**5.2.3.109   reframe()**

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const index_set< LO, HI > glucat::reframe (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs,
    matrix_multi< Scalar_T, LO, HI > & lhs_reframed,
    matrix_multi< Scalar_T, LO, HI > & rhs_reframed ) [inline]
```

Find a common frame for operands of a binary operator.

Definition at line 350 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame.

Referenced by operator\*(), and operator/().

**5.2.3.110   reverse()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↔
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::reverse (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Reversion, eg. {1}\*{2} -> {2}\*{1}.

Definition at line 459 of file clifford\_algebra\_imp.h.

**5.2.3.111   scalar()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↔
_T , const index_t LO, const index_t HI>
Scalar_T glucat::scalar (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Scalar part.

Definition at line 392 of file clifford\_algebra\_imp.h.

Referenced by atan(), clifford\_exp(), cos(), cosh(), exp(), glucat::framed\_multi< Scalar\_T, LO, HI >::fast(), sin(), sinh(), tan(), and tanh().



**5.2.3.112** `sign_of_square()`

```
int glucat::sign_of_square (
    index_t j ) [inline]
```

Square of generator {j}.

Square of generator index j.

Definition at line 960 of file `index_set_imp.h`.

**5.2.3.113** `sin()` [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↔
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::sin (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false )
```

Sine of multivector with specified complexifier.

Definition at line 872 of file `clifford_algebra_imp.h`.

References `check_complex()`, `exp()`, `PyClical::i`, `PyClical::pi`, and `scalar()`.

Referenced by `glucat::numeric_traits< Scalar_T >::sin()`, `sin()`, and `tan()`.

**5.2.3.114** `sin()` [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↔
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::sin (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Sine of multivector.

Definition at line 896 of file `clifford_algebra_imp.h`.

References `complexifier()`, and `sin()`.

### 5.2.3.115 sinh()

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::sinh (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Hyperbolic sine of multivector.

Definition at line 826 of file `clifford_algebra_imp.h`.

References `exp()`, and `scalar()`.

Referenced by `glucat::numeric_traits< Scalar_T >::sinh()`, and `tanh()`.

### 5.2.3.116 sqrt() [1/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::sqrt (
    const matrix_multi< Scalar_T, LO, HI > & val,
    const matrix_multi< Scalar_T, LO, HI > & i,
    bool prechecked )
```

Square root of multivector with specified complexifier.

Definition at line 1600 of file `matrix_multi_imp.h`.

References `check_complex()`, `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >::function_precision`, `PyClical::i`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, matrix_multi< Scalar_T, LO, HI > >::isnan()`, `matrix_sqrt()`, `precision_demoted`, and `precision_promoted`.

### 5.2.3.117 sqrt() [2/4]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::sqrt (
    const framed_multi< Scalar_T, LO, HI > & val,
    const framed_multi< Scalar_T, LO, HI > & i,
    bool prechecked )
```

Square root of multivector with specified complexifier.

Definition at line 1924 of file `framed_multi_imp.h`.

References `check_complex()`, `PyClical::i`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::isnan()`, `glucat::clifford_algebra< Scalar_T, index_set< LO, HI >, framed_multi< Scalar_T, LO, HI > >::scalar()`, and `sqrt()`.

**5.2.3.118** `sqrt()` [3/4]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::sqrt (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Square root of multivector with specified complexifier.

Definition at line 589 of file `clifford_algebra_imp.h`.

References `PyClical::i`.

Referenced by `acosh()`, `asinh()`, `matrix_sqrt()`, `glucat::framed_multi< Scalar_T, LO, HI >::random()`, and `sqrt()`.

**5.2.3.119** `sqrt()` [4/4]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::sqrt (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Square root of multivector.

Definition at line 597 of file `clifford_algebra_imp.h`.

References `complexifier()`, and `sqrt()`.

**5.2.3.120** `star()` [1/3]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::star (
    const matrix_multi< Scalar_T, LO, HI > & lhs,
    const matrix_multi< Scalar_T, LO, HI > & rhs ) [inline]
```

Hestenes scalar product.

Definition at line 654 of file `matrix_multi_imp.h`.

**5.2.3.121** `star()` [2/3]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::star (
    const framed_multi< Scalar_T, LO, HI > & lhs,
    const framed_multi< Scalar_T, LO, HI > & rhs )
```

Hestenes scalar product.

Definition at line 855 of file `framed_multi_imp.h`.

**5.2.3.122 star()** [3/3]

```
template<template< typename, const index_t, const index_t > class Multivector, template<
typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO,
const index_t HI>
Scalar_T glucat::star (
    const Multivector< Scalar_T, LO, HI > & lhs,
    const RHS< Scalar_T, LO, HI > & rhs ) [inline]
```

Hestenes scalar product.

Definition at line 258 of file clifford\_algebra\_imp.h.

**5.2.3.123 tan()** [1/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::tan (
    const Multivector< Scalar_T, LO, HI > & val,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Tangent of multivector with specified complexifier.

Definition at line 977 of file clifford\_algebra\_imp.h.

References check\_complex(), cos(), PyClical::i, scalar(), and sin().

Referenced by glucat::numeric\_traits< Scalar\_T >::tan(), and tan().

**5.2.3.124 tan()** [2/2]

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar↵
_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::tan (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Tangent of multivector.

Definition at line 996 of file clifford\_algebra\_imp.h.

References complexifier(), and tan().

**5.2.3.125 tanh()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::tanh (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Hyperbolic tangent of multivector.

Definition at line 933 of file clifford\_algebra\_imp.h.

References cosh(), scalar(), and sinh().

Referenced by glucat::numeric\_traits< Scalar\_T >::tanh().

**5.2.3.126 to\_demote()**

```
template<typename Scalar_T >
numeric_traits<Scalar_T>::demoted::type glucat::to_demote (
    const Scalar_T & val ) [inline]
```

Cast to demote.

Definition at line 134 of file scalar\_imp.h.

References glucat::numeric\_traits< Scalar\_T >::to\_scalar\_t().

**5.2.3.127 to\_promote()**

```
template<typename Scalar_T >
numeric_traits<Scalar_T>::promoted::type glucat::to_promote (
    const Scalar_T & val ) [inline]
```

Cast to promote.

Definition at line 124 of file scalar\_imp.h.

References glucat::numeric\_traits< Scalar\_T >::to\_scalar\_t().

**5.2.3.128 try\_catch()** [1/2]

```
int glucat::try_catch (
    intfn f )
```

Exception catching for functions returning int.

Definition at line 49 of file try\_catch.h.

References PyClical::e().

Referenced by glucat::control\_t::call().

**5.2.3.129 try\_catch()** [2/2]

```
int glucat::try_catch (
    int(intfn f,
    int arg )
```

Exception catching for functions of int returning int.

Definition at line 64 of file try\_catch.h.

References PyClical::e().

**5.2.3.130 vector\_part()**

```
template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>
const std::vector< Scalar_T > glucat::vector_part (
    const Multivector< Scalar_T, LO, HI > & val ) [inline]
```

Vector part of multivector, as a vector\_t with respect to frame()

Definition at line 443 of file clifford\_algebra\_imp.h.

**5.2.4 Variable Documentation****5.2.4.1 BITS\_PER\_SET\_VALUE**

```
const index_t glucat::BITS_PER_SET_VALUE = std::numeric_limits<set_value_t>::digits
```

Number of bits in set\_value\_t.

Definition at line 103 of file global.h.

**5.2.4.2 DEFAULT\_Basis\_Max\_Count**

```
const unsigned int glucat::DEFAULT_Basis_Max_Count = 12
```

Definition at line 130 of file global.h.

#### 5.2.4.3 DEFAULT\_Div\_Max\_Steps

```
const unsigned int glucat::DEFAULT_Div_Max_Steps = 4
```

Definition at line 126 of file global.h.

#### 5.2.4.4 DEFAULT\_Fast\_Size\_Threshold

```
const unsigned int glucat::DEFAULT_Fast_Size_Threshold = 1 << 6
```

Definition at line 131 of file global.h.

#### 5.2.4.5 DEFAULT\_Function\_Precision

```
const precision_t glucat::DEFAULT_Function_Precision = precision_same
```

Definition at line 134 of file global.h.

#### 5.2.4.6 DEFAULT\_HI

```
const index_t glucat::DEFAULT_HI = index_t(BITS_PER_SET_VALUE / 2)
```

Default highest index in an index set.

Definition at line 111 of file global.h.

#### 5.2.4.7 DEFAULT\_Inv\_Fast\_Dim\_Threshold

```
const unsigned int glucat::DEFAULT_Inv_Fast_Dim_Threshold = 1 << 3
```

Definition at line 132 of file global.h.

#### 5.2.4.8 DEFAULT\_Log\_Max\_Inner\_Steps

```
const unsigned int glucat::DEFAULT_Log_Max_Inner_Steps = 32
```

Definition at line 129 of file global.h.

#### 5.2.4.9 DEFAULT\_Log\_Max\_Outer\_Steps

```
const unsigned int glucat::DEFAULT_Log_Max_Outer_Steps = 256
```

Definition at line 128 of file global.h.

#### 5.2.4.10 DEFAULT\_Mult\_Matrix\_Threshold

```
const unsigned int glucat::DEFAULT_Mult_Matrix_Threshold = 8
```

Definition at line 125 of file global.h.

#### 5.2.4.11 DEFAULT\_Products\_Size\_Threshold

```
const unsigned int glucat::DEFAULT_Products_Size_Threshold = 1 << 22
```

Definition at line 133 of file global.h.

#### 5.2.4.12 DEFAULT\_Sqrt\_Max\_Steps

```
const unsigned int glucat::DEFAULT_Sqrt_Max_Steps = 256
```

Definition at line 127 of file global.h.

#### 5.2.4.13 DEFAULT\_TRUNCATION

```
const double glucat::DEFAULT_TRUNCATION = std::numeric_limits<float>::epsilon()
```

Default for truncation.

Definition at line 114 of file global.h.

#### 5.2.4.14 l\_ln2

```
const long double glucat::l_ln2 = 0.6931471805599453094172321214581766L [static]
```

Definition at line 41 of file long\_double.h.

Referenced by glucat::numeric\_traits< Scalar\_T >::ln\_2().



## 5.2.4.15 l\_pi

```
const long double glucat::l_pi = 3.1415926535897932384626433832795029L [static]
```

Definition at line 40 of file long\_double.h.

Referenced by glucat::numeric\_traits< Scalar\_T >::pi().

## 5.2.4.16 MS\_PER\_S

```
const double glucat::MS_PER_S = 1000.0
```

Timing constant: deprecated here - moved to [test/timing.h](#).

Definition at line 83 of file global.h.

## 5.3 glucat::gen Namespace Reference

### Classes

- class [generator\\_table](#)  
*Table of generators for specific signatures.*

### Typedefs

- typedef std::pair< [index\\_t](#), [index\\_t](#) > [signature\\_t](#)  
*A signature is a pair of indices, p, q, with p == frame.max(), q == -frame.min()*

### Variables

- static const [index\\_t](#) [offset\\_to\\_super](#) [] = {0,-1, 0,-1,-2, 3, 2, 1}  
*Offsets between the current signature and that of the real superalgebra.*

### 5.3.1 Typedef Documentation

## 5.3.1.1 signature\_t

```
typedef std::pair<index\_t, index\_t> glucat::gen::signature\_t
```

A signature is a pair of indices, p, q, with p == frame.max(), q == -frame.min()

Definition at line 43 of file generation.h.

### 5.3.2 Variable Documentation

#### 5.3.2.1 offset\_to\_super

```
const index_t glucat::gen::offset_to_super[] = {0,-1, 0,-1,-2, 3, 2, 1} [static]
```

Offsets between the current signature and that of the real superalgebra.

Definition at line 81 of file generation.h.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::fast\_framed\_multi(), glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi(), and glucat::gen::generator\_table< Matrix\_T >::operator()().

## 5.4 glucat::matrix Namespace Reference

### Classes

- struct [eig\\_genus](#)  
*Structure containing classification of eigenvalues.*

### Enumerations

- enum [eig\\_case\\_t](#) { [safe\\_eig\\_case](#), [negative\\_eig\\_case](#), [both\\_eig\\_case](#) }  
*Classification of eigenvalues of a matrix.*

### Functions

- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [kron](#) (const LHS\_T &lhs, const RHS\_T &rhs)  
*Kronecker tensor product of matrices - as per Matlab kron.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [mono\\_kron](#) (const LHS\_T &lhs, const RHS\_T &rhs)  
*Sparse Kronecker tensor product of monomial matrices.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [nork](#) (const LHS\_T &lhs, const RHS\_T &rhs, const bool mono=true)  
*Left inverse of Kronecker product.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [signed\\_perm\\_nork](#) (const LHS\_T &lhs, const RHS\_T &rhs)  
*Left inverse of Kronecker product where lhs is a signed permutation matrix.*
- template<typename Matrix\_T >  
Matrix\_T::size\_type [nnz](#) (const Matrix\_T &m)  
*Number of non-zeros.*
- template<typename Matrix\_T >  
bool [isnan](#) (const Matrix\_T &m)  
*Not a Number.*

- `template<typename Matrix_T >`  
`const Matrix_T unit (const typename Matrix_T::size_type n)`  
*Unit matrix - as per Matlab eye.*
- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T::expression_type mono\_prod (const ublas::matrix_expression< LHS_T > &lhs, const ublas::matrix_expression< RHS_T > &rhs)`  
*Product of monomial matrices.*
- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T::expression_type sparse\_prod (const ublas::matrix_expression< LHS_T > &lhs, const ublas::matrix_expression< RHS_T > &rhs)`  
*Product of sparse matrices.*
- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T::expression_type prod (const ublas::matrix_expression< LHS_T > &lhs, const ublas::matrix_expression< RHS_T > &rhs)`  
*Product of matrices.*
- `template<typename Scalar_T , typename LHS_T , typename RHS_T >`  
`Scalar_T inner (const LHS_T &lhs, const RHS_T &rhs)`  
*Inner product:  $\sum(x(i,j)*y(i,j))/x.nrows()$*
- `template<typename Matrix_T >`  
`Matrix_T::value_type norm\_frob2 (const Matrix_T &val)`  
*Square of Frobenius norm.*
- `template<typename Matrix_T >`  
`Matrix_T::value_type trace (const Matrix_T &val)`  
*Matrix trace.*
- `template<typename Matrix_T >`  
`ublas::vector< std::complex< double > > eigenvalues (const Matrix_T &val)`  
*Eigenvalues of a matrix.*
- `template<typename Matrix_T >`  
`eig\_genus< Matrix_T > classify\_eigenvalues (const Matrix_T &val)`  
*Classify the eigenvalues of a matrix.*
- `template<typename LHS_T , typename RHS_T >`  
`void nork\_range (RHS_T &result, const typename LHS_T::const_iterator2 lhs_it2, const RHS_T &rhs, const typename RHS_T::size_type res_s1, const typename RHS_T::size_type res_s2)`  
*Utility routine for nork: calculate result for a range of indices.*
- `template<typename Matrix_T >`  
`static ublas::matrix< double, ublas::column_major > to\_lapack (const Matrix_T &val)`  
*Convert matrix to LAPACK format.*

### 5.4.1 Enumeration Type Documentation

#### 5.4.1.1 `eig_case_t`

enum `glucat::matrix::eig_case_t`

Classification of eigenvalues of a matrix.

Enumerator

<code>safe_eig_case</code>	
<code>negative_eig_case</code>	
<code>both_eig_case</code>	

Definition at line 127 of file matrix.h.

## 5.4.2 Function Documentation

### 5.4.2.1 classify\_eigenvalues()

```
template<typename Matrix_T >
eig_genus< Matrix_T > glucat::matrix::classify_eigenvalues (
    const Matrix_T & val )
```

Classify the eigenvalues of a matrix.

Definition at line 526 of file matrix\_imp.h.

References glucat::abs(), both\_eig\_case, eigenvalues(), glucat::imag(), glucat::matrix::eig\_genus< Matrix\_T >::m\_eig\_case, glucat::matrix::eig\_genus< Matrix\_T >::m\_safe\_arg, negative\_eig\_case, glucat::norm(), glucat::numeric\_traits< Scalar\_T >::pi(), PyClical::pi, glucat::real(), and safe\_eig\_case.

Referenced by glucat::matrix\_log(), and glucat::matrix\_sqrt().

### 5.4.2.2 eigenvalues()

```
template<typename Matrix_T >
ublas::vector< std::complex< double > > glucat::matrix::eigenvalues (
    const Matrix_T & val )
```

Eigenvalues of a matrix.

Definition at line 493 of file matrix\_imp.h.

References to\_lapack().

Referenced by classify\_eigenvalues().

### 5.4.2.3 inner()

```
template<typename Scalar_T , typename LHS_T , typename RHS_T >
Scalar_T glucat::matrix::inner (
    const LHS_T & lhs,
    const RHS_T & rhs )
```

Inner product:  $\sum(x(i,j)*y(i,j))/x.nrows()$

Inner product:  $\sum(lhs(i,j)*rhs(i,j))/lhs.nrows()$

Definition at line 391 of file matrix\_imp.h.

## 5.4.2.4 isnan()

```
template<typename Matrix_T >
bool glucat::matrix::isnan (
    const Matrix_T & m )
```

Not a Number.

Definition at line 292 of file matrix\_imp.h.

Referenced by glucat::matrix\_log(), and glucat::operator/().

## 5.4.2.5 kron()

```
template<typename LHS_T , typename RHS_T >
const RHS_T glucat::matrix::kron (
    const LHS_T & lhs,
    const RHS_T & rhs )
```

Kronecker tensor product of matrices - as per Matlab kron.

Definition at line 73 of file matrix\_imp.h.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast().

## 5.4.2.6 mono\_kron()

```
template<typename LHS_T , typename RHS_T >
const RHS_T glucat::matrix::mono_kron (
    const LHS_T & lhs,
    const RHS_T & rhs )
```

Sparse Kronecker tensor product of monomial matrices.

Definition at line 116 of file matrix\_imp.h.

Referenced by glucat::gen::generator\_table< Matrix\_T >::gen\_from\_pm1\_qm1().

## 5.4.2.7 mono\_prod()

```
template<typename LHS_T , typename RHS_T >
const RHS_T::expression_type glucat::matrix::mono_prod (
    const ublas::matrix_expression< LHS_T > & lhs,
    const ublas::matrix_expression< RHS_T > & rhs )
```

Product of monomial matrices.

Definition at line 326 of file matrix\_imp.h.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element(), glucat::gen::generator\_table< Matrix\_T >::gen\_from\_pm4\_qp4(), glucat::gen::generator\_table< Matrix\_T >::gen\_from\_pp4\_qm4(), and glucat::gen::generator\_table< Matrix\_T >::gen\_from\_qp1\_pm1().

#### 5.4.2.8 nnz()

```
template<typename Matrix_T >
Matrix_T::size_type glucat::matrix::nnz (
    const Matrix_T & m )
```

Number of non-zeros.

Definition at line 269 of file matrix\_imp.h.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi().

#### 5.4.2.9 nork()

```
template<typename LHS_T , typename RHS_T >
const RHS_T glucat::matrix::nork (
    const LHS_T & lhs,
    const RHS_T & rhs,
    const bool mono = true )
```

Left inverse of Kronecker product.

Definition at line 188 of file matrix\_imp.h.

References norm\_frob2().

#### 5.4.2.10 nork\_range()

```
template<typename LHS_T , typename RHS_T >
void glucat::matrix::nork_range (
    RHS_T & result,
    const typename LHS_T::const_iterator2 lhs_it2,
    const RHS_T & rhs,
    const typename RHS_T::size_type res_s1,
    const typename RHS_T::size_type res_s2 )
```

Utility routine for nork: calculate result for a range of indices.

Definition at line 155 of file matrix\_imp.h.

References glucat::numeric\_traits< Scalar\_T >::to\_scalar\_t().

## 5.4.2.11 norm\_frob2()

```
template<typename Matrix_T >
Matrix_T::value_type glucat::matrix::norm_frob2 (
    const Matrix_T & val )
```

Square of Frobenius norm.

Definition at line 413 of file matrix\_imp.h.

Referenced by nork().

## 5.4.2.12 prod()

```
template<typename LHS_T , typename RHS_T >
const RHS_T::expression_type glucat::matrix::prod (
    const ublas::matrix_expression< LHS_T > & lhs,
    const ublas::matrix_expression< RHS_T > & rhs ) [inline]
```

Product of matrices.

Definition at line 373 of file matrix\_imp.h.

## 5.4.2.13 signed\_perm\_nork()

```
template<typename LHS_T , typename RHS_T >
const RHS_T glucat::matrix::signed_perm_nork (
    const LHS_T & lhs,
    const RHS_T & rhs )
```

Left inverse of Kronecker product where lhs is a signed permutation matrix.

Definition at line 237 of file matrix\_imp.h.

Referenced by glucat::fast().

## 5.4.2.14 sparse\_prod()

```
template<typename LHS_T , typename RHS_T >
const RHS_T::expression_type glucat::matrix::sparse_prod (
    const ublas::matrix_expression< LHS_T > & lhs,
    const ublas::matrix_expression< RHS_T > & rhs ) [inline]
```

Product of sparse matrices.

Definition at line 362 of file matrix\_imp.h.

#### 5.4.2.15 to\_lapack()

```
template<typename Matrix_T >
static ublas::matrix<double, ublas::column_major> glucat::matrix::to_lapack (
    const Matrix_T & val ) [static]
```

Convert matrix to LAPACK format.

Definition at line 461 of file matrix\_imp.h.

Referenced by eigenvalues().

#### 5.4.2.16 trace()

```
template<typename Matrix_T >
Matrix_T::value_type glucat::matrix::trace (
    const Matrix_T & val )
```

Matrix trace.

Definition at line 437 of file matrix\_imp.h.

References glucat::numeric\_traits< Scalar\_T >::NaN().

#### 5.4.2.17 unit()

```
template<typename Matrix_T >
const Matrix_T glucat::matrix::unit (
    const typename Matrix_T::size_type n ) [inline]
```

Unit matrix - as per Matlab eye.

Definition at line 317 of file matrix\_imp.h.

## 5.5 glucat::timing Namespace Reference

### Functions

- static double [elapsed](#) (clock\_t cpu\_time)  
*Elapsed time in milliseconds.*

### Variables

- const double [MS\\_PER\\_SEC](#) = 1000.0  
*Timing constant: milliseconds per second.*
- const double [MS\\_PER\\_CLOCK](#) = [MS\\_PER\\_SEC](#) / double(CLOCKS\_PER\_SEC)  
*Timing constant: milliseconds per clock.*
- const int [EXTRA\\_TRIALS](#) = 2  
*Timing constant: trial expansion factor.*



### 5.5.1 Function Documentation

#### 5.5.1.1 elapsed()

```
static double glucat::timing::elapsed (  
    clock_t cpu_time ) [inline], [static]
```

Elapsed time in milliseconds.

Definition at line 51 of file timing.h.

References `MS_PER_CLOCK`.

### 5.5.2 Variable Documentation

#### 5.5.2.1 EXTRA\_TRIALS

```
const int glucat::timing::EXTRA_TRIALS = 2
```

Timing constant: trial expansion factor.

Definition at line 45 of file timing.h.

#### 5.5.2.2 MS\_PER\_CLOCK

```
const double glucat::timing::MS_PER_CLOCK = MS\_PER\_SEC / double(CLOCKS_PER_SEC)
```

Timing constant: milliseconds per clock.

Definition at line 42 of file timing.h.

Referenced by `elapsed()`.

#### 5.5.2.3 MS\_PER\_SEC

```
const double glucat::timing::MS_PER_SEC = 1000.0
```

Timing constant: milliseconds per second.

Definition at line 39 of file timing.h.

## 5.6 PyClical Namespace Reference

### Classes

- class [clifford](#)
- class [index\\_set](#)

### Functions

- def [index\\_set\\_hidden\\_doctests](#) ()
- def [clifford\\_hidden\\_doctests](#) ()
- def [e](#) (obj)
- def [istpq](#) (p, q)
- def [\\_test](#) ()

### Variables

- string [\\_\\_version\\_\\_](#) = "0.8.2"
- [obj](#)
- [i](#)
- [ixt](#)
- [fill](#)
- float [tau](#) = atan([clifford](#)(1.0)) \* 8.0
- float [pi](#) = [tau](#) / 2.0
- [cl](#) = [clifford](#)
- [ist](#) = [index\\_set](#)
- def [ninf3](#) = [e](#)(4) + [e](#)(-1)
- def [nbar3](#) = [e](#)(4) - [e](#)(-1)

### 5.6.1 Function Documentation

#### 5.6.1.1 [\\_test\(\)](#)

```
def PyClical._test ( ) [private]
```

Definition at line 1913 of file PyClical.pyx.

## 5.6.1.2 clifford\_hidden\_doctests()

```
def PyClical.clifford_hidden_doctests ( )
```

Tests for functions that Doctest cannot see.

For clifford.\_\_cinit\_\_: Construct an object of type clifford.

```
>>> print clifford(2)
2
>>> print clifford(2L)
2
>>> print clifford(2.0)
2
>>> print clifford(1.0e-1)
0.1
>>> print clifford("2")
2
>>> print clifford("2{1,2,3}")
2{1,2,3}
>>> print clifford(clifford("2{1,2,3}"))
2{1,2,3}
>>> print clifford("-{1}")
-{1}
>>> print clifford(2,index_set({1,2}))
2{1,2}
>>> print clifford([2,3],index_set({1,2}))
2{1}+3{2}
>>> print clifford([1,2])
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from <type 'list'>.
>>> print clifford(None)
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from <type 'NoneType'>.
>>> print clifford(None,[1,2])
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from (<type 'NoneType'>, <type 'list'>).
>>> print clifford([1,2],[1,2])
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from (<type 'list'>, <type 'list'>).
>>> print clifford("")
Traceback (most recent call last):
...
ValueError: Cannot initialize clifford object from invalid string ''.
>>> print clifford("{")
Traceback (most recent call last):
...
ValueError: Cannot initialize clifford object from invalid string '{'.
>>> print clifford("{1}")
Traceback (most recent call last):
...
ValueError: Cannot initialize clifford object from invalid string '{1'.
>>> print clifford("{+")
Traceback (most recent call last):
...
ValueError: Cannot initialize clifford object from invalid string '{+'.
>>> print clifford("-")
Traceback (most recent call last):
...
ValueError: Cannot initialize clifford object from invalid string '-'.
>>> print clifford("{1}+")
Traceback (most recent call last):
...
ValueError: Cannot initialize clifford object from invalid string '{1}+'.

```

For clifford.\_\_richcmp\_\_: Compare objects of type clifford.

```

>>> clifford("{1}") == clifford("1{1}")
True
>>> clifford("{1}") != clifford("1.0{1}")
False
>>> clifford("{1}") != clifford("1.0")
True
>>> clifford("{1,2}") == None
False
>>> clifford("{1,2}") != None
True
>>> None == clifford("{1,2}")
False
>>> None != clifford("{1,2}")
True

```

Definition at line 1243 of file PyClicl.pyx.

### 5.6.1.3 e()

```

def PyClicl.e (
    obj )

```

Abbreviation for `clifford(index_set(obj))`.

```

>>> print e(1)
{1}
>>> print e(-1)
{-1}
>>> print e(0)
1

```

Definition at line 1887 of file PyClicl.pyx.

Referenced by `glucat::matrix_multi< Scalar_T, LO, HI >.basis_element()`, `clifford_to_str()`, `glucat::framed_multi< Scalar_T, LO, HI >.framed_multi()`, `glucat::matrix_multi< Scalar_T, LO, HI >.matrix_multi()`, and `glucat.try_catch()`.

### 5.6.1.4 index\_set\_hidden\_doctests()

```

def PyClicl.index_set_hidden_doctests ( )

```

Tests for functions that Doctest cannot see.

For `index_set.__cinit__`: Construct `index_set`.

```

>>> print index_set(1)
{1}
>>> print index_set({1,2})
{1,2}
>>> print index_set(index_set({1,2}))
{1,2}
>>> print index_set({1,2})
{1,2}
>>> print index_set({1,2,1})
{1,2}
>>> print index_set({1,2,1})
{1,2}

```

```

>>> print index_set("")
{}
>>> print index_set("{}")
Traceback (most recent call last):
...
ValueError: Cannot initialize index_set object from invalid string '{}'.
>>> print index_set("{1}")
Traceback (most recent call last):
...
ValueError: Cannot initialize index_set object from invalid string '{1}'.
>>> print index_set("{1,2,100}")
Traceback (most recent call last):
...
ValueError: Cannot initialize index_set object from invalid string '{1,2,100}'.
>>> print index_set({1,2,100})
Traceback (most recent call last):
...
IndexError: Cannot initialize index_set object from invalid set([1, 2, 100]).
>>> print index_set([1,2])
Traceback (most recent call last):
...
TypeError: Cannot initialize index_set object from <type 'list'>.

For index_set.__richcmp__: Compare two objects of class index_set.

>>> index_set(1) == index_set({1})
True
>>> index_set({1}) != index_set({1})
False
>>> index_set({1}) != index_set({2})
True
>>> index_set({1}) == index_set({2})
False
>>> index_set({1}) < index_set({2})
True
>>> index_set({1}) <= index_set({2})
True
>>> index_set({1}) > index_set({2})
False
>>> index_set({1}) >= index_set({2})
False
>>> None == index_set({1,2})
False
>>> None != index_set({1,2})
True
>>> None < index_set({1,2})
False
>>> None <= index_set({1,2})
False
>>> None > index_set({1,2})
False
>>> None >= index_set({1,2})
False
>>> index_set({1,2}) == None
False
>>> index_set({1,2}) != None
True
>>> index_set({1,2}) < None
False
>>> index_set({1,2}) <= None
False
>>> index_set({1,2}) > None
False
>>> index_set({1,2}) >= None
False

```

Definition at line 404 of file PyClical.pyx.

#### 5.6.1.5 istpq()

```
def PyClical.istpq (
    p,
    q )
```

Abbreviation for `index_set({-q,...p})`.

```
>>> print istpq(2,3)
{-3,-2,-1,1,2}
```

Definition at line 1900 of file PyClical.pyx.

### 5.6.2 Variable Documentation

#### 5.6.2.1 \_\_version\_\_

```
string PyClical.__version__ = "0.8.2" [private]
```

Definition at line 32 of file PyClical.pyx.

#### 5.6.2.2 cl

```
PyClical.cl = clifford
```

Definition at line 1859 of file PyClical.pyx.

Referenced by `cga3.agc3()`, `cga3.cga3()`, and `cga3.cga3std()`.

#### 5.6.2.3 fill

```
PyClical.fill
```

Definition at line 1815 of file PyClical.pyx.

Referenced by `glucat::matrix_multi< Scalar_T, LO, HI >.random()`, and `glucat::framed_multi< Scalar_T, LO, HI >.random()`.

## 5.6.2.4 i

PyClical.i

Definition at line 1542 of file PyClical.pyx.

Referenced by `glucat.acos()`, `glucat.acosh()`, `glucat.asin()`, `glucat.asinh()`, `glucat.atan()`, `glucat.atanh()`, `glucat.check_complex()`, `glucat.cos()`, `glucat.log()`, `glucat.matrix_log()`, `glucat.matrix_sqrt()`, `glucat.operator<<()`, `glucat.operator>>()`, `glucat.sin()`, `glucat.sqrt()`, and `glucat.tan()`.

## 5.6.2.5 ist

PyClical.ist = `index_set`

Definition at line 1879 of file PyClical.pyx.

Referenced by `cga3.agc3()`, `glucat::matrix_multi< Scalar_T, LO, HI >.basis_element()`, `glucat::framed_multi< Scalar_T, LO, HI >.centre_pm4_qp4()`, `glucat::framed_multi< Scalar_T, LO, HI >.centre_pp4_qm4()`, `glucat::framed_multi< Scalar_T, LO, HI >.centre_qp1_pm1()`, `cga3.cga3()`, `cga3.cga3std()`, `glucat::framed_multi< Scalar_T, LO, HI >.divide()`, `glucat::framed_multi< Scalar_T, LO, HI >.framed_multi()`, `index_set_to_repr()`, `index_set_to_str()`, `glucat::matrix_multi< Scalar_T, LO, HI >.matrix_multi()`, `glucat.max_pos()`, `glucat.min_neg()`, `glucat.operator<<()`, and `glucat.operator>>()`.

## 5.6.2.6 ixt

PyClical.ixt

Definition at line 1815 of file PyClical.pyx.

## 5.6.2.7 nbar3

`def PyClical.nbar3 =  $e(4) - e(-1)$`

Definition at line 1910 of file PyClical.pyx.

## 5.6.2.8 ninf3

`def PyClical.ninf3 =  $e(4) + e(-1)$`

Definition at line 1909 of file PyClical.pyx.

Referenced by `cga3.cga3()`, and `cga3.cga3std()`.

### 5.6.2.9 obj

`PyClical.obj`

Definition at line 1542 of file `PyClical.pyx`.

### 5.6.2.10 pi

```
float PyClical.pi = tau / 2.0
```

Definition at line 1857 of file `PyClical.pyx`.

Referenced by `glucat::matrix.classify_eigenvalues()`, `glucat.cos()`, `glucat.log()`, `glucat.matrix_log()`, and `glucat.sin()`.

### 5.6.2.11 tau

```
float PyClical.tau = atan(clifford(1.0)) * 8.0
```

Definition at line 1856 of file `PyClical.pyx`.

## 5.7 std Namespace Reference

### Classes

- struct `numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > >`  
*Numeric limits for framed\_multi inherit limits for the corresponding scalar type.*
- struct `numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > >`  
*Numeric limits for matrix\_multi inherit limits for the corresponding scalar type.*



## Chapter 6

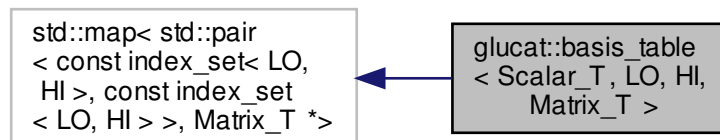
# Class Documentation

### 6.1 `glucat::basis_table< Scalar_T, LO, HI, Matrix_T >` Class Template Reference

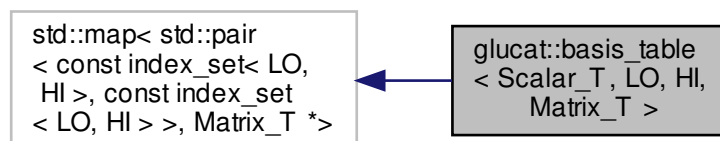
Table of basis elements used as a cache by `basis_element()`

```
#include <matrix_multi_imp.h>
```

Inheritance diagram for `glucat::basis_table< Scalar_T, LO, HI, Matrix_T >`:



Collaboration diagram for `glucat::basis_table< Scalar_T, LO, HI, Matrix_T >`:



## Static Public Member Functions

- static [basis\\_table](#) & [basis](#) ()  
*Single instance of basis table.*

## Private Member Functions

- [basis\\_table](#) ()
- [~basis\\_table](#) ()
- [basis\\_table](#) (const [basis\\_table](#) &)
- [basis\\_table](#) & [operator=](#) (const [basis\\_table](#) &)

## Friends

- class [friend\\_for\\_private\\_destructor](#)

### 6.1.1 Detailed Description

```
template<typename Scalar_T, const index_t LO, const index_t HI, typename Matrix_T>
class glucat::basis_table< Scalar_T, LO, HI, Matrix_T >
```

Table of basis elements used as a cache by [basis\\_element](#)()

Definition at line 1218 of file [matrix\\_multi\\_imp.h](#).

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 [basis\\_table](#)() [1/2]

```
template<typename Scalar_T , const index_t LO, const index_t HI, typename Matrix_T >
glucat::basis_table< Scalar_T, LO, HI, Matrix_T >::basis_table ( ) [inline], [private]
```

Definition at line 1228 of file [matrix\\_multi\\_imp.h](#).

#### 6.1.2.2 [~basis\\_table](#)()

```
template<typename Scalar_T , const index_t LO, const index_t HI, typename Matrix_T >
glucat::basis_table< Scalar_T, LO, HI, Matrix_T >::~basis_table ( ) [inline], [private]
```

Definition at line 1229 of file [matrix\\_multi\\_imp.h](#).

### 6.1.2.3 basis\_table() [2/2]

```
template<typename Scalar_T , const index_t LO, const index_t HI, typename Matrix_T >
glucat::basis_table< Scalar_T, LO, HI, Matrix_T >::basis_table (
    const basis_table< Scalar_T, LO, HI, Matrix_T > & ) [private]
```

## 6.1.3 Member Function Documentation

### 6.1.3.1 basis()

```
template<typename Scalar_T , const index_t LO, const index_t HI, typename Matrix_T >
static basis_table& glucat::basis_table< Scalar_T, LO, HI, Matrix_T >::basis ( ) [inline],
[static]
```

Single instance of basis table.

Definition at line 1224 of file matrix\_multi\_imp.h.

### 6.1.3.2 operator=()

```
template<typename Scalar_T , const index_t LO, const index_t HI, typename Matrix_T >
basis_table& glucat::basis_table< Scalar_T, LO, HI, Matrix_T >::operator= (
    const basis_table< Scalar_T, LO, HI, Matrix_T > & ) [private]
```

## 6.1.4 Friends And Related Function Documentation

### 6.1.4.1 friend\_for\_private\_destructor

```
template<typename Scalar_T , const index_t LO, const index_t HI, typename Matrix_T >
friend class friend_for_private_destructor [friend]
```

Friend declaration to avoid compiler warning: "... only defines a private destructor and has no friends" Ref: Carlos O'Ryan, ACE <http://doc.ece.uci.edu>

Definition at line 1236 of file matrix\_multi\_imp.h.

The documentation for this class was generated from the following file:

- glucat/[matrix\\_multi\\_imp.h](#)

## 6.2 glucat::bool\_to\_type< truth\_value > Class Template Reference

Bool to type.

```
#include <global.h>
```

### Private Types

- enum { [value](#) = truth\_value }

### 6.2.1 Detailed Description

```
template<bool truth_value>  
class glucat::bool_to_type< truth_value >
```

Bool to type.

Definition at line 69 of file global.h.

### 6.2.2 Member Enumeration Documentation

#### 6.2.2.1 anonymous enum

```
template<bool truth_value>  
anonymous enum [private]
```

#### Enumerator

value	
-------	--

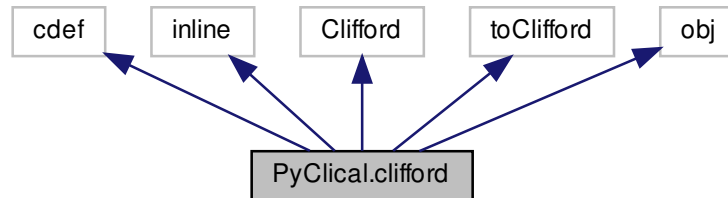
Definition at line 72 of file global.h.

The documentation for this class was generated from the following file:

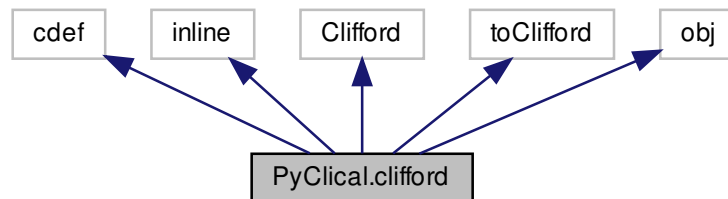
- [glucat/global.h](#)

## 6.3 PyClical.clifford Class Reference

Inheritance diagram for PyClical.clifford:



Collaboration diagram for PyClical.clifford:



### Public Member Functions

- def `__cinit__` (self, other=0, `ixt`=None)
- def `__dealloc__` (self)
- def `__contains__` (self, x)
- def `__iter__` (self)
- def `reframe` (self, `ixt`)
- def `__richcmp__` (lhs, rhs, int, op)
- def `__getitem__` (self, `ixt`)
- def `__neg__` (self)
- def `__pos__` (self)
- def `__add__` (lhs, rhs)
- def `__iadd__` (self, rhs)
- def `__sub__` (lhs, rhs)
- def `__isub__` (self, rhs)
- def `__mul__` (lhs, rhs)
- def `__imul__` (self, rhs)
- def `__mod__` (lhs, rhs)
- def `__imod__` (self, rhs)

- def `__and__` (lhs, rhs)
- def `__iand__` (self, rhs)
- def `__xor__` (lhs, rhs)
- def `__ixor__` (self, rhs)
- def `__div__` (lhs, rhs)
- def `__idiv__` (self, rhs)
- def `inv` (self)
- def `__or__` (lhs, rhs)
- def `__ior__` (self, rhs)
- def `__pow__` (self, m, dummy)
- def `pow` (self, m)
- def `outer_pow` (self, m)
- def `__call__` (self, grade)
- def `scalar` (self)
- def `pure` (self)
- def `even` (self)
- def `odd` (self)
- def `vector_part` (self, frm=None)
- def `involute` (self)
- def `reverse` (self)
- def `conj` (self)
- def `quad` (self)
- def `norm` (self)
- def `abs` (self)
- def `max_abs` (self)
- def `truncated` (self, limit)
- def `isnan` (self)
- def `frame` (self)
- def `__repr__` (self)
- def `__str__` (self)

## Public Attributes

- `instance`

### 6.3.1 Detailed Description

Python class `clifford` wraps C++ class `Clifford`.

Definition at line 532 of file `PyClical.pyx`.

### 6.3.2 Member Function Documentation

**6.3.2.1 `__add__()`**

```
def PyClical.clifford.__add__ (
    lhs,
    rhs )

Geometric sum.

>>> print clifford(1) + clifford("{2}")
1+{2}
>>> print clifford("{1}") + clifford("{2}")
{1}+{2}
```

Definition at line 739 of file PyClical.pyx.

**6.3.2.2 `__and__()`**

```
def PyClical.clifford.__and__ (
    lhs,
    rhs )

Inner product.

>>> print clifford("{1}") & clifford("{2}")
0
>>> print clifford(2) & clifford("{2}")
0
>>> print clifford("{1}") & clifford("{1}")
1
>>> print clifford("{1}") & clifford("{1,2}")
{2}
```

Definition at line 835 of file PyClical.pyx.

**6.3.2.3 `__call__()`**

```
def PyClical.clifford.__call__ (
    self,
    grade )

Pure grade-vector part.

>>> print clifford("{1}") (1)
{1}
>>> print clifford("{1}") (0)
0
>>> print clifford("1+{1}+{1,2}") (0)
1
>>> print clifford("1+{1}+{1,2}") (1)
{1}
>>> print clifford("1+{1}+{1,2}") (2)
{1,2}
>>> print clifford("1+{1}+{1,2}") (3)
0
```

Definition at line 1019 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

#### 6.3.2.4 `__cinit__()`

```
def PyClical.clifford.__cinit__ (
    self,
    other = 0,
    ixt = None )
```

Construct an object of type clifford.

```
>>> print clifford(2)
2
>>> print clifford(2L)
2
>>> print clifford(2.0)
2
>>> print clifford(1.0e-1)
0.1
>>> print clifford("2")
2
>>> print clifford("2{1,2,3}")
2{1,2,3}
>>> print clifford(clifford("2{1,2,3}"))
2{1,2,3}
>>> print clifford("-{1}")
-{1}
>>> print clifford(2,index_set ({1,2}))
2{1,2}
>>> print clifford([2,3],index_set ({1,2}))
2{1}+3{2}
```

Definition at line 563 of file PyClical.pyx.

#### 6.3.2.5 `__contains__()`

```
def PyClical.clifford.__contains__ (
    self,
    x )
```

Not applicable.

```
>>> x=clifford(index_set ({-3,4,7})); -3 in x
Traceback (most recent call last):
...
TypeError: Not applicable.
```

Definition at line 626 of file PyClical.pyx.

#### 6.3.2.6 `__dealloc__()`

```
def PyClical.clifford.__dealloc__ (
    self )
```

Clean up by deallocating the instance of C++ class Clifford.

Definition at line 620 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.



**6.3.2.7 `__div__()`**

```
def PyClical.clifford.__div__ (
    lhs,
    rhs )
```

Geometric quotient.

```
>>> print clifford("{1}") / clifford("{2}")
{1,2}
>>> print clifford(2) / clifford("{2}")
2{2}
>>> print clifford("{1}") / clifford("{1}")
1
>>> print clifford("{1}") / clifford("{1,2}")
-{2}
```

Definition at line 895 of file PyClical.pyx.

**6.3.2.8 `__getitem__()`**

```
def PyClical.clifford.__getitem__ (
    self,
    ixt )
```

Subscripting: map from index set to scalar coordinate.

```
>>> clifford("{1}") [index_set(1)]
1.0
>>> clifford("{1}") [index_set({1})]
1.0
>>> clifford("{1}") [index_set({1,2})]
0.0
>>> clifford("2{1,2}") [index_set({1,2})]
2.0
```

Definition at line 706 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

**6.3.2.9 `__iadd__()`**

```
def PyClical.clifford.__iadd__ (
    self,
    rhs )
```

Geometric sum.

```
>>> x = clifford(1); x += clifford("{2}"); print x
1+{2}
```

Definition at line 750 of file PyClical.pyx.

**6.3.2.10** `__iand__()`

```
def PyClical.clifford.__iand__ (
    self,
    rhs )
```

Inner product.

```
>>> x = clifford("{1}"); x &= clifford("{2}"); print x
0
>>> x = clifford(2); x &= clifford("{2}"); print x
0
>>> x = clifford("{1}"); x &= clifford("{1}"); print x
1
>>> x = clifford("{1}"); x &= clifford("{1,2}"); print x
{2}
```

Definition at line 850 of file PyClical.pyx.

**6.3.2.11** `__idiv__()`

```
def PyClical.clifford.__idiv__ (
    self,
    rhs )
```

Geometric quotient.

```
>>> x = clifford("{1}"); x /= clifford("{2}"); print x
{1,2}
>>> x = clifford(2); x /= clifford("{2}"); print x
2{2}
>>> x = clifford("{1}"); x /= clifford("{1}"); print x
1
>>> x = clifford("{1}"); x /= clifford("{1,2}"); print x
-{2}
```

Definition at line 910 of file PyClical.pyx.

**6.3.2.12** `__imod__()`

```
def PyClical.clifford.__imod__ (
    self,
    rhs )
```

Contraction.

```
>>> x = clifford("{1}"); x %= clifford("{2}"); print x
0
>>> x = clifford(2); x %= clifford("{2}"); print x
2{2}
>>> x = clifford("{1}"); x %= clifford("{1}"); print x
1
>>> x = clifford("{1}"); x %= clifford("{1,2}"); print x
{2}
```

Definition at line 820 of file PyClical.pyx.

**6.3.2.13** `__imul__()`

```
def PyClical.clifford.__imul__ (
    self,
    rhs )
```

Geometric product.

```
>>> x = clifford(2); x *= clifford("{2}"); print x
2{2}
>>> x = clifford("{1}"); x *= clifford("{2}"); print x
{1,2}
>>> x = clifford("{1}"); x *= clifford("{1,2}"); print x
{2}
```

Definition at line 792 of file PyClical.pyx.

**6.3.2.14** `__ior__()`

```
def PyClical.clifford.__ior__ (
    self,
    rhs )
```

Transform left hand side, using right hand side as a transformation.

```
>>> x=clifford("{1,2}") * pi/2; y=clifford("{1}"); y|=x; print y
-{1}
>>> x=clifford("{1,2}") * pi/2; y=clifford("{1}"); y|=exp(x); print y
-{1}
```

Definition at line 949 of file PyClical.pyx.

**6.3.2.15** `__isub__()`

```
def PyClical.clifford.__isub__ (
    self,
    rhs )
```

Geometric difference.

```
>>> x = clifford(1); x -= clifford("{2}"); print x
1-{2}
```

Definition at line 770 of file PyClical.pyx.

**6.3.2.16** `__iter__()`

```
def PyClical.clifford.__iter__ (
    self )
```

Not applicable.

```
>>> for a in clifford(index_set({-3,4,7})): print a,
Traceback (most recent call last):
...
TypeError: Not applicable.
```

Definition at line 637 of file PyClical.pyx.

**6.3.2.17** `__ixor__()`

```
def PyClical.clifford.__ixor__ (
    self,
    rhs )
```

Outer product.

```
>>> x = clifford("{1}"); x ^= clifford("{2}"); print x
{1,2}
>>> x = clifford(2); x ^= clifford("{2}"); print x
2{2}
>>> x = clifford("{1}"); x ^= clifford("{1}"); print x
0
>>> x = clifford("{1}"); x ^= clifford("{1,2}"); print x
0
```

Definition at line 880 of file PyClical.pyx.

**6.3.2.18** `__mod__()`

```
def PyClical.clifford.__mod__ (
    lhs,
    rhs )
```

Contraction.

```
>>> print clifford("{1}") % clifford("{2}")
0
>>> print clifford(2) % clifford("{2}")
2{2}
>>> print clifford("{1}") % clifford("{1}")
1
>>> print clifford("{1}") % clifford("{1,2}")
{2}
```

Definition at line 805 of file PyClical.pyx.

**6.3.2.19 `__mul__()`**

```
def PyClical.clifford.__mul__ (
    lhs,
    rhs )
```

Geometric product.

```
>>> print clifford("{1}") * clifford("{2}")
{1,2}
>>> print clifford(2) * clifford("{2}")
2{2}
>>> print clifford("{1}") * clifford("{1,2}")
{2}
```

Definition at line 779 of file PyClical.pyx.

**6.3.2.20 `__neg__()`**

```
def PyClical.clifford.__neg__ (
    self )
```

Unary `-`.

```
>>> print -clifford("{1}")
-{1}
```

Definition at line 721 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

**6.3.2.21 `__or__()`**

```
def PyClical.clifford.__or__ (
    lhs,
    rhs )
```

Transform left hand side, using right hand side as a transformation.

```
>>> x=clifford("{1,2}") * pi/2; y=clifford("{1}"); print y|x
-{1}
>>> x=clifford("{1,2}") * pi/2; y=clifford("{1}"); print y|exp(x)
-{1}
```

Definition at line 938 of file PyClical.pyx.

**6.3.2.22 \_\_pos\_\_()**

```
def PyClical.clifford.__pos__ (
    self )
```

Unary +.

```
>>> print +clifford("{1}")
{1}
```

Definition at line 730 of file PyClical.pyx.

**6.3.2.23 \_\_pow\_\_()**

```
def PyClical.clifford.__pow__ (
    self,
    m,
    dummy )
```

Power: self to the m.

```
>>> x=clifford("{1}"); print x ** 2
1
>>> x=clifford("2"); print x ** 2
4
>>> x=clifford("2+{1}"); print x ** 0
1
>>> x=clifford("2+{1}"); print x ** 1
2+{1}
>>> x=clifford("2+{1}"); print x ** 2
5+4{1}
>>> i=clifford("{1,2}");print exp(pi/2) * (i ** i)
1
```

Definition at line 960 of file PyClical.pyx.

References PyClical.clifford.pow().

**6.3.2.24 \_\_repr\_\_()**

```
def PyClical.clifford.__repr__ (
    self )
```

The "official" string representation of self.

```
>>> clifford("1+3{-1}+2{1,2}+4{-2,7}").__repr__()
'clifford("1+3{-1}+2{1,2}+4{-2,7}")'
```

Definition at line 1225 of file PyClical.pyx.

References clifford\_to\_repr().

**6.3.2.25** `__richcmp__()`

```
def PyClical.clifford.__richcmp__ (
    lhs,
    rhs,
    int,
    op )
```

Compare objects of type clifford.

```
>>> clifford("{1}") == clifford("1{1}")
True
>>> clifford("{1}") != clifford("1.0{1}")
False
>>> clifford("{1}") != clifford("1.0")
True
>>> clifford("{1,2}") == None
False
>>> clifford("{1,2}") != None
True
>>> None == clifford("{1,2}")
False
>>> None != clifford("{1,2}")
True
```

Definition at line 671 of file PyClical.pyx.

**6.3.2.26** `__str__()`

```
def PyClical.clifford.__str__ (
    self )
```

The “informal” string representation of self.

```
>>> clifford("1+3{-1}+2{1,2}+4{-2,7}").__str__()
'1+3{-1}+2{1,2}+4{-2,7}'
```

Definition at line 1234 of file PyClical.pyx.

References `clifford_to_str()`.

**6.3.2.27** `__sub__()`

```
def PyClical.clifford.__sub__ (
    lhs,
    rhs )
```

Geometric difference.

```
>>> print clifford(1) - clifford("{2}")
1-{2}
>>> print clifford("{1}") - clifford("{2}")
{1}-{2}
```

Definition at line 759 of file PyClical.pyx.

**6.3.2.28 `__xor__()`**

```
def PyClical.clifford.__xor__ (
    lhs,
    rhs )
```

Outer product.

```
>>> print clifford("{1}") ^ clifford("{2}")
{1,2}
>>> print clifford(2) ^ clifford("{2}")
2{2}
>>> print clifford("{1}") ^ clifford("{1}")
0
>>> print clifford("{1}") ^ clifford("{1,2}")
0
```

Definition at line 865 of file PyClical.pyx.

**6.3.2.29 `abs()`**

```
def PyClical.clifford.abs (
    self )
```

Absolute value: square root of norm.

```
>>> clifford("1+{-1}+{1,2}+{1,2,3}").abs()
2.0
```

Definition at line 1174 of file PyClical.pyx.

References `glucat.abs()`.

**6.3.2.30 `conj()`**

```
def PyClical.clifford.conj (
    self )
```

Conjugation, reverse o involute == involute o reverse.

```
>>> print (clifford("{1}")).conj()
-1
>>> print (clifford("{2}") * clifford("{1}")).conj()
{1,2}
>>> print (clifford("{1}") * clifford("{2}")).conj()
-1,2
>>> print clifford("1+{1}+{1,2}").conj()
1-{1}-{1,2}
```

Definition at line 1137 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.



**6.3.2.31 even()**

```
def PyClical.clifford.even (
    self )
```

Even part of multivector, sum of even grade terms.

```
>>> print clifford("1+{1}+{1,2}").even()
1+{1,2}
```

Definition at line 1060 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

**6.3.2.32 frame()**

```
def PyClical.clifford.frame (
    self )
```

Subalgebra generated by all generators of terms of given multivector.

```
>>> print clifford("1+3{-1}+2{1,2}+4{-2,7}").frame()
{-2,-1,1,2,7}
>>> s=clifford("1+3{-1}+2{1,2}+4{-2,7}").frame(); type(s)
<type 'PyClical.index_set'>
```

Definition at line 1214 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

**6.3.2.33 inv()**

```
def PyClical.clifford.inv (
    self )
```

Geometric multiplicative inverse.

```
>>> x = clifford("{1}"); print x.inv()
{1}
>>> x = clifford(2); print x.inv()
0.5
>>> x = clifford("{1,2}"); print x.inv()
-{1,2}
```

Definition at line 925 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

**6.3.2.34 involute()**

```
def PyClical.clifford.involute (
    self )
```

Main involution, each {i} is replaced by -{i} in each term,  
eg. clifford("{1}") -> -clifford("{1}").

```
>>> print clifford("{1}").involute()
-{1}
>>> print (clifford("{2}") * clifford("{1}")).involute()
-{1,2}
>>> print (clifford("{1}") * clifford("{2}")).involute()
{1,2}
>>> print clifford("1+{1}+{1,2}").involute()
1-{1}+{1,2}
```

Definition at line 1106 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

**6.3.2.35 isnan()**

```
def PyClical.clifford.isnan (
    self )
```

Check if a multivector contains any IEEE NaN values.

```
>>> clifford().isnan()
False
```

Definition at line 1205 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

**6.3.2.36 max\_abs()**

```
def PyClical.clifford.max_abs (
    self )
```

Maximum of absolute values of components of multivector: multivector infinity norm.

```
>>> clifford("1+{-1}+{1,2}+{1,2,3}").max_abs()
1.0
>>> clifford("3+2{1}+{1,2}").max_abs()
3.0
```

Definition at line 1183 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

**6.3.2.37 norm()**

```
def PyClical.clifford.norm (
    self )
```

Norm == sum of squares of coordinates.

```
>>> clifford("1+{1}+{1,2}").norm()
3.0
>>> clifford("1+{-1}+{1,2}+{1,2,3}").norm()
4.0
```

Definition at line 1163 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

**6.3.2.38 odd()**

```
def PyClical.clifford.odd (
    self )
```

Odd part of multivector, sum of odd grade terms.

```
>>> print clifford("1+{1}+{1,2}").odd()
{1}
```

Definition at line 1069 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

**6.3.2.39 outer\_pow()**

```
def PyClical.clifford.outer_pow (
    self,
    m )
```

Outer product power.

```
>>> x=clifford("2+{1}"); print x.outer_pow(0)
1
>>> x=clifford("2+{1}"); print x.outer_pow(1)
2+{1}
>>> x=clifford("2+{1}"); print x.outer_pow(2)
4+4{1}
>>> print clifford("1+{1}+{1,2}").outer_pow(3)
1+3{1}+3{1,2}
```

Definition at line 1003 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

### 6.3.2.40 pow()

```
def PyClical.clifford.pow (
    self,
    m )

Power: self to the m.

>>> x=clifford("{1}"); print x.pow(2)
1
>>> x=clifford("2"); print x.pow(2)
4
>>> x=clifford("2+{1}"); print x.pow(0)
1
>>> x=clifford("2+{1}"); print x.pow(1)
2+{1}
>>> x=clifford("2+{1}"); print x.pow(2)
5+4{1}
>>> print clifford("1+{1}+{1,2}").pow(3)
1+3{1}+3{1,2}
>>> i=clifford("{1,2}");print exp(pi/2) * i.pow(i)
1
```

Definition at line 979 of file PyClical.pyx.

References `glucat.exp()`, `PyClical.index_set.instance`, `PyClical.clifford.instance`, and `glucat.log()`.

Referenced by `PyClical.clifford.__pow__()`.

### 6.3.2.41 pure()

```
def PyClical.clifford.pure (
    self )

Pure part.

>>> print clifford("1+{1}+{1,2}").pure()
{1}+{1,2}
>>> print clifford("{1,2}").pure()
{1,2}
```

Definition at line 1049 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

### 6.3.2.42 quad()

```
def PyClical.clifford.quad (
    self )

Quadratic form == (rev(x)*x)(0).

>>> print clifford("1+{1}+{1,2}").quad()
3.0
>>> print clifford("1+{-1}+{1,2}+{1,2,3}").quad()
2.0
```

Definition at line 1152 of file PyClical.pyx.

References `PyClical.index_set.instance`, and `PyClical.clifford.instance`.

**6.3.2.43 reframe()**

```
def PyClical.clifford.reframe (
    self,
    ixt )
```

Put self into a larger frame, containing the union of self.frame() and index set ixt. This can be used to make multiplication faster, by multiplying within a common frame.

```
>>> clifford("2+3{1}").reframe(index_set({1,2,3}))
clifford("2+3{1}")
>>> s=index_set({1,2,3});t=index_set({-3,-2,-1});x=random_clifford(s); x.reframe(t).frame() == (s|t);
True
```

Definition at line 648 of file PyClical.pyx.

**6.3.2.44 reverse()**

```
def PyClical.clifford.reverse (
    self )
```

Reversion, eg. clifford("{1}")\*clifford("{2}") -> clifford("{2}")\*clifford("{1}").

```
>>> print clifford("{1}").reverse()
{1}
>>> print (clifford("{2}") * clifford("{1}")).reverse()
{1,2}
>>> print (clifford("{1}") * clifford("{2}")).reverse()
-{1,2}
>>> print clifford("1+{1}+{1,2}").reverse()
1+{1}-{1,2}
```

Definition at line 1122 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

**6.3.2.45 scalar()**

```
def PyClical.clifford.scalar (
    self )
```

Scalar part.

```
>>> clifford("1+{1}+{1,2}").scalar()
1.0
>>> clifford("{1,2}").scalar()
0.0
```

Definition at line 1038 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

### 6.3.2.46 truncated()

```
def PyClical.clifford.truncated (
    self,
    limit )
```

Remove all terms of self with relative size smaller than limit.

```
>>> clifford("1e8+{1}+1e-8{1,2}").truncated(1.0e-6)
clifford("100000000")
>>> clifford("1e4+{1}+1e-4{1,2}").truncated(1.0e-6)
clifford("10000+{1}")
```

Definition at line 1194 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

### 6.3.2.47 vector\_part()

```
def PyClical.clifford.vector_part (
    self,
    frm = None )
```

Vector part of multivector, as a Python list, with respect to frm.

```
>>> print clifford("1+2{1}+3{2}+4{1,2}").vector_part()
[2.0, 3.0]
>>> print clifford("1+2{1}+3{2}+4{1,2}").vector_part(index_set({-1,1,2}))
[0.0, 2.0, 3.0]
```

Definition at line 1078 of file PyClical.pyx.

References PyClical.index\_set.instance, and PyClical.clifford.instance.

## 6.3.3 Member Data Documentation

### 6.3.3.1 instance

PyClical.clifford.instance

Definition at line 592 of file PyClical.pyx.

Referenced by PyClical.clifford.\_\_call\_\_(), PyClical.clifford.\_\_dealloc\_\_(), PyClical.clifford.\_\_getitem\_\_(), PyClical.clifford.\_\_neg\_\_(), PyClical.clifford.conj(), PyClical.clifford.even(), PyClical.clifford.frame(), PyClical.clifford.inv(), PyClical.clifford.involute(), PyClical.clifford.isnan(), PyClical.clifford.max\_abs(), PyClical.clifford.norm(), PyClical.clifford.odd(), PyClical.clifford.outer\_pow(), PyClical.clifford.pow(), PyClical.clifford.pure(), PyClical.clifford.quad(), PyClical.clifford.reverse(), PyClical.clifford.scalar(), PyClical.clifford.truncated(), and PyClical.clifford.vector\_part().

The documentation for this class was generated from the following file:

- [pyclical/PyClical.pyx](#)

## 6.4 glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T > Class Template Reference

clifford\_algebra<> declares the operations of a Clifford algebra

```
#include <clifford_algebra.h>
```

### Public Types

- typedef Scalar\_T [scalar\\_t](#)
- typedef Index\_Set\_T [index\\_set\\_t](#)
- typedef Multivector\_T [multivector\\_t](#)
- typedef std::pair< const [index\\_set\\_t](#), Scalar\_T > [pair\\_t](#)
- typedef std::vector< Scalar\_T > [vector\\_t](#)

### Public Member Functions

- virtual [~clifford\\_algebra](#) ()
- virtual bool [operator==](#) (const [multivector\\_t](#) &val) const =0  
*Test for equality of multivectors.*
- virtual bool [operator==](#) (const Scalar\_T &scr) const =0  
*Test for equality of multivector and scalar.*
- virtual [multivector\\_t](#) & [operator+=](#) (const [multivector\\_t](#) &rhs)=0  
*Geometric sum.*
- virtual [multivector\\_t](#) & [operator+=](#) (const Scalar\_T &scr)=0  
*Geometric sum of multivector and scalar.*
- virtual [multivector\\_t](#) & [operator-=](#) (const [multivector\\_t](#) &rhs)=0  
*Geometric difference.*
- virtual const [multivector\\_t](#) [operator-](#) () const =0  
*Unary -.*
- virtual [multivector\\_t](#) & [operator\\*=](#) (const Scalar\_T &scr)=0  
*Product of multivector and scalar.*
- virtual [multivector\\_t](#) & [operator\\*=](#) (const [multivector\\_t](#) &rhs)=0  
*Geometric product.*
- virtual [multivector\\_t](#) & [operator%=>](#) (const [multivector\\_t](#) &rhs)=0  
*Contraction.*
- virtual [multivector\\_t](#) & [operator &=>](#) (const [multivector\\_t](#) &rhs)=0  
*Inner product.*
- virtual [multivector\\_t](#) & [operator^=>](#) (const [multivector\\_t](#) &rhs)=0  
*Outer product.*
- virtual [multivector\\_t](#) & [operator/=](#) (const Scalar\_T &scr)=0  
*Quotient of multivector and scalar.*
- virtual [multivector\\_t](#) & [operator/=](#) (const [multivector\\_t](#) &rhs)=0  
*Geometric quotient.*
- virtual [multivector\\_t](#) & [operator|=>](#) (const [multivector\\_t](#) &rhs)=0  
*Transformation via twisted adjoint action.*
- virtual const [multivector\\_t](#) [inv](#) () const =0  
*Geometric multiplicative inverse.*
- virtual const [multivector\\_t](#) [pow](#) (int m) const =0

- *\*this to the m*
- virtual const [multivector\\_t](#) [outer\\_pow](#) (int m) const =0  
*Outer product power.*
- virtual const [index\\_set\\_t](#) [frame](#) () const =0  
*Subalgebra generated by all generators of terms of given multivector.*
- virtual [index\\_t](#) [grade](#) () const =0  
*Maximum of the grades of each term.*
- virtual Scalar\_T [operator\[\]](#) (const [index\\_set\\_t](#) ist) const =0  
*Subscripting: map from index set to scalar coordinate.*
- virtual const [multivector\\_t](#) [operator\(\)](#) ([index\\_t](#) grade) const =0  
*Pure grade-vector part.*
- virtual Scalar\_T [scalar](#) () const =0  
*Scalar part.*
- virtual const [multivector\\_t](#) [pure](#) () const =0  
*Pure part.*
- virtual const [multivector\\_t](#) [even](#) () const =0  
*Even part of multivector, sum of even grade terms.*
- virtual const [multivector\\_t](#) [odd](#) () const =0  
*Odd part of multivector, sum of odd grade terms.*
- virtual const [vector\\_t](#) [vector\\_part](#) () const =0  
*Vector part of multivector, as a vector\_t with respect to [frame\(\)](#)*
- virtual const [vector\\_t](#) [vector\\_part](#) (const [index\\_set\\_t](#) frm, const bool prechecked) const =0  
*Vector part of multivector, as a vector\_t with respect to frm.*
- virtual const [multivector\\_t](#) [involute](#) () const =0  
*Main involution, each {i} is replaced by -{i} in each term, eg. {1} -> -{1}.*
- virtual const [multivector\\_t](#) [reverse](#) () const =0  
*Reversion, eg. {1}\*{2} -> {2}\*{1}.*
- virtual const [multivector\\_t](#) [conj](#) () const =0  
*Conjugation, reverse o involute == involute o reverse.*
- virtual Scalar\_T [quad](#) () const =0  
*Scalar\_T quadratic form == (rev(x)\*x)(0)*
- virtual Scalar\_T [norm](#) () const =0  
*Scalar\_T norm == sum of norm of coordinates.*
- virtual Scalar\_T [max\\_abs](#) () const =0  
*Maximum of absolute values of components of multivector: multivector infinity norm.*
- virtual const [multivector\\_t](#) [truncated](#) (const Scalar\_T &limit=Scalar\_T(DEFAULT\_TRUNCATION)) const =0  
*Remove all terms with relative size smaller than limit.*
- virtual bool [isnan](#) () const =0  
*Check if a multivector contains any IEEE NaN values.*
- virtual void [write](#) (const std::string &msg="") const =0  
*Write formatted multivector to output.*
- virtual void [write](#) (std::ofstream &ofile, const std::string &msg="") const =0  
*Write formatted multivector to file.*

## Static Public Member Functions

- static const std::string [classname](#) ()



### 6.4.1 Detailed Description

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
class glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >
```

clifford\_algebra<> declares the operations of a Clifford algebra

Definition at line 42 of file clifford\_algebra.h.

### 6.4.2 Member Typedef Documentation

#### 6.4.2.1 index\_set\_t

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
typedef Index_Set_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::index_set_t
```

Definition at line 46 of file clifford\_algebra.h.

#### 6.4.2.2 multivector\_t

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
typedef Multivector_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::multivector_t
```

Definition at line 47 of file clifford\_algebra.h.

#### 6.4.2.3 pair\_t

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
typedef std::pair< const index_set_t, Scalar_T > glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::pair_t
```

Definition at line 48 of file clifford\_algebra.h.

#### 6.4.2.4 scalar\_t

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
typedef Scalar_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::scalar_t
```

Definition at line 45 of file clifford\_algebra.h.

#### 6.4.2.5 vector\_t

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
typedef std::vector<Scalar_T> glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T
>::vector_t
```

Definition at line 49 of file clifford\_algebra.h.

### 6.4.3 Constructor & Destructor Documentation

#### 6.4.3.1 ~clifford\_algebra()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::~~clifford_algebra (
) [inline], [virtual]
```

Definition at line 53 of file clifford\_algebra.h.

### 6.4.4 Member Function Documentation

#### 6.4.4.1 classname()

```
template<typename Scalar_T , typename Index_Set_T , typename Multivector_T >
const std::string glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::classname
( ) [static]
```

Definition at line 66 of file clifford\_algebra\_imp.h.

#### 6.4.4.2 conj()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::conj ( ) const [pure virtual]
```

Conjugation, reverse o involute == involute o reverse.

#### 6.4.4.3 even()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::even ( ) const [pure virtual]
```

Even part of multivector, sum of even grade terms.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi().

#### 6.4.4.4 frame()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const index_set_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::frame ( ) const [pure virtual]
```

Subalgebra generated by all generators of terms of given multivector.

#### 6.4.4.5 grade()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual index_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::grade ( )
const [pure virtual]
```

Maximum of the grades of each term.

#### 6.4.4.6 inv()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::inv ( ) const [pure virtual]
```

Geometric multiplicative inverse.

#### 6.4.4.7 involute()

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::involute ( ) const [pure virtual]
```

Main involution, each {i} is replaced by -{i} in each term, eg. {1} -> -{1}.

**6.4.4.8 isnan()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual bool glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::isnan ( ) const
[pure virtual]
```

Check if a multivector contains any IEEE NaN values.

**6.4.4.9 max\_abs()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual Scalar_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::max_abs ( )
const [pure virtual]
```

Maximum of absolute values of components of multivector: multivector infinity norm.

**6.4.4.10 norm()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual Scalar_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::norm ( )
const [pure virtual]
```

Scalar\_T norm == sum of norm of coordinates.

**6.4.4.11 odd()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::odd ( ) const [pure virtual]
```

Odd part of multivector, sum of odd grade terms.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi().

**6.4.4.12 operator &=()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::operator&= (
    const multivector_t & rhs ) [pure virtual]
```

Inner product.

**6.4.4.13 operator%=( )**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator%= (
    const multivector_t & rhs ) [pure virtual]
```

Contraction.

**6.4.4.14 operator()( )**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator() (
    index_t grade ) const [pure virtual]
```

Pure grade-vector part.

**6.4.4.15 operator\*=( ) [1/2]**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator*= (
    const Scalar_T & scr ) [pure virtual]
```

Product of multivector and scalar.

**6.4.4.16 operator\*=( ) [2/2]**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator*= (
    const multivector_t & rhs ) [pure virtual]
```

Geometric product.

**6.4.4.17 operator+=( ) [1/2]**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator+= (
    const multivector_t & rhs ) [pure virtual]
```

Geometric sum.

**6.4.4.18 operator+=( )** [2/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator+= (
    const Scalar_T & scr ) [pure virtual]
```

Geometric sum of multivector and scalar.

**6.4.4.19 operator-( )**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator- ( ) const [pure virtual]
```

Unary -.

**6.4.4.20 operator-=( )**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator-= (
    const multivector_t & rhs ) [pure virtual]
```

Geometric difference.

**6.4.4.21 operator/=( )** [1/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator/= (
    const Scalar_T & scr ) [pure virtual]
```

Quotient of multivector and scalar.

**6.4.4.22 operator/=( )** [2/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator/= (
    const multivector_t & rhs ) [pure virtual]
```

Geometric quotient.

**6.4.4.23 operator==( )** [1/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual bool glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::operator== (
    const multivector_t & val ) const [pure virtual]
```

Test for equality of multivectors.

**6.4.4.24 operator==( )** [2/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual bool glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::operator== (
    const Scalar_T & scr ) const [pure virtual]
```

Test for equality of multivector and scalar.

**6.4.4.25 operator[]()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual Scalar_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::operator[]
(
    const index_set_t ist ) const [pure virtual]
```

Subscripting: map from index set to scalar coordinate.

**6.4.4.26 operator^=()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator^= (
    const multivector_t & rhs ) [pure virtual]
```

Outer product.

**6.4.4.27 operator" |=()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual multivector_t& glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::operator|= (
    const multivector_t & rhs ) [pure virtual]
```

Transformation via twisted adjoint action.

**6.4.4.28 outer\_pow()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::outer_pow (
    int m ) const [pure virtual]
```

Outer product power.

**6.4.4.29 pow()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::pow (
    int m ) const [pure virtual]
```

\*this to the m

**6.4.4.30 pure()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::pure ( ) const [pure virtual]
```

Pure part.

**6.4.4.31 quad()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual Scalar_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::quad ( )
const [pure virtual]
```

Scalar\_T quadratic form == (rev(x)\*x)(0)

**6.4.4.32 reverse()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↔
::reverse ( ) const [pure virtual]
```

Reversion, eg. {1}\*{2} -> {2}\*{1}.



**6.4.4.33 scalar()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual Scalar_T glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::scalar ( )
const [pure virtual]
```

Scalar part.

**6.4.4.34 truncated()**

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const multivector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::truncated (
    const Scalar_T & limit = Scalar_T(DEFAULT_TRUNCATION) ) const [pure virtual]
```

Remove all terms with relative size smaller than limit.

**6.4.4.35 vector\_part()** [1/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const vector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::vector_part ( ) const [pure virtual]
```

Vector part of multivector, as a vector\_t with respect to [frame\(\)](#)

**6.4.4.36 vector\_part()** [2/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual const vector_t glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >↵
::vector_part (
    const index_set_t frm,
    const bool prechecked ) const [pure virtual]
```

Vector part of multivector, as a vector\_t with respect to frm.

**6.4.4.37 write()** [1/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual void glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >::write (
    const std::string & msg = "" ) const [pure virtual]
```

Write formatted multivector to output.

**6.4.4.38** `write()` [2/2]

```
template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
virtual void glucat::clifford\_algebra< Scalar_T, Index_Set_T, Multivector_T >::write (
    std::ofstream & ofile,
    const std::string & msg = "" ) const [pure virtual]
```

Write formatted multivector to file.

The documentation for this class was generated from the following files:

- [glucat/clifford\\_algebra.h](#)
- [glucat/clifford\\_algebra\\_imp.h](#)

**6.5** `glucat::compare_types< LHS_T, RHS_T >` Class Template Reference

Type comparison.

```
#include <global.h>
```

**Public Types**

- enum { [are\\_same](#) = false }

**6.5.1** Detailed Description

```
template<typename LHS_T, typename RHS_T>
class glucat::compare\_types< LHS_T, RHS_T >
```

Type comparison.

Definition at line 54 of file `global.h`.

**6.5.2** Member Enumeration Documentation**6.5.2.1** anonymous enum

```
template<typename LHS_T , typename RHS_T >
anonymous enum
```

**Enumerator**

<code>are_same</code>	
-----------------------	--

Definition at line 57 of file global.h.

The documentation for this class was generated from the following file:

- [glucat/global.h](#)

## 6.6 glucat::compare\_types< T, T > Class Template Reference

```
#include <global.h>
```

### Public Types

- enum { [are\\_same](#) = true }

### 6.6.1 Detailed Description

```
template<typename T>
class glucat::compare_types< T, T >
```

Definition at line 60 of file global.h.

### 6.6.2 Member Enumeration Documentation

#### 6.6.2.1 anonymous enum

```
template<typename T >
anonymous enum
```

#### Enumerator

<a href="#">are_same</a>	
--------------------------	--

Definition at line 63 of file global.h.

The documentation for this class was generated from the following file:

- [glucat/global.h](#)

## 6.7 glucat::control\_t Class Reference

Parameters to control tests.

```
#include <control.h>
```

## Public Member Functions

- int [call](#) ([intfn](#) f) const  
*Call a function that returns int.*
- int [call](#) ([intintfn](#) f, int arg) const  
*Call a function of int that returns int.*

## Static Public Member Functions

- static const [control\\_t](#) & [control](#) (int argc, char \*\*argv)
- static bool [verbose](#) ()  
*Produce more detailed output from tests.*

## Private Member Functions

- bool [valid](#) () const
- bool [catch\\_exceptions](#) () const
- [control\\_t](#) (int argc, char \*\*argv)  
*Constructor from program arguments.*
- [control\\_t](#) ()
- [~control\\_t](#) ()
- [control\\_t](#) (const [control\\_t](#) &)
- [control\\_t](#) & [operator=](#) (const [control\\_t](#) &)

## Private Attributes

- bool [m\\_valid](#)  
*Test parameters are valid.*
- bool [m\\_catch\\_exceptions](#)  
*Catch exceptions.*

## Static Private Attributes

- static bool [m\\_verbose\\_output](#) = false  
*Produce more detailed output from tests.*

## Friends

- class [friend\\_for\\_private\\_destructor](#)

### 6.7.1 Detailed Description

Parameters to control tests.

Definition at line 39 of file control.h.

## 6.7.2 Constructor & Destructor Documentation

### 6.7.2.1 control\_t() [1/3]

```
glucat::control_t::control_t (
    int argc,
    char ** argv ) [private]
```

Constructor from program arguments.

Test control constructor from program arguments.

Definition at line 89 of file control.h.

References GLUCAT\_PACKAGE\_NAME, GLUCAT\_VERSION, m\_catch\_exceptions, m\_valid, m\_verbose\_output, and valid().

### 6.7.2.2 control\_t() [2/3]

```
glucat::control_t::control_t ( ) [inline], [private]
```

Definition at line 59 of file control.h.

### 6.7.2.3 ~control\_t()

```
glucat::control_t::~~control_t ( ) [inline], [private]
```

Definition at line 60 of file control.h.

### 6.7.2.4 control\_t() [3/3]

```
glucat::control_t::control_t (
    const control_t & ) [private]
```

## 6.7.3 Member Function Documentation

#### 6.7.3.1 `call()` [1/2]

```
int glucat::control_t::call (
    intfn f ) const [inline]
```

Call a function that returns int.

Definition at line 137 of file control.h.

References `catch_exceptions()`, `glucat::try_catch()`, and `valid()`.

#### 6.7.3.2 `call()` [2/2]

```
int glucat::control_t::call (
    intintfn f,
    int arg ) const [inline]
```

Call a function of int that returns int.

Definition at line 151 of file control.h.

References `catch_exceptions()`, `glucat::try_catch()`, and `valid()`.

#### 6.7.3.3 `catch_exceptions()`

```
bool glucat::control_t::catch_exceptions ( ) const [inline], [private]
```

Definition at line 49 of file control.h.

References `m_catch_exceptions`.

Referenced by `call()`.

#### 6.7.3.4 `control()`

```
static const control_t& glucat::control_t::control (
    int argc,
    char ** argv ) [inline], [static]
```

Single instance Ref: Scott Meyers, "Effective C++" Second Edition, Addison-Wesley, 1998.

Definition at line 71 of file control.h.

#### 6.7.3.5 operator=()

```
control_t& glucat::control_t::operator= (
    const control_t & ) [private]
```

#### 6.7.3.6 valid()

```
bool glucat::control_t::valid ( ) const [inline], [private]
```

Definition at line 44 of file control.h.

References m\_valid.

Referenced by call(), and control\_t().

#### 6.7.3.7 verbose()

```
static bool glucat::control_t::verbose ( ) [inline], [static]
```

Produce more detailed output from tests.

Definition at line 80 of file control.h.

References m\_verbose\_output.

### 6.7.4 Friends And Related Function Documentation

#### 6.7.4.1 friend\_for\_private\_destructor

```
friend class friend_for_private_destructor [friend]
```

Friend declaration to avoid compiler warning: "... only defines a private destructor and has no friends" Ref: Carlos O'Ryan, ACE <http://doc.ece.uci.edu>

Definition at line 67 of file control.h.

### 6.7.5 Member Data Documentation

#### 6.7.5.1 m\_catch\_exceptions

```
bool glucat::control_t::m_catch_exceptions [private]
```

Catch exceptions.

Definition at line 48 of file control.h.

Referenced by `catch_exceptions()`, and `control_t()`.

#### 6.7.5.2 m\_valid

```
bool glucat::control_t::m_valid [private]
```

Test parameters are valid.

Definition at line 43 of file control.h.

Referenced by `control_t()`, and `valid()`.

#### 6.7.5.3 m\_verbose\_output

```
bool glucat::control_t::m_verbose_output = false [static], [private]
```

Produce more detailed output from tests.

Definition at line 53 of file control.h.

Referenced by `control_t()`, and `verbose()`.

The documentation for this class was generated from the following file:

- [test/control.h](#)

## 6.8 glucat::CTAssertion< bool > Struct Template Reference

Compile time assertion.

```
#include <global.h>
```



### 6.8.1 Detailed Description

```
template<bool>
struct glucat::CTAssertion< bool >
```

Compile time assertion.

Definition at line 46 of file global.h.

The documentation for this struct was generated from the following file:

- [glucat/global.h](#)

## 6.9 glucat::CTAssertion< true > Struct Template Reference

```
#include <global.h>
```

### 6.9.1 Detailed Description

```
template<>
struct glucat::CTAssertion< true >
```

Definition at line 47 of file global.h.

The documentation for this struct was generated from the following file:

- [glucat/global.h](#)

## 6.10 glucat::numeric\_traits< Scalar\_T >::demoted<> Struct Template Reference

Demoted type for long double.

```
#include <long_double.h>
```

### Public Types

- typedef long double [type](#)
- typedef float [type](#)

### 6.10.1 Detailed Description

```
template<typename Scalar_T>
template<>
struct glucat::numeric_traits< Scalar_T >::demoted<>
```

Demoted type for long double.

Demoted type.

Definition at line 47 of file long\_double.h.

## 6.10.2 Member Typedef Documentation

### 6.10.2.1 `type` [1/2]

```
template<typename Scalar_T >
typedef long double glucat::numeric_traits< Scalar_T >::demoted<>::type
```

Definition at line 49 of file long\_double.h.

### 6.10.2.2 `type` [2/2]

```
template<typename Scalar_T >
typedef float glucat::numeric_traits< Scalar_T >::demoted<>::type
```

Definition at line 147 of file scalar.h.

The documentation for this struct was generated from the following files:

- glucat/long\_double.h
- glucat/scalar.h

## 6.11 `glucat::matrix::eig_genus< Matrix_T >` Struct Template Reference

Structure containing classification of eigenvalues.

```
#include <matrix.h>
```

### Public Types

- typedef `Matrix_T::value_type` `Scalar_T`

### Public Attributes

- `eig_case_t m_eig_case`  
What kind of eigenvalues does the matrix contain?
- `Scalar_T m_safe_arg`  
Argument such that  $\exp(\pi \cdot m\_safe\_arg)$  lies between arguments of eigenvalues.

### 6.11.1 Detailed Description

```
template<typename Matrix_T>
struct glucat::matrix::eig_genus< Matrix_T >
```

Structure containing classification of eigenvalues.

Definition at line 131 of file matrix.h.

### 6.11.2 Member Typedef Documentation

#### 6.11.2.1 Scalar\_T

```
template<typename Matrix_T>
typedef Matrix_T::value_type glucat::matrix::eig_genus< Matrix_T >::Scalar_T
```

Definition at line 133 of file matrix.h.

### 6.11.3 Member Data Documentation

#### 6.11.3.1 m\_eig\_case

```
template<typename Matrix_T>
eig_case_t glucat::matrix::eig_genus< Matrix_T >::m_eig_case
```

What kind of eigenvalues does the matrix contain?

Definition at line 135 of file matrix.h.

Referenced by glucat::matrix::classify\_eigenvalues(), glucat::matrix\_log(), and glucat::matrix\_sqrt().

#### 6.11.3.2 m\_safe\_arg

```
template<typename Matrix_T>
Scalar_T glucat::matrix::eig_genus< Matrix_T >::m_safe_arg
```

Argument such that  $\exp(\pi m\_safe\_arg)$  lies between arguments of eigenvalues.

Definition at line 137 of file matrix.h.

Referenced by glucat::matrix::classify\_eigenvalues(), glucat::matrix\_log(), and glucat::matrix\_sqrt().

The documentation for this struct was generated from the following file:

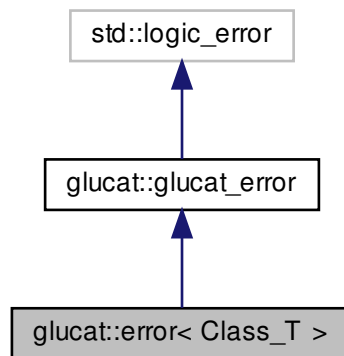
- glucat/matrix.h

## 6.12 glucat::error< Class\_T > Class Template Reference

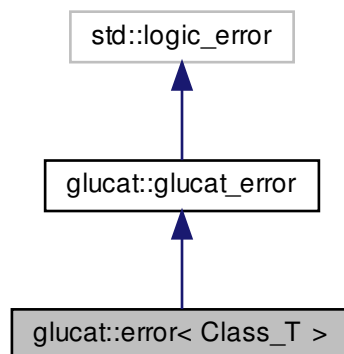
Specific exception class.

```
#include <errors.h>
```

Inheritance diagram for glucat::error< Class\_T >:



Collaboration diagram for glucat::error< Class\_T >:



### Public Member Functions

- [error](#) (const std::string &msg)  
*Specific exception class.*
- [error](#) (const std::string &context, const std::string &msg)
- virtual const std::string [heading](#) () const throw ()
- virtual const std::string [classname](#) () const throw ()
- virtual void [print\\_error\\_msg](#) () const

## Additional Inherited Members

### 6.12.1 Detailed Description

```
template<class Class_T>
class glucat::error< Class_T >
```

Specific exception class.

Definition at line 57 of file errors.h.

### 6.12.2 Constructor & Destructor Documentation

#### 6.12.2.1 error() [1/2]

```
template<class Class_T >
glucat::error< Class_T >::error (
    const std::string & msg )
```

Specific exception class.

Definition at line 39 of file errors\_imp.h.

#### 6.12.2.2 error() [2/2]

```
template<class Class_T >
glucat::error< Class_T >::error (
    const std::string & context,
    const std::string & msg )
```

Definition at line 45 of file errors\_imp.h.

### 6.12.3 Member Function Documentation

#### 6.12.3.1 classname()

```
template<class Class_T >
const std::string glucat::error< Class_T >::classname ( ) const throw ( ) [virtual]
```

Implements [glucat::glucat\\_error](#).

Definition at line 58 of file errors\_imp.h.

### 6.12.3.2 heading()

```
template<class Class_T >
const std::string glucat::error< Class_T >::heading ( ) const throw ( ) [virtual]
```

Implements [glucat::glucat\\_error](#).

Definition at line 52 of file errors\_imp.h.

### 6.12.3.3 print\_error\_msg()

```
template<class Class_T >
void glucat::error< Class_T >::print_error_msg ( ) const [virtual]
```

Implements [glucat::glucat\\_error](#).

Definition at line 64 of file errors\_imp.h.

The documentation for this class was generated from the following files:

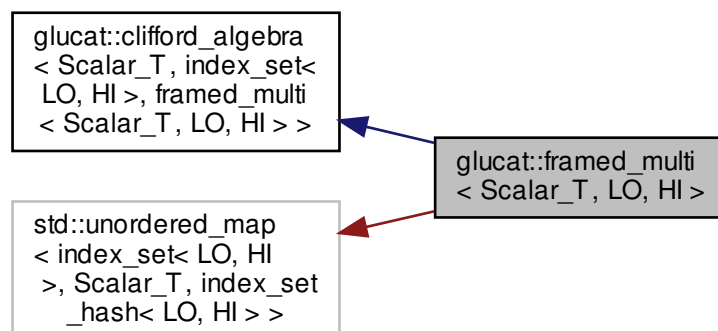
- [glucat/errors.h](#)
- [glucat/errors\\_imp.h](#)

## 6.13 glucat::framed\_multi< Scalar\_T, LO, HI > Class Template Reference

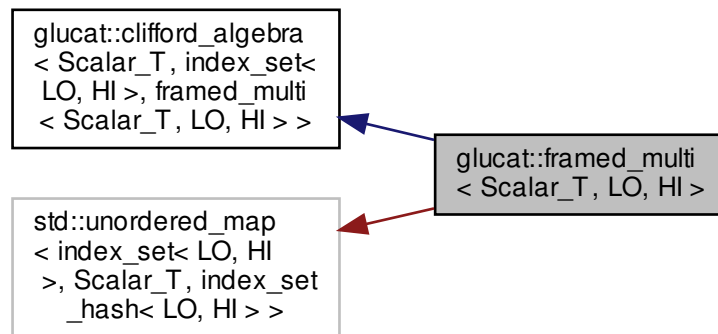
A framed\_multi<Scalar\_T,LO,HI> is a framed approximation to a multivector.

```
#include <framed_multi.h>
```

Inheritance diagram for glucat::framed\_multi< Scalar\_T, LO, HI >:



Collaboration diagram for glucat::framed\_multi< Scalar\_T, LO, HI >:



## Classes

- class [hash\\_size\\_t](#)
- class [var\\_term](#)

*Variable term.*

## Public Types

- typedef [framed\\_multi](#) [multivector\\_t](#)
- typedef [multivector\\_t](#) [framed\\_multi\\_t](#)
- typedef [Scalar\\_T](#) [scalar\\_t](#)
- typedef [index\\_set](#)< [LO](#), [HI](#) > [index\\_set\\_t](#)
- typedef std::pair< const [index\\_set\\_t](#), [Scalar\\_T](#) > [term\\_t](#)
- typedef std::vector< [Scalar\\_T](#) > [vector\\_t](#)
- typedef [error](#)< [multivector\\_t](#) > [error\\_t](#)
- typedef [matrix\\_multi](#)< [Scalar\\_T](#), [LO](#), [HI](#) > [matrix\\_multi\\_t](#)

## Public Member Functions

- [~framed\\_multi](#) ()  
*Destructor.*
- [framed\\_multi](#) ()  
*Default constructor.*
- template<typename Other\_Scalar\_T >  
[framed\\_multi](#) (const [framed\\_multi](#)< Other\_Scalar\_T, [LO](#), [HI](#) > &val)  
*Construct a multivector from a multivector with a different scalar type.*
- template<typename Other\_Scalar\_T >  
[framed\\_multi](#) (const [framed\\_multi](#)< Other\_Scalar\_T, [LO](#), [HI](#) > &val, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a given multivector.*
- [framed\\_multi](#) (const [framed\\_multi\\_t](#) &val, const [index\\_set\\_t](#) frm, const bool prechecked=false)

- Construct a multivector, within a given frame, from a given multivector.*

  - `framed_multi` (const `index_set_t` ist, const `Scalar_T` &crd=`Scalar_T`(1))

*Construct a multivector from an index set and a scalar coordinate.*

  - `framed_multi` (const `index_set_t` ist, const `Scalar_T` &crd, const `index_set_t` frm, const bool prechecked=false)

*Construct a multivector, within a given frame, from an index set and a scalar coordinate.*

  - `framed_multi` (const `Scalar_T` &scr, const `index_set_t` frm=`index_set_t`())

*Construct a multivector from a scalar (within a frame, if given)*

  - `framed_multi` (const int scr, const `index_set_t` frm=`index_set_t`())

*Construct a multivector from an int (within a frame, if given)*

  - `framed_multi` (const `vector_t` &vec, const `index_set_t` frm, const bool prechecked=false)

*Construct a multivector, within a given frame, from a given vector.*

  - `framed_multi` (const std::string &str)

*Construct a multivector from a string: eg: "3+2{1,2}-6.1e-2{2,3}".*

  - `framed_multi` (const std::string &str, const `index_set_t` frm, const bool prechecked=false)

*Construct a multivector, within a given frame, from a string: eg: "3+2{1,2}-6.1e-2{2,3}".*

  - `framed_multi` (const char \*str)

*Construct a multivector from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".*

  - `framed_multi` (const char \*str, const `index_set_t` frm, const bool prechecked=false)

*Construct a multivector, within a given frame, from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".*

  - template<typename Other\_Scalar\_T >  
`framed_multi` (const `matrix_multi`< Other\_Scalar\_T, LO, HI > &val)

*Construct a multivector from a matrix\_multi\_t.*

  - template<typename Other\_Scalar\_T >  
const `matrix_multi`< Other\_Scalar\_T, LO, HI > `fast_matrix_multi` (const `index_set_t` frm) const

*Use generalized FFT to construct a matrix\_multi\_t.*

  - const `framed_multi_t` `fast_framed_multi` () const

*Use inverse generalized FFT to construct a framed\_multi\_t.*

  - `_GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS` unsigned long `nbr_terms` () const

*Number of terms.*

  - `multivector_t` & `operator+=` (const `term_t` &term)

*Add a term, if non-zero.*

## Static Public Member Functions

- static const std::string `classname` ()
- Class name used in messages.*
- static const `framed_multi_t` `random` (const `index_set_t` frm, `Scalar_T` fill=`Scalar_T`(1))
- Random multivector within a frame.*

## Private Types

- typedef class `var_term` `var_term_t`
- typedef `matrix_multi_t::matrix_t` `matrix_t`
- typedef std::map< `index_set_t`, `Scalar_T`, std::less< const `index_set_t` > > `sorted_map_t`
- typedef std::unordered\_map< `index_set_t`, `Scalar_T`, `index_set_hash`< LO, HI > > `map_t`
- typedef std::pair< const `multivector_t`, const `multivector_t` > `framed_pair_t`
- typedef map\_t::size\_type `size_type`
- typedef map\_t::iterator `iterator`
- typedef map\_t::const\_iterator `const_iterator`



## Private Member Functions

- `framed_multi` (const `hash_size_t` &hash\_size)  
*Private constructor using hash\_size.*
- `multivector_t fold` (const `index_set_t` frm) const  
*Subalgebra isomorphism: fold each term within the given frame.*
- `multivector_t unfold` (const `index_set_t` frm) const  
*Subalgebra isomorphism: unfold each term within the given frame.*
- `multivector_t & centre_pm4_qp4` (`index_t` &p, `index_t` &q)  
*Subalgebra isomorphism:  $R_{\{p,q\}}$  to  $R_{\{p-4,q+4\}}$ .*
- `multivector_t & centre_pp4_qm4` (`index_t` &p, `index_t` &q)  
*Subalgebra isomorphism:  $R_{\{p,q\}}$  to  $R_{\{p+4,q-4\}}$ .*
- `multivector_t & centre_qp1_pm1` (`index_t` &p, `index_t` &q)  
*Subalgebra isomorphism:  $R_{\{p,q\}}$  to  $R_{\{q+1,p-1\}}$ .*
- const `framed_pair_t divide` (const `index_set_t` ist) const  
*Divide multivector into part divisible by `index_set` and remainder.*
- const `matrix_t fast` (const `index_t` level, const bool odd) const  
*Generalized FFT from framed\_multi\_t to matrix\_t.*

## Friends

- template<typename Other\_Scalar\_T , const `index_t` Other\_LO, const `index_t` Other\_HI>  
class `matrix_multi`
- template<typename Other\_Scalar\_T , const `index_t` Other\_LO, const `index_t` Other\_HI>  
class `framed_multi`
- const `framed_multi_t operator*` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- const `framed_multi_t operator^` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- const `framed_multi_t operator &` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- const `framed_multi_t operator%` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- `Scalar_T star` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- const `framed_multi_t operator/` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- const `framed_multi_t operator|` (const `framed_multi_t` &lhs, const `framed_multi_t` &rhs)
- std::istream & `operator>>` (std::istream &s, `multivector_t` &val)
- std::ostream & `operator<<` (std::ostream &os, const `multivector_t` &val)
- std::ostream & `operator<<` (std::ostream &os, const `term_t` &term)
- const `framed_multi_t exp` (const `framed_multi_t` &val)

### 6.13.1 Detailed Description

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
class glucat::framed_multi< Scalar_T, LO, HI >
```

A `framed_multi<Scalar_T,LO,HI>` is a framed approximation to a multivector.

Definition at line 65 of file `framed_multi.h`.

### 6.13.2 Member Typedef Documentation

### 6.13.2.1 const\_iterator

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef map_t::const_iterator glucat::framed_multi< Scalar_T, LO, HI >::const_iterator [private]
```

Definition at line 196 of file framed\_multi.h.

### 6.13.2.2 error\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef error<multivector_t> glucat::framed_multi< Scalar_T, LO, HI >::error_t
```

Definition at line 155 of file framed\_multi.h.

### 6.13.2.3 framed\_multi\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef multivector_t glucat::framed_multi< Scalar_T, LO, HI >::framed_multi_t
```

Definition at line 150 of file framed\_multi.h.

### 6.13.2.4 framed\_pair\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef std::pair< const multivector_t, const multivector_t > glucat::framed_multi< Scalar_T,
LO, HI >::framed_pair_t [private]
```

Definition at line 193 of file framed\_multi.h.

### 6.13.2.5 index\_set\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef index_set<LO,HI> glucat::framed_multi< Scalar_T, LO, HI >::index_set_t
```

Definition at line 152 of file framed\_multi.h.

#### 6.13.2.6 iterator

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef map_t::iterator glucat::framed_multi< Scalar_T, LO, HI >::iterator [private]
```

Definition at line 195 of file framed\_multi.h.

#### 6.13.2.7 map\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef std::unordered_map< index_set_t, Scalar_T, index_set_hash<LO,HI> > glucat::framed_multi<
Scalar_T, LO, HI >::map_t [private]
```

Definition at line 175 of file framed\_multi.h.

#### 6.13.2.8 matrix\_multi\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef matrix_multi<Scalar_T,LO,HI> glucat::framed_multi< Scalar_T, LO, HI >::matrix_multi_t
```

Definition at line 156 of file framed\_multi.h.

#### 6.13.2.9 matrix\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef matrix_multi_t::matrix_t glucat::framed_multi< Scalar_T, LO, HI >::matrix_t [private]
```

Definition at line 165 of file framed\_multi.h.

#### 6.13.2.10 multivector\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef framed_multi glucat::framed_multi< Scalar_T, LO, HI >::multivector_t
```

Definition at line 149 of file framed\_multi.h.

**6.13.2.11 scalar\_t**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef Scalar_T glucat::framed_multi< Scalar_T, LO, HI >::scalar_t
```

Definition at line 151 of file framed\_multi.h.

**6.13.2.12 size\_type**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef map_t::size_type glucat::framed_multi< Scalar_T, LO, HI >::size_type [private]
```

Definition at line 194 of file framed\_multi.h.

**6.13.2.13 sorted\_map\_t**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef std::map< index_set_t, Scalar_T, std::less<const index_set_t> > glucat::framed_multi<
Scalar_T, LO, HI >::sorted_map_t [private]
```

Definition at line 172 of file framed\_multi.h.

**6.13.2.14 term\_t**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef std::pair<const index_set_t, Scalar_T> glucat::framed_multi< Scalar_T, LO, HI >↵
::term_t
```

Definition at line 153 of file framed\_multi.h.

**6.13.2.15 var\_term\_t**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef class var_term glucat::framed_multi< Scalar_T, LO, HI >::var_term_t [private]
```

Definition at line 164 of file framed\_multi.h.

## 6.13.2.16 vector\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef std::vector<Scalar_T> glucat::framed_multi< Scalar_T, LO, HI >::vector_t
```

Definition at line 154 of file framed\_multi.h.

## 6.13.3 Constructor &amp; Destructor Documentation

## 6.13.3.1 ~framed\_multi()

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::~~framed_multi ( ) [inline]
```

Destructor.

Definition at line 202 of file framed\_multi.h.

## 6.13.3.2 framed\_multi() [1/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi ( )
```

Default constructor.

Definition at line 67 of file framed\_multi\_imp.h.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi().

## 6.13.3.3 framed\_multi() [2/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const hash_size_t & hash_size ) [private]
```

Private constructor using hash\_size.

Definition at line 74 of file framed\_multi\_imp.h.

**6.13.3.4 framed\_multi()** [3/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const framed_multi< Other_Scalar_T, LO, HI > & val )
```

Construct a multivector from a multivector with a different scalar type.

Definition at line 82 of file framed\_multi\_imp.h.

**6.13.3.5 framed\_multi()** [4/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const framed_multi< Other_Scalar_T, LO, HI > & val,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a given multivector.

Definition at line 97 of file framed\_multi\_imp.h.

**6.13.3.6 framed\_multi()** [5/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const framed_multi_t & val,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a given multivector.

Definition at line 112 of file framed\_multi\_imp.h.

**6.13.3.7 framed\_multi()** [6/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const index_set_t ist,
    const Scalar_T & crd = Scalar_T(1) )
```

Construct a multivector from an index set and a scalar coordinate.

Definition at line 120 of file framed\_multi\_imp.h.

References PyClical::ist.

**6.13.3.8 framed\_multi()** [7/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const index_set_t ist,
    const Scalar_T & crd,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from an index set and a scalar coordinate.

Definition at line 130 of file framed\_multi\_imp.h.

References PyClical::ist.

**6.13.3.9 framed\_multi()** [8/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const Scalar_T & scr,
    const index_set_t frm = index_set_t() )
```

Construct a multivector from a scalar (within a frame, if given)

Definition at line 143 of file framed\_multi\_imp.h.

**6.13.3.10 framed\_multi()** [9/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const int scr,
    const index_set_t frm = index_set_t() )
```

Construct a multivector from an int (within a frame, if given)

Definition at line 153 of file framed\_multi\_imp.h.

**6.13.3.11 framed\_multi()** [10/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const vector_t & vec,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a given vector.

Definition at line 163 of file framed\_multi\_imp.h.

References glucat::index\_set< LO, HI >::count(), glucat::index\_set< LO, HI >::max(), and glucat::index\_set< LO, HI >::min().

**6.13.3.12 framed\_multi()** [11/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const std::string & str )
```

Construct a multivector from a string: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 186 of file framed\_multi\_imp.h.

**6.13.3.13 framed\_multi()** [12/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const std::string & str,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a string: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 202 of file framed\_multi\_imp.h.

**6.13.3.14 framed\_multi()** [13/15]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const char * str ) [inline]
```

Construct a multivector from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 238 of file framed\_multi.h.

References glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi().

**6.13.3.15 framed\_multi()** [14/15]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const char * str,
    const index_set_t frm,
    const bool prechecked = false ) [inline]
```

Construct a multivector, within a given frame, from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 241 of file framed\_multi.h.

References glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi().



## 6.13.3.16 framed\_multi() [15/15]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::framed_multi< Scalar_T, LO, HI >::framed_multi (
    const matrix_multi< Other_Scalar_T, LO, HI > & val )
```

Construct a multivector from a matrix\_multi\_t.

Definition at line 215 of file framed\_multi\_imp.h.

References \_GLUCAT\_HASH\_SIZE\_T, glucat::abs(), glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element(), glucat::index\_set< LO, HI >::count(), PyClical::e(), glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::frame(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::inv\_fast\_dim\_threshold, PyClical::ist, glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix, glucat::matrix::nnz(), and glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, matrix\_multi< Scalar\_T, LO, HI > >::norm().

## 6.13.4 Member Function Documentation

## 6.13.4.1 centre\_pm4\_qp4()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
framed_multi< Scalar_T, LO, HI > & glucat::framed_multi< Scalar_T, LO, HI >::centre_pm4_qp4 (
    index_t & p,
    index_t & q ) [private]
```

Subalgebra isomorphism:  $R_{\{p,q\}}$  to  $R_{\{p-4,q+4\}}$ .

Definition at line 1655 of file framed\_multi\_imp.h.

References PyClical::ist.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::fast\_framed\_multi(), and glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi().

## 6.13.4.2 centre\_pp4\_qm4()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
framed_multi< Scalar_T, LO, HI > & glucat::framed_multi< Scalar_T, LO, HI >::centre_pp4_qm4 (
    index_t & p,
    index_t & q ) [private]
```

Subalgebra isomorphism:  $R_{\{p,q\}}$  to  $R_{\{p+4,q-4\}}$ .

Definition at line 1700 of file framed\_multi\_imp.h.

References PyClical::ist.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::fast\_framed\_multi(), and glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi().

#### 6.13.4.3 centre\_qp1\_pm1()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
framed_multi< Scalar_T, LO, HI > & glucat::framed_multi< Scalar_T, LO, HI >::centre_qp1_pm1 (
    index_t & p,
    index_t & q ) [private]
```

Subalgebra isomorphism:  $R_{\{p,q\}}$  to  $R_{\{q+1,p-1\}}$ .

Definition at line 1745 of file framed\_multi\_imp.h.

References PyClical::ist.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::fast\_framed\_multi(), and glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi().

#### 6.13.4.4 classname()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const std::string glucat::framed_multi< Scalar_T, LO, HI >::classname ( ) [static]
```

Class name used in messages.

Definition at line 53 of file framed\_multi\_imp.h.

#### 6.13.4.5 divide()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const std::pair< const framed_multi< Scalar_T, LO, HI >, const framed_multi< Scalar_T, LO, HI > >
glucat::framed_multi< Scalar_T, LO, HI >::divide (
    const index_set_t ist ) const [private]
```

Divide multivector into part divisible by [index\\_set](#) and remainder.

Divide multivector into quotient with terms divisible by index set, and remainder.

Definition at line 1781 of file framed\_multi\_imp.h.

References PyClical::ist.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast().

## 6.13.4.6 fast()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI >::matrix_t glucat::framed_multi< Scalar_T, LO, HI >↵
::fast (
    const index_t level,
    const bool odd ) const [private]
```

Generalized FFT from framed\_multi\_t to matrix\_t.

Definition at line 1800 of file framed\_multi\_imp.h.

References glucat::framed\_multi< Scalar\_T, LO, HI >::divide(), glucat::framed\_multi< Scalar\_T, LO, HI >::fast(), glucat::matrix::kron(), glucat::odd(), and glucat::scalar().

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast(), and glucat::framed\_multi< Scalar\_T, LO, HI >↵::fast\_matrix\_multi().

## 6.13.4.7 fast\_framed\_multi()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::framed_multi< Scalar_T, LO, HI >::fast_↵
framed_multi ( ) const [inline]
```

Use inverse generalized FFT to construct a framed\_multi\_t.

Definition at line 1898 of file framed\_multi\_imp.h.

## 6.13.4.8 fast\_matrix\_multi()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
const matrix_multi< Other_Scalar_T, LO, HI > glucat::framed_multi< Scalar_T, LO, HI >::fast_↵
_matrix_multi (
    const index_set_t frm ) const
```

Use generalized FFT to construct a matrix\_multi\_t.

Definition at line 1866 of file framed\_multi\_imp.h.

References glucat::framed\_multi< Scalar\_T, LO, HI >::centre\_pm4\_qp4(), glucat::framed\_multi< Scalar\_T, LO, HI >::centre\_pp4\_qm4(), glucat::framed\_multi< Scalar\_T, LO, HI >::centre\_qp1\_pm1(), glucat::index\_set< L↵O, HI >::count\_neg(), glucat::index\_set< LO, HI >::count\_pos(), glucat::clifford\_algebra< Scalar\_T, Index\_Set↵\_T, Multivector\_T >::even(), glucat::framed\_multi< Scalar\_T, LO, HI >::fast(), glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >::odd(), glucat::gen::offset\_to\_super, and glucat::pos\_mod().

#### 6.13.4.9 fold()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
framed_multi< Scalar_T, LO, HI > glucat::framed_multi< Scalar_T, LO, HI >::fold (
    const index_set_t frm ) const [private]
```

Subalgebra isomorphism: fold each term within the given frame.

Definition at line 1614 of file framed\_multi\_imp.h.

References glucat::index\_set< LO, HI >::is\_contiguous().

#### 6.13.4.10 nbr\_terms()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
unsigned long glucat::framed_multi< Scalar_T, LO, HI >::nbr_terms ( ) const
```

Number of terms.

Definition at line 1545 of file framed\_multi\_imp.h.

#### 6.13.4.11 operator+=()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
framed_multi< Scalar_T, LO, HI > & glucat::framed_multi< Scalar_T, LO, HI >::operator+= (
    const term_t & term ) [inline]
```

Add a term, if non-zero.

Insert a term into a multivector, add terms with same index set.

Geometric sum.

Geometric sum of multivector and scalar.

Definition at line 329 of file framed\_multi\_imp.h.

#### 6.13.4.12 random()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const framed_multi< Scalar_T, LO, HI > glucat::framed_multi< Scalar_T, LO, HI >::random (
    const index_set_t frm,
    Scalar_T fill = Scalar_T(1) ) [static]
```

Random multivector within a frame.

Definition at line 1273 of file framed\_multi\_imp.h.

References glucat::index\_set< LO, HI >::count(), PyClical::fill, and glucat::sqrt().

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::random().

6.13.4.13 `unfold()`

```
template<typename Scalar_T , const index_t LO, const index_t HI>
framed_multi< Scalar_T, LO, HI > glucat::framed_multi< Scalar_T, LO, HI >::unfold (
    const index_set_t frm ) const [private]
```

Subalgebra isomorphism: unfold each term within the given frame.

Definition at line 1634 of file framed\_multi\_imp.h.

References `glucat::index_set< LO, HI >::is_contiguous()`.

Referenced by `glucat::matrix_multi< Scalar_T, LO, HI >::fast_framed_multi()`.

## 6.13.5 Friends And Related Function Documentation

6.13.5.1 `exp`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
const framed_multi_t exp (
    const framed_multi_t & val ) [friend]
```

6.13.5.2 `framed_multi`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
friend class framed_multi [friend]
```

Definition at line 160 of file framed\_multi.h.

6.13.5.3 `matrix_multi`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
friend class matrix_multi [friend]
```

Definition at line 158 of file framed\_multi.h.

#### 6.13.5.4 operator &

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const framed_multi_t operator& (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

#### 6.13.5.5 operator%

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const framed_multi_t operator% (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

#### 6.13.5.6 operator\*

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const framed_multi_t operator* (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

#### 6.13.5.7 operator/

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const framed_multi_t operator/ (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

#### 6.13.5.8 operator<< [1/2]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
std::ostream& operator<< (
    std::ostream & os,
    const multivector_t & val ) [friend]
```

## 6.13.5.9 operator&lt;&lt; [2/2]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
std::ostream& operator<< (
    std::ostream & os,
    const term_t & term ) [friend]
```

## 6.13.5.10 operator&gt;&gt;

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
std::istream& operator>> (
    std::istream & s,
    multivector_t & val ) [friend]
```

## 6.13.5.11 operator^

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const framed_multi_t operator^ (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

## 6.13.5.12 operator" |

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const framed_multi_t operator| (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

## 6.13.5.13 star

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
Scalar_T star (
    const framed_multi_t & lhs,
    const framed_multi_t & rhs ) [friend]
```

The documentation for this class was generated from the following files:

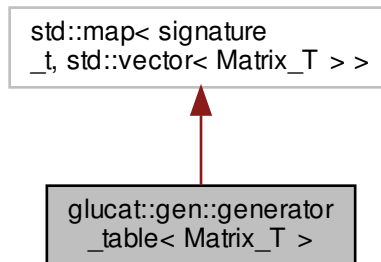
- [glucat/framed\\_multi.h](#)
- [glucat/framed\\_multi\\_imp.h](#)

## 6.14 glucat::gen::generator\_table< Matrix\_T > Class Template Reference

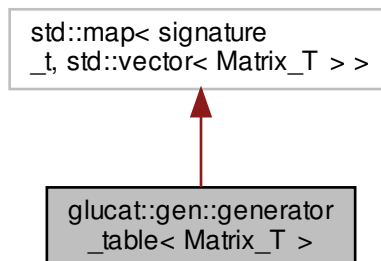
Table of generators for specific signatures.

```
#include <generation.h>
```

Inheritance diagram for glucat::gen::generator\_table< Matrix\_T >:



Collaboration diagram for glucat::gen::generator\_table< Matrix\_T >:



### Public Member Functions

- `const Matrix_T * operator()` (const `index_t` p, const `index_t` q)  
*Pointer to generators for a specific signature.*

### Static Public Member Functions

- static `generator_table< Matrix_T > & generator` ()  
*Single instance of generator table.*



## Private Member Functions

- `const std::vector< Matrix_T > & gen_vector` (const `index_t` p, const `index_t` q)  
*Construct a vector of generators for a specific signature.*
- `void gen_from_pm1_qm1` (const std::vector< Matrix\_T > &old, const `signature_t` sig)  
*Construct generators for p,q given generators for p-1,q-1.*
- `void gen_from_pm4_qp4` (const std::vector< Matrix\_T > &old, const `signature_t` sig)  
*Construct generators for p,q given generators for p-4,q+4.*
- `void gen_from_pp4_qm4` (const std::vector< Matrix\_T > &old, const `signature_t` sig)  
*Construct generators for p,q given generators for p+4,q-4.*
- `void gen_from_qp1_pm1` (const std::vector< Matrix\_T > &old, const `signature_t` sig)  
*Construct generators for p,q given generators for q+1,p-1.*
- `generator_table` ()
- `~generator_table` ()
- `generator_table` (const `generator_table` &)
- `generator_table & operator=` (const `generator_table` &)

## Friends

- class `friend_for_private_destructor`

### 6.14.1 Detailed Description

```
template<class Matrix_T>
class glucat::gen::generator_table< Matrix_T >
```

Table of generators for specific signatures.

Definition at line 47 of file generation.h.

### 6.14.2 Constructor & Destructor Documentation

#### 6.14.2.1 generator\_table() [1/2]

```
template<class Matrix_T>
glucat::gen::generator_table< Matrix_T >::generator_table ( ) [inline], [private]
```

Definition at line 69 of file generation.h.

#### 6.14.2.2 ~generator\_table()

```
template<class Matrix_T>
glucat::gen::generator_table< Matrix_T >::~~generator_table ( ) [inline], [private]
```

Definition at line 70 of file generation.h.

### 6.14.2.3 generator\_table() [2/2]

```
template<class Matrix_T>
glucat::gen::generator_table< Matrix_T >::generator_table (
    const generator_table< Matrix_T > & ) [private]
```

## 6.14.3 Member Function Documentation

### 6.14.3.1 gen\_from\_pm1\_qm1()

```
template<class Matrix_T >
void glucat::gen::generator_table< Matrix_T >::gen_from_pm1_qm1 (
    const std::vector< Matrix_T > & old,
    const signature_t sig ) [private]
```

Construct generators for p,q given generators for p-1,q-1.

Definition at line 127 of file generation\_imp.h.

References glucat::matrix::mono\_kron().

### 6.14.3.2 gen\_from\_pm4\_qp4()

```
template<class Matrix_T >
void glucat::gen::generator_table< Matrix_T >::gen_from_pm4_qp4 (
    const std::vector< Matrix_T > & old,
    const signature_t sig ) [private]
```

Construct generators for p,q given generators for p-4,q+4.

Definition at line 164 of file generation\_imp.h.

References glucat::matrix::mono\_prod().

### 6.14.3.3 gen\_from\_pp4\_qm4()

```
template<class Matrix_T >
void glucat::gen::generator_table< Matrix_T >::gen_from_pp4_qm4 (
    const std::vector< Matrix_T > & old,
    const signature_t sig ) [private]
```

Construct generators for p,q given generators for p+4,q-4.

Definition at line 195 of file generation\_imp.h.

References glucat::matrix::mono\_prod().

## 6.14.3.4 gen\_from\_qp1\_pm1()

```
template<class Matrix_T >
void glucat::gen::generator_table< Matrix_T >::gen_from_qp1_pm1 (
    const std::vector< Matrix_T > & old,
    const signature_t sig ) [private]
```

Construct generators for p,q given generators for q+1,p-1.

Definition at line 225 of file generation\_imp.h.

References glucat::matrix::mono\_prod().

## 6.14.3.5 gen\_vector()

```
template<class Matrix_T >
const std::vector< Matrix_T > & glucat::gen::generator_table< Matrix_T >::gen_vector (
    const index_t p,
    const index_t q ) [private]
```

Construct a vector of generators for a specific signature.

Definition at line 80 of file generation\_imp.h.

References glucat::pos\_mod().

## 6.14.3.6 generator()

```
template<class Matrix_T >
generator_table< Matrix_T > & glucat::gen::generator_table< Matrix_T >::generator ( ) [static]
```

Single instance of generator table.

Definition at line 50 of file generation\_imp.h.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element().

## 6.14.3.7 operator&gt;()()

```
template<class Matrix_T >
const Matrix_T * glucat::gen::generator_table< Matrix_T >::operator() (
    const index_t p,
    const index_t q ) [inline]
```

Pointer to generators for a specific signature.

Definition at line 59 of file generation\_imp.h.

References glucat::gen::offset\_to\_super, and glucat::pos\_mod().

#### 6.14.3.8 operator=()

```
template<class Matrix_T>
generator_table& glucat::gen::generator_table< Matrix_T >::operator= (
    const generator_table< Matrix_T > & ) [private]
```

### 6.14.4 Friends And Related Function Documentation

#### 6.14.4.1 friend\_for\_private\_destructor

```
template<class Matrix_T>
friend class friend_for_private_destructor [friend]
```

Friend declaration to avoid compiler warning: "... only defines a private destructor and has no friends" Ref: Carlos O'Ryan, ACE <http://doc.ece.uci.edu>

Definition at line 77 of file generation.h.

The documentation for this class was generated from the following files:

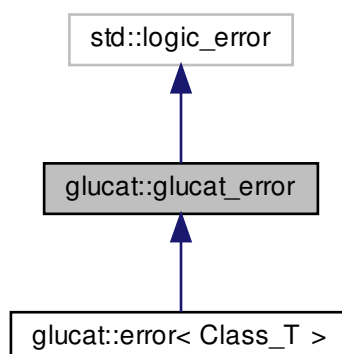
- [glucat/generation.h](#)
- [glucat/generation\\_imp.h](#)

## 6.15 glucat::glucat\_error Class Reference

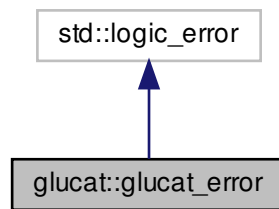
Abstract exception class.

```
#include <errors.h>
```

Inheritance diagram for glucat::glucat\_error:



Collaboration diagram for glucat::glucat\_error:



### Public Member Functions

- [glucat\\_error](#) (const std::string &context, const std::string &msg)
- [~glucat\\_error](#) () throw ()
- virtual const std::string [heading](#) () const =0 throw ()
- virtual const std::string [classname](#) () const =0 throw ()
- virtual void [print\\_error\\_msg](#) () const =0

### Public Attributes

- std::string [name](#)

#### 6.15.1 Detailed Description

Abstract exception class.

Definition at line 41 of file errors.h.

#### 6.15.2 Constructor & Destructor Documentation

##### 6.15.2.1 glucat\_error()

```
glucat::glucat_error::glucat_error (  
    const std::string & context,  
    const std::string & msg ) [inline]
```

Definition at line 44 of file errors.h.

#### 6.15.2.2 ~glucat\_error()

```
glucat::glucat_error::~~glucat_error ( ) throw )    [inline]
```

Definition at line 47 of file errors.h.

### 6.15.3 Member Function Documentation

#### 6.15.3.1 classname()

```
virtual const std::string glucat::glucat_error::classname ( ) const throw )    [pure virtual]
```

Implemented in [glucat::error< Class\\_T >](#).

#### 6.15.3.2 heading()

```
virtual const std::string glucat::glucat_error::heading ( ) const throw )    [pure virtual]
```

Implemented in [glucat::error< Class\\_T >](#).

#### 6.15.3.3 print\_error\_msg()

```
virtual void glucat::glucat_error::print_error_msg ( ) const    [pure virtual]
```

Implemented in [glucat::error< Class\\_T >](#).

### 6.15.4 Member Data Documentation

#### 6.15.4.1 name

```
std::string glucat::glucat_error::name
```

Definition at line 52 of file errors.h.

The documentation for this class was generated from the following file:

- [glucat/errors.h](#)

## 6.16 glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size\_t Class Reference

### Public Member Functions

- [hash\\_size\\_t](#) (size\_t hash\_size)
- [size\\_t operator\(\)](#) () const

### Private Attributes

- [size\\_t n](#)

#### 6.16.1 Detailed Description

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
class glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t
```

Definition at line 180 of file framed\_multi.h.

#### 6.16.2 Constructor & Destructor Documentation

##### 6.16.2.1 hash\_size\_t()

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t::hash_size_t (
    size_t hash_size ) [inline]
```

Definition at line 183 of file framed\_multi.h.

#### 6.16.3 Member Function Documentation

##### 6.16.3.1 operator>()

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
size_t glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t::operator() ( ) const [inline]
```

Definition at line 186 of file framed\_multi.h.

References [glucat::framed\\_multi< Scalar\\_T, LO, HI >::hash\\_size\\_t::n](#).

## 6.16.4 Member Data Documentation

### 6.16.4.1 n

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
size_t glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t::n [private]
```

Definition at line 189 of file framed\_multi.h.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size\_t::operator()().

The documentation for this class was generated from the following file:

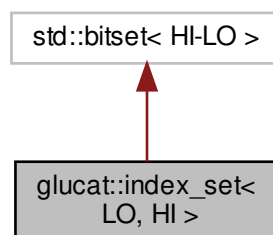
- glucat/framed\_multi.h

## 6.17 glucat::index\_set< LO, HI > Class Template Reference

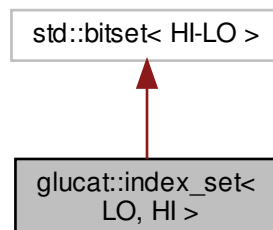
Index set class based on std::bitset<> in Gnu standard C++ library.

```
#include <index_set.h>
```

Inheritance diagram for glucat::index\_set< LO, HI >:



Collaboration diagram for glucat::index\_set< LO, HI >:





## Classes

- class [reference](#)  
*Index set member reference.*

## Public Types

- typedef [index\\_set](#) [index\\_set\\_t](#)
- typedef std::pair< [index\\_t](#), [index\\_t](#) > [index\\_pair\\_t](#)

## Public Member Functions

- [index\\_set](#) ()  
*Default constructor creates an empty set.*
- [index\\_set](#) (const [bitset\\_t](#) bst)  
*Constructor from [bitset\\_t](#).*
- [index\\_set](#) (const [index\\_t](#) idx)  
*Constructor from [index](#).*
- [index\\_set](#) (const [set\\_value\\_t](#) folded\_val, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Constructor from set value of an index set folded within the given frame.*
- [index\\_set](#) (const [index\\_pair\\_t](#) &range, const bool prechecked=false)  
*Constructor from range of indices from *range.first* to *range.second*.*
- [index\\_set](#) (const std::string &str)  
*Constructor from string.*
- bool [operator==](#) (const [index\\_set\\_t](#) rhs) const  
*Equality.*
- bool [operator!=](#) (const [index\\_set\\_t](#) rhs) const  
*Inequality.*
- [index\\_set\\_t](#) [operator~](#) () const  
*Set complement: not.*
- [index\\_set\\_t](#) & [operator^](#) = (const [index\\_set\\_t](#) rhs)  
*Symmetric set difference: exclusive or.*
- [index\\_set\\_t](#) & [operator &=](#) (const [index\\_set\\_t](#) rhs)  
*Set intersection: and.*
- [index\\_set\\_t](#) & [operator|=](#) (const [index\\_set\\_t](#) rhs)  
*Set union: or.*
- bool [operator\[\]](#) (const [index\\_t](#) idx) const  
*Subscripting: Test *idx* for membership: test value of bit *idx*.*
- bool [test](#) (const [index\\_t](#) idx) const  
*Test *idx* for membership: test value of bit *idx*.*
- [index\\_set\\_t](#) & [set](#) ()  
*Include all indices except 0: set all bits except 0.*
- [index\\_set\\_t](#) & [set](#) (const [index\\_t](#) idx)  
*Include *idx*: Set bit at *idx* if *idx* != 0.*
- [index\\_set\\_t](#) & [set](#) (const [index\\_t](#) idx, const int val)  
*Set membership of *idx* to *val* if *idx* != 0: Set bit at *idx* to *val* if *idx* != 0.*
- [index\\_set\\_t](#) & [reset](#) ()  
*Make set empty: Set all bits to 0.*
- [index\\_set\\_t](#) & [reset](#) (const [index\\_t](#) idx)  
*Exclude *idx*: Set bit at *idx* to 0.*

- `index_set_t & flip ()`  
*Set complement, except 0: flip all bits, except 0.*
- `index_set_t & flip (const index_t idx)`  
*Complement membership of idx if idx != 0: flip bit at idx if idx != 0.*
- `index_t count () const`  
*Cardinality: Number of indices included in set.*
- `index_t count_neg () const`  
*Number of negative indices included in set.*
- `index_t count_pos () const`  
*Number of positive indices included in set.*
- `index_t min () const`  
*Minimum member.*
- `index_t max () const`  
*Maximum member.*
- `bool operator< (const index_set_t rhs) const`  
*Less than operator used for comparisons, map, etc.*
- `bool is_contiguous () const`  
*Determine if the index set is contiguous, ie. has no gaps.*
- `const index_set_t fold () const`  
*Fold this index set within itself as a frame.*
- `const index_set_t fold (const index_set_t frm, const bool prechecked=false) const`  
*Fold this index set within the given frame.*
- `const index_set_t unfold (const index_set_t frm, const bool prechecked=false) const`  
*Unfold this index set within the given frame.*
- `set_value_t value_of_fold (const index_set_t frm) const`  
*The set value of the fold of this index set within the given frame.*
- `int sign_of_mult (const index_set_t ist) const`  
*Sign of geometric product of two Clifford basis elements.*
- `int sign_of_square () const`  
*Sign of geometric square of a Clifford basis element.*
- `size_t hash_fn () const`  
*Hash function.*
- `reference operator[] (index_t idx)`  
*Subscripting: Element access.*

## Static Public Member Functions

- `static const std::string classname ()`

## Static Public Attributes

- `static const index_t v_lo = LO`
- `static const index_t v_hi = HI`

## Private Types

- `typedef std::bitset< HI-LO > bitset_t`
- `typedef error< index_set > error_t`

## Private Member Functions

- `BOOST_STATIC_ASSERT` ((LO<=0) &&(0<=HI) &&(LO< HI) &&(-LO< \_GLUCAT\_BITS\_PER\_ULONG) &&(HI< \_GLUCAT\_BITS\_PER\_ULONG) &&(HI-LO<=\_GLUCAT\_BITS\_PER\_ULONG))
- `bool lex_less_than` (const `index_set_t` rhs) const  
*Lexicographic ordering of two sets: \*this < rhs.*

## Friends

- class `reference`
- const `index_set_t` `operator^` (const `index_set_t` &lhs, const `index_set_t` &rhs)
- const `index_set_t` `operator &` (const `index_set_t` &lhs, const `index_set_t` &rhs)
- const `index_set_t` `operator|` (const `index_set_t` &lhs, const `index_set_t` &rhs)
- int `compare` (const `index_set_t` &lhs, const `index_set_t` &rhs)

### 6.17.1 Detailed Description

```
template<const index_t LO, const index_t HI>
class glucat::index_set< LO, HI >
```

Index set class based on `std::bitset<>` in Gnu standard C++ library.

Definition at line 45 of file `index_set.h`.

### 6.17.2 Member Typedef Documentation

#### 6.17.2.1 `bitset_t`

```
template<const index_t LO, const index_t HI>
typedef std::bitset<HI-LO> glucat::index_set< LO, HI >::bitset_t [private]
```

Definition at line 81 of file `index_set.h`.

#### 6.17.2.2 `error_t`

```
template<const index_t LO, const index_t HI>
typedef error<index_set> glucat::index_set< LO, HI >::error_t [private]
```

Definition at line 82 of file `index_set.h`.

### 6.17.2.3 index\_pair\_t

```
template<const index_t LO, const index_t HI>
typedef std::pair<index_t, index_t> glucat::index_set< LO, HI >::index_pair_t
```

Definition at line 85 of file index\_set.h.

### 6.17.2.4 index\_set\_t

```
template<const index_t LO, const index_t HI>
typedef index_set glucat::index_set< LO, HI >::index_set_t
```

Definition at line 84 of file index\_set.h.

## 6.17.3 Constructor & Destructor Documentation

### 6.17.3.1 index\_set() [1/6]

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::index_set ( ) [inline]
```

Default constructor creates an empty set.

Definition at line 92 of file index\_set.h.

### 6.17.3.2 index\_set() [2/6]

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::index_set (
    const bitset_t bst )
```

Constructor from bitset\_t.

Definition at line 61 of file index\_set\_imp.h.

**6.17.3.3 index\_set()** [3/6]

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::index_set (
    const index_t idx )
```

Constructor from index.

Constructor from index value.

Definition at line 55 of file index\_set\_imp.h.

**6.17.3.4 index\_set()** [4/6]

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::index_set (
    const set_value_t folded_val,
    const index_set_t frm,
    const bool prechecked = false )
```

Constructor from set value of an index set folded within the given frame.

Definition at line 68 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::count(), glucat::index\_set< LO, HI >::fold(), glucat::index\_set< LO, HI >::min(), and glucat::index\_set< LO, HI >::unfold().

**6.17.3.5 index\_set()** [5/6]

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::index_set (
    const index_pair_t & range,
    const bool prechecked = false )
```

Constructor from range of indices from range.first to range.second.

Definition at line 82 of file index\_set\_imp.h.

**6.17.3.6 index\_set()** [6/6]

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::index_set (
    const std::string & str )
```

Constructor from string.

Definition at line 102 of file index\_set\_imp.h.

## 6.17.4 Member Function Documentation

### 6.17.4.1 BOOST\_STATIC\_ASSERT()

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::BOOST_STATIC_ASSERT (
    (LO<=0) && (0<=HI) && (LO< HI) && (-LO< _GLUCAT_BITS_PER_ULONG) && (HI< _GLUCAT_BITS_PER_ULONG) && (HI-LO<=_GLUCAT_BITS_PER_ULONG) ) [private]
```

### 6.17.4.2 classname()

```
template<const index_t LO, const index_t HI>
const std::string glucat::index_set< LO, HI >::classname ( ) [inline], [static]
```

Definition at line 49 of file index\_set\_imp.h.

### 6.17.4.3 count()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::count ( ) const [inline]
```

Cardinality: Number of indices included in set.

Definition at line 344 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::count\_neg(), glucat::index\_set< LO, HI >::count\_pos(), glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi(), glucat::index\_set< LO, HI >::index\_set(), glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi(), glucat::index\_set< LO, HI >::operator<(), and glucat::framed\_multi< Scalar\_T, LO, HI >::random().

### 6.17.4.4 count\_neg()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::count_neg ( ) const [inline]
```

Number of negative indices included in set.

Definition at line 364 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::count().

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi(), and glucat::folded\_dim().

## 6.17.4.5 count\_pos()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::count_pos ( ) const [inline]
```

Number of positive indices included in set.

Definition at line 376 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::count().

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fast\_matrix\_multi(), and glucat::folded\_dim().

## 6.17.4.6 flip() [1/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::flip ( ) [inline]
```

Set complement, except 0: flip all bits, except 0.

Definition at line 319 of file index\_set\_imp.h.

## 6.17.4.7 flip() [2/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::flip (
    const index_t idx ) [inline]
```

Complement membership of idx if idx != 0: flip bit at idx if idx != 0.

Definition at line 330 of file index\_set\_imp.h.

## 6.17.4.8 fold() [1/2]

```
template<const index_t LO, const index_t HI>
const index_set< LO, HI > glucat::index_set< LO, HI >::fold ( ) const [inline]
```

Fold this index set within itself as a frame.

Definition at line 748 of file index\_set\_imp.h.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element(), glucat::index\_set< LO, HI >::index\_set(), and glucat::index\_set< LO, HI >::value\_of\_fold().

**6.17.4.9 fold()** [2/2]

```
template<const index_t LO, const index_t HI>
const index_set< LO, HI > glucat::index_set< LO, HI >::fold (
    const index_set_t frm,
    const bool prechecked = false ) const
```

Fold this index set within the given frame.

Definition at line 756 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::max(), glucat::index\_set< LO, HI >::min(), glucat::index\_set< LO, HI >::set(), and glucat::index\_set< LO, HI >::test().

**6.17.4.10 hash\_fn()**

```
template<const index_t LO, const index_t HI>
size_t glucat::index_set< LO, HI >::hash_fn ( ) const [inline]
```

Hash function.

Definition at line 948 of file index\_set\_imp.h.

Referenced by glucat::index\_set\_hash< LO, HI >::operator()().

**6.17.4.11 is\_contiguous()**

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::is_contiguous ( ) const [inline]
```

Determine if the index set is contiguous, ie. has no gaps.

Determine if the index set is contiguous, ie. has no gaps when 0 is included.

Definition at line 732 of file index\_set\_imp.h.

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::fold(), and glucat::framed\_multi< Scalar\_T, LO, HI >::unfold().

**6.17.4.12 lex\_less\_than()**

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::lex_less_than (
    const index_set_t rhs ) const [inline], [private]
```

Lexicographic ordering of two sets: \*this < rhs.

Definition at line 588 of file index\_set\_imp.h.

Referenced by glucat::compare().



## 6.17.4.13 max()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::max ( ) const
```

Maximum member.

Maximum member, or 0 if none.

Definition at line 550 of file index\_set\_imp.h.

Referenced by PyClical.index\_set::\_\_iter\_\_(), glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element(), glucat::index\_set< LO, HI >::fold(), glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi(), glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi(), and glucat::index\_set< LO, HI >::unfold().

## 6.17.4.14 min()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::min ( ) const
```

Minimum member.

Minimum member, or 0 if none.

Definition at line 461 of file index\_set\_imp.h.

Referenced by PyClical.index\_set::\_\_iter\_\_(), glucat::matrix\_multi< Scalar\_T, LO, HI >::basis\_element(), glucat::index\_set< LO, HI >::fold(), glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi(), glucat::index\_set< LO, HI >::index\_set(), glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi(), glucat::index\_set< LO, HI >::unfold(), and glucat::index\_set< LO, HI >::value\_of\_fold().

## 6.17.4.15 operator &amp;=()

```
template<const index_t LO, const index_t HI>
index_set_t& glucat::index_set< LO, HI >::operator&= (
    const index_set_t rhs )
```

Set intersection: and.

## 6.17.4.16 operator!=(=)

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::operator!= (
    const index_set_t rhs ) const [inline]
```

Inequality.

Definition at line 130 of file index\_set\_imp.h.

**6.17.4.17 operator<()**

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::operator< (
    const index_set_t rhs ) const [inline]
```

Less than operator used for comparisons, map, etc.

Definition at line 597 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::count().

**6.17.4.18 operator==()**

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::operator== (
    const index_set_t rhs ) const [inline]
```

Equality.

Definition at line 119 of file index\_set\_imp.h.

**6.17.4.19 operator[]()** [1/2]

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::operator[] (
    const index_t idx ) const [inline]
```

Subscripting: Test idx for membership: test value of bit idx.

Definition at line 232 of file index\_set\_imp.h.

**6.17.4.20 operator[]()** [2/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI >::reference glucat::index_set< LO, HI >::operator[] (
    index_t idx ) [inline]
```

Subscripting: Element access.

Definition at line 224 of file index\_set\_imp.h.

**6.17.4.21 operator^=()**

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::operator^= (
    const index_set_t rhs ) [inline]
```

Symmetric set difference: exclusive or.

Definition at line 149 of file index\_set\_imp.h.

**6.17.4.22 operator" |=()**

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::operator|= (
    const index_set_t rhs ) [inline]
```

Set union: or.

Definition at line 199 of file index\_set\_imp.h.

**6.17.4.23 operator~()**

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > glucat::index_set< LO, HI >::operator~ ( ) const [inline]
```

Set complement: not.

Definition at line 141 of file index\_set\_imp.h.

**6.17.4.24 reset()** [1/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::reset ( ) [inline]
```

Make set empty: Set all bits to 0.

Definition at line 294 of file index\_set\_imp.h.

**6.17.4.25 reset()** [2/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::reset (
    const index_t idx ) [inline]
```

Exclude idx: Set bit at idx to 0.

Definition at line 305 of file index\_set\_imp.h.

**6.17.4.26 set()** [1/3]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::set ( ) [inline]
```

Include all indices except 0: set all bits except 0.

Definition at line 255 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::fold(), glucat::operator>>(), and glucat::index\_set< LO, HI >↵::unfold().

**6.17.4.27 set()** [2/3]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::set (
    const index_t idx ) [inline]
```

Include idx: Set bit at idx if idx != 0.

Definition at line 266 of file index\_set\_imp.h.

**6.17.4.28 set()** [3/3]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI > & glucat::index_set< LO, HI >::set (
    const index_t idx,
    const int val ) [inline]
```

Set membership of idx to val if idx != 0: Set bit at idx to val if idx != 0.

Definition at line 280 of file index\_set\_imp.h.

## 6.17.4.29 sign\_of\_mult()

```
template<const index_t LO, const index_t HI>
int glucat::index_set< LO, HI >::sign_of_mult (
    const index_set_t ist ) const
```

Sign of geometric product of two Clifford basis elements.

Definition at line 879 of file index\_set\_imp.h.

References glucat::inverse\_gray(), and glucat::inverse\_reversed\_gray().

## 6.17.4.30 sign\_of\_square()

```
template<const index_t LO, const index_t HI>
int glucat::index_set< LO, HI >::sign_of_square ( ) const [inline]
```

Sign of geometric square of a Clifford basis element.

Definition at line 928 of file index\_set\_imp.h.

## 6.17.4.31 test()

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::test (
    const index_t idx ) const [inline]
```

Test idx for membership: test value of bit idx.

Definition at line 240 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::fold(), and glucat::index\_set< LO, HI >::unfold().

## 6.17.4.32 unfold()

```
template<const index_t LO, const index_t HI>
const index_set< LO, HI > glucat::index_set< LO, HI >::unfold (
    const index_set_t frm,
    const bool prechecked = false ) const
```

Unfold this index set within the given frame.

Definition at line 794 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::max(), glucat::index\_set< LO, HI >::min(), glucat::index\_set< LO, HI >::set(), and glucat::index\_set< LO, HI >::test().

Referenced by glucat::index\_set< LO, HI >::index\_set().

#### 6.17.4.33 value\_of\_fold()

```
template<const index_t LO, const index_t HI>
set_value_t glucat::index_set< LO, HI >::value_of_fold (
    const index_set_t frm ) const [inline]
```

The set value of the fold of this index set within the given frame.

Definition at line 828 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::fold(), and glucat::index\_set< LO, HI >::min().

### 6.17.5 Friends And Related Function Documentation

#### 6.17.5.1 compare

```
template<const index_t LO, const index_t HI>
int compare (
    const index_set_t & lhs,
    const index_set_t & rhs ) [friend]
```

#### 6.17.5.2 operator &

```
template<const index_t LO, const index_t HI>
const index_set_t operator& (
    const index_set_t & lhs,
    const index_set_t & rhs ) [friend]
```

#### 6.17.5.3 operator^

```
template<const index_t LO, const index_t HI>
const index_set_t operator^ (
    const index_set_t & lhs,
    const index_set_t & rhs ) [friend]
```

#### 6.17.5.4 operator" |

```
template<const index_t LO, const index_t HI>
const index_set_t operator| (
    const index_set_t & lhs,
    const index_set_t & rhs ) [friend]
```

## 6.17.5.5 reference

```
template<const index_t LO, const index_t HI>
friend class reference [friend]
```

Definition at line 173 of file index\_set.h.

## 6.17.6 Member Data Documentation

## 6.17.6.1 v\_hi

```
template<const index_t LO, const index_t HI>
const index_t glucat::index_set< LO, HI >::v_hi = HI [static]
```

Definition at line 88 of file index\_set.h.

## 6.17.6.2 v\_lo

```
template<const index_t LO, const index_t HI>
const index_t glucat::index_set< LO, HI >::v_lo = LO [static]
```

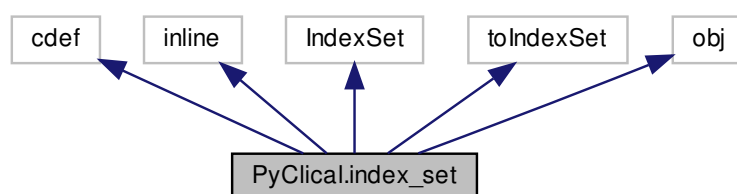
Definition at line 87 of file index\_set.h.

The documentation for this class was generated from the following files:

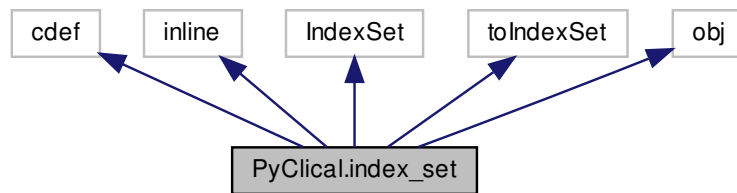
- glucat/index\_set.h
- glucat/index\_set\_imp.h

## 6.18 PyClical.index\_set Class Reference

Inheritance diagram for PyClical.index\_set:



Collaboration diagram for PyClical.index\_set:



## Public Member Functions

- `def __cinit__(self, other=0)`
- `def __dealloc__(self)`
- `def __richcmp__(lhs, rhs, int, op)`
- `def __setitem__(self, idx, val)`
- `def __getitem__(self, idx)`
- `def __contains__(self, idx)`
- `def __iter__(self)`
- `def __invert__(self)`
- `def __xor__(lhs, rhs)`
- `def __ixor__(self, rhs)`
- `def __and__(lhs, rhs)`
- `def __iand__(self, rhs)`
- `def __or__(lhs, rhs)`
- `def __ior__(self, rhs)`
- `def count(self)`
- `def count_neg(self)`
- `def count_pos(self)`
- `def min(self)`
- `def max(self)`
- `def hash_fn(self)`
- `def sign_of_mult(self, rhs)`
- `def sign_of_square(self)`
- `def __repr__(self)`
- `def __str__(self)`

## Public Attributes

- `instance`

## 6.18.1 Detailed Description

Return the C++ `IndexSet` instance wrapped by `index_set(obj)`.

Python class `index_set` wraps C++ class `IndexSet`.

Definition at line 39 of file `PyClical.pyx`.



## 6.18.2 Member Function Documentation

### 6.18.2.1 `__and__()`

```
def PyClical.index_set.__and__ (
    lhs,
    rhs )
```

Set intersection: and.

```
>>> print index_set({1}) & index_set({2})
{}
>>> print index_set({1,2}) & index_set({2})
{2}
```

Definition at line 269 of file PyClical.pyx.

### 6.18.2.2 `__cinit__()`

```
def PyClical.index_set.__cinit__ (
    self,
    other = 0 )
```

Construct an object of type index\_set.

```
>>> print index_set(1)
{1}
>>> print index_set({1,2})
{1,2}
>>> print index_set(index_set({1,2}))
{1,2}
>>> print index_set({1,2})
{1,2}
>>> print index_set({1,2,1})
{1,2}
>>> print index_set("{1,2,1}")
{1,2}
>>> print index_set("")
{}
```

Definition at line 73 of file PyClical.pyx.

### 6.18.2.3 `__contains__()`

```
def PyClical.index_set.__contains__ (
    self,
    idx )
```

Check that an `index_set` object contains the index `idx`: `idx` in `self`.

```
>>> 1 in index_set({1})
True
>>> 2 in index_set({1})
False
>>> -1 in index_set({2})
False
>>> 1 in index_set({2})
False
>>> 2 in index_set({2})
True
>>> 33 in index_set({2})
False
```

Definition at line 208 of file `PyClical.pyx`.

References `PyClical.index_set.instance`.

### 6.18.2.4 `__dealloc__()`

```
def PyClical.index_set.__dealloc__ (
    self )
```

Clean up by deallocating the instance of C++ class `IndexSet`.

Definition at line 114 of file `PyClical.pyx`.

References `PyClical.index_set.instance`.

### 6.18.2.5 `__getitem__()`

```
def PyClical.index_set.__getitem__ (
    self,
    idx )
```

Get the value of an `index_set` object at an index.

```
>>> index_set({1})[1]
True
>>> index_set({1})[2]
False
>>> index_set({2})[-1]
False
>>> index_set({2})[1]
False
>>> index_set({2})[2]
True
>>> index_set({2})[33]
False
```

Definition at line 189 of file `PyClical.pyx`.

References `PyClical.index_set.instance`.

#### 6.18.2.6 \_\_iand\_\_()

```
def PyClical.index_set.__iand__ (
    self,
    rhs )

Set intersection: and.

>>> x = index_set({1}); x &= index_set({2}); print x
{}
>>> x = index_set({1,2}); x &= index_set({2}); print x
{2}
```

Definition at line 280 of file PyClical.pyx.

#### 6.18.2.7 \_\_invert\_\_()

```
def PyClical.index_set.__invert__ (
    self )

Set complement: not.

>>> print ~index_set({-16,-15,-14,-13,-12,-11,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,1,2,3,4,5,6,7,8,9,10,11,12,13,14,
{-32,-31,-30,-29,-28,-27,-26,-25,-24,-23,-22,-21,-20,-19,-18,-17,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,
```

Definition at line 238 of file PyClical.pyx.

References `PyClical.index_set.instance`.

#### 6.18.2.8 \_\_ior\_\_()

```
def PyClical.index_set.__ior__ (
    self,
    rhs )

Set union: or.

>>> x = index_set({1}); x |= index_set({2}); print x
{1,2}
>>> x = index_set({1,2}); x |= index_set({2}); print x
{1,2}
```

Definition at line 302 of file PyClical.pyx.

#### 6.18.2.9 `__iter__()`

```
def PyClical.index_set.__iter__ (
    self )
```

Iterate over the indices of an `index_set`.

```
>>> for i in index_set({-3,4,7}): print i,
-3 4 7
```

Definition at line 227 of file `PyClical.pyx`.

References `glucat::index_set< LO, HI >.max()`, `PyClical.index_set.max()`, `glucat::index_set< LO, HI >.min()`, and `PyClical.index_set.min()`.

#### 6.18.2.10 `__ixor__()`

```
def PyClical.index_set.__ixor__ (
    self,
    rhs )
```

Symmetric set difference: exclusive or.

```
>>> x = index_set({1}); x ^= index_set({2}); print x
{1,2}
>>> x = index_set({1,2}); x ^= index_set({2}); print x
{1}
```

Definition at line 258 of file `PyClical.pyx`.

#### 6.18.2.11 `__or__()`

```
def PyClical.index_set.__or__ (
    lhs,
    rhs )
```

Set union: or.

```
>>> print index_set({1}) | index_set({2})
{1,2}
>>> print index_set({1,2}) | index_set({2})
{1,2}
```

Definition at line 291 of file `PyClical.pyx`.

#### 6.18.2.12 `__repr__()`

```
def PyClical.index_set.__repr__ (
    self )
```

The “official” string representation of self.

```
>>> index_set({1,2}).__repr__()
'index_set({1,2})'
>>> repr(index_set({1,2}))
'index_set({1,2})'
```

Definition at line 382 of file `PyClical.pyx`.

References `index_set_to_repr()`.

#### 6.18.2.13 `__richcmp__()`

```
def PyClical.index_set.__richcmp__ (
    lhs,
    rhs,
    int,
    op )
```

Compare two objects of class `index_set`.

```
>>> index_set(1) == index_set({1})
True
>>> index_set({1}) != index_set({1})
False
>>> index_set({1}) != index_set({2})
True
>>> index_set({1}) == index_set({2})
False
>>> index_set({1}) < index_set({2})
True
>>> index_set({1}) <= index_set({2})
True
>>> index_set({1}) > index_set({2})
False
>>> index_set({1}) >= index_set({2})
False
```

Definition at line 120 of file `PyClical.pyx`.

#### 6.18.2.14 `__setitem__()`

```
def PyClical.index_set.__setitem__ (
    self,
    idx,
    val )
```

Set the value of an `index_set` object at index `idx` to value `val`.

```
>>> s=index_set({1}); s[2] = True; print s
{1,2}
>>> s=index_set({1,2}); s[1] = False; print s
{2}
```

Definition at line 177 of file `PyClical.pyx`.

References `PyClical.index_set.instance`.

#### 6.18.2.15 `__str__()`

```
def PyClical.index_set.__str__ (
    self )
```

The "informal" string representation of self.

```
>>> index_set({1,2}).__str__()
'{1,2}'
>>> str(index_set({1,2}))
'{1,2}'
```

Definition at line 393 of file PyClical.pyx.

References `index_set_to_str()`.

#### 6.18.2.16 `__xor__()`

```
def PyClical.index_set.__xor__ (
    lhs,
    rhs )
```

Symmetric set difference: exclusive or.

```
>>> print index_set({1}) ^ index_set({2})
{1,2}
>>> print index_set({1,2}) ^ index_set({2})
{1}
```

Definition at line 247 of file PyClical.pyx.

#### 6.18.2.17 `count()`

```
def PyClical.index_set.count (
    self )
```

Cardinality: Number of indices included in set.

```
>>> index_set({-1,1,2}).count()
3
```

Definition at line 313 of file PyClical.pyx.

References `PyClical.index_set.instance`.

#### 6.18.2.18 count\_neg()

```
def PyClical.index_set.count_neg (
    self )
```

Number of negative indices included in set.

```
>>> index_set({-1,1,2}).count_neg()
1
```

Definition at line 322 of file PyClical.pyx.

References `PyClical.index_set.instance`.

#### 6.18.2.19 count\_pos()

```
def PyClical.index_set.count_pos (
    self )
```

Number of positive indices included in set.

```
>>> index_set({-1,1,2}).count_pos()
2
```

Definition at line 331 of file PyClical.pyx.

References `PyClical.index_set.instance`.

#### 6.18.2.20 hash\_fn()

```
def PyClical.index_set.hash_fn (
    self )
```

Hash function.

Definition at line 358 of file PyClical.pyx.

References `PyClical.index_set.instance`.

#### 6.18.2.21 max()

```
def PyClical.index_set.max (
    self )
```

Maximum member.

```
>>> index_set({-1,1,2}).max()
2
```

Definition at line 349 of file PyClical.pyx.

References PyClical.index\_set.instance.

Referenced by PyClical.index\_set.\_\_iter\_\_().

#### 6.18.2.22 min()

```
def PyClical.index_set.min (
    self )
```

Minimum member.

```
>>> index_set({-1,1,2}).min()
-1
```

Definition at line 340 of file PyClical.pyx.

References PyClical.index\_set.instance.

Referenced by PyClical.index\_set.\_\_iter\_\_().

#### 6.18.2.23 sign\_of\_mult()

```
def PyClical.index_set.sign_of_mult (
    self,
    rhs )
```

Sign of geometric product of two Clifford basis elements.

```
>>> s = index_set({1,2}); t=index_set({-1}); s.sign_of_mult(t)
1
```

Definition at line 364 of file PyClical.pyx.

References PyClical.index\_set.instance.



## 6.18.2.24 sign\_of\_square()

```
def PyClical.index_set.sign_of_square (
    self )
```

Sign of geometric square of a Clifford basis element.

```
>>> s = index_set({1,2}); s.sign_of_square()
-1
```

Definition at line 373 of file PyClical.pyx.

References PyClical.index\_set.instance.

## 6.18.3 Member Data Documentation

## 6.18.3.1 instance

PyClical.index\_set.instance

Definition at line 94 of file PyClical.pyx.

Referenced by PyClical.clifford.\_\_call\_\_(), PyClical.index\_set.\_\_contains\_\_(), PyClical.index\_set.\_\_dealloc\_\_(), PyClical.clifford.\_\_dealloc\_\_(), PyClical.index\_set.\_\_getitem\_\_(), PyClical.clifford.\_\_getitem\_\_(), PyClical.index\_set.\_\_invert\_\_(), PyClical.clifford.\_\_neg\_\_(), PyClical.index\_set.\_\_setitem\_\_(), PyClical.clifford.conj(), PyClical.index\_set.count(), PyClical.index\_set.count\_neg(), PyClical.index\_set.count\_pos(), PyClical.clifford.even(), PyClical.clifford.frame(), PyClical.index\_set.hash\_fn(), PyClical.clifford.inv(), PyClical.clifford.involute(), PyClical.clifford.isnan(), PyClical.index\_set.max(), PyClical.clifford.max\_abs(), PyClical.index\_set.min(), PyClical.clifford.norm(), PyClical.clifford.odd(), PyClical.clifford.outer\_pow(), PyClical.clifford.pow(), PyClical.clifford.pure(), PyClical.clifford.quad(), PyClical.clifford.reverse(), PyClical.clifford.scalar(), PyClical.index\_set.sign\_of\_mult(), PyClical.index\_set.sign\_of\_square(), PyClical.clifford.truncated(), and PyClical.clifford.vector\_part().

The documentation for this class was generated from the following file:

- [pyclical/PyClical.pyx](#)

## 6.19 glucat::index\_set\_hash&lt; LO, HI &gt; Class Template Reference

```
#include <framed_multi.h>
```

## Public Types

- typedef [index\\_set](#)< LO, HI > [index\\_set\\_t](#)

## Public Member Functions

- `size_t operator() (index_set_t val) const`

### 6.19.1 Detailed Description

```
template<const index_t LO, const index_t HI>  
class glucat::index_set_hash< LO, HI >
```

Definition at line 126 of file framed\_multi.h.

### 6.19.2 Member Typedef Documentation

#### 6.19.2.1 index\_set\_t

```
template<const index_t LO, const index_t HI>  
typedef index_set<LO,HI> glucat::index_set_hash< LO, HI >::index_set_t
```

Definition at line 129 of file framed\_multi.h.

### 6.19.3 Member Function Documentation

#### 6.19.3.1 operator()()

```
template<const index_t LO, const index_t HI>  
size_t glucat::index_set_hash< LO, HI >::operator() (  
    index_set_t val ) const [inline]
```

Definition at line 130 of file framed\_multi.h.

References `glucat::index_set< LO, HI >::hash_fn()`.

The documentation for this class was generated from the following file:

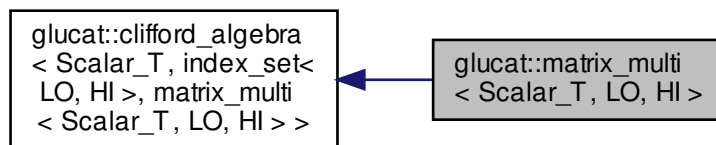
- `glucat/framed_multi.h`

## 6.20 glucat::matrix\_multi< Scalar\_T, LO, HI > Class Template Reference

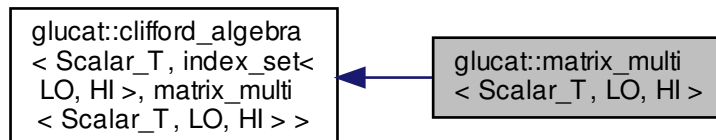
A matrix\_multi<Scalar\_T,LO,HI> is a matrix approximation to a multivector.

```
#include <framed_multi.h>
```

Inheritance diagram for glucat::matrix\_multi< Scalar\_T, LO, HI >:



Collaboration diagram for glucat::matrix\_multi< Scalar\_T, LO, HI >:



### Public Types

- typedef [matrix\\_multi](#) [multivector\\_t](#)
- typedef [multivector\\_t](#) [matrix\\_multi\\_t](#)
- typedef [Scalar\\_T](#) [scalar\\_t](#)
- typedef [index\\_set< LO, HI >](#) [index\\_set\\_t](#)
- typedef [std::pair< const index\\_set\\_t, Scalar\\_T >](#) [term\\_t](#)
- typedef [std::vector< Scalar\\_T >](#) [vector\\_t](#)
- typedef [error< multivector\\_t >](#) [error\\_t](#)
- typedef [framed\\_multi< Scalar\\_T, LO, HI >](#) [framed\\_multi\\_t](#)

## Public Member Functions

- [~matrix\\_multi](#) ()  
*Destructor.*
- [matrix\\_multi](#) ()  
*Default constructor.*
- `template<typename Other_Scalar_T >`  
[matrix\\_multi](#) (const [matrix\\_multi](#)< Other\_Scalar\_T, LO, HI > &val)  
*Construct a multivector from a multivector with a different scalar type.*
- `template<typename Other_Scalar_T >`  
[matrix\\_multi](#) (const [matrix\\_multi](#)< Other\_Scalar\_T, LO, HI > &val, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a given multivector.*
- [matrix\\_multi](#) (const [multivector\\_t](#) &val, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a given multivector.*
- [matrix\\_multi](#) (const [index\\_set\\_t](#) ist, const Scalar\_T &crd=Scalar\_T(1))  
*Construct a multivector from an index set and a scalar coordinate.*
- [matrix\\_multi](#) (const [index\\_set\\_t](#) ist, const Scalar\_T &crd, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from an index set and a scalar coordinate.*
- [matrix\\_multi](#) (const Scalar\_T &scr, const [index\\_set\\_t](#) frm=[index\\_set\\_t](#)())  
*Construct a multivector from a scalar (within a frame, if given)*
- [matrix\\_multi](#) (const int scr, const [index\\_set\\_t](#) frm=[index\\_set\\_t](#)())  
*Construct a multivector from an int (within a frame, if given)*
- [matrix\\_multi](#) (const [vector\\_t](#) &vec, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a given vector.*
- [matrix\\_multi](#) (const std::string &str)  
*Construct a multivector from a string: eg: "3+2{1,2}-6.1e-2{2,3}".*
- [matrix\\_multi](#) (const std::string &str, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a string: eg: "3+2{1,2}-6.1e-2{2,3}".*
- [matrix\\_multi](#) (const char \*str)  
*Construct a multivector from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".*
- [matrix\\_multi](#) (const char \*str, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".*
- `template<typename Other_Scalar_T >`  
[matrix\\_multi](#) (const [framed\\_multi](#)< Other\_Scalar\_T, LO, HI > &val)  
*Construct a multivector from a framed\_multi\_t.*
- `template<typename Other_Scalar_T >`  
[matrix\\_multi](#) (const [framed\\_multi](#)< Other\_Scalar\_T, LO, HI > &val, const [index\\_set\\_t](#) frm, const bool prechecked=false)  
*Construct a multivector, within a given frame, from a framed\_multi\_t.*
- const [matrix\\_multi\\_t](#) [fast\\_matrix\\_multi](#) (const [index\\_set\\_t](#) frm) const  
*Use generalized FFT to construct a matrix\_multi\_t.*
- `template<typename Other_Scalar_T >`  
const [framed\\_multi](#)< Other\_Scalar\_T, LO, HI > [fast\\_framed\\_multi](#) () const  
*Use inverse generalized FFT to construct a framed\_multi\_t.*
- [\\_GLUCAT\\_CLIFFORD\\_ALGEBRA\\_OPERATIONS](#) [multivector\\_t](#) & [operator=](#) (const [multivector\\_t](#) &rhs)  
*Assignment operator.*
- [multivector\\_t](#) & [operator+=](#) (const [term\\_t](#) &rhs)  
*Add a term, if non-zero.*

## Static Public Member Functions

- static const std::string [classname](#) ()  
*Class name used in messages.*
- static const [matrix\\_multi\\_t](#) [random](#) (const [index\\_set\\_t](#) frm, Scalar\_T fill=Scalar\_T(1))  
*Random multivector within a frame.*

## Private Types

- typedef ublas::row\_major [orientation\\_t](#)
- typedef ublas::compressed\_matrix< int, [orientation\\_t](#) > [basis\\_matrix\\_t](#)
- typedef ublas::compressed\_matrix< Scalar\_T, [orientation\\_t](#) > [matrix\\_t](#)
- typedef matrix\_t::size\_type [matrix\\_index\\_t](#)

## Private Member Functions

- template<typename Matrix\_T >  
[matrix\\_multi](#) (const Matrix\_T &mtx, const [index\\_set\\_t](#) frm)  
*Construct a multivector within a given frame from a given matrix.*
- [matrix\\_multi](#) (const [matrix\\_t](#) &mtx, const [index\\_set\\_t](#) frm)  
*Construct a multivector within a given frame from a given matrix.*
- const [basis\\_matrix\\_t](#) [basis\\_element](#) (const [index\\_set](#)< LO, HI > &ist) const  
*Create a basis element matrix within the current frame.*

## Private Attributes

- [index\\_set\\_t](#) [m\\_frame](#)  
*Index set representing the frame for the subalgebra which contains the multivector.*
- [matrix\\_t](#) [m\\_matrix](#)  
*Matrix value representing the multivector within the folded frame.*

## Friends

- template<typename Other\_Scalar\_T , const index\_t Other\_LO, const index\_t Other\_HI>  
class [framed\\_multi](#)
- template<typename Other\_Scalar\_T , const index\_t Other\_LO, const index\_t Other\_HI>  
class [matrix\\_multi](#)
- const [matrix\\_multi\\_t](#) operator\* (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- const [matrix\\_multi\\_t](#) operator^ (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- const [matrix\\_multi\\_t](#) operator & (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- const [matrix\\_multi\\_t](#) operator% (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- Scalar\_T star (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- const [matrix\\_multi\\_t](#) operator/ (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- const [matrix\\_multi\\_t](#) operator| (const [matrix\\_multi\\_t](#) &lhs, const [matrix\\_multi\\_t](#) &rhs)
- std::istream & operator>> (std::istream &s, [multivector\\_t](#) &val)
- std::ostream & operator<< (std::ostream &os, const [multivector\\_t](#) &val)
- std::ostream & operator<< (std::ostream &os, const [term\\_t](#) &term)

- `template<typename Other_Scalar_T, const index_t Other_LO, const index_t Other_HI>`  
`const index\_set< Other_LO, Other_HI > reframe (const matrix\_multi< Other_Scalar_T, Other_LO, Other_HI`  
`> &lhs, const matrix\_multi< Other_Scalar_T, Other_LO, Other_HI > &rhs, matrix\_multi< Other_Scalar_↵`  
`T, Other_LO, Other_HI > &lhs_reframed, matrix\_multi< Other_Scalar_T, Other_LO, Other_HI > &rhs_↵`  
`reframed)`
- `template<typename Other_Scalar_T, const index_t Other_LO, const index_t Other_HI>`  
`const matrix\_multi< Other_Scalar_T, Other_LO, Other_HI > matrix\_sqrt (const matrix\_multi< Other_↵`  
`Scalar_T, Other_LO, Other_HI > &val, const matrix\_multi< Other_Scalar_T, Other_LO, Other_HI > &i)`
- `template<typename Other_Scalar_T, const index_t Other_LO, const index_t Other_HI>`  
`const matrix\_multi< Other_Scalar_T, Other_LO, Other_HI > matrix\_log (const matrix\_multi< Other_↵`  
`Scalar_T, Other_LO, Other_HI > &val, const matrix\_multi< Other_Scalar_T, Other_LO, Other_HI > &i)`

### 6.20.1 Detailed Description

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
class glucat::matrix_multi< Scalar_T, LO, HI >
```

A `matrix_multi<Scalar_T,LO,HI>` is a matrix approximation to a multivector.

Definition at line 68 of file `framed_multi.h`.

### 6.20.2 Member Typedef Documentation

#### 6.20.2.1 `basis_matrix_t`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef ublas::compressed_matrix< int, orientation\_t > glucat::matrix\_multi< Scalar_T, LO, HI
>::basis_matrix_t [private]
```

Definition at line 152 of file `matrix_multi.h`.

#### 6.20.2.2 `error_t`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
typedef error<multivector\_t> glucat::matrix\_multi< Scalar_T, LO, HI >::error_t
```

Definition at line 142 of file `matrix_multi.h`.

### 6.20.2.3 framed\_multi\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef framed_multi<Scalar_T,LO,HI> glucat::matrix_multi< Scalar_T, LO, HI >::framed_multi_t
```

Definition at line 143 of file matrix\_multi.h.

### 6.20.2.4 index\_set\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef index_set<LO,HI> glucat::matrix_multi< Scalar_T, LO, HI >::index_set_t
```

Definition at line 139 of file matrix\_multi.h.

### 6.20.2.5 matrix\_index\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef matrix_t::size_type glucat::matrix_multi< Scalar_T, LO, HI >::matrix_index_t [private]
```

Definition at line 159 of file matrix\_multi.h.

### 6.20.2.6 matrix\_multi\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef multivector_t glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi_t
```

Definition at line 137 of file matrix\_multi.h.

### 6.20.2.7 matrix\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef ublas::compressed_matrix< Scalar_T, orientation_t > glucat::matrix_multi< Scalar_T,
LO, HI >::matrix_t [private]
```

Definition at line 157 of file matrix\_multi.h.

### 6.20.2.8 multivector\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef matrix_multi glucat::matrix_multi< Scalar_T, LO, HI >::multivector_t
```

Definition at line 136 of file matrix\_multi.h.

### 6.20.2.9 orientation\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef ublas::row_major glucat::matrix_multi< Scalar_T, LO, HI >::orientation_t [private]
```

Definition at line 150 of file matrix\_multi.h.

### 6.20.2.10 scalar\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef Scalar_T glucat::matrix_multi< Scalar_T, LO, HI >::scalar_t
```

Definition at line 138 of file matrix\_multi.h.

### 6.20.2.11 term\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef std::pair<const index_set_t, Scalar_T> glucat::matrix_multi< Scalar_T, LO, HI >::term_t
```

Definition at line 140 of file matrix\_multi.h.

### 6.20.2.12 vector\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef std::vector<Scalar_T> glucat::matrix_multi< Scalar_T, LO, HI >::vector_t
```

Definition at line 141 of file matrix\_multi.h.



### 6.20.3 Constructor & Destructor Documentation

#### 6.20.3.1 ~matrix\_multi()

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
glucat::matrix_multi< Scalar_T, LO, HI >::~~matrix_multi ( ) [inline]
```

Destructor.

Definition at line 165 of file matrix\_multi.h.

#### 6.20.3.2 matrix\_multi() [1/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi ( )
```

Default constructor.

Definition at line 97 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi().

#### 6.20.3.3 matrix\_multi() [2/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const matrix_multi< Other_Scalar_T, LO, HI > & val )
```

Construct a multivector from a multivector with a different scalar type.

Definition at line 106 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix.

**6.20.3.4 matrix\_multi()** [3/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const matrix_multi< Other_Scalar_T, LO, HI > & val,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a given multivector.

Definition at line 128 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, and glucat::matrix\_multi< Scalar\_T, LO, HI >↔::m\_matrix.

**6.20.3.5 matrix\_multi()** [4/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const multivector_t & val,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a given multivector.

Definition at line 156 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, and glucat::matrix\_multi< Scalar\_T, LO, HI >↔::m\_matrix.

**6.20.3.6 matrix\_multi()** [5/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const index_set_t ist,
    const Scalar_T & crd = Scalar_T(1) )
```

Construct a multivector from an index set and a scalar coordinate.

Definition at line 168 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, and glucat::matrix\_multi< Scalar\_T, LO, HI >↔::m\_matrix.

**6.20.3.7 matrix\_multi()** [6/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const index_set_t ist,
    const Scalar_T & crd,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from an index set and a scalar coordinate.

Definition at line 180 of file matrix\_multi\_imp.h.

References PyClical::ist, and glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix.

**6.20.3.8 matrix\_multi()** [7/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const Scalar_T & scr,
    const index_set_t frm = index_set_t() )
```

Construct a multivector from a scalar (within a frame, if given)

Definition at line 194 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix.

**6.20.3.9 matrix\_multi()** [8/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const int scr,
    const index_set_t frm = index_set_t() )
```

Construct a multivector from an int (within a frame, if given)

Definition at line 206 of file matrix\_multi\_imp.h.

**6.20.3.10 matrix\_multi()** [9/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const vector_t & vec,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a given vector.

Definition at line 212 of file matrix\_multi\_imp.h.

References glucat::index\_set< LO, HI >::count(), glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix, glucat::index\_set< LO, HI >::max(), and glucat::index\_set< LO, HI >::min().

**6.20.3.11 matrix\_multi()** [10/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const std::string & str )
```

Construct a multivector from a string: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 239 of file matrix\_multi\_imp.h.

**6.20.3.12 matrix\_multi()** [11/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const std::string & str,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a string: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 245 of file matrix\_multi\_imp.h.

**6.20.3.13 matrix\_multi()** [12/17]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const char * str ) [inline]
```

Construct a multivector from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 196 of file matrix\_multi.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi().

**6.20.3.14 matrix\_multi()** [13/17]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↵
LT_HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const char * str,
    const index_set_t frm,
    const bool prechecked = false ) [inline]
```

Construct a multivector, within a given frame, from a char\*: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 199 of file matrix\_multi.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi().

## 6.20.3.15 matrix\_multi() [14/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const framed_multi< Other_Scalar_T, LO, HI > & val )
```

Construct a multivector from a framed\_multi\_t.

Definition at line 252 of file matrix\_multi\_imp.h.

References PyClical::e(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::fast\_size\_threshold, glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, and glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix.

## 6.20.3.16 matrix\_multi() [15/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const framed_multi< Other_Scalar_T, LO, HI > & val,
    const index_set_t frm,
    const bool prechecked = false )
```

Construct a multivector, within a given frame, from a framed\_multi\_t.

Definition at line 279 of file matrix\_multi\_imp.h.

References PyClical::e(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::fast\_size\_threshold, and glucat::clifford\_algebra< Scalar\_T, index\_set< LO, HI >, framed\_multi< Scalar\_T, LO, HI > >::frame().

## 6.20.3.17 matrix\_multi() [16/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Matrix_T >
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const Matrix_T & mtx,
    const index_set_t frm ) [private]
```

Construct a multivector within a given frame from a given matrix.

Definition at line 307 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix.

### 6.20.3.18 matrix\_multi() [17/17]

```
template<typename Scalar_T , const index_t LO, const index_t HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
    const matrix_t & mtx,
    const index_set_t frm ) [private]
```

Construct a multivector within a given frame from a given matrix.

Definition at line 328 of file matrix\_multi\_imp.h.

## 6.20.4 Member Function Documentation

### 6.20.4.1 basis\_element()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI >::basis_matrix_t glucat::matrix_multi< Scalar_T, LO, HI
>::basis_element (
    const index_set< LO, HI > & ist ) const [private]
```

Create a basis element matrix within the current frame.

Definition at line 1243 of file matrix\_multi\_imp.h.

References glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::basis\_max\_count, PyClical::e(), glucat::index\_set< LO, HI >::fold(), glucat::gen::generator\_table< Matrix\_T >::generator(), PyClical::ist, glucat::index\_set< LO, HI >::max(), glucat::index\_set< LO, HI >::min(), glucat::matrix::mono\_prod(), and glucat::offset\_level().

Referenced by glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi().

### 6.20.4.2 classname()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const std::string glucat::matrix_multi< Scalar_T, LO, HI >::classname ( ) [static]
```

Class name used in messages.

Definition at line 69 of file matrix\_multi\_imp.h.

## 6.20.4.3 fast\_framed\_multi()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
template<typename Other_Scalar_T >
const framed_multi< Other_Scalar_T, LO, HI > glucat::matrix_multi< Scalar_T, LO, HI >::fast↵
_framed_multi ( ) const
```

Use inverse generalized FFT to construct a framed\_multi\_t.

Definition at line 1166 of file matrix\_multi\_imp.h.

References glucat::framed\_multi< Scalar\_T, LO, HI >::centre\_pm4\_qp4(), glucat::framed\_multi< Scalar\_T, LO, HI >::centre\_pp4\_qm4(), glucat::framed\_multi< Scalar\_T, LO, HI >::centre\_qp1\_pm1(), glucat::gen::offset\_to\_↵super, glucat::pos\_mod(), and glucat::framed\_multi< Scalar\_T, LO, HI >::unfold().

## 6.20.4.4 fast\_matrix\_multi()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::matrix_multi< Scalar_T, LO, HI >::fast↵
matrix_multi (
    const index_set_t frm ) const [inline]
```

Use generalized FFT to construct a matrix\_multi\_t.

Definition at line 1153 of file matrix\_multi\_imp.h.

## 6.20.4.5 operator+=( )

```
template<typename Scalar_T , const index_t LO, const index_t HI>
matrix_multi< Scalar_T, LO, HI > & glucat::matrix_multi< Scalar_T, LO, HI >::operator+= (
    const term_t & rhs ) [inline]
```

Add a term, if non-zero.

Geometric sum.

Geometric sum of multivector and scalar.

Definition at line 470 of file matrix\_multi\_imp.h.

## 6.20.4.6 operator=( )

```
template<typename Scalar_T , const index_t LO, const index_t HI>
matrix_multi< Scalar_T, LO, HI > & glucat::matrix_multi< Scalar_T, LO, HI >::operator= (
    const multivector_t & rhs )
```

Assignment operator.

Definition at line 336 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, and glucat::matrix\_multi< Scalar\_T, LO, HI >↵::m\_matrix.

#### 6.20.4.7 random()

```
template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::matrix_multi< Scalar_T, LO, HI >::random (
    const index_set_t frm,
    Scalar_T fill = Scalar_T(1) ) [static]
```

Random multivector within a frame.

Definition at line 996 of file matrix\_multi\_imp.h.

References PyClical::fill, and glucat::framed\_multi< Scalar\_T, LO, HI >::random().

### 6.20.5 Friends And Related Function Documentation

#### 6.20.5.1 framed\_multi

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
friend class framed_multi [friend]
```

Definition at line 145 of file matrix\_multi.h.

#### 6.20.5.2 matrix\_log

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
const matrix_multi<Other_Scalar_T,Other_LO,Other_HI> matrix_log (
    const matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & val,
    const matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & i ) [friend]
```

#### 6.20.5.3 matrix\_multi

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
friend class matrix_multi [friend]
```

Definition at line 147 of file matrix\_multi.h.



## 6.20.5.4 matrix\_sqrt

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
const matrix_multi<Other_Scalar_T,Other_LO,Other_HI> matrix_sqrt (
    const matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & val,
    const matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & i ) [friend]
```

## 6.20.5.5 operator &amp;

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const matrix_multi_t operator& (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

## 6.20.5.6 operator%

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const matrix_multi_t operator% (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

## 6.20.5.7 operator\*

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const matrix_multi_t operator* (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

## 6.20.5.8 operator/

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const matrix_multi_t operator/ (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

**6.20.5.9 operator<< [1/2]**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
std::ostream& operator<< (
    std::ostream & os,
    const multivector_t & val ) [friend]
```

**6.20.5.10 operator<< [2/2]**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
std::ostream& operator<< (
    std::ostream & os,
    const term_t & term ) [friend]
```

**6.20.5.11 operator>>**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
std::istream& operator>> (
    std::istream & s,
    multivector_t & val ) [friend]
```

**6.20.5.12 operator^**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const matrix_multi_t operator^ (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

**6.20.5.13 operator" |**

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
const matrix_multi_t operator| (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

6.20.5.14 `reframe`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
template<typename Other_Scalar_T , const index_t Other_LO, const index_t Other_HI>
const index_set<Other_LO,Other_HI> reframe (
    const matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & lhs,
    const matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & rhs,
    matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & lhs_reframed,
    matrix_multi< Other_Scalar_T, Other_LO, Other_HI > & rhs_reframed ) [friend]
```

6.20.5.15 `star`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
Scalar_T star (
    const matrix_multi_t & lhs,
    const matrix_multi_t & rhs ) [friend]
```

## 6.20.6 Member Data Documentation

6.20.6.1 `m_frame`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
index_set_t glucat::matrix_multi< Scalar_T, LO, HI >::m_frame [private]
```

Index set representing the frame for the subalgebra which contains the multivector.

Definition at line 275 of file `matrix_multi.h`.

Referenced by `glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi()`, `glucat::operator*()`, `glucat::operator/()`, `glucat::matrix_multi< Scalar_T, LO, HI >::operator=()`, and `glucat::reframe()`.

6.20.6.2 `m_matrix`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
matrix_t glucat::matrix_multi< Scalar_T, LO, HI >::m_matrix [private]
```

Matrix value representing the multivector within the folded frame.

Definition at line 277 of file `matrix_multi.h`.

Referenced by `glucat::framed_multi< Scalar_T, LO, HI >::framed_multi()`, `glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi()`, `glucat::operator*()`, and `glucat::matrix_multi< Scalar_T, LO, HI >::operator=()`.

The documentation for this class was generated from the following files:

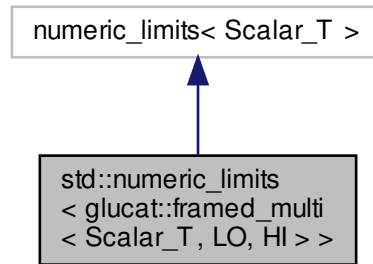
- `glucat/framed_multi.h`
- `glucat/matrix_multi.h`
- `glucat/matrix_multi_imp.h`

## 6.21 `std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > >` Struct Template Reference

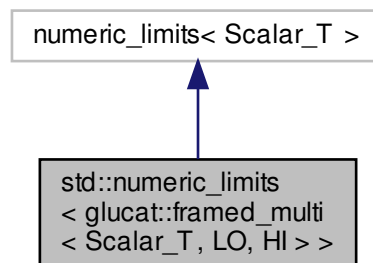
Numeric limits for `framed_multi` inherit limits for the corresponding scalar type.

```
#include <framed_multi.h>
```

Inheritance diagram for `std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > >`:



Collaboration diagram for `std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > >`:



### 6.21.1 Detailed Description

```
template<typename Scalar_T, const glucat::index_t LO, const glucat::index_t HI>
struct std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > >
```

Numeric limits for `framed_multi` inherit limits for the corresponding scalar type.

Definition at line 374 of file `framed_multi.h`.

The documentation for this struct was generated from the following file:

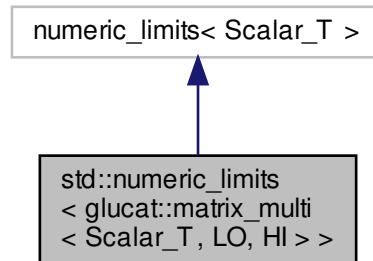
- [glucat/framed\\_multi.h](#)

## 6.22 std::numeric\_limits< glucat::matrix\_multi< Scalar\_T, LO, HI > > Struct Template Reference

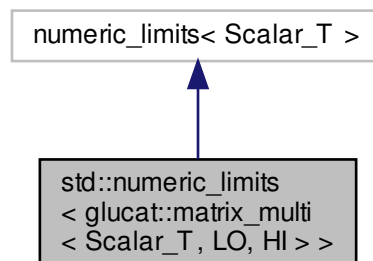
Numeric limits for matrix\_multi inherit limits for the corresponding scalar type.

```
#include <matrix_multi.h>
```

Inheritance diagram for std::numeric\_limits< glucat::matrix\_multi< Scalar\_T, LO, HI > >:



Collaboration diagram for std::numeric\_limits< glucat::matrix\_multi< Scalar\_T, LO, HI > >:



### 6.22.1 Detailed Description

```
template<typename Scalar_T, const glucat::index_t LO, const glucat::index_t HI>
struct std::numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > >
```

Numeric limits for matrix\_multi inherit limits for the corresponding scalar type.

Definition at line 293 of file matrix\_multi.h.

The documentation for this struct was generated from the following file:

- glucat/[matrix\\_multi.h](#)

## 6.23 glucat::numeric\_traits< Scalar\_T > Class Template Reference

Extra traits which extend numeric limits.

```
#include <scalar.h>
```

### Classes

- struct [demoted](#)  
*Demoted type for long double.*
- struct [promoted](#)  
*Promoted type.*

### Public Member Functions

- `template<>`  
`long double pi ()`  
*Pi for long double.*
- `template<>`  
`long double ln\_2 ()`  
*log(2) for long double*
- `template<>`  
`float to\_scalar\_t (const Other_Scalar_T &val)`  
*Extra traits which extend numeric limits.*
- `template<>`  
`double to\_scalar\_t (const Other_Scalar_T &val)`  
*Cast to double.*
- `template<>`  
`long double to\_scalar\_t (const dd_real &val)`  
*Cast to long double.*
- `template<>`  
`long double to\_scalar\_t (const qd_real &val)`  
*Cast to long double.*
- `template<>`  
`dd_real to\_scalar\_t (const long double &val)`  
*Cast to dd\_real.*
- `template<>`  
`dd_real to\_scalar\_t (const qd_real &val)`  
*Cast to dd\_real.*
- `template<>`  
`qd_real to\_scalar\_t (const long double &val)`  
*Cast to qd\_real.*
- `template<>`  
`qd_real to\_scalar\_t (const dd_real &val)`  
*Cast to qd\_real.*

## Static Public Member Functions

- static bool [isInf](#) (const Scalar\_T &val)  
*Smart isinf.*
- static bool [isNaN](#) (const Scalar\_T &val)  
*Smart isnan.*
- static bool [isNaN\\_or\\_isInf](#) (const Scalar\_T &val)  
*Smart isnan or isinf.*
- static Scalar\_T [NaN](#) ()  
*Smart NaN.*
- static int [to\\_int](#) (const Scalar\_T &val)  
*Cast to int.*
- static double [to\\_double](#) (const Scalar\_T &val)  
*Cast to double.*
- template<typename Other\_Scalar\_T >  
static Scalar\_T [to\\_scalar\\_t](#) (const Other\_Scalar\_T &val)  
*Cast to Scalar\_T.*
- static Scalar\_T [fmod](#) (const Scalar\_T &lhs, const Scalar\_T &rhs)  
*Modulo function for scalar.*
- static Scalar\_T [conj](#) (const Scalar\_T &val)  
*Complex conjugate of scalar.*
- static Scalar\_T [real](#) (const Scalar\_T &val)  
*Real part of scalar.*
- static Scalar\_T [imag](#) (const Scalar\_T &val)  
*Imaginary part of scalar.*
- static Scalar\_T [abs](#) (const Scalar\_T &val)  
*Absolute value of scalar.*
- static Scalar\_T [pi](#) ()  
*Pi.*
- static Scalar\_T [ln\\_2](#) ()  
*log(2)*
- static Scalar\_T [pow](#) (const Scalar\_T &val, int n)  
*Integer power.*
- static Scalar\_T [sqrt](#) (const Scalar\_T &val)  
*Square root of scalar.*
- static Scalar\_T [exp](#) (const Scalar\_T &val)  
*Exponential.*
- static Scalar\_T [log](#) (const Scalar\_T &val)  
*Logarithm of scalar.*
- static Scalar\_T [log2](#) (const Scalar\_T &val)  
*Log base 2.*
- static Scalar\_T [cos](#) (const Scalar\_T &val)  
*Cosine of scalar.*
- static Scalar\_T [acos](#) (const Scalar\_T &val)  
*Inverse cosine of scalar.*
- static Scalar\_T [cosh](#) (const Scalar\_T &val)  
*Hyperbolic cosine of scalar.*
- static Scalar\_T [sin](#) (const Scalar\_T &val)  
*Sine of scalar.*
- static Scalar\_T [asin](#) (const Scalar\_T &val)  
*Inverse sine of scalar.*

- static Scalar\_T [sinh](#) (const Scalar\_T &val)  
*Hyperbolic sine of scalar.*
- static Scalar\_T [tan](#) (const Scalar\_T &val)  
*Tangent of scalar.*
- static Scalar\_T [atan](#) (const Scalar\_T &val)  
*Inverse tangent of scalar.*
- static Scalar\_T [tanh](#) (const Scalar\_T &val)  
*Hyperbolic tangent of scalar.*

### Static Private Member Functions

- static bool [isInf](#) (const Scalar\_T &val, [bool\\_to\\_type](#)< false >)  
*Smart isinf specialised for Scalar\_T without infinity.*
- static bool [isInf](#) (const Scalar\_T &val, [bool\\_to\\_type](#)< true >)  
*Smart isinf specialised for Scalar\_T with infinity.*
- static bool [isNaN](#) (const Scalar\_T &val, [bool\\_to\\_type](#)< false >)  
*Smart isnan specialised for Scalar\_T without quiet NaN.*
- static bool [isNaN](#) (const Scalar\_T &val, [bool\\_to\\_type](#)< true >)  
*Smart isnan specialised for Scalar\_T with quiet NaN.*

### 6.23.1 Detailed Description

```
template<typename Scalar_T>
class glucat::numeric_traits< Scalar_T >
```

Extra traits which extend numeric limits.

Definition at line 46 of file scalar.h.

### 6.23.2 Member Function Documentation

#### 6.23.2.1 [abs\(\)](#)

```
template<typename Scalar_T >
static Scalar_T glucat::numeric\_traits< Scalar_T >::abs (
    const Scalar_T & val ) [inline], [static]
```

Absolute value of scalar.

Definition at line 181 of file scalar.h.

References [UBLAS\\_ABS](#).



#### 6.23.2.2 acos()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::acos (
    const Scalar_T & val ) [inline], [static]
```

Inverse cosine of scalar.

Definition at line 244 of file scalar.h.

References glucat::acos().

#### 6.23.2.3 asin()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::asin (
    const Scalar_T & val ) [inline], [static]
```

Inverse sine of scalar.

Definition at line 265 of file scalar.h.

References glucat::asin().

#### 6.23.2.4 atan()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::atan (
    const Scalar_T & val ) [inline], [static]
```

Inverse tangent of scalar.

Definition at line 286 of file scalar.h.

References glucat::atan().

#### 6.23.2.5 conj()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::conj (
    const Scalar_T & val ) [inline], [static]
```

Complex conjugate of scalar.

Definition at line 160 of file scalar.h.

#### 6.23.2.6 cos()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::cos (
    const Scalar_T & val ) [inline], [static]
```

Cosine of scalar.

Definition at line 237 of file scalar.h.

References glucat::cos().

#### 6.23.2.7 cosh()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::cosh (
    const Scalar_T & val ) [inline], [static]
```

Hyperbolic cosine of scalar.

Definition at line 251 of file scalar.h.

References glucat::cosh().

#### 6.23.2.8 exp()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::exp (
    const Scalar_T & val ) [inline], [static]
```

Exponential.

Definition at line 216 of file scalar.h.

References glucat::exp().

#### 6.23.2.9 fmod()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::fmod (
    const Scalar_T & lhs,
    const Scalar_T & rhs ) [inline], [static]
```

Modulo function for scalar.

Definition at line 153 of file scalar.h.

**6.23.2.10** `imag()`

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::imag (
    const Scalar_T & val ) [inline], [static]
```

Imaginary part of scalar.

Definition at line 174 of file scalar.h.

**6.23.2.11** `isInf()` [1/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isInf (
    const Scalar_T & val,
    bool_to_type< false > ) [inline], [static], [private]
```

Smart isinf specialised for Scalar\_T without infinity.

Definition at line 53 of file scalar.h.

Referenced by `glucat::numeric_traits< Scalar_T >::isInf()`, and `glucat::numeric_traits< Scalar_T >::isNaN_or_isInf()`.

**6.23.2.12** `isInf()` [2/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isInf (
    const Scalar_T & val,
    bool_to_type< true > ) [inline], [static], [private]
```

Smart isinf specialised for Scalar\_T with infinity.

Definition at line 60 of file scalar.h.

References `_GLUCAT_ISINF`.

**6.23.2.13** `isInf()` [3/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isInf (
    const Scalar_T & val ) [inline], [static]
```

Smart isinf.

Definition at line 82 of file scalar.h.

References `glucat::numeric_traits< Scalar_T >::isInf()`.

**6.23.2.14 isNaN()** [1/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isNaN (
    const Scalar_T & val,
    bool_to_type< false > ) [inline], [static], [private]
```

Smart isnan specialised for Scalar\_T without quiet NaN.

Definition at line 67 of file scalar.h.

Referenced by glucat::numeric\_traits< Scalar\_T >::isNaN(), and glucat::numeric\_traits< Scalar\_T >::isNaN\_or\_isInf().

**6.23.2.15 isNaN()** [2/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isNaN (
    const Scalar_T & val,
    bool_to_type< true > ) [inline], [static], [private]
```

Smart isnan specialised for Scalar\_T with quiet NaN.

Definition at line 74 of file scalar.h.

References \_GLUCAT\_ISNAN.

**6.23.2.16 isNaN()** [3/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isNaN (
    const Scalar_T & val ) [inline], [static]
```

Smart isnan.

Definition at line 92 of file scalar.h.

References glucat::numeric\_traits< Scalar\_T >::isNaN().

**6.23.2.17 isNaN\_or\_isInf()**

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isNaN_or_isInf (
    const Scalar_T & val ) [inline], [static]
```

Smart isnan or isinf.

Definition at line 102 of file scalar.h.

References glucat::numeric\_traits< Scalar\_T >::isInf(), and glucat::numeric\_traits< Scalar\_T >::isNaN().

**6.23.2.18** `ln_2()` [1/2]

```
template<>
long double glucat::numeric_traits< long double >::ln_2 ( ) [inline]
```

`log(2)` for long double

Definition at line 83 of file `long_double.h`.

References `glucat::l_ln2`.

**6.23.2.19** `ln_2()` [2/2]

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::ln_2 ( ) [inline], [static]
```

`log(2)`

Definition at line 195 of file `scalar.h`.

Referenced by `glucat::numeric_traits< Scalar_T >::log2()`.

**6.23.2.20** `log()`

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::log (
    const Scalar_T & val ) [inline], [static]
```

Logarithm of scalar.

Definition at line 223 of file `scalar.h`.

References `glucat::log()`.

Referenced by `glucat::numeric_traits< Scalar_T >::log2()`.

**6.23.2.21** `log2()`

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::log2 (
    const Scalar_T & val ) [inline], [static]
```

Log base 2.

Definition at line 230 of file `scalar.h`.

References `glucat::numeric_traits< Scalar_T >::ln_2()`, and `glucat::numeric_traits< Scalar_T >::log()`.

Referenced by `glucat::log2()`.

#### 6.23.2.22 NaN()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::NaN ( ) [inline], [static]
```

Smart NaN.

Definition at line 114 of file scalar.h.

References glucat::log().

Referenced by glucat::matrix::trace().

#### 6.23.2.23 pi() [1/2]

```
template<>
long double glucat::numeric_traits< long double >::pi ( ) [inline]
```

Pi for long double.

Definition at line 75 of file long\_double.h.

References glucat::l\_pi.

#### 6.23.2.24 pi() [2/2]

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::pi ( ) [inline], [static]
```

Pi.

Definition at line 188 of file scalar.h.

Referenced by glucat::matrix::classify\_eigenvalues().

#### 6.23.2.25 pow()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::pow (
    const Scalar_T & val,
    int n ) [inline], [static]
```

Integer power.

Definition at line 202 of file scalar.h.

References glucat::pow().

#### 6.23.2.26 real()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::real (
    const Scalar_T & val ) [inline], [static]
```

Real part of scalar.

Definition at line 167 of file scalar.h.

#### 6.23.2.27 sin()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::sin (
    const Scalar_T & val ) [inline], [static]
```

Sine of scalar.

Definition at line 258 of file scalar.h.

References glucat::sin().

#### 6.23.2.28 sinh()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::sinh (
    const Scalar_T & val ) [inline], [static]
```

Hyperbolic sine of scalar.

Definition at line 272 of file scalar.h.

References glucat::sinh().

#### 6.23.2.29 sqrt()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::sqrt (
    const Scalar_T & val ) [inline], [static]
```

Square root of scalar.

Definition at line 209 of file scalar.h.

References UBLAS\_SQRT.

Referenced by glucat::abs().

#### 6.23.2.30 tan()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::tan (
    const Scalar_T & val ) [inline], [static]
```

Tangent of scalar.

Definition at line 279 of file scalar.h.

References glucat::tan().

#### 6.23.2.31 tanh()

```
template<typename Scalar_T >
static Scalar_T glucat::numeric_traits< Scalar_T >::tanh (
    const Scalar_T & val ) [inline], [static]
```

Hyperbolic tangent of scalar.

Definition at line 293 of file scalar.h.

References glucat::tanh().

#### 6.23.2.32 to\_double()

```
template<typename Scalar_T >
static double glucat::numeric_traits< Scalar_T >::to_double (
    const Scalar_T & val ) [inline], [static]
```

Cast to double.

Definition at line 132 of file scalar.h.

Referenced by glucat::operator<<(), and glucat::numeric\_traits< Scalar\_T >::to\_scalar\_t().

#### 6.23.2.33 to\_int()

```
template<typename Scalar_T >
static int glucat::numeric_traits< Scalar_T >::to_int (
    const Scalar_T & val ) [inline], [static]
```

Cast to int.

Definition at line 125 of file scalar.h.



**6.23.2.34 to\_scalar\_t()** [1/9]

```
template<>
float glucat::numeric_traits< float >::to_scalar_t (
    const Other_Scalar_T & val ) [inline]
```

Extra traits which extend numeric limits.

Cast to float

Definition at line 52 of file scalar\_imp.h.

References glucat::numeric\_traits< Scalar\_T >::to\_double().

**6.23.2.35 to\_scalar\_t()** [2/9]

```
template<>
double glucat::numeric_traits< double >::to_scalar_t (
    const Other_Scalar_T & val ) [inline]
```

Cast to double.

Definition at line 61 of file scalar\_imp.h.

References glucat::numeric\_traits< Scalar\_T >::to\_double().

**6.23.2.36 to\_scalar\_t()** [3/9]

```
template<>
long double glucat::numeric_traits< long double >::to_scalar_t (
    const dd_real & val ) [inline]
```

Cast to long double.

Definition at line 71 of file scalar\_imp.h.

**6.23.2.37 to\_scalar\_t()** [4/9]

```
template<>
long double glucat::numeric_traits< long double >::to_scalar_t (
    const qd_real & val ) [inline]
```

Cast to long double.

Definition at line 80 of file scalar\_imp.h.

**6.23.2.38 to\_scalar\_t()** [ 5/9]

```
template<>
dd_real glucat::numeric_traits< dd_real >::to_scalar_t (
    const long double & val ) [inline]
```

Cast to dd\_real.

Definition at line 89 of file scalar\_imp.h.

**6.23.2.39 to\_scalar\_t()** [ 6/9]

```
template<>
dd_real glucat::numeric_traits< dd_real >::to_scalar_t (
    const qd_real & val ) [inline]
```

Cast to dd\_real.

Definition at line 98 of file scalar\_imp.h.

**6.23.2.40 to\_scalar\_t()** [ 7/9]

```
template<>
qd_real glucat::numeric_traits< qd_real >::to_scalar_t (
    const long double & val ) [inline]
```

Cast to qd\_real.

Definition at line 107 of file scalar\_imp.h.

**6.23.2.41 to\_scalar\_t()** [ 8/9]

```
template<>
qd_real glucat::numeric_traits< qd_real >::to_scalar_t (
    const dd_real & val ) [inline]
```

Cast to qd\_real.

Definition at line 116 of file scalar\_imp.h.

6.23.2.42 `to_scalar_t()` [ 9/9 ]

```
template<typename Scalar_T >
template<typename Other_Scalar_T >
static Scalar_T glucat::numeric\_traits< Scalar\_T >::to\_scalar\_t (
    const Other_Scalar_T & val ) [inline], [static]
```

Cast to `Scalar_T`.

Definition at line 140 of file `scalar.h`.

Referenced by `glucat::matrix::nork_range()`, `glucat::to_demote()`, and `glucat::to_promote()`.

The documentation for this class was generated from the following file:

- [glucat/scalar.h](#)

6.24 `glucat::numeric_traits< Scalar_T >::promoted` Struct Reference

Promoted type.

```
#include <scalar.h>
```

## Public Types

- typedef double [type](#)

## 6.24.1 Detailed Description

```
template<typename Scalar_T>
struct glucat::numeric_traits< Scalar_T >::promoted
```

Promoted type.

Definition at line 144 of file `scalar.h`.

## 6.24.2 Member Typedef Documentation

6.24.2.1 `type`

```
template<typename Scalar_T >
typedef double glucat::numeric\_traits< Scalar\_T >::promoted::type
```

Definition at line 144 of file `scalar.h`.

The documentation for this struct was generated from the following file:

- [glucat/scalar.h](#)

## 6.25 glucat::random\_generator< Scalar\_T > Class Template Reference

Random number generator with single instance per Scalar\_T.

```
#include <random.h>
```

### Public Member Functions

- Scalar\_T [uniform](#) ()
- Scalar\_T [normal](#) ()

### Static Public Member Functions

- static [random\\_generator](#) & [generator](#) ()  
*Single instance of Random number generator.*

### Private Member Functions

- [random\\_generator](#) (const [random\\_generator](#) &)
- [random\\_generator](#) & [operator=](#) (const [random\\_generator](#) &)
- [random\\_generator](#) ()
- [~random\\_generator](#) ()

### Private Attributes

- std::mt19937 [uint\\_gen](#)
- std::uniform\_real\_distribution< double > [uniform\\_dist](#)
- std::normal\_distribution< double > [normal\\_dist](#)

### Static Private Attributes

- static const unsigned long [seed](#) = 19590921UL

### Friends

- class [friend\\_for\\_private\\_destructor](#)

#### 6.25.1 Detailed Description

```
template<typename Scalar_T>  
class glucat::random_generator< Scalar_T >
```

Random number generator with single instance per Scalar\_T.

Definition at line 47 of file random.h.

## 6.25.2 Constructor & Destructor Documentation

### 6.25.2.1 random\_generator() [1/2]

```
template<typename Scalar_T >  
glucat::random_generator< Scalar_T >::random_generator (   
    const random_generator< Scalar_T > & ) [private]
```

### 6.25.2.2 random\_generator() [2/2]

```
template<typename Scalar_T >  
glucat::random_generator< Scalar_T >::random_generator ( ) [inline], [private]
```

Definition at line 83 of file random.h.

References glucat::random\_generator< Scalar\_T >::seed.

### 6.25.2.3 ~random\_generator()

```
template<typename Scalar_T >  
glucat::random_generator< Scalar_T >::~~random_generator ( ) [inline], [private]
```

Definition at line 87 of file random.h.

## 6.25.3 Member Function Documentation

### 6.25.3.1 generator()

```
template<typename Scalar_T >  
static random_generator& glucat::random_generator< Scalar_T >::generator ( ) [inline], [static]
```

Single instance of Random number generator.

Definition at line 51 of file random.h.

#### 6.25.3.2 normal()

```
template<typename Scalar_T >
Scalar_T glucat::random_generator< Scalar_T >::normal ( ) [inline]
```

Definition at line 93 of file random.h.

References glucat::random\_generator< Scalar\_T >::normal\_dist.

#### 6.25.3.3 operator=()

```
template<typename Scalar_T >
random_generator& glucat::random_generator< Scalar_T >::operator= (
    const random_generator< Scalar_T > & ) [private]
```

#### 6.25.3.4 uniform()

```
template<typename Scalar_T >
Scalar_T glucat::random_generator< Scalar_T >::uniform ( ) [inline]
```

Definition at line 91 of file random.h.

References glucat::random\_generator< Scalar\_T >::uniform\_dist.

### 6.25.4 Friends And Related Function Documentation

#### 6.25.4.1 friend\_for\_private\_destructor

```
template<typename Scalar_T >
friend class friend_for_private_destructor [friend]
```

Friend declaration to avoid compiler warning: "... only defines a private destructor and has no friends" Ref: Carlos O'Ryan, ACE <http://doc.ece.uci.edu>

Definition at line 56 of file random.h.

### 6.25.5 Member Data Documentation

#### 6.25.5.1 normal\_dist

```
template<typename Scalar_T >
std::normal_distribution<double> glucat::random_generator< Scalar_T >::normal_dist [private]
```

Definition at line 81 of file random.h.

Referenced by glucat::random\_generator< Scalar\_T >::normal().

#### 6.25.5.2 seed

```
template<typename Scalar_T >
const unsigned long glucat::random_generator< Scalar_T >::seed = 19590921UL [static], [private]
```

Definition at line 59 of file random.h.

Referenced by glucat::random\_generator< Scalar\_T >::random\_generator().

#### 6.25.5.3 uint\_gen

```
template<typename Scalar_T >
std::mt19937 glucat::random_generator< Scalar_T >::uint_gen [private]
```

Definition at line 79 of file random.h.

#### 6.25.5.4 uniform\_dist

```
template<typename Scalar_T >
std::uniform_real_distribution<double> glucat::random_generator< Scalar_T >::uniform_dist
[private]
```

Definition at line 80 of file random.h.

Referenced by glucat::random\_generator< Scalar\_T >::uniform().

The documentation for this class was generated from the following file:

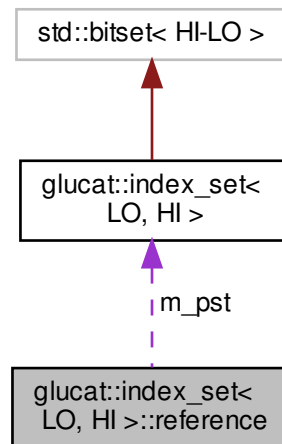
- glucat/[random.h](#)

## 6.26 glucat::index\_set< LO, HI >::reference Class Reference

Index set member reference.

```
#include <index_set.h>
```

Collaboration diagram for glucat::index\_set< LO, HI >::reference:



### Public Member Functions

- [reference](#) ([index\\_set\\_t](#) &ist, [index\\_t](#) idx)  
*index\_set reference*
- [~reference](#) ()
- [reference](#) & [operator=](#) (const bool x)  
*for b[i] = x;*
- [reference](#) & [operator=](#) (const [reference](#) &j)  
*for b[i] = b[j];*
- bool [operator~](#) () const  
*Flips a bit.*
- [operator bool](#) () const  
*for x = b[i];*
- [reference](#) & [flip](#) ()  
*for b[i].flip();*

### Private Member Functions

- [reference](#) ()  
*Private default constructor is left undefined.*



## Private Attributes

- [index\\_set\\_t](#) \* m\_pst
- [index\\_t](#) m\_idx

## Friends

- class [index\\_set](#)

### 6.26.1 Detailed Description

```
template<const index_t LO, const index_t HI>
class glucat::index_set< LO, HI >::reference
```

Index set member reference.

Definition at line 177 of file [index\\_set.h](#).

### 6.26.2 Constructor & Destructor Documentation

#### 6.26.2.1 [reference\(\)](#) [1/2]

```
template<const index_t LO, const index_t HI>
glucat::index\_set< LO, HI >::reference::reference ( ) [private]
```

Private default constructor is left undefined.

#### 6.26.2.2 [reference\(\)](#) [2/2]

```
template<const index_t LO, const index_t HI>
glucat::index\_set< LO, HI >::reference::reference (
    index\_set\_t & ist,
    index\_t idx ) [inline]
```

[index\\_set](#) reference

Definition at line 983 of file [index\\_set\\_imp.h](#).

### 6.26.2.3 ~reference()

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::reference::~reference ( ) [inline]
```

Definition at line 184 of file index\_set.h.

## 6.26.3 Member Function Documentation

### 6.26.3.1 flip()

```
template<const index_t LO, const index_t HI>
index_set< LO, HI >::reference & glucat::index_set< LO, HI >::reference::flip ( ) [inline]
```

for b[i].flip();

Definition at line 1036 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::reference::flip().

Referenced by glucat::index\_set< LO, HI >::reference::flip().

### 6.26.3.2 operator bool()

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::reference::operator bool ( ) const [inline]
```

for x = b[i];

Definition at line 1028 of file index\_set\_imp.h.

### 6.26.3.3 operator=() [1/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI >::reference & glucat::index_set< LO, HI >::reference::operator= (
    const bool x ) [inline]
```

for b[i] = x;

Definition at line 993 of file index\_set\_imp.h.

**6.26.3.4 operator=()** [2/2]

```
template<const index_t LO, const index_t HI>
index_set< LO, HI >::reference & glucat::index_set< LO, HI >::reference::operator= (
    const reference & j ) [inline]
```

for b[i] = b[j];

Definition at line 1007 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::reference::m\_idx, and glucat::index\_set< LO, HI >::reference::m\_pst.

**6.26.3.5 operator~()**

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::reference::operator~ ( ) const [inline]
```

Flips a bit.

flips the bit

Definition at line 1021 of file index\_set\_imp.h.

**6.26.4 Friends And Related Function Documentation****6.26.4.1 index\_set**

```
template<const index_t LO, const index_t HI>
friend class index_set [friend]
```

Definition at line 178 of file index\_set.h.

**6.26.5 Member Data Documentation****6.26.5.1 m\_idx**

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::reference::m_idx [private]
```

Definition at line 198 of file index\_set.h.

Referenced by glucat::index\_set< LO, HI >::reference::operator=().

### 6.26.5.2 m\_pst

```
template<const index_t LO, const index_t HI>
index_set_t* glucat::index_set< LO, HI >::reference::m_pst [private]
```

Definition at line 197 of file index\_set.h.

Referenced by glucat::index\_set< LO, HI >::reference::operator=().

The documentation for this class was generated from the following files:

- glucat/index\_set.h
- glucat/index\_set\_imp.h

## 6.27 glucat::sorted\_range< Map\_T, Sorted\_Map\_T > Class Template Reference

Sorted range for use with output.

```
#include <framed_multi_imp.h>
```

### Public Types

- typedef Map\_T [map\\_t](#)
- typedef Sorted\_Map\_T [sorted\\_map\\_t](#)
- typedef Sorted\_Map\_T::const\_iterator [sorted\\_iterator](#)

### Public Member Functions

- [sorted\\_range](#) (Sorted\_Map\_T &sorted\_val, const Map\_T &val)

### Public Attributes

- [sorted\\_iterator](#) [sorted\\_begin](#)
- [sorted\\_iterator](#) [sorted\\_end](#)

### 6.27.1 Detailed Description

```
template<typename Map_T, typename Sorted_Map_T>
class glucat::sorted_range< Map_T, Sorted_Map_T >
```

Sorted range for use with output.

Definition at line 1326 of file framed\_multi\_imp.h.

### 6.27.2 Member Typedef Documentation

### 6.27.2.1 map\_t

```
template<typename Map_T, typename Sorted_Map_T>
typedef Map_T glucat::sorted_range< Map_T, Sorted_Map_T >::map_t
```

Definition at line 1329 of file framed\_multi\_imp.h.

### 6.27.2.2 sorted\_iterator

```
template<typename Map_T, typename Sorted_Map_T>
typedef Sorted_Map_T::const_iterator glucat::sorted_range< Map_T, Sorted_Map_T >::sorted_iterator
```

Definition at line 1331 of file framed\_multi\_imp.h.

### 6.27.2.3 sorted\_map\_t

```
template<typename Map_T, typename Sorted_Map_T>
typedef Sorted_Map_T glucat::sorted_range< Map_T, Sorted_Map_T >::sorted_map_t
```

Definition at line 1330 of file framed\_multi\_imp.h.

## 6.27.3 Constructor & Destructor Documentation

### 6.27.3.1 sorted\_range()

```
template<typename Map_T, typename Sorted_Map_T>
glucat::sorted_range< Map_T, Sorted_Map_T >::sorted_range (
    Sorted_Map_T & sorted_val,
    const Map_T & val ) [inline]
```

Definition at line 1333 of file framed\_multi\_imp.h.

References glucat::sorted\_range< Map\_T, Sorted\_Map\_T >::sorted\_begin, and glucat::sorted\_range< Map\_T, Sorted\_Map\_T >::sorted\_end.

## 6.27.4 Member Data Documentation

#### 6.27.4.1 sorted\_begin

```
template<typename Map_T, typename Sorted_Map_T>
sorted_iterator glucat::sorted_range< Map_T, Sorted_Map_T >::sorted_begin
```

Definition at line 1343 of file framed\_multi\_imp.h.

Referenced by glucat::operator<<(), and glucat::sorted\_range< Map\_T, Sorted\_Map\_T >::sorted\_range().

#### 6.27.4.2 sorted\_end

```
template<typename Map_T, typename Sorted_Map_T>
sorted_iterator glucat::sorted_range< Map_T, Sorted_Map_T >::sorted_end
```

Definition at line 1344 of file framed\_multi\_imp.h.

Referenced by glucat::operator<<(), and glucat::sorted\_range< Map\_T, Sorted\_Map\_T >::sorted\_range().

The documentation for this class was generated from the following file:

- glucat/framed\_multi\_imp.h

## 6.28 glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T > Class Template Reference

```
#include <framed_multi_imp.h>
```

### Public Types

- typedef Sorted\_Map\_T [map\\_t](#)
- typedef Sorted\_Map\_T [sorted\\_map\\_t](#)
- typedef Sorted\_Map\_T::const\_iterator [sorted\\_iterator](#)

### Public Member Functions

- [sorted\\_range](#) (Sorted\_Map\_T &sorted\_val, const Sorted\_Map\_T &val)

### Public Attributes

- [sorted\\_iterator](#) [sorted\\_begin](#)
- [sorted\\_iterator](#) [sorted\\_end](#)

#### 6.28.1 Detailed Description

```
template<typename Sorted_Map_T>
class glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >
```

Definition at line 1348 of file framed\_multi\_imp.h.

## 6.28.2 Member Typedef Documentation

### 6.28.2.1 map\_t

```
template<typename Sorted_Map_T >
typedef Sorted_Map_T glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >::map_t
```

Definition at line 1351 of file framed\_multi\_imp.h.

### 6.28.2.2 sorted\_iterator

```
template<typename Sorted_Map_T >
typedef Sorted_Map_T::const_iterator glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >↔
::sorted_iterator
```

Definition at line 1353 of file framed\_multi\_imp.h.

### 6.28.2.3 sorted\_map\_t

```
template<typename Sorted_Map_T >
typedef Sorted_Map_T glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >::sorted_map_t
```

Definition at line 1352 of file framed\_multi\_imp.h.

## 6.28.3 Constructor & Destructor Documentation

### 6.28.3.1 sorted\_range()

```
template<typename Sorted_Map_T >
glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >::sorted_range (
    Sorted_Map_T & sorted_val,
    const Sorted_Map_T & val ) [inline]
```

Definition at line 1355 of file framed\_multi\_imp.h.

## 6.28.4 Member Data Documentation

#### 6.28.4.1 sorted\_begin

```
template<typename Sorted_Map_T >
sorted_iterator glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >::sorted_begin
```

Definition at line 1359 of file framed\_multi\_imp.h.

#### 6.28.4.2 sorted\_end

```
template<typename Sorted_Map_T >
sorted_iterator glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >::sorted_end
```

Definition at line 1360 of file framed\_multi\_imp.h.

The documentation for this class was generated from the following file:

- [glucat/framed\\_multi\\_imp.h](#)

### 6.29 glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision > Struct Template Reference

Tuning policy.

```
#include <global.h>
```

#### Public Types

- enum { [mult\\_matrix\\_threshold](#) = Mult\_Matrix\_Threshold }  
*Minimum index count needed to invoke matrix multiplication algorithm.*
- enum { [div\\_max\\_steps](#) = Div\_Max\_Steps }  
*Maximum steps of iterative refinement in division algorithm.*
- enum { [sqrt\\_max\\_steps](#) = Sqrt\_Max\_Steps }  
*Maximum number of steps in square root iteration.*
- enum { [log\\_max\\_outer\\_steps](#) = Log\_Max\_Outer\_Steps }  
*Maximum number of incomplete square roots in cascade log algorithm.*
- enum { [log\\_max\\_inner\\_steps](#) = Log\_Max\_Inner\_Steps }  
*Maximum number of steps in incomplete square root within cascade log algorithm.*
- enum { [basis\\_max\\_count](#) = Basis\_Max\_Count }  
*Maximum index count of folded frames in basis cache.*
- enum { [fast\\_size\\_threshold](#) = Fast\_Size\_Threshold }  
*Minimum map size needed to invoke generalized FFT.*
- enum { [inv\\_fast\\_dim\\_threshold](#) = Inv\_Fast\_Dim\_Threshold }  
*Minimum matrix dimension needed to invoke inverse generalized FFT.*
- enum { [products\\_size\\_threshold](#) = Products\_Size\_Threshold }  
*Minimum size needed for to invoke faster products algorithms.*



- static const [precision\\_t function\\_precision](#) = Function\_Precision  
*Precision used for exp, log and sqrt functions.*

### 6.29.1 Detailed Description

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int Div_Max_Steps = DEFAULT_↵
Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps, unsigned int Log_Max_Outer_Steps = DEFAULT_↵
Log_Max_Outer_Steps, unsigned int Log_Max_Inner_Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count =
DEFAULT_Basis_Max_Count, unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
_Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_Products_Size_Threshold,
precision_t Function_Precision = DEFAULT_Function_Precision>
struct glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps,
Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_Threshold, Products_Size_Threshold, Function_Precision >
```

Tuning policy.

Definition at line 151 of file global.h.

## 6.29.2 Member Enumeration Documentation

### 6.29.2.1 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum
```

Minimum matrix dimension needed to invoke inverse generalized FFT.

Enumerator

inv_fast_dim_threshold	
------------------------	--

Definition at line 174 of file global.h.

### 6.29.2.2 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
```

```

unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum

```

Minimum size needed for to invoke faster products algorithms.

#### Enumerator

products_size_threshold	
-------------------------	--

Definition at line 177 of file global.h.

#### 6.29.2.3 anonymous enum

```

template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum

```

Minimum index count needed to invoke matrix multiplication algorithm.

#### Enumerator

mult_matrix_threshold	
-----------------------	--

Definition at line 155 of file global.h.

#### 6.29.2.4 anonymous enum

```

template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum

```

Maximum steps of iterative refinement in division algorithm.

Enumerator

div_max_steps	
---------------	--

Definition at line 158 of file global.h.

6.29.2.5 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps, unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count, unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum
```

Maximum number of steps in square root iteration.

Enumerator

sqrt_max_steps	
----------------	--

Definition at line 161 of file global.h.

6.29.2.6 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps, unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count, unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum
```

Maximum number of incomplete square roots in cascade log algorithm.

Enumerator

log_max_outer_steps	
---------------------	--

Definition at line 164 of file global.h.

### 6.29.2.7 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum
```

Maximum number of steps in incomplete square root within cascade log algorithm.

#### Enumerator

log_max_inner_steps	
---------------------	--

Definition at line 166 of file global.h.

### 6.29.2.8 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum
```

Maximum index count of folded frames in basis cache.

#### Enumerator

basis_max_count	
-----------------	--

Definition at line 169 of file global.h.

### 6.29.2.9 anonymous enum

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
anonymous enum
```

Minimum map size needed to invoke generalized FFT.

## Enumerator

fast_size_threshold	
---------------------	--

Definition at line 172 of file global.h.

### 6.29.3 Member Data Documentation

#### 6.29.3.1 function\_precision

```
template<unsigned int Mult_Matrix_Threshold = DEFAULT_Mult_Matrix_Threshold, unsigned int
Div_Max_Steps = DEFAULT_Div_Max_Steps, unsigned int Sqrt_Max_Steps = DEFAULT_Sqrt_Max_Steps,
unsigned int Log_Max_Outer_Steps = DEFAULT_Log_Max_Outer_Steps, unsigned int Log_Max_Inner_↵
Steps = DEFAULT_Log_Max_Inner_Steps, unsigned int Basis_Max_Count = DEFAULT_Basis_Max_Count,
unsigned int Fast_Size_Threshold = DEFAULT_Fast_Size_Threshold, unsigned int Inv_Fast_Dim_↵
Threshold = DEFAULT_Inv_Fast_Dim_Threshold, unsigned int Products_Size_Threshold = DEFAULT_↵
Products_Size_Threshold, precision_t Function_Precision = DEFAULT_Function_Precision>
const precision\_t glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_↵
_Max_Outer_Steps, Log_Max_Inner_Steps, Basis_Max_Count, Fast_Size_Threshold, Inv_Fast_Dim_↵
_Threshold, Products_Size_Threshold, Function_Precision >::function_precision = Function_↵
Precision [static]
```

Precision used for exp, log and sqrt functions.

Definition at line 180 of file global.h.

Referenced by `glucat::exp()`, `glucat::log()`, and `glucat::sqrt()`.

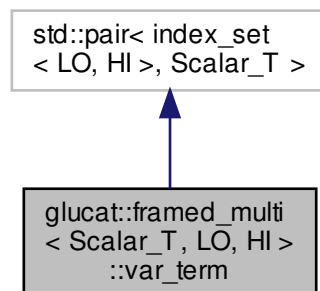
The documentation for this struct was generated from the following file:

- [glucat/global.h](#)

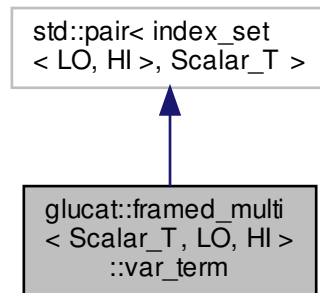
## 6.30 glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term Class Reference

Variable term.

Inheritance diagram for `glucat::framed_multi< Scalar_T, LO, HI >::var_term`:



Collaboration diagram for `glucat::framed_multi< Scalar_T, LO, HI >::var_term`:



## Public Types

- typedef `std::pair< index\_set< LO, HI >, Scalar\_T >` [var\\_pair\\_t](#)

## Public Member Functions

- [~var\\_term](#) ()  
*Destructor.*
- [var\\_term](#) ()  
*Default constructor.*
- [var\\_term](#) (const [index\\_set\\_t](#) ist, const [Scalar\\_T](#) &crd=[Scalar\\_T](#)(1))  
*Construct a variable term from an index set and a scalar coordinate.*
- [var\\_term\\_t](#) & [operator\\*=](#) (const [term\\_t](#) &rhs)  
*Product of variable term and term.*

## Static Public Member Functions

- static const `std::string` [classname](#) ()  
*Class name used in messages.*

### 6.30.1 Detailed Description

```

template<typename Scalar\_T = double, const index\_t LO = DEFAULT\_LO, const index\_t HI = DEFAULT\_HI>
class glucat::framed_multi< Scalar\_T, LO, HI >::var_term

```

Variable term.

Definition at line 308 of file `framed_multi.h`.

## 6.30.2 Member Typedef Documentation

### 6.30.2.1 var\_pair\_t

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
typedef std::pair<index_set<LO,HI>, Scalar_T> glucat::framed_multi< Scalar_T, LO, HI >::var_term::var_pair_t
```

Definition at line 312 of file framed\_multi.h.

## 6.30.3 Constructor & Destructor Documentation

### 6.30.3.1 ~var\_term()

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::var_term::~~var_term ( ) [inline]
```

Destructor.

Definition at line 318 of file framed\_multi.h.

### 6.30.3.2 var\_term() [1/2]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::var_term::var_term ( ) [inline]
```

Default constructor.

Definition at line 320 of file framed\_multi.h.

### 6.30.3.3 var\_term() [2/2]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::var_term::var_term (
    const index_set_t ist,
    const Scalar_T & crd = Scalar_T(1) ) [inline]
```

Construct a variable term from an index set and a scalar coordinate.

Definition at line 324 of file framed\_multi.h.

## 6.30.4 Member Function Documentation

### 6.30.4.1 `classname()`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
static const std::string glucat::framed\_multi< Scalar_T, LO, HI >::var_term::classname ( )
[inline], [static]
```

Class name used in messages.

Definition at line 315 of file `framed_multi.h`.

### 6.30.4.2 `operator*=( )`

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
LT_HI>
var\_term\_t& glucat::framed\_multi< Scalar_T, LO, HI >::var_term::operator*= (
    const term\_t & rhs ) [inline]
```

Product of variable term and term.

Definition at line 328 of file `framed_multi.h`.

The documentation for this class was generated from the following file:

- [glucat/framed\\_multi.h](#)



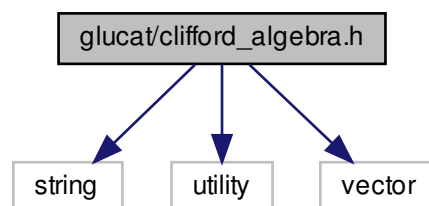
## Chapter 7

# File Documentation

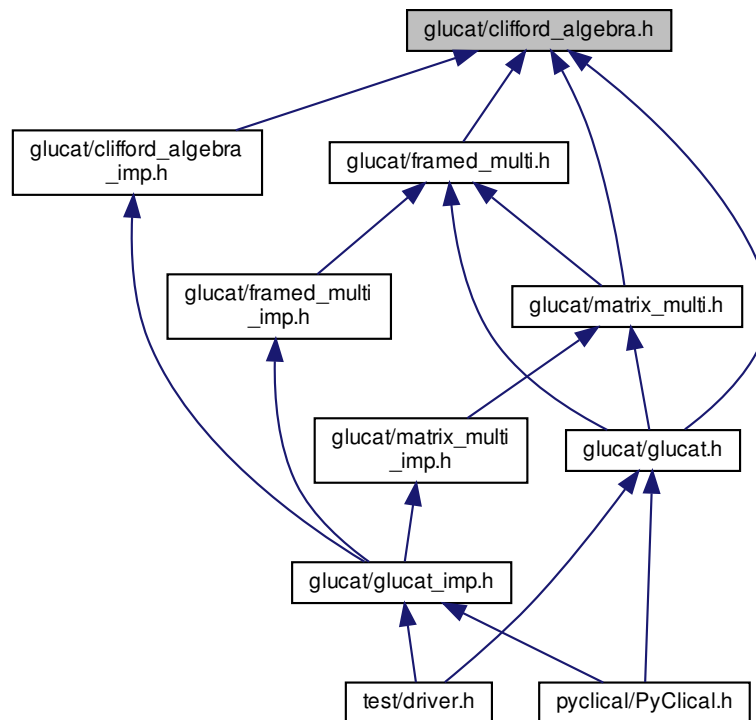
### 7.1 glucat/clifford\_algebra.h File Reference

```
#include <string>
#include <utility>
#include <vector>
```

Include dependency graph for clifford\_algebra.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::clifford\\_algebra< Scalar\\_T, Index\\_Set\\_T, Multivector\\_T >](#)  
*clifford\_algebra<> declares the operations of a Clifford algebra*

## Namespaces

- [glucat](#)

## Macros

- [#define \\_GLUCAT\\_CLIFFORD\\_ALGEBRA\\_OPERATIONS](#)

## Functions

- [template<template< typename, const index\\_t, const index\\_t > class Multivector, template< typename, const index\\_t, const index\\_t > class RHS, typename Scalar\\_T, const index\\_t LO, const index\\_t HI>](#)  
[bool glucat::operator!=](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const RHS< Scalar\_T, LO, HI > &rhs)  
*Test for inequality of multivectors.*

- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 bool [glucat::operator!=](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const Scalar\_T &scr)  
*Test for inequality of multivector and scalar.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 bool [glucat::operator!=](#) (const Scalar\_T &scr, const Multivector< Scalar\_T, LO, HI > &rhs)  
*Test for inequality of scalar and multivector.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator+](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const Scalar\_T &scr)  
*Geometric sum of multivector and scalar.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator+](#) (const Scalar\_T &scr, const Multivector< Scalar\_T, LO, HI > &rhs)  
*Geometric sum of scalar and multivector.*
- template<template< typename, const index\_t, const index\_t > class Multivector, template< typename, const index\_t, const index\_t > class RHS, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator+](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const RHS< Scalar\_T, LO, HI > &rhs)  
*Geometric sum.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator-](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const Scalar\_T &scr)  
*Geometric difference of multivector and scalar.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator-](#) (const Scalar\_T &scr, const Multivector< Scalar\_T, LO, HI > &rhs)  
*Geometric difference of scalar and multivector.*
- template<template< typename, const index\_t, const index\_t > class Multivector, template< typename, const index\_t, const index\_t > class RHS, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator-](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const RHS< Scalar\_T, LO, HI > &rhs)  
*Geometric difference.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator\\*](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const Scalar\_T &scr)  
*Product of multivector and scalar.*
- template<template< typename, const index\_t, const index\_t > class Multivector, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator\\*](#) (const Scalar\_T &scr, const Multivector< Scalar\_T, LO, HI > &rhs)  
*Product of scalar and multivector.*
- template<template< typename, const index\_t, const index\_t > class Multivector, template< typename, const index\_t, const index\_t > class RHS, typename Scalar\_T , const index\_t LO, const index\_t HI>  
 const Multivector< Scalar\_T, LO, HI > [glucat::operator\\*](#) (const Multivector< Scalar\_T, LO, HI > &lhs, const RHS< Scalar\_T, LO, HI > &rhs)  
*Geometric product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator^ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Outer product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator & (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Inner product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator% (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Left contraction.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::star (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Hestenes scalar product.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`

*Quotient of multivector and scalar.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`

*Quotient of scalar and multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Geometric quotient.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator| (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Transformation via twisted adjoint action.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::inv (const Multivector< Scalar_T, LO, HI > &val)`

*Geometric multiplicative inverse.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)`

*Integer power of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::pow (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Multivector power of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::outer\_pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)`

*Outer product power of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`Scalar_T glucat::scalar (const Multivector< Scalar_T, LO, HI > &val)`

*Scalar part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`Scalar_T glucat::real (const Multivector< Scalar_T, LO, HI > &val)`

*Real part: synonym for scalar part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`Scalar_T glucat::imag (const Multivector< Scalar_T, LO, HI > &val)`

*Imaginary part: deprecated (always 0)*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const Multivector< Scalar_T, LO, HI > glucat::pure (const Multivector< Scalar_T, LO, HI > &val)`

*Pure part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const Multivector< Scalar_T, LO, HI > glucat::even (const Multivector< Scalar_T, LO, HI > &val)`

*Even part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const Multivector< Scalar_T, LO, HI > glucat::odd (const Multivector< Scalar_T, LO, HI > &val)`

*Odd part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const std::vector< Scalar_T > glucat::vector_part (const Multivector< Scalar_T, LO, HI > &val)`

*Vector part of multivector, as a vector\_t with respect to frame()*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const Multivector< Scalar_T, LO, HI > glucat::involute (const Multivector< Scalar_T, LO, HI > &val)`

*Main involution, each {i} is replaced by -{i} in each term, eg. {1}\*{2} -> (-{2})\*(-{1})*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const Multivector< Scalar_T, LO, HI > glucat::reverse (const Multivector< Scalar_T, LO, HI > &val)`

*Reversion, eg. {1}\*{2} -> {2}\*{1}.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`const Multivector< Scalar_T, LO, HI > glucat::conj (const Multivector< Scalar_T, LO, HI > &val)`

*Conjugation, rev o invo == invo o rev.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`Scalar_T glucat::quad (const Multivector< Scalar_T, LO, HI > &val)`

*Scalar\_T quadratic form == (rev(x)\*x)(0)*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`Scalar_T glucat::norm (const Multivector< Scalar_T, LO, HI > &val)`

*Scalar\_T norm == sum of norm of coordinates.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`

`Scalar_T glucat::abs (const Multivector< Scalar_T, LO, HI > &val)`

*Absolute value == sqrt(norm)*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 Scalar_T glucat::max\_abs (const Multivector< Scalar_T, LO, HI > &val)  
Maximum of absolute values of components of multivector: multivector infinity norm.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::complexifier (const Multivector< Scalar_T, LO, HI > &val)  
Square root of -1 which commutes with all members of the frame of the given multivector.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::elliptic (const Multivector< Scalar_T, LO, HI > &val)`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::sqrt (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)  
Square root of multivector with specified complexifier.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::sqrt (const Multivector< Scalar_T, LO, HI > &val)  
Square root of multivector.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::clifford\_exp (const Multivector< Scalar_T, LO, HI > &val)  
Exponential of multivector.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::log (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)  
Natural logarithm of multivector with specified complexifier.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::log (const Multivector< Scalar_T, LO, HI > &val)  
Natural logarithm of multivector.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::cos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)  
Cosine of multivector with specified complexifier.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::cos (const Multivector< Scalar_T, LO, HI > &val)  
Cosine of multivector.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::acos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)  
Inverse cosine of multivector with specified complexifier.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::acos (const Multivector< Scalar_T, LO, HI > &val)  
Inverse cosine of multivector.`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>  
 const Multivector< Scalar_T, LO, HI > glucat::cosh (const Multivector< Scalar_T, LO, HI > &val)  
Hyperbolic cosine of multivector.`

- Generated by Doxygen

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::atan (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse tangent of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::tanh (const Multivector< Scalar_T, LO, HI > &val)`  
*Hyperbolic tangent of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::atanh (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Inverse hyperbolic tangent of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::atanh (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse hyperbolic tangent of multivector.*

## 7.1.1 Macro Definition Documentation

### 7.1.1.1 \_GLUCAT\_CLIFFORD\_ALGEBRA\_OPERATIONS

```
#define _GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS
```

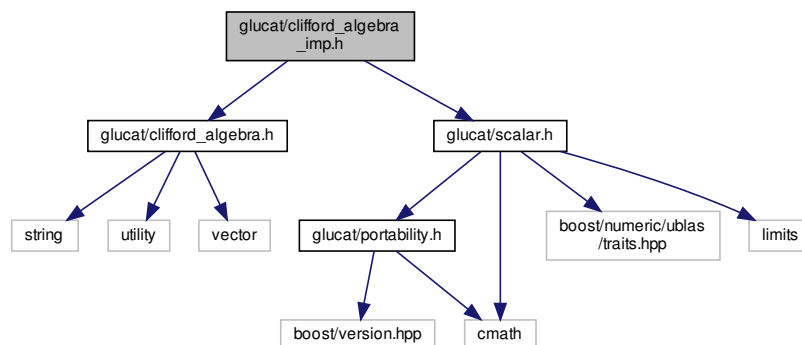
Definition at line 134 of file `clifford_algebra.h`.

## 7.2 `glucat/clifford_algebra_imp.h` File Reference

```
#include <glucat/clifford_algebra.h>
```

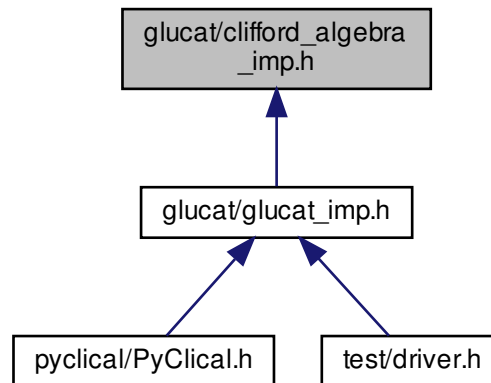
```
#include <glucat/scalar.h>
```

Include dependency graph for `clifford_algebra_imp.h`:





This graph shows which files directly or indirectly include this file:



## Namespaces

- [glucat](#)

## Functions

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T, const index_t LO, const index_t HI>`  
`bool glucat::operator!= (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`  
*Test for inequality of multivectors.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>`  
`bool glucat::operator!= (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`  
*Test for inequality of multivector and scalar.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>`  
`bool glucat::operator!= (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Test for inequality of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator+ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`  
*Geometric sum of multivector and scalar.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T, const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator+ (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`  
*Geometric sum of scalar and multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T, const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator+ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Geometric sum.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator- (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`

*Geometric difference of multivector and scalar.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator- (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`

*Geometric difference of scalar and multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator- (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Geometric difference.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator\* (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`

*Product of multivector and scalar.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator\* (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)`

*Product of scalar and multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator\* (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Geometric product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator^ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Outer product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator & (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Inner product.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator% (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Left contraction.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::star (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)`

*Hestenes scalar product.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)`

*Quotient of multivector and scalar.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Scalar_T &scr, const Multivector< Scalar_T,`  
`LO, HI > &rhs)`

*Quotient of scalar and multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t >`  
`class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const`  
`RHS< Scalar_T, LO, HI > &rhs)`

*Geometric quotient.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t >`  
`class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::operator| (const Multivector< Scalar_T, LO, HI > &lhs, const`  
`RHS< Scalar_T, LO, HI > &rhs)`

*Transformation via twisted adjoint action.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::inv (const Multivector< Scalar_T, LO, HI > &val)`

*Geometric multiplicative inverse.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)`

*Integer power of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t >`  
`class RHS, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::pow (const Multivector< Scalar_T, LO, HI > &lhs, const RHS<`  
`Scalar_T, LO, HI > &rhs)`

*Multivector power of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::outer_pow (const Multivector< Scalar_T, LO, HI > &lhs, int`  
`rhs)`

*Outer product power of multivector.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`Scalar_T glucat::scalar (const Multivector< Scalar_T, LO, HI > &val)`

*Scalar part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`Scalar_T glucat::real (const Multivector< Scalar_T, LO, HI > &val)`

*Real part: synonym for scalar part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`Scalar_T glucat::imag (const Multivector< Scalar_T, LO, HI > &val)`

*Imaginary part: deprecated (always 0)*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::pure (const Multivector< Scalar_T, LO, HI > &val)`

*Pure part.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t`  
`HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::even (const Multivector< Scalar_T, LO, HI > &val)`

*Even part.*

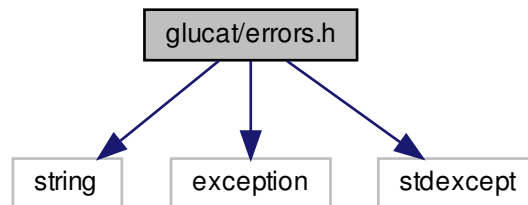
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::odd (const Multivector< Scalar_T, LO, HI > &val)`  
*Odd part.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const std::vector< Scalar_T > glucat::vector\_part (const Multivector< Scalar_T, LO, HI > &val)`  
*Vector part of multivector, as a vector\_t with respect to frame()*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::involute (const Multivector< Scalar_T, LO, HI > &val)`  
*Main involution, each {i} is replaced by -{i} in each term, eg. {1}\*{2} -> (-{2})\*(-{1})*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::reverse (const Multivector< Scalar_T, LO, HI > &val)`  
*Reversion, eg. {1}\*{2} -> {2}\*{1}.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::conj (const Multivector< Scalar_T, LO, HI > &val)`  
*Conjugation, rev o invo == invo o rev.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::quad (const Multivector< Scalar_T, LO, HI > &val)`  
*Scalar\_T quadratic form == (rev(x)\*x)(0)*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::norm (const Multivector< Scalar_T, LO, HI > &val)`  
*Scalar\_T norm == sum of norm of coordinates.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::abs (const Multivector< Scalar_T, LO, HI > &val)`  
*Absolute value == sqrt(norm)*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::max\_abs (const Multivector< Scalar_T, LO, HI > &val)`  
*Maximum of absolute values of components of multivector: multivector infinity norm.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::complexifier (const Multivector< Scalar_T, LO, HI > &val)`  
*Square root of -1 which commutes with all members of the frame of the given multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::elliptic (const Multivector< Scalar_T, LO, HI > &val)`
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`static void glucat::check\_complex (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Check that i is a valid complexifier for val.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::sqrt (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Square root of multivector with specified complexifier.*

- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::sqrt (const Multivector< Scalar_T, LO, HI > &val)`  
*Square root of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::clifford_exp (const Multivector< Scalar_T, LO, HI > &val)`  
*Exponential of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::log (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Natural logarithm of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::log (const Multivector< Scalar_T, LO, HI > &val)`  
*Natural logarithm of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::cosh (const Multivector< Scalar_T, LO, HI > &val)`  
*Hyperbolic cosine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::acosh (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Inverse hyperbolic cosine of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::acosh (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse hyperbolic cosine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::cos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Cosine of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::cos (const Multivector< Scalar_T, LO, HI > &val)`  
*Cosine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::acos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*Inverse cosine of multivector with specified complexifier.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::acos (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse cosine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::sinh (const Multivector< Scalar_T, LO, HI > &val)`  
*Hyperbolic sine of multivector.*
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::asinh (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`

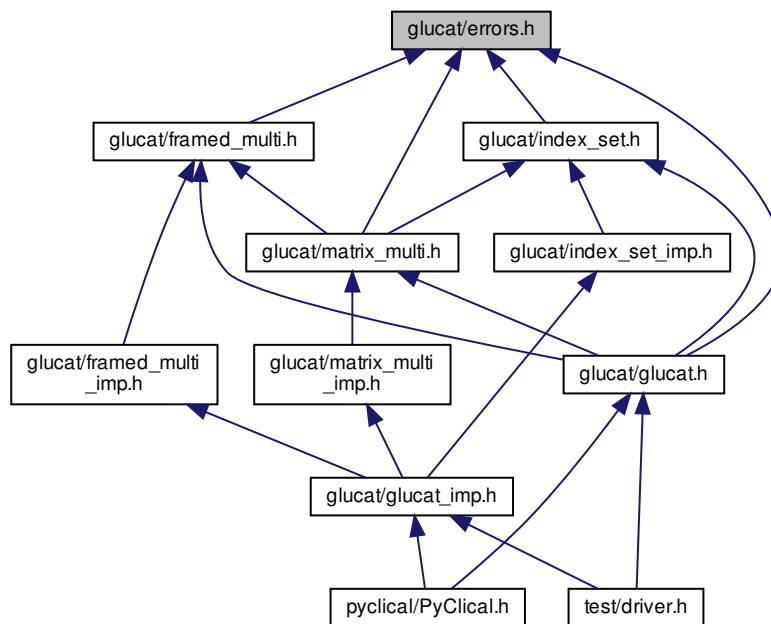


## 7.3 glucat/errors.h File Reference

```
#include <string>
#include <exception>
#include <stdexcept>
Include dependency graph for errors.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

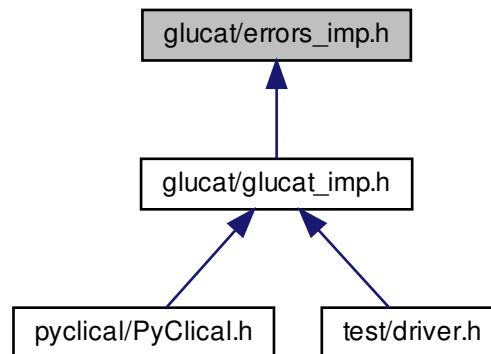
- class [glucat::glucat\\_error](#)  
*Abstract exception class.*
- class [glucat::error< Class\\_T >](#)  
*Specific exception class.*

## Namespaces

- [glucat](#)

## 7.4 glucat/errors\_imp.h File Reference

This graph shows which files directly or indirectly include this file:



## Namespaces

- [glucat](#)

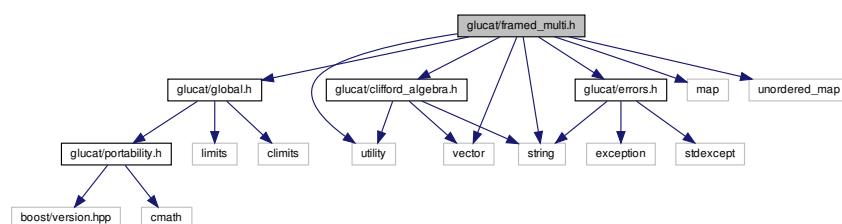
## 7.5 glucat/framed\_multi.h File Reference

```

#include "glucat/global.h"
#include "glucat/errors.h"
#include "glucat/clifford_algebra.h"
#include <string>
#include <utility>
#include <map>
#include <vector>
#include <unordered_map>

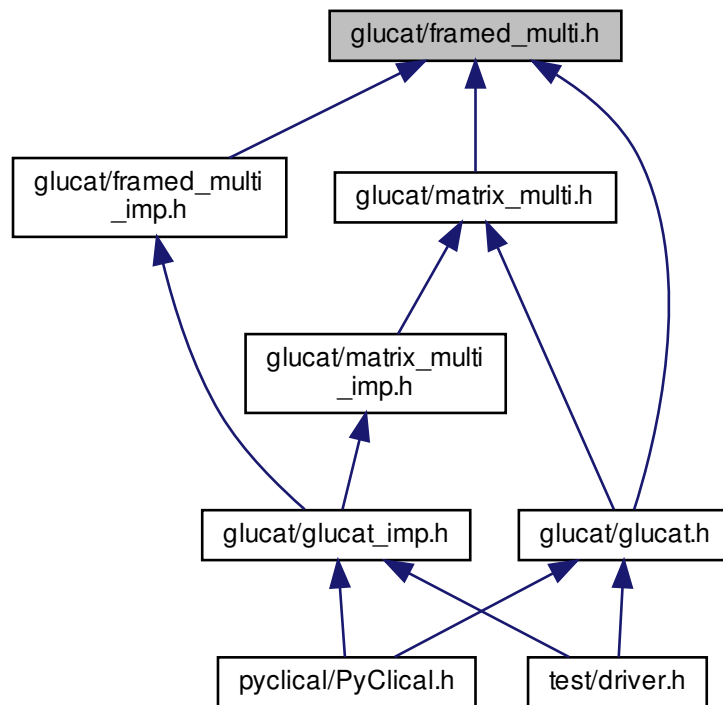
```

Include dependency graph for `framed_multi.h`:





This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::framed\\_multi< Scalar\\_T, LO, HI >](#)  
A *framed\_multi<Scalar\_T,LO,HI>* is a framed approximation to a multivector.
- class [glucat::matrix\\_multi< Scalar\\_T, LO, HI >](#)  
A *matrix\_multi<Scalar\_T,LO,HI>* is a matrix approximation to a multivector.
- class [glucat::index\\_set\\_hash< LO, HI >](#)
- class [glucat::framed\\_multi< Scalar\\_T, LO, HI >](#)  
A *framed\_multi<Scalar\_T,LO,HI>* is a framed approximation to a multivector.
- class [glucat::framed\\_multi< Scalar\\_T, LO, HI >::hash\\_size\\_t](#)
- class [glucat::framed\\_multi< Scalar\\_T, LO, HI >::var\\_term](#)  
Variable term.
- struct [std::numeric\\_limits< glucat::framed\\_multi< Scalar\\_T, LO, HI > >](#)  
Numeric limits for *framed\_multi* inherit limits for the corresponding scalar type.

## Namespaces

- [glucat](#)
- [std](#)

## Macros

- `#define _GLUCAT_MAP_IS_HASH`

## Functions

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator\* (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Geometric product.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator^ (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Outer product.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator & (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Inner product.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator% (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Left contraction.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::star (const framed_multi< Scalar_T, LO, HI > &lhs, const framed_multi< Scalar_T, LO, HI`  
`> &rhs)`  
*Hestenes scalar product.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator/ (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Geometric quotient.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator| (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Transformation via twisted adjoint action.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::istream & glucat::operator>> (std::istream &s, framed_multi< Scalar_T, LO, HI > &val)`  
*Read multivector from input.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::ostream & glucat::operator<< (std::ostream &os, const framed_multi< Scalar_T, LO, HI > &val)`  
*Write multivector to output.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::ostream & glucat::operator<< (std::ostream &os, const std::pair< const index_set< LO, HI >, Scalar_T`  
`> &term)`  
*Write term to output.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::exp (const framed_multi< Scalar_T, LO, HI > &val)`  
*Exponential of multivector.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static Scalar_T glucat::crd\_of\_mult (const std::pair< const index_set< LO, HI >, Scalar_T > &lhs, const`  
`std::pair< const index_set< LO, HI >, Scalar_T > &rhs)`  
*Coordinate of product of terms.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const std::pair< const index_set< LO, HI >, Scalar_T > glucat::operator\* (const std::pair< const index_↵`  
`set< LO, HI >, Scalar_T > &lhs, const std::pair< const index_set< LO, HI >, Scalar_T > &rhs)`

*Product of terms.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::sqrt (const framed_multi< Scalar_T, LO, HI > &val, const framed_multi< Scalar_T, LO, HI > &i, bool prechecked)`

*Square root of multivector with specified complexifier.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::log (const framed_multi< Scalar_T, LO, HI > &val, const framed_multi< Scalar_T, LO, HI > &i, bool prechecked)`

*Natural logarithm of multivector with specified complexifier.*

## 7.5.1 Macro Definition Documentation

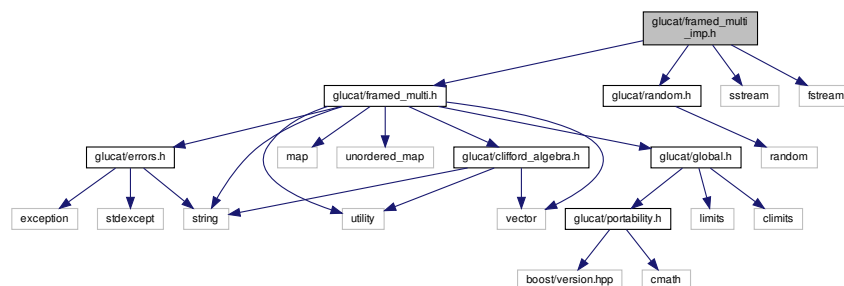
### 7.5.1.1 \_GLUCAT\_MAP\_IS\_HASH

```
#define _GLUCAT_MAP_IS_HASH
```

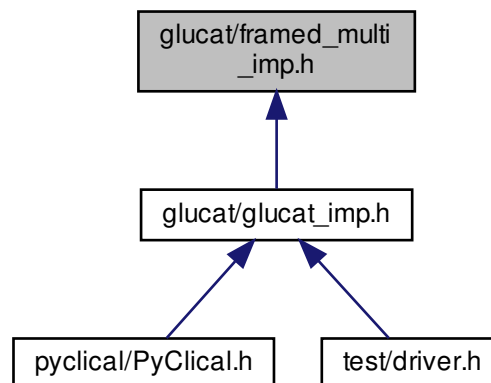
Definition at line 55 of file framed\_multi.h.

## 7.6 glucat/framed\_multi\_imp.h File Reference

```
#include "glucat/framed_multi.h"
#include "glucat/random.h"
#include <sstream>
#include <fstream>
Include dependency graph for framed_multi_imp.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::sorted\\_range< Map\\_T, Sorted\\_Map\\_T >](#)  
*Sorted range for use with output.*
- class [glucat::sorted\\_range< Sorted\\_Map\\_T, Sorted\\_Map\\_T >](#)

## Namespaces

- [glucat](#)

## Macros

- [#define \\_GLUCAT\\_HASH\\_N\(x\)](#)
- [#define \\_GLUCAT\\_HASH\\_SIZE\\_T\(x\)](#)

## Functions

- [template<typename Scalar\\_T , const index\\_t LO, const index\\_t HI>](#)  
[const framed\\_multi< Scalar\\_T, LO, HI >](#) [glucat::operator\\*](#) ([const framed\\_multi< Scalar\\_T, LO, HI >](#) &lhs,  
[const framed\\_multi< Scalar\\_T, LO, HI >](#) &rhs)  
*Geometric product.*
- [template<typename Scalar\\_T , const index\\_t LO, const index\\_t HI>](#)  
[const framed\\_multi< Scalar\\_T, LO, HI >](#) [glucat::operator^](#) ([const framed\\_multi< Scalar\\_T, LO, HI >](#) &lhs,  
[const framed\\_multi< Scalar\\_T, LO, HI >](#) &rhs)  
*Outer product.*
- [template<typename Scalar\\_T , const index\\_t LO, const index\\_t HI>](#)  
[const framed\\_multi< Scalar\\_T, LO, HI >](#) [glucat::operator &](#) ([const framed\\_multi< Scalar\\_T, LO, HI >](#) &lhs,  
[const framed\\_multi< Scalar\\_T, LO, HI >](#) &rhs)  
*Inner product.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator% (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Left contraction.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::star (const framed_multi< Scalar_T, LO, HI > &lhs, const framed_multi< Scalar_T, LO, HI`  
`> &rhs)`  
*Hestenes scalar product.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator/ (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Geometric quotient.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::operator| (const framed_multi< Scalar_T, LO, HI > &lhs,`  
`const framed_multi< Scalar_T, LO, HI > &rhs)`  
*Transformation via twisted adjoint action.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::ostream & glucat::operator<< (std::ostream &os, const framed_multi< Scalar_T, LO, HI > &val)`  
*Write multivector to output.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::ostream & glucat::operator<< (std::ostream &os, const std::pair< const index_set< LO, HI >, Scalar_T`  
`> &term)`  
*Write term to output.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::istream & glucat::operator>> (std::istream &s, framed_multi< Scalar_T, LO, HI > &val)`  
*Read multivector from input.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static Scalar_T glucat::crd_of_mult (const std::pair< const index_set< LO, HI >, Scalar_T > &lhs, const`  
`std::pair< const index_set< LO, HI >, Scalar_T > &rhs)`  
*Coordinate of product of terms.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const std::pair< const index_set< LO, HI >, Scalar_T > glucat::operator* (const std::pair< const index_↔`  
`set< LO, HI >, Scalar_T > &lhs, const std::pair< const index_set< LO, HI >, Scalar_T > &rhs)`  
*Product of terms.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::sqrt (const framed_multi< Scalar_T, LO, HI > &val, const`  
`framed_multi< Scalar_T, LO, HI > &i, bool prechecked)`  
*Square root of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::exp (const framed_multi< Scalar_T, LO, HI > &val)`  
*Exponential of multivector.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const framed_multi< Scalar_T, LO, HI > glucat::log (const framed_multi< Scalar_T, LO, HI > &val, const`  
`framed_multi< Scalar_T, LO, HI > &i, bool prechecked)`  
*Natural logarithm of multivector with specified complexifier.*

### 7.6.1 Macro Definition Documentation

### 7.6.1.1 \_GLUCAT\_HASH\_N

```
#define _GLUCAT_HASH_N(  
    x )
```

Definition at line 60 of file framed\_multi\_imp.h.

### 7.6.1.2 \_GLUCAT\_HASH\_SIZE\_T

```
#define _GLUCAT_HASH_SIZE_T(  
    x )
```

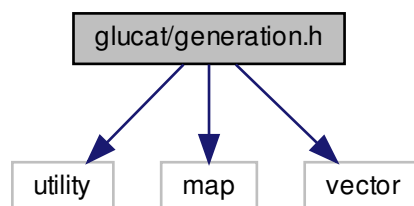
Definition at line 61 of file framed\_multi\_imp.h.

Referenced by `glucat::framed_multi< Scalar_T, LO, HI >::framed_multi()`, `glucat::operator &()`, `glucat::operator%()`, `glucat::operator*()`, and `glucat::operator^()`.

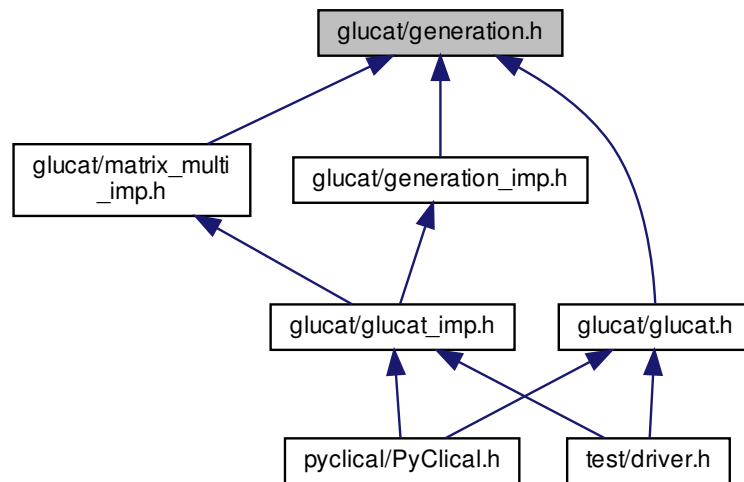
## 7.7 glucat/generation.h File Reference

```
#include <utility>  
#include <map>  
#include <vector>
```

Include dependency graph for generation.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::gen::generator\\_table< Matrix\\_T >](#)  
*Table of generators for specific signatures.*

## Namespaces

- [glucat](#)
- [glucat::gen](#)

## Typedefs

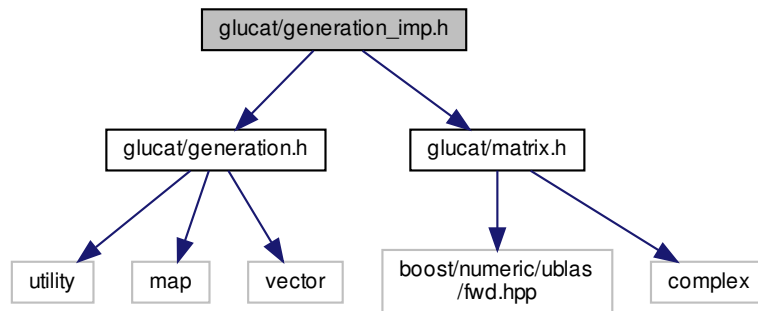
- typedef std::pair< index\_t, index\_t > [glucat::gen::signature\\_t](#)  
*A signature is a pair of indices, p, q, with p == frame.max(), q == -frame.min()*

## Variables

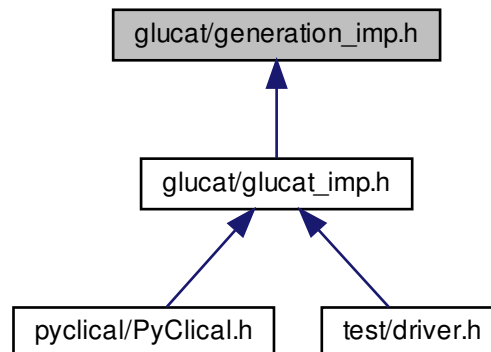
- static const index\_t [glucat::gen::offset\\_to\\_super](#) [] = {0,-1, 0,-1,-2, 3, 2, 1}  
*Offsets between the current signature and that of the real superalgebra.*

## 7.8 glucat/generation\_imp.h File Reference

```
#include "glucat/generation.h"
#include "glucat/matrix.h"
Include dependency graph for generation_imp.h:
```



This graph shows which files directly or indirectly include this file:



### Namespaces

- [glucat](#)
- [glucat::gen](#)

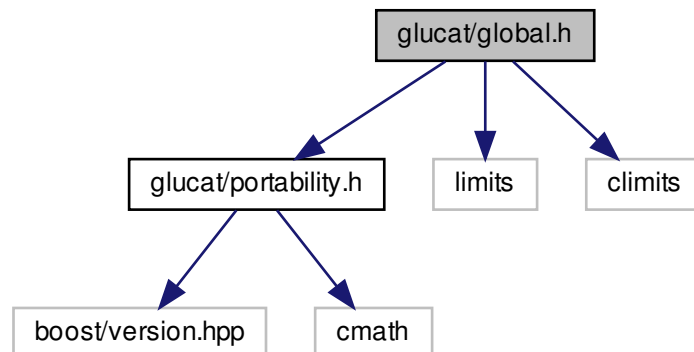
## 7.9 glucat/global.h File Reference

```
#include "glucat/portability.h"
#include <limits>
```

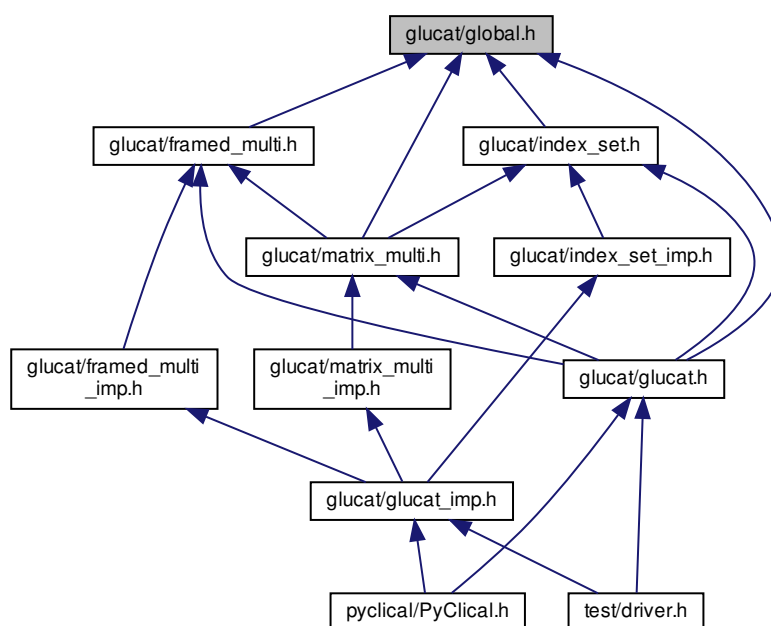


```
#include <climits>
```

Include dependency graph for global.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct `glucat::CTAssertion< bool >`  
*Compile time assertion.*
- struct `glucat::CTAssertion< true >`

- class `glucat::compare_types< LHS_T, RHS_T >`  
*Type comparison.*
- class `glucat::compare_types< T, T >`
- class `glucat::bool_to_type< truth_value >`  
*Bool to type.*
- struct `glucat::tuning< Mult_Matrix_Threshold, Div_Max_Steps, Sqrt_Max_Steps, Log_Max_Outer_Steps, Log_Max_Inner_Steps >`  
*Tuning policy.*

## Namespaces

- `glucat`

## Macros

- `#define _GLUCAT_CTAssert(expr, msg) namespace { struct msg { glucat::CTAssertion<(expr)> ERROR↵  
_##msg; }; }`

## Typedefs

- typedef int `glucat::index_t`  
*Size of index\_t should be enough to represent LO, HI.*
- typedef unsigned long `glucat::set_value_t`  
*Size of set\_value\_t should be enough to contain index\_set<LO,HI>*

## Enumerations

- enum `glucat::precision_t` { `glucat::precision_demoted`, `glucat::precision_same`, `glucat::precision_promoted` }  
*Precision policy.*

## Functions

- `glucat::_GLUCAT_CTAssert` (std::numeric\_limits< unsigned char >::radix==2, CannotDetermineBitsPer↵  
Char) const index\_t BITS\_PER\_CHAR  
*If radix of unsigned char is not 2, we can't easily determine number of bits from sizeof.*
- `glucat::_GLUCAT_CTAssert` (\_GLUCAT\_BITS\_PER\_ULONG==BITS\_PER\_SET\_VALUE, BitsPerULong↵  
DoesNotMatchSetValueT) const index\_t DEFAULT\_LO  
*Default lowest index in an index set.*
- template<typename LHS\_T, typename RHS\_T >  
LHS\_T `glucat::pos_mod` (LHS\_T lhs, RHS\_T rhs)  
*Modulo function which works reliably for lhs < 0.*

## Variables

- const double `glucat::MS_PER_S` = 1000.0  
*Timing constant: deprecated here - moved to [test/timing.h](#).*
- const index\_t `glucat::BITS_PER_SET_VALUE` = std::numeric\_limits<set\_value\_t>::digits  
*Number of bits in set\_value\_t.*
- const index\_t `glucat::DEFAULT_HI` = index\_t(BITS\_PER\_SET\_VALUE / 2)  
*Default highest index in an index set.*
- const double `glucat::DEFAULT_TRUNCATION` = std::numeric\_limits<float>::epsilon()  
*Default for truncation.*
- const unsigned int `glucat::DEFAULT_Mult_Matrix_Threshold` = 8
- const unsigned int `glucat::DEFAULT_Div_Max_Steps` = 4
- const unsigned int `glucat::DEFAULT_Sqrt_Max_Steps` = 256
- const unsigned int `glucat::DEFAULT_Log_Max_Outer_Steps` = 256
- const unsigned int `glucat::DEFAULT_Log_Max_Inner_Steps` = 32
- const unsigned int `glucat::DEFAULT_Basis_Max_Count` = 12
- const unsigned int `glucat::DEFAULT_Fast_Size_Threshold` = 1 << 6
- const unsigned int `glucat::DEFAULT_Inv_Fast_Dim_Threshold` = 1 << 3
- const unsigned int `glucat::DEFAULT_Products_Size_Threshold` = 1 << 22
- const precision\_t `glucat::DEFAULT_Function_Precision` = precision\_same

## 7.9.1 Macro Definition Documentation

### 7.9.1.1 \_GLUCAT\_CTAssert

```
#define _GLUCAT_CTAssert(  
    expr,  
    msg ) namespace { struct msg { glucat::CTAssertion<(expr)> ERROR_##msg; }; }
```

Definition at line 48 of file global.h.

## 7.10 glucat/glucat.h File Reference

```
#include "glucat/portability.h"  
#include "glucat/global.h"  
#include "glucat/errors.h"  
#include "glucat/index_set.h"  
#include "glucat/scalar.h"  
#include "glucat/long_double.h"  
#include "glucat/qd.h"  
#include "glucat/clifford_algebra.h"  
#include "glucat/framed_multi.h"  
#include "glucat/generation.h"  
#include "glucat/matrix.h"
```



## Macros

- `#define GLUCAT_HAVE_INTTYPES_H 1`
- `#define GLUCAT_HAVE_MEMORY_H 1`
- `#define GLUCAT_HAVE_STDINT_H 1`
- `#define GLUCAT_HAVE_STDLIB_H 1`
- `#define GLUCAT_HAVE_STRINGS_H 1`
- `#define GLUCAT_HAVE_STRING_H 1`
- `#define GLUCAT_HAVE_SYS_STAT_H 1`
- `#define GLUCAT_HAVE_SYS_TYPES_H 1`
- `#define GLUCAT_HAVE_UNISTD_H 1`
- `#define GLUCAT_PACKAGE "glucat"`
- `#define GLUCAT_PACKAGE_BUGREPORT ""`
- `#define GLUCAT_PACKAGE_NAME "glucat"`
- `#define GLUCAT_PACKAGE_STRING "glucat 0.8.2"`
- `#define GLUCAT_PACKAGE_TARNAME "glucat"`
- `#define GLUCAT_PACKAGE_URL ""`
- `#define GLUCAT_PACKAGE_VERSION "0.8.2"`
- `#define GLUCAT_STDC_HEADERS 1`
- `#define GLUCAT_VERSION "0.8.2"`

### 7.11.1 Macro Definition Documentation

#### 7.11.1.1 GLUCAT\_HAVE\_INTTYPES\_H

```
#define GLUCAT_HAVE_INTTYPES_H 1
```

Definition at line 10 of file `glucat_config.h`.

#### 7.11.1.2 GLUCAT\_HAVE\_MEMORY\_H

```
#define GLUCAT_HAVE_MEMORY_H 1
```

Definition at line 18 of file `glucat_config.h`.

#### 7.11.1.3 GLUCAT\_HAVE\_STDINT\_H

```
#define GLUCAT_HAVE_STDINT_H 1
```

Definition at line 23 of file `glucat_config.h`.

#### 7.11.1.4 GLUCAT\_HAVE\_STDLIB\_H

```
#define GLUCAT_HAVE_STDLIB_H 1
```

Definition at line 28 of file `glucat_config.h`.

#### 7.11.1.5 GLUCAT\_HAVE\_STRING\_H

```
#define GLUCAT_HAVE_STRING_H 1
```

Definition at line 38 of file `glucat_config.h`.

#### 7.11.1.6 GLUCAT\_HAVE\_STRINGS\_H

```
#define GLUCAT_HAVE_STRINGS_H 1
```

Definition at line 33 of file `glucat_config.h`.

#### 7.11.1.7 GLUCAT\_HAVE\_SYS\_STAT\_H

```
#define GLUCAT_HAVE_SYS_STAT_H 1
```

Definition at line 43 of file `glucat_config.h`.

#### 7.11.1.8 GLUCAT\_HAVE\_SYS\_TYPES\_H

```
#define GLUCAT_HAVE_SYS_TYPES_H 1
```

Definition at line 48 of file `glucat_config.h`.

#### 7.11.1.9 GLUCAT\_HAVE\_UNISTD\_H

```
#define GLUCAT_HAVE_UNISTD_H 1
```

Definition at line 53 of file `glucat_config.h`.

#### 7.11.1.10 GLUCAT\_PACKAGE

```
#define GLUCAT_PACKAGE "glucat"
```

Definition at line 58 of file glucat\_config.h.

#### 7.11.1.11 GLUCAT\_PACKAGE\_BUGREPORT

```
#define GLUCAT_PACKAGE_BUGREPORT ""
```

Definition at line 63 of file glucat\_config.h.

#### 7.11.1.12 GLUCAT\_PACKAGE\_NAME

```
#define GLUCAT_PACKAGE_NAME "glucat"
```

Definition at line 68 of file glucat\_config.h.

Referenced by glucat::control\_t::control\_t().

#### 7.11.1.13 GLUCAT\_PACKAGE\_STRING

```
#define GLUCAT_PACKAGE_STRING "glucat 0.8.2"
```

Definition at line 73 of file glucat\_config.h.

#### 7.11.1.14 GLUCAT\_PACKAGE\_TARNAME

```
#define GLUCAT_PACKAGE_TARNAME "glucat"
```

Definition at line 78 of file glucat\_config.h.

#### 7.11.1.15 GLUCAT\_PACKAGE\_URL

```
#define GLUCAT_PACKAGE_URL ""
```

Definition at line 83 of file glucat\_config.h.

#### 7.11.1.16 GLUCAT\_PACKAGE\_VERSION

```
#define GLUCAT_PACKAGE_VERSION "0.8.2"
```

Definition at line 88 of file `glucat_config.h`.

#### 7.11.1.17 GLUCAT\_STDC\_HEADERS

```
#define GLUCAT_STDC_HEADERS 1
```

Definition at line 93 of file `glucat_config.h`.

#### 7.11.1.18 GLUCAT\_VERSION

```
#define GLUCAT_VERSION "0.8.2"
```

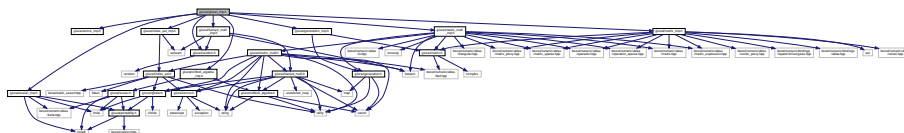
Definition at line 98 of file `glucat_config.h`.

Referenced by `glucat::control_t::control_t()`.

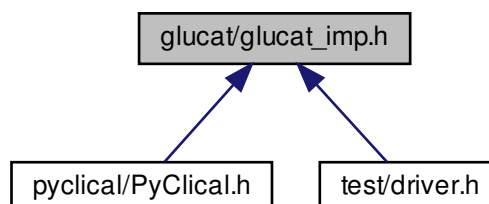
## 7.12 glucat/glucat\_imp.h File Reference

```
#include "glucat/errors_imp.h"
#include "glucat/index_set_imp.h"
#include "glucat/scalar_imp.h"
#include "glucat/clifford_algebra_imp.h"
#include "glucat/random.h"
#include "glucat/framed_multi_imp.h"
#include "glucat/matrix_imp.h"
#include "glucat/generation_imp.h"
#include "glucat/matrix_multi_imp.h"
```

Include dependency graph for `glucat_imp.h`:



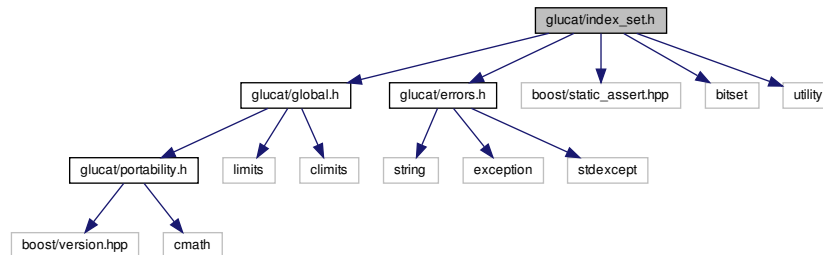
This graph shows which files directly or indirectly include this file:



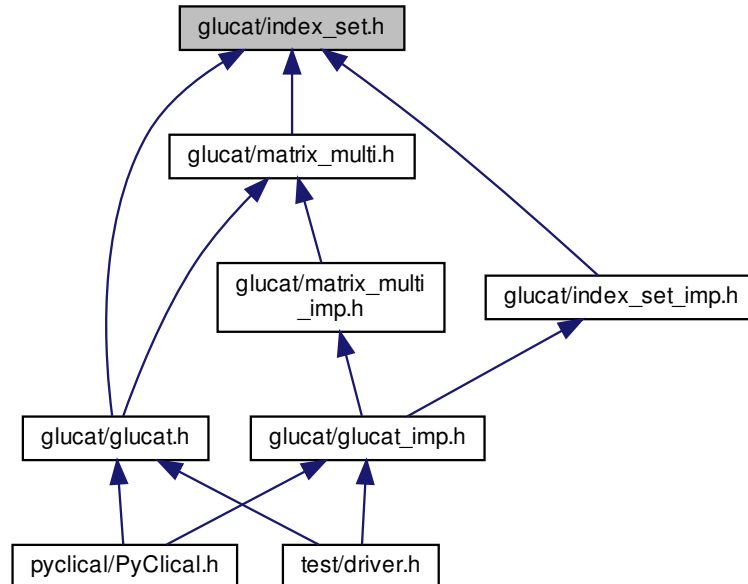


## 7.13 glucat/index\_set.h File Reference

```
#include "glucat/global.h"
#include "glucat/errors.h"
#include <boost/static_assert.hpp>
#include <bitset>
#include <utility>
Include dependency graph for index_set.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class `glucat::index_set< LO, HI >`

*Index set class based on `std::bitset<>` in Gnu standard C++ library.*

- class [glucat::index\\_set< LO, HI >](#)  
*Index set class based on `std::bitset<>` in Gnu standard C++ library.*
- class [glucat::index\\_set< LO, HI >::reference](#)  
*Index set member reference.*

## Namespaces

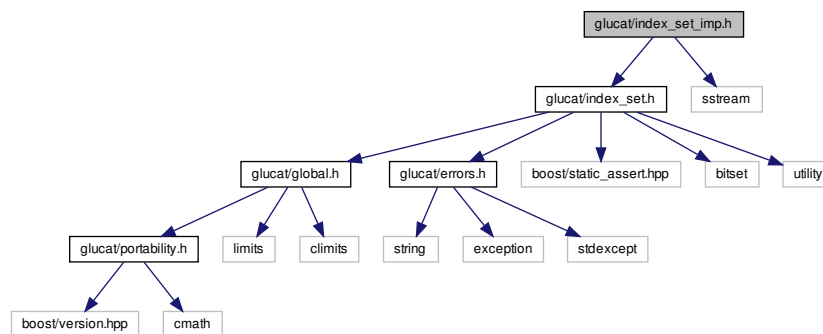
- [glucat](#)

## Functions

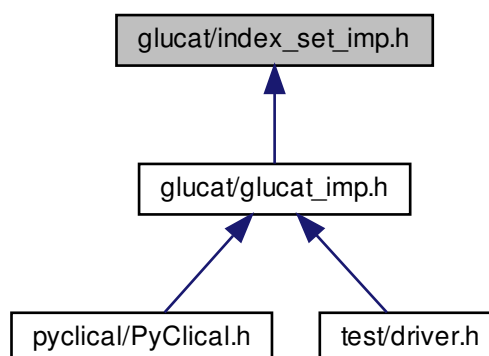
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator^ (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`  
*Symmetric set difference: exclusive or.*
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator & (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`  
*Set intersection: and.*
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator| (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`  
*Set union: or.*
- `template<const index_t LO, const index_t HI>`  
`int glucat::compare (const index_set< LO, HI > &a, const index_set< LO, HI > &b)`  
*"lexicographic compare" eg. {3,4,5} is less than {3,7,8}*
- `glucat::GLUCAT\_CTAssert (sizeof(set_value_t) >=sizeof(std::bitset< DEFAULT_HI-DEFAULT_LO >), Default_index_set_too_big_for_value) template< const index_t LO`  
*Size of `set_value_t` should be enough to contain `bitset<DEFAULT_HI-DEFAULT_LO>`*
- `const index_t HI std::ostream & glucat::operator<< (std::ostream &os, const index_set< LO, HI > &ist)`  
*Write out index set.*
- `template<const index_t LO, const index_t HI>`  
`std::istream & glucat::operator>> (std::istream &s, index_set< LO, HI > &ist)`  
*Read in index set.*
- `int glucat::sign\_of\_square (index_t j)`  
*Square of generator {j}.*
- `template<const index_t LO, const index_t HI>`  
`index_t glucat::min\_neg (const index_set< LO, HI > &ist)`  
*Minimum negative index, or 0 if none.*
- `template<const index_t LO, const index_t HI>`  
`index_t glucat::max\_pos (const index_set< LO, HI > &ist)`  
*Maximum positive index, or 0 if none.*

## 7.14 glucat/index\_set\_imp.h File Reference

```
#include "glucat/index_set.h"
#include <sstream>
Include dependency graph for index_set_imp.h:
```



This graph shows which files directly or indirectly include this file:



### Namespaces

- [glucat](#)

### Functions

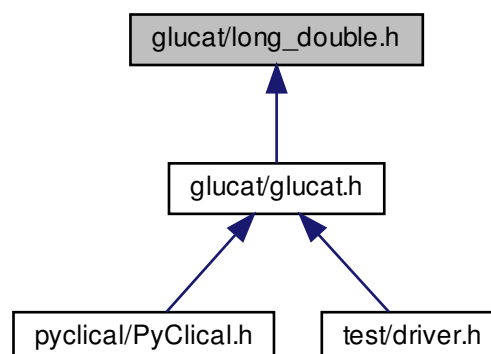
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator^ (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`

*Symmetric set difference: exclusive or.*

- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator & (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`  
*Set intersection: and.*
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator | (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`  
*Set union: or.*
- `template<const index_t LO, const index_t HI>`  
`int glucat::compare (const index_set< LO, HI > &a, const index_set< LO, HI > &b)`  
*"lexicographic compare" eg. {3,4,5} is less than {3,7,8}*
- `const index_t HI std::ostream & glucat::operator << (std::ostream &os, const index_set< LO, HI > &ist)`  
*Write out index set.*
- `template<const index_t LO, const index_t HI>`  
`std::istream & glucat::operator >> (std::istream &s, index_set< LO, HI > &ist)`  
*Read in index set.*
- `static unsigned long glucat::inverse\_reversed\_gray (unsigned long x)`  
*Inverse reversed Gray code.*
- `static unsigned long glucat::inverse\_gray (unsigned long x)`  
*Inverse Gray code.*
- `int glucat::sign\_of\_square (index_t j)`  
*Square of generator {j}.*
- `template<const index_t LO, const index_t HI>`  
`index_t glucat::min\_neg (const index_set< LO, HI > &ist)`  
*Minimum negative index, or 0 if none.*
- `template<const index_t LO, const index_t HI>`  
`index_t glucat::max\_pos (const index_set< LO, HI > &ist)`  
*Maximum positive index, or 0 if none.*

## 7.15 glucat/long\_double.h File Reference

This graph shows which files directly or indirectly include this file:



## Classes

- struct `glucat::numeric_traits< Scalar_T >::demoted<>`  
*Demoted type for long double.*

## Namespaces

- `glucat`

## Variables

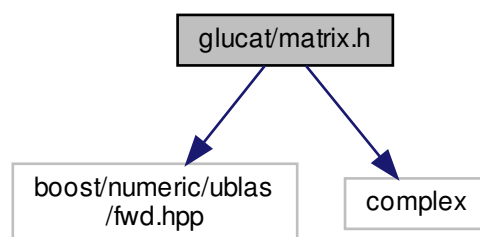
- static const long double `glucat::l_pi` = 3.1415926535897932384626433832795029L
- static const long double `glucat::l_ln2` = 0.6931471805599453094172321214581766L

## 7.16 glucat/matrix.h File Reference

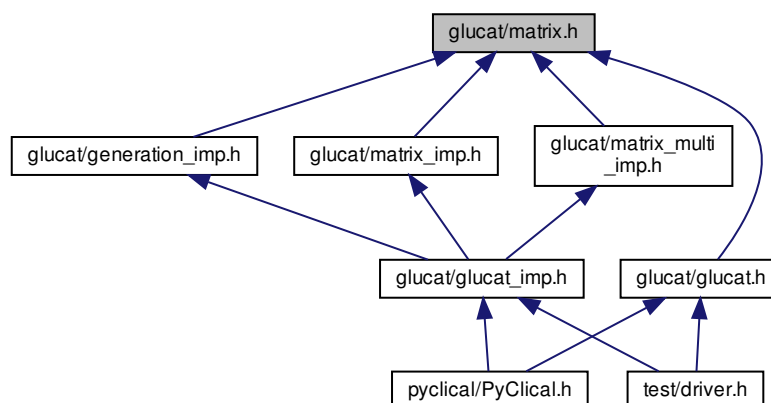
```
#include <boost/numeric/ublas/fwd.hpp>
```

```
#include <complex>
```

Include dependency graph for matrix.h:



This graph shows which files directly or indirectly include this file:



## Classes

- struct [glucat::matrix::eig\\_genus](#)< [Matrix\\_T](#) >  
*Structure containing classification of eigenvalues.*

## Namespaces

- [glucat](#)
- [glucat::matrix](#)

## Enumerations

- enum [glucat::matrix::eig\\_case\\_t](#) { [glucat::matrix::safe\\_eig\\_case](#), [glucat::matrix::negative\\_eig\\_case](#), [glucat::matrix::both\\_eig\\_case](#) }  
*Classification of eigenvalues of a matrix.*

## Functions

- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [glucat::matrix::kron](#) (const LHS\_T &lhs, const RHS\_T &rhs)  
*Kronecker tensor product of matrices - as per Matlab kron.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [glucat::matrix::mono\\_kron](#) (const LHS\_T &lhs, const RHS\_T &rhs)  
*Sparse Kronecker tensor product of monomial matrices.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [glucat::matrix::nork](#) (const LHS\_T &lhs, const RHS\_T &rhs, const bool mono=true)  
*Left inverse of Kronecker product.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T [glucat::matrix::signed\\_perm\\_nork](#) (const LHS\_T &lhs, const RHS\_T &rhs)  
*Left inverse of Kronecker product where lhs is a signed permutation matrix.*
- template<typename Matrix\_T >  
Matrix\_T::size\_type [glucat::matrix::nnz](#) (const Matrix\_T &m)  
*Number of non-zeros.*
- template<typename Matrix\_T >  
bool [glucat::matrix::isnan](#) (const Matrix\_T &m)  
*Not a Number.*
- template<typename Matrix\_T >  
const Matrix\_T [glucat::matrix::unit](#) (const typename Matrix\_T::size\_type n)  
*Unit matrix - as per Matlab eye.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T::expression\_type [glucat::matrix::mono\\_prod](#) (const ublas::matrix\_expression< LHS\_T > &lhs, const ublas::matrix\_expression< RHS\_T > &rhs)  
*Product of monomial matrices.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T::expression\_type [glucat::matrix::sparse\\_prod](#) (const ublas::matrix\_expression< LHS\_T > &lhs, const ublas::matrix\_expression< RHS\_T > &rhs)  
*Product of sparse matrices.*
- template<typename LHS\_T, typename RHS\_T >  
const RHS\_T::expression\_type [glucat::matrix::prod](#) (const ublas::matrix\_expression< LHS\_T > &lhs, const ublas::matrix\_expression< RHS\_T > &rhs)  
*Product of matrices.*

- `template<typename Scalar_T , typename LHS_T , typename RHS_T >`  
`Scalar_T glucat::matrix::inner (const LHS_T &lhs, const RHS_T &rhs)`  
*Inner product:  $\sum(x(i,j)*y(i,j))/x.nrows()$*
- `template<typename Matrix_T >`  
`Matrix_T::value_type glucat::matrix::norm_frob2 (const Matrix_T &val)`  
*Square of Frobenius norm.*
- `template<typename Matrix_T >`  
`Matrix_T::value_type glucat::matrix::trace (const Matrix_T &val)`  
*Matrix trace.*
- `template<typename Matrix_T >`  
`ublas::vector< std::complex< double > > glucat::matrix::eigenvalues (const Matrix_T &val)`  
*Eigenvalues of a matrix.*
- `template<typename Matrix_T >`  
`eig_genus< Matrix_T > glucat::matrix::classify_eigenvalues (const Matrix_T &val)`  
*Classify the eigenvalues of a matrix.*

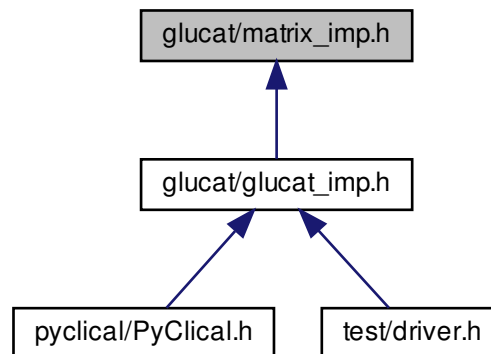
## 7.17 glucat/matrix\_imp.h File Reference

```
#include "glucat/matrix.h"
#include <boost/numeric/ublas/vector.hpp>
#include <boost/numeric/ublas/vector_proxy.hpp>
#include <boost/numeric/ublas/matrix.hpp>
#include <boost/numeric/ublas/matrix_expression.hpp>
#include <boost/numeric/ublas/matrix_proxy.hpp>
#include <boost/numeric/ublas/matrix_sparse.hpp>
#include <boost/numeric/ublas/operation.hpp>
#include <boost/numeric/ublas/operation_sparse.hpp>
#include <boost/numeric/bindings/lapack/driver/gees.hpp>
#include <boost/numeric/bindings/ublas.hpp>
#include <set>
```

Include dependency graph for matrix\_imp.h:



This graph shows which files directly or indirectly include this file:



## Namespaces

- [glucat](#)
- [glucat::matrix](#)

## Functions

- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T glucat::matrix::kron (const LHS_T &lhs, const RHS_T &rhs)`  
*Kronecker tensor product of matrices - as per Matlab kron.*
- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T glucat::matrix::mono\_kron (const LHS_T &lhs, const RHS_T &rhs)`  
*Sparse Kronecker tensor product of monomial matrices.*
- `template<typename LHS_T , typename RHS_T >`  
`void glucat::matrix::nork\_range (RHS_T &result, const typename LHS_T::const_iterator2 lhs_it2, const RHS_T &rhs, const typename RHS_T::size_type res_s1, const typename RHS_T::size_type res_s2)`  
*Utility routine for nork: calculate result for a range of indices.*
- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T glucat::matrix::nork (const LHS_T &lhs, const RHS_T &rhs, const bool mono=true)`  
*Left inverse of Kronecker product.*
- `template<typename LHS_T , typename RHS_T >`  
`const RHS_T glucat::matrix::signed\_perm\_nork (const LHS_T &lhs, const RHS_T &rhs)`  
*Left inverse of Kronecker product where lhs is a signed permutation matrix.*
- `template<typename Matrix_T >`  
`Matrix_T::size_type glucat::matrix::nnz (const Matrix_T &m)`  
*Number of non-zeros.*
- `template<typename Matrix_T >`  
`bool glucat::matrix::isnans (const Matrix_T &m)`  
*Not a Number.*
- `template<typename Matrix_T >`  
`const Matrix_T glucat::matrix::unit (const typename Matrix_T::size_type n)`



- `template<typename LHS_T, typename RHS_T >`  
`const RHS_T::expression_type glucat::matrix::mono_prod (const ublas::matrix_expression< LHS_T > &lhs,`  
`const ublas::matrix_expression< RHS_T > &rhs)`

- `template<typename LHS_T, typename RHS_T >`  
`const RHS_T::expression_type glucat::matrix::sparse_prod (const ublas::matrix_expression< LHS_T >`  
`&lhs, const ublas::matrix_expression< RHS_T > &rhs)`

- `template<typename LHS_T, typename RHS_T >`  
`const RHS_T::expression_type glucatt::matrix::prod (const ublas::matrix_expression< LHS_T > &lhs, const`  
`ublas::matrix_expression< RHS_T > &rhs)`

- `template<typename Scalar_T, typename LHS_T, typename RHS_T >`  
`Scalar_T glucat::matrix::inner (const LHS_T &lhs, const RHS_T &rhs)`

- `template<typename Matrix_T >`  
`Matrix_T::value_type glucat::matrix::norm_frob2 (const Matrix_T &val)`

- `template<typename Matrix_T>`  
`Matrix_T::value_type glucat::matrix::trace (const Matrix_T &val)`

- `template<typename Matrix_T>`  
`static ublas::matrix< double, ublas::column_major> glucatt::matrix::to\_lapack (const Matrix_T &val)`

- `template<typename Matrix_T>`  
`ublas::vector< std::complex< double > > glucat::matrix::eigenvalues (const Matrix_T &val)`

- `template<typename Matrix_T>`  
`eig_genus< Matrix_T> glucat::matrix::classify\_eigenvalues (const Matrix_T &val)`

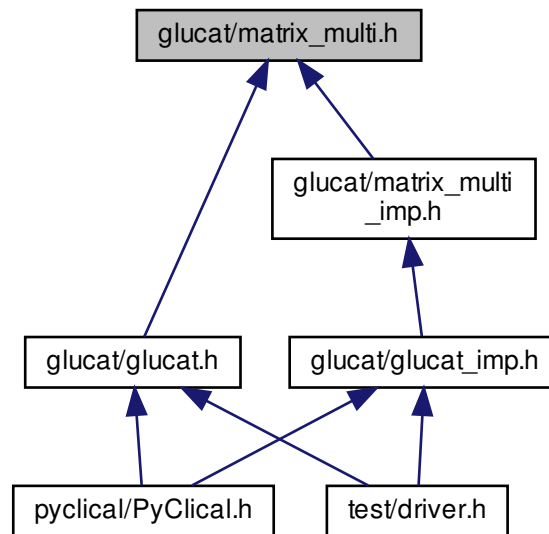
## 7.18 glucat/matrix\_multi.h File Reference

```
#include "glucat/global.h"
#include "glucat/errors.h"
#include "glucat/index_set.h"
#include "glucat/clifford_algebra.h"
#include "glucat/framed_multi.h"
#include <boost/numeric/ublas/fwd.hpp>
#include <fstream>
#include <string>
#include <utility>
#include <vector>
```

```

graph TD
    glucomath_main_h[glucomath_main.h] --> glucomath_index_h[glucomath_index.h]
    glucomath_main_h --> glucomath_framework_h[glucomath_framework.h]
    glucomath_main_h --> boost_numeric_ublas_lwd_hpp[boost/numeric/ublas/lwd.hpp]
    glucomath_main_h --> tstream[tstream]
    glucomath_index_h --> bitset[bitset]
    glucomath_index_h --> boost_static_assert_hpp[boost/static_assert.hpp]
    glucomath_index_h --> glucomath_global_h[glucomath_global.h]
    glucomath_index_h --> glucomath_vendors_h[glucomath_vendors.h]
    glucomath_framework_h --> unordered_map[unordered_map]
    glucomath_framework_h --> map[map]
    glucomath_framework_h --> glucomath_algebra_h[glucomath_algebra.h]
    glucomath_framework_h --> string[string]
    glucomath_framework_h --> utility[utility]
    glucomath_framework_h --> vector[vector]
    boost_numeric_ublas_lwd_hpp --> vector
    tstream --> vector
    glucomath_global_h --> limits[limits]
    glucomath_global_h --> climits[climits]
    glucomath_global_h --> glucomath_portability_h[glucomath_portability.h]
    glucomath_global_h --> exception[exception]
    glucomath_global_h --> stdexcept[stdexcept]
    glucomath_global_h --> string
    glucomath_vendors_h --> string
    glucomath_portability_h --> boost_version_hpp[boost/version.hpp]
    glucomath_portability_h --> cmath[cmath]
  
```

This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::framed\\_multi< Scalar\\_T, LO, HI >](#)  
A *framed\_multi< Scalar\_T, LO, HI >* is a framed approximation to a multivector.
- class [glucat::matrix\\_multi< Scalar\\_T, LO, HI >](#)  
A *matrix\_multi< Scalar\_T, LO, HI >* is a matrix approximation to a multivector.
- class [glucat::matrix\\_multi< Scalar\\_T, LO, HI >](#)  
A *matrix\_multi< Scalar\_T, LO, HI >* is a matrix approximation to a multivector.
- struct [std::numeric\\_limits< glucat::matrix\\_multi< Scalar\\_T, LO, HI > >](#)  
Numeric limits for *matrix\_multi* inherit limits for the corresponding scalar type.

## Namespaces

- [glucat](#)
- [std](#)

## Functions

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator\* (const matrix_multi< Scalar_T, LO, HI > &lhs,`  
`const matrix_multi< Scalar_T, LO, HI > &rhs)`  
*Geometric product.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator^ (const matrix_multi< Scalar_T, LO, HI > &lhs,`  
`const matrix_multi< Scalar_T, LO, HI > &rhs)`

*Outer product.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator & (const matrix_multi< Scalar_T, LO, HI > &lhs,`  
`const matrix_multi< Scalar_T, LO, HI > &rhs)`

*Inner product.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator% (const matrix_multi< Scalar_T, LO, HI > &lhs,`  
`const matrix_multi< Scalar_T, LO, HI > &rhs)`

*Left contraction.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`Scalar_T glucat::star (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI >`  
`&rhs)`

*Hestenes scalar product.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator/ (const matrix_multi< Scalar_T, LO, HI > &lhs, const`  
`matrix_multi< Scalar_T, LO, HI > &rhs)`

*Geometric quotient.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator| (const matrix_multi< Scalar_T, LO, HI > &lhs, const`  
`matrix_multi< Scalar_T, LO, HI > &rhs)`

*Transformation via twisted adjoint action.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::istream & glucat::operator>> (std::istream &s, matrix_multi< Scalar_T, LO, HI > &val)`

*Read multivector from input.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`std::ostream & glucat::operator<< (std::ostream &os, const matrix_multi< Scalar_T, LO, HI > &val)`

*Write multivector to output.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::reframe (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_↔`  
`multi< Scalar_T, LO, HI > &rhs, matrix_multi< Scalar_T, LO, HI > &lhs_reframed, matrix_multi< Scalar_T,`  
`LO, HI > &rhs_reframed)`

*Find a common frame for operands of a binary operator.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::sqrt (const matrix_multi< Scalar_T, LO, HI > &val, const`  
`matrix_multi< Scalar_T, LO, HI > &i, bool prechecked)`

*Square root of multivector with specified complexifier.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::matrix_sqrt (const matrix_multi< Scalar_T, LO, HI > &val,`  
`const matrix_multi< Scalar_T, LO, HI > &i)`

*Square root of multivector with specified complexifier.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::log (const matrix_multi< Scalar_T, LO, HI > &val, const`  
`matrix_multi< Scalar_T, LO, HI > &i, bool prechecked)`

*Natural logarithm of multivector with specified complexifier.*

- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::matrix_log (const matrix_multi< Scalar_T, LO, HI > &val,`  
`const matrix_multi< Scalar_T, LO, HI > &i)`

*Natural logarithm of multivector with specified complexifier.*

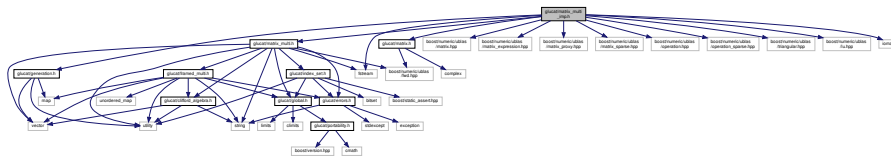
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::exp (const matrix_multi< Scalar_T, LO, HI > &val)`

*Exponential of multivector.*

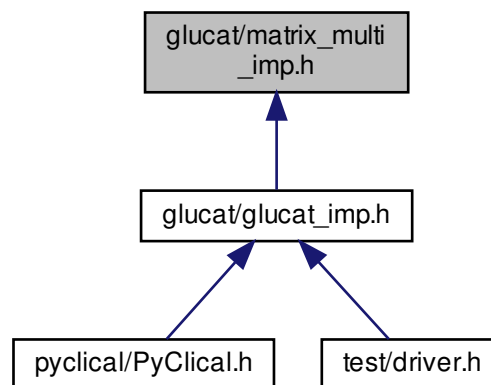
## 7.19 glucat/matrix\_multi\_imp.h File Reference

```
#include "glucat/matrix_multi.h"
#include "glucat/matrix.h"
#include "glucat/generation.h"
#include <boost/numeric/ublas/matrix.hpp>
#include <boost/numeric/ublas/matrix_expression.hpp>
#include <boost/numeric/ublas/matrix_proxy.hpp>
#include <boost/numeric/ublas/matrix_sparse.hpp>
#include <boost/numeric/ublas/operation.hpp>
#include <boost/numeric/ublas/operation_sparse.hpp>
#include <boost/numeric/ublas/triangular.hpp>
#include <boost/numeric/ublas/lu.hpp>
#include <fstream>
#include <iomanip>
```

Include dependency graph for matrix\_multi\_imp.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class [glucat::basis\\_table< Scalar\\_T, LO, HI, Matrix\\_T >](#)  
*Table of basis elements used as a cache by basis\_element()*

### Namespaces

- [glucat](#)

## Functions

- `index_t glucat::offset_level` (const index\_t p, const index\_t q)  
*Determine the log2 dim corresponding to signature p, q.*
- `template<typename Matrix_Index_T, const index_t LO, const index_t HI>`  
`static Matrix_Index_T glucat::folded_dim` (const index\_set< LO, HI > &sub)  
*Determine the matrix dimension of the fold of a subalgebra.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::reframe` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs, matrix\_multi< Scalar\_T, LO, HI > &lhs\_reframed, matrix\_multi< Scalar\_T, LO, HI > &rhs\_reframed)  
*Find a common frame for operands of a binary operator.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator*` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Geometric product.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator^` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Outer product.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator &` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Inner product.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator%` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Left contraction.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`Scalar_T glucat::star` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Hestenes scalar product.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator/` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Geometric quotient.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::operator|` (const matrix\_multi< Scalar\_T, LO, HI > &lhs, const matrix\_multi< Scalar\_T, LO, HI > &rhs)  
*Transformation via twisted adjoint action.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`std::ostream & glucat::operator<<` (std::ostream &os, const matrix\_multi< Scalar\_T, LO, HI > &val)  
*Write multivector to output.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`std::istream & glucat::operator>>` (std::istream &s, matrix\_multi< Scalar\_T, LO, HI > &val)  
*Read multivector from input.*
- `template<typename Multivector_T, typename Matrix_T, typename Basis_Matrix_T>`  
`static Multivector_T glucat::fast` (const Matrix\_T &X, index\_t level)  
*Inverse generalized Fast Fourier Transform.*
- `template<typename Scalar_T, const index_t LO, const index_t HI>`  
`static const matrix_multi< Scalar_T, LO, HI > glucat::pade_approx` (const int array\_size, const Scalar\_T a[], const Scalar\_T b[], const matrix\_multi< Scalar\_T, LO, HI > &X)  
*Pade' approximation.*

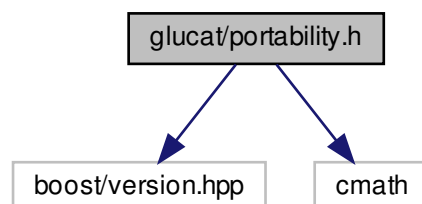
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static void glucat::db\_step (matrix_multi< Scalar_T, LO, HI > &M, matrix_multi< Scalar_T, LO, HI > &Y)`  
*Single step of product form of Denman-Beavers square root iteration.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix_multi< Scalar_T, LO, HI > glucat::db\_sqrt (const matrix_multi< Scalar_T, LO, HI > &val)`  
*Product form of Denman-Beavers square root iteration.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::sqrt (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i, bool prechecked)`  
*Square root of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::matrix\_sqrt (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i)`  
*Square root of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix_multi< Scalar_T, LO, HI > glucat::pade\_log (const matrix_multi< Scalar_T, LO, HI > &val)`  
*Pade' approximation of log.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`static const matrix_multi< Scalar_T, LO, HI > glucat::cascade\_log (const matrix_multi< Scalar_T, LO, HI > &val)`  
*Incomplete square root cascade and Pade' approximation of log.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::log (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i, bool prechecked)`  
*Natural logarithm of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::matrix\_log (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i)`  
*Natural logarithm of multivector with specified complexifier.*
- `template<typename Scalar_T , const index_t LO, const index_t HI>`  
`const matrix_multi< Scalar_T, LO, HI > glucat::exp (const matrix_multi< Scalar_T, LO, HI > &val)`  
*Exponential of multivector.*

## 7.20 `glucat/portability.h` File Reference

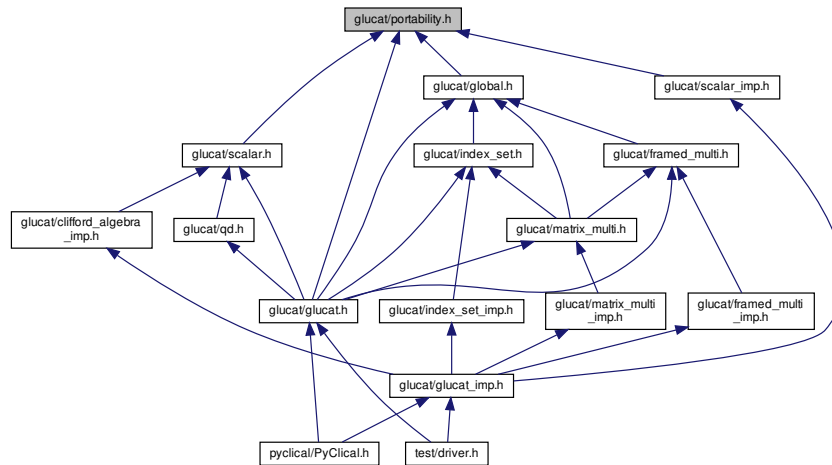
```
#include <boost/version.hpp>
```

```
#include <cmath>
```

Include dependency graph for `portability.h`:



This graph shows which files directly or indirectly include this file:



## Macros

- `#define _GLUCAT_ISNAN(x) (x != x)`
- `#define _GLUCAT_ISINF(x) (!_GLUCAT_ISNAN(x) && _GLUCAT_ISNAN(x-x))`
- `#define UBLAS_ABS abs`
- `#define UBLAS_SQRT sqrt`

## 7.20.1 Macro Definition Documentation

### 7.20.1.1 \_GLUCAT\_ISINF

```
#define _GLUCAT_ISINF(
    x ) ( !_GLUCAT_ISNAN(x) && _GLUCAT_ISNAN(x-x) )
```

Definition at line 48 of file portability.h.

Referenced by `glucat::numeric_traits< Scalar_T >::isInf()`.

### 7.20.1.2 \_GLUCAT\_ISNAN

```
#define _GLUCAT_ISNAN(
    x ) ( x != x )
```

Definition at line 47 of file portability.h.

Referenced by `glucat::numeric_traits< Scalar_T >::isNaN()`.

### 7.20.1.3 UBLAS\_ABS

```
#define UBLAS_ABS abs
```

Definition at line 56 of file portability.h.

Referenced by `glucat::numeric_traits< Scalar_T >::abs()`.

### 7.20.1.4 UBLAS\_SQRT

```
#define UBLAS_SQRT sqrt
```

Definition at line 57 of file portability.h.

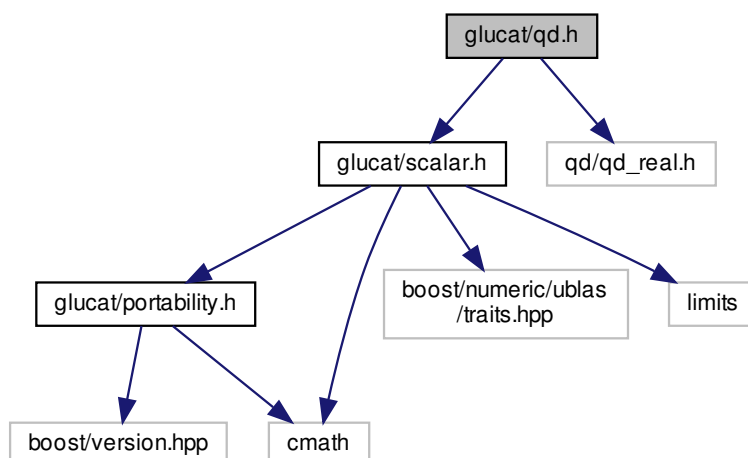
Referenced by `glucat::numeric_traits< Scalar_T >::sqrt()`.

## 7.21 glucat/qd.h File Reference

```
#include "glucat/scalar.h"
```

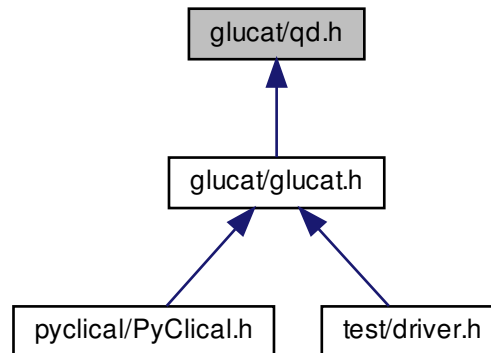
```
#include <qd/qd_real.h>
```

Include dependency graph for qd.h:





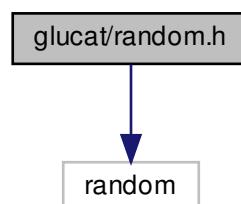
This graph shows which files directly or indirectly include this file:



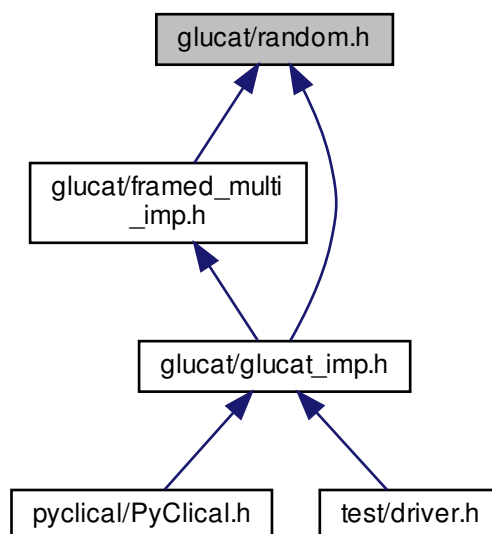
## 7.22 glucat/random.h File Reference

```
#include <random>
```

Include dependency graph for random.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::random\\_generator< Scalar\\_T >](#)  
*Random number generator with single instance per Scalar\_T.*

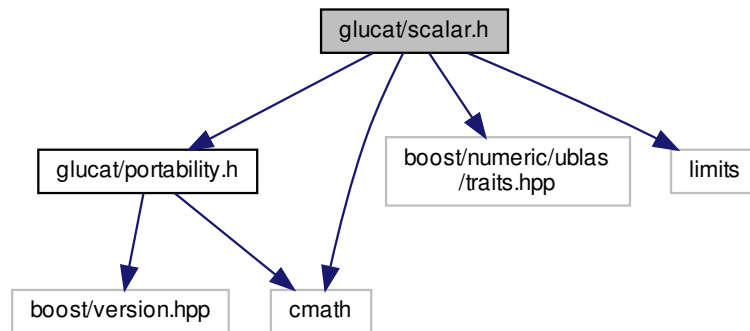
## Namespaces

- [glucat](#)

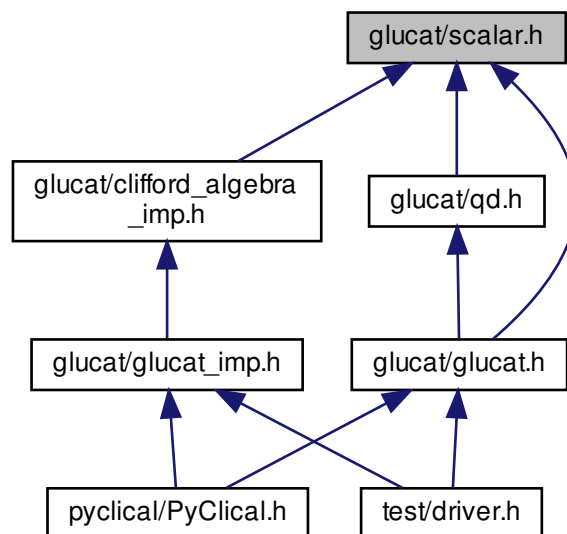
## 7.23 glucat/scalar.h File Reference

```
#include "glucat/portability.h"  
#include <boost/numeric/ublas/traits.hpp>  
#include <cmath>  
#include <limits>
```

Include dependency graph for scalar.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class `glucat::numeric_traits< Scalar_T >`  
Extra traits which extend numeric limits.
- struct `glucat::numeric_traits< Scalar_T >::promoted`  
Promoted type.
- struct `glucat::numeric_traits< Scalar_T >::demoted<>`  
Demoted type for long double.

## Namespaces

- [glucat](#)

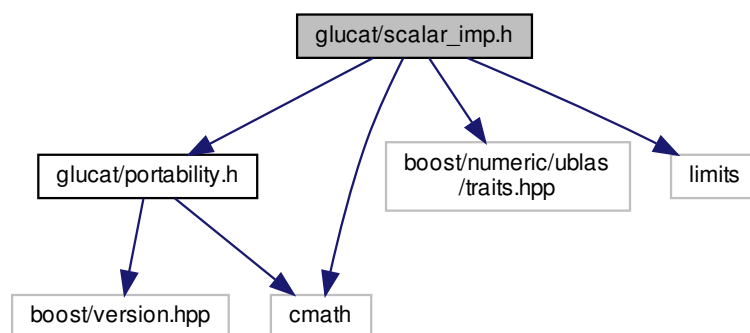
## Functions

- `template<typename Scalar_T >`  
`Scalar_T glucat::log2 (const Scalar_T &x)`  
*Log base 2 of scalar.*

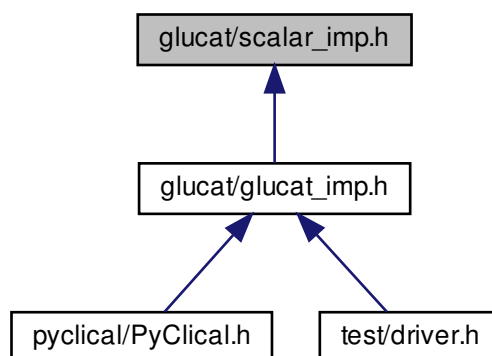
## 7.24 glucat/scalar\_imp.h File Reference

```
#include "glucat/portability.h"
#include <boost/numeric/ublas/traits.hpp>
#include <cmath>
#include <limits>
```

Include dependency graph for scalar\_imp.h:



This graph shows which files directly or indirectly include this file:



## Namespaces

- [glucat](#)

## Functions

- `template<typename Scalar_T >`  
`numeric_traits< Scalar_T >::promoted::type` [glucat::to\\_promote](#) (const Scalar\_T &val)  
*Cast to promote.*
- `template<typename Scalar_T >`  
`numeric_traits< Scalar_T >::demoted::type` [glucat::to\\_demote](#) (const Scalar\_T &val)  
*Cast to demote.*

## 7.25 pyclical/glucat.pxd File Reference

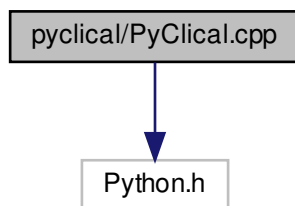
### Namespaces

- [glucat](#)

## 7.26 pyclical/PyClical.cpp File Reference

```
#include "Python.h"
```

Include dependency graph for PyClical.cpp:



## Macros

- `#define` [PY\\_SSIZE\\_T\\_CLEAN](#)

### 7.26.1 Macro Definition Documentation

### 7.26.1.1 PY\_SSIZE\_T\_CLEAN

```
#define PY_SSIZE_T_CLEAN
```

Definition at line 57 of file PyClical.cpp.

## 7.27 pyclical/PyClical.h File Reference

```
#include "glucat/glucat.h"
#include "glucat/glucat_imp.h"
#include <iostream>
#include <sstream>
#include <iomanip>
```

Include dependency graph for PyClical.h:



### Namespaces

- [cga3](#)  
*Definitions for 3D Conformal Geometric Algebra [DL].*

### Typedefs

- typedef [glucat::tuning](#)< [glucat::DEFAULT\\_Mult\\_Matrix\\_Threshold](#), [glucat::DEFAULT\\_Div\\_Max\\_Steps](#), [glucat::DEFAULT\\_Sqrt\\_Max\\_Steps](#), [glucat::DEFAULT\\_Log\\_Max\\_Outer\\_Steps](#), [glucat::DEFAULT\\_Log\\_Max\\_Inner\\_Steps](#), [glucat::DEFAULT\\_Basis\\_Max\\_Count](#), [glucat::DEFAULT\\_Fast\\_Size\\_Threshold](#), [glucat::DEFAULT\\_Inv\\_Fast\\_Dim\\_Threshold](#), [glucat::DEFAULT\\_Products\\_Size\\_Threshold](#), [glucat::precision\\_promoted](#) > [Tune\\_P](#)
- typedef std::string [String](#)
- typedef [index\\_set](#)< [lo\\_ndx](#), [hi\\_ndx](#) > [IndexSet](#)
- typedef double [scalar\\_t](#)
- typedef [matrix\\_multi](#)< [scalar\\_t](#) > [Clifford](#)

### Functions

- template<typename [Scalar\\_T](#) >  
[PyObject](#) \* [PyFloat\\_FromDouble](#) ([Scalar\\_T](#) v)
- template<typename [Index\\_Set\\_T](#) >  
[String](#) [index\\_set\\_to\\_repr](#) (const [Index\\_Set\\_T](#) &ist)  
*The "official" string representation of [Index\\_Set\\_T](#) ist.*
- template<typename [Index\\_Set\\_T](#) >  
[String](#) [index\\_set\\_to\\_str](#) (const [Index\\_Set\\_T](#) &ist)  
*The "informal" string representation of [Index\\_Set\\_T](#) ist.*
- template<typename [Multivector\\_T](#) >  
[String](#) [clifford\\_to\\_repr](#) (const [Multivector\\_T](#) &mv)

The "official" string representation of `Multivector_T` mv.

- template<typename `Multivector_T`>  
[String clifford\\_to\\_str](#) (const `Multivector_T` &mv)

The "informal" string representation of `Multivector_T` mv.

- template<typename `Multivector_T`>  
`Multivector_T` [cga3::cga3](#) (const `Multivector_T` &x)

Convert Euclidean 3D vector to Conformal Geometric Algebra null vector [DL (10.50)].

- template<typename `Multivector_T`>  
`Multivector_T` [cga3::cga3std](#) (const `Multivector_T` &X)

Convert CGA3 null vector to standard Conformal Geometric Algebra null vector [DL (10.52)].

- template<typename `Multivector_T`>  
`Multivector_T` [cga3::agc3](#) (const `Multivector_T` &X)

Convert CGA3 null vector to Euclidean 3D vector [DL (10.50)].

## Variables

- const [index\\_t lo\\_ndx](#) = DEFAULT\_LO
- const [index\\_t hi\\_ndx](#) = DEFAULT\_HI

### 7.27.1 Typedef Documentation

#### 7.27.1.1 Clifford

```
typedef matrix\_multi<scalar\_t> Clifford
```

Definition at line 160 of file `PyClical.h`.

#### 7.27.1.2 IndexSet

```
typedef index\_set<lo\_ndx,hi\_ndx> IndexSet
```

Definition at line 157 of file `PyClical.h`.

#### 7.27.1.3 scalar\_t

```
typedef double scalar\_t
```

Definition at line 159 of file `PyClical.h`.

#### 7.27.1.4 String

```
typedef std::string String
```

Definition at line 65 of file PyClical.h.

#### 7.27.1.5 Tune\_P

```
typedef glucat::tuning< glucat::DEFAULT_Mult_Matrix_Threshold, glucat::DEFAULT_Div_Max_Steps,
glucal::DEFAULT_Sqrt_Max_Steps, glucat::DEFAULT_Log_Max_Outer_Steps, glucat::DEFAULT_Log_Max_Inner_Steps,
glucat::DEFAULT_Basis_Max_Count, glucat::DEFAULT_Fast_Size_Threshold, glucat::DEFAULT_Inv_Fast_Dim_Threshold,
glucat::DEFAULT_Products_Size_Threshold, glucat::precision_promoted > Tune_P
```

Definition at line 49 of file PyClical.h.

### 7.27.2 Function Documentation

#### 7.27.2.1 clifford\_to\_repr()

```
template<typename Multivector_T >
String clifford_to_repr (
    const Multivector_T & mv ) [inline]
```

The "official" string representation of Multivector\_T mv.

Definition at line 87 of file PyClical.h.

Referenced by PyClical.clifford::\_\_repr\_\_().

#### 7.27.2.2 clifford\_to\_str()

```
template<typename Multivector_T >
String clifford_to_str (
    const Multivector_T & mv ) [inline]
```

The "informal" string representation of Multivector\_T mv.

Definition at line 98 of file PyClical.h.

References glucat::abs(), and PyClical::e().

Referenced by PyClical.clifford::\_\_str\_\_().



### 7.27.2.3 index\_set\_to\_repr()

```
template<typename Index_Set_T >
String index_set_to_repr (
    const Index_Set_T & ist ) [inline]
```

The "official" string representation of Index\_Set\_T ist.

Definition at line 69 of file PyClical.h.

References PyClical::ist.

Referenced by PyClical.index\_set::\_\_repr\_\_().

### 7.27.2.4 index\_set\_to\_str()

```
template<typename Index_Set_T >
String index_set_to_str (
    const Index_Set_T & ist ) [inline]
```

The "informal" string representation of Index\_Set\_T ist.

Definition at line 78 of file PyClical.h.

References PyClical::ist.

Referenced by PyClical.index\_set::\_\_str\_\_().

### 7.27.2.5 PyFloat\_FromDouble()

```
template<typename Scalar_T >
PyObject* PyFloat_FromDouble (
    Scalar_T v ) [inline]
```

Create a PyFloatObject object from Scalar\_T v. Needed because Scalar\_T might not be the same as double.

Definition at line 59 of file PyClical.h.

## 7.27.3 Variable Documentation

### 7.27.3.1 hi\_ndx

```
const index_t hi_ndx = DEFAULT_HI
```

Definition at line 156 of file PyClical.h.

### 7.27.3.2 lo\_ndx

```
const index_t lo_ndx = DEFAULT_LO
```

Definition at line 155 of file PyClical.h.

## 7.28 pyclical/PyClical.pxd File Reference

### Namespaces

- [PyClical](#)

## 7.29 pyclical/PyClical.pyx File Reference

### Classes

- class [PyClical.index\\_set](#)
- class [PyClical.index\\_set](#)
- class [PyClical.clifford](#)
- class [PyClical.clifford](#)

### Namespaces

- [PyClical](#)

### Functions

- def [PyClical.index\\_set\\_hidden\\_doctests](#) ()
- def [PyClical.clifford\\_hidden\\_doctests](#) ()
- def [PyClical.e](#) (obj)
- def [PyClical.istpq](#) (p, q)
- def [PyClical.\\_test](#) ()

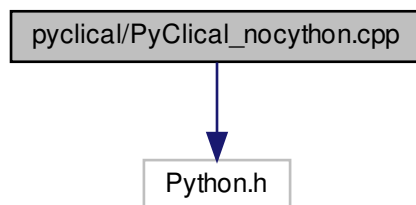
### Variables

- string [PyClical.\\_\\_version\\_\\_](#) = "0.8.2"
- [PyClical.obj](#)
- [PyClical.i](#)
- [PyClical.ixt](#)
- [PyClical.fill](#)
- float [PyClical.tau](#) = atan(clifford(1.0)) \* 8.0
- float [PyClical.pi](#) = tau / 2.0
- [PyClical.cl](#) = clifford
- [PyClical.ist](#) = index\_set
- def [PyClical.ninf3](#) = e(4) + e(-1)
- def [PyClical.nbar3](#) = e(4) - e(-1)

## 7.30 pyclical/PyClical\_nocython.cpp File Reference

```
#include "Python.h"
```

Include dependency graph for PyClical\_nocython.cpp:



### Macros

- `#define PY_SSIZE_T_CLEAN`

#### 7.30.1 Macro Definition Documentation

##### 7.30.1.1 PY\_SSIZE\_T\_CLEAN

```
#define PY_SSIZE_T_CLEAN
```

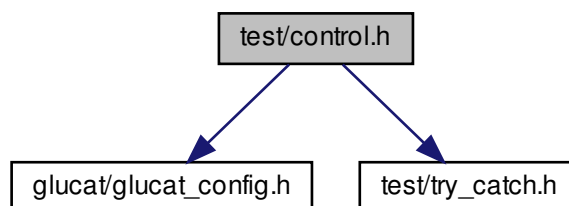
Definition at line 52 of file PyClical\_nocython.cpp.

## 7.31 test/control.h File Reference

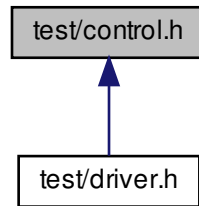
```
#include "glucat/glucat_config.h"
```

```
#include "test/try_catch.h"
```

Include dependency graph for control.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [glucat::control\\_t](#)  
*Parameters to control tests.*

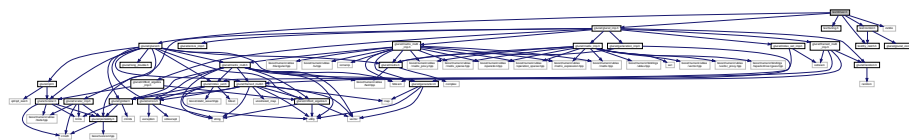
## Namespaces

- [glucat](#)

## 7.32 test/driver.h File Reference

```
#include "glucat/glucat.h"
#include "test/tuning.h"
#include "glucat/glucat_imp.h"
#include "test/try_catch.h"
#include "test/control.h"
#include <cstdio>
```

Include dependency graph for driver.h:



## 7.33 test/timing.h File Reference

### Namespaces

- [glucat](#)
- [glucat::timing](#)

## Functions

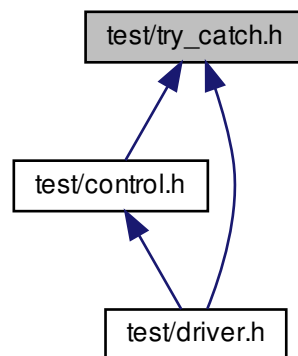
- static double `glucat::timing::elapsed` (clock\_t cpu\_time)  
*Elapsed time in milliseconds.*

## Variables

- const double `glucat::timing::MS_PER_SEC` = 1000.0  
*Timing constant: milliseconds per second.*
- const double `glucat::timing::MS_PER_CLOCK` = MS\_PER\_SEC / double(CLOCKS\_PER\_SEC)  
*Timing constant: milliseconds per clock.*
- const int `glucat::timing::EXTRA_TRIALS` = 2  
*Timing constant: trial expansion factor.*

## 7.34 test/try\_catch.h File Reference

This graph shows which files directly or indirectly include this file:



## Namespaces

- `glucat`

## Typedefs

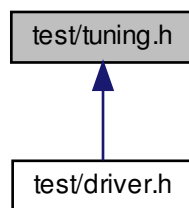
- typedef int(\* `glucat::intfn`) ()  
*For exception catching: pointer to function returning int.*
- typedef int(\* `glucat::intintfn`) (int)  
*For exception catching: pointer to function of int returning int.*

## Functions

- int [glucat::try\\_catch](#) (intfn f)  
*Exception catching for functions returning int.*
- int [glucat::try\\_catch](#) (intintfn f, int arg)  
*Exception catching for functions of int returning int.*

## 7.35 test/tuning.h File Reference

This graph shows which files directly or indirectly include this file:



## Macros

- `#define \_\_TEST\_TUNING\_DEFAULT\_CONSTANT(SUFFIX) const unsigned int Test_Tuning_##SUFFIX = glucat::DEFAULT_##SUFFIX`

## Typedefs

- typedef [glucat::precision\\_t](#) [precision\\_t](#)
- typedef [glucat::tuning](#) < Test\_Tuning\_Mult\_Matrix\_Threshold, Test\_Tuning\_Div\_Max\_Steps, Test\_Tuning\_Sqrt\_Max\_Steps, Test\_Tuning\_Log\_Max\_Outer\_Steps, Test\_Tuning\_Log\_Max\_Inner\_Steps, Test\_Tuning\_Basis\_Max\_Count, Test\_Tuning\_Fast\_Size\_Threshold, Test\_Tuning\_Inv\_Fast\_Dim\_Threshold, Test\_Tuning\_Products\_Size\_Threshold, [Test\\_Tuning\\_Function\\_Precision](#) > [Tune\\_P](#)  
*Tuning policy.*

## Functions

- [\\_GLUCAT\\_CTAssert](#) (std::numeric\_limits< unsigned int >::radix==2, CannotSetThresholds) const unsigned int Test\_Tuning\_Int\_Digits
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Mult\_Matrix\_Threshold)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Div\_Max\_Steps)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Sqrt\_Max\_Steps)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Log\_Max\_Outer\_Steps)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Log\_Max\_Inner\_Steps)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Basis\_Max\_Count)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Fast\_Size\_Threshold)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Inv\_Fast\_Dim\_Threshold)
- [\\_\\_TEST\\_TUNING\\_DEFAULT\\_CONSTANT](#) (Products\_Size\_Threshold)

## Variables

- const unsigned int `Test_Tuning_Max_Threshold` = 1 << `Test_Tuning_Int_Digits`
- const `precision_t` `Test_Tuning_Function_Precision` = `glucat::DEFAULT_Function_Precision`

## 7.35.1 Macro Definition Documentation

### 7.35.1.1 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT

```
#define __TEST_TUNING_DEFAULT_CONSTANT(  
    SUFFIX ) const unsigned int Test_Tuning_##SUFFIX = glucat::DEFAULT_##SUFFIX
```

Definition at line 41 of file tuning.h.

## 7.35.2 Typedef Documentation

### 7.35.2.1 precision\_t

```
typedef glucat::precision_t precision_t
```

Definition at line 39 of file tuning.h.

### 7.35.2.2 Tune\_P

```
typedef glucat::tuning< Test_Tuning_Mult_Matrix_Threshold, Test_Tuning_Div_Max_Steps, Test_↵  
_Tuning_Sqrt_Max_Steps, Test_Tuning_Log_Max_Outer_Steps, Test_Tuning_Log_Max_Inner_Steps,  
Test_Tuning_Basis_Max_Count, Test_Tuning_Fast_Size_Threshold, Test_Tuning_Inv_Fast_Dim_↵  
Threshold, Test_Tuning_Products_Size_Threshold, Test_Tuning_Function_Precision > Tune_P
```

Tuning policy.

Definition at line 126 of file tuning.h.

## 7.35.3 Function Documentation

**7.35.3.1 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [1/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Mult_Matrix_Threshold )
```

**7.35.3.2 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [2/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Div_Max_Steps )
```

**7.35.3.3 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [3/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Sqrt_Max_Steps )
```

**7.35.3.4 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [4/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Log_Max_Outer_Steps )
```

**7.35.3.5 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [5/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Log_Max_Inner_Steps )
```

**7.35.3.6 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [6/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Basis_Max_Count )
```

**7.35.3.7 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT()** [7/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Fast_Size_Threshold )
```



7.35.3.8 `__TEST_TUNING_DEFAULT_CONSTANT()` [8/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Inv_Fast_Dim_Threshold )
```

7.35.3.9 `__TEST_TUNING_DEFAULT_CONSTANT()` [9/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Products_Size_Threshold )
```

7.35.3.10 `_GLUCAT_CTAssert()`

```
_GLUCAT_CTAssert (
    std::numeric_limits< unsigned int >::radix  = =2,
    CannotSetThresholds ) const
```

## 7.35.4 Variable Documentation

7.35.4.1 `Test_Tuning_Function_Precision`

```
const precision_t Test_Tuning_Function_Precision = glucat::DEFAULT_Function_Precision
```

Definition at line 110 of file tuning.h.

7.35.4.2 `Test_Tuning_Max_Threshold`

```
const unsigned int Test_Tuning_Max_Threshold = 1 << Test_Tuning_Int_Digits
```

Definition at line 37 of file tuning.h.

## 7.36 test/undefine.h File Reference



# Index

`_GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS`  
    `clifford_algebra.h`, [238](#)  
`_GLUCAT_CTAssert`  
    `global.h`, [257](#)  
    `glucat`, [23](#)  
    `tuning.h`, [295](#)  
`_GLUCAT_HASH_N`  
    `framed_multi_imp.h`, [251](#)  
`_GLUCAT_HASH_SIZE_T`  
    `framed_multi_imp.h`, [252](#)  
`_GLUCAT_ISINF`  
    `portability.h`, [277](#)  
`_GLUCAT_ISNAN`  
    `portability.h`, [277](#)  
`_GLUCAT_MAP_IS_HASH`  
    `framed_multi.h`, [249](#)  
`_TEST_TUNING_DEFAULT_CONSTANT`  
    `tuning.h`, [293–295](#)  
`__add__`  
    `PyClical::clifford`, [84](#)  
`__and__`  
    `PyClical::clifford`, [85](#)  
    `PyClical::index_set`, [167](#)  
`__call__`  
    `PyClical::clifford`, [85](#)  
`__cinit__`  
    `PyClical::clifford`, [85](#)  
    `PyClical::index_set`, [167](#)  
`__contains__`  
    `PyClical::clifford`, [86](#)  
    `PyClical::index_set`, [167](#)  
`__dealloc__`  
    `PyClical::clifford`, [86](#)  
    `PyClical::index_set`, [168](#)  
`__div__`  
    `PyClical::clifford`, [86](#)  
`__getitem__`  
    `PyClical::clifford`, [87](#)  
    `PyClical::index_set`, [168](#)  
`__iadd__`  
    `PyClical::clifford`, [87](#)  
`__iand__`  
    `PyClical::clifford`, [87](#)  
    `PyClical::index_set`, [168](#)  
`__idiv__`  
    `PyClical::clifford`, [88](#)  
`__imod__`  
    `PyClical::clifford`, [88](#)  
`__imul__`  
    `PyClical::clifford`, [88](#)  
`__invert__`  
    `PyClical::index_set`, [169](#)  
`__ior__`  
    `PyClical::clifford`, [89](#)  
    `PyClical::index_set`, [169](#)  
`__isub__`  
    `PyClical::clifford`, [89](#)  
`__iter__`  
    `PyClical::clifford`, [89](#)  
    `PyClical::index_set`, [169](#)  
`__ixor__`  
    `PyClical::clifford`, [90](#)  
    `PyClical::index_set`, [170](#)  
`__mod__`  
    `PyClical::clifford`, [90](#)  
`__mul__`  
    `PyClical::clifford`, [90](#)  
`__neg__`  
    `PyClical::clifford`, [91](#)  
`__or__`  
    `PyClical::clifford`, [91](#)  
    `PyClical::index_set`, [170](#)  
`__pos__`  
    `PyClical::clifford`, [91](#)  
`__pow__`  
    `PyClical::clifford`, [92](#)  
`__repr__`  
    `PyClical::clifford`, [92](#)  
    `PyClical::index_set`, [170](#)  
`__richcmp__`  
    `PyClical::clifford`, [92](#)  
    `PyClical::index_set`, [171](#)  
`__setitem__`  
    `PyClical::index_set`, [171](#)  
`__str__`  
    `PyClical::clifford`, [93](#)  
    `PyClical::index_set`, [171](#)  
`__sub__`  
    `PyClical::clifford`, [93](#)  
`__version__`  
    `PyClical`, [76](#)  
`__xor__`  
    `PyClical::clifford`, [93](#)  
    `PyClical::index_set`, [172](#)  
`__test__`  
    `PyClical`, [72](#)  
`~basis_table`  
    `glucat::basis_table`, [80](#)

- ~clifford\_algebra
  - glucat::clifford\_algebra, 104
- ~control\_t
  - glucat::control\_t, 115
- ~framed\_multi
  - glucat::framed\_multi, 131
- ~generator\_table
  - glucat::gen::generator\_table, 143
- ~glucat\_error
  - glucat::glucat\_error, 147
- ~matrix\_multi
  - glucat::matrix\_multi, 183
- ~random\_generator
  - glucat::random\_generator, 211
- ~reference
  - glucat::index\_set::reference, 215
- ~var\_term
  - glucat::framed\_multi::var\_term, 229
- abs
  - glucat, 23
  - glucat::numeric\_traits, 198
  - PyClical::clifford, 94
- acos
  - glucat, 23, 24
  - glucat::numeric\_traits, 198
- acosh
  - glucat, 24
- agc3
  - cga3, 9
- asin
  - glucat, 25
  - glucat::numeric\_traits, 199
- asinh
  - glucat, 25, 26
- atan
  - glucat, 26
  - glucat::numeric\_traits, 199
- atanh
  - glucat, 27
- BITS\_PER\_SET\_VALUE
  - glucat, 60
- BOOST\_STATIC\_ASSERT
  - glucat::index\_set, 156
- basis
  - glucat::basis\_table, 81
- basis\_element
  - glucat::matrix\_multi, 188
- basis\_matrix\_t
  - glucat::matrix\_multi, 180
- basis\_table
  - glucat::basis\_table, 80
- bitset\_t
  - glucat::index\_set, 153
- call
  - glucat::control\_t, 115, 116
- cascade\_log
  - glucat, 27
- catch\_exceptions
  - glucat::control\_t, 116
- centre\_pm4\_qp4
  - glucat::framed\_multi, 135
- centre\_pp4\_qm4
  - glucat::framed\_multi, 135
- centre\_qp1\_pm1
  - glucat::framed\_multi, 135
- cga3, 9
  - agc3, 9
  - cga3, 9
  - cga3std, 10
- cga3std
  - cga3, 10
- check\_complex
  - glucat, 28
- cl
  - PyClical, 76
- classify\_eigenvalues
  - glucat::matrix, 66
- classname
  - glucat::clifford\_algebra, 104
  - glucat::error, 123
  - glucat::framed\_multi, 136
  - glucat::framed\_multi::var\_term, 230
  - glucat::glucat\_error, 148
  - glucat::index\_set, 156
  - glucat::matrix\_multi, 188
- Clifford
  - PyClical.h, 285
- clifford\_algebra.h
  - \_GLUCAT\_CLIFFORD\_ALGEBRA\_OPERATIO↵NS, 238
- clifford\_exp
  - glucat, 28
- clifford\_hidden\_doctests
  - PyClical, 72
- clifford\_to\_repr
  - PyClical.h, 286
- clifford\_to\_str
  - PyClical.h, 286
- compare
  - glucat, 28
  - glucat::index\_set, 164
- complexifier
  - glucat, 29
- conj
  - glucat, 29
  - glucat::clifford\_algebra, 104
  - glucat::numeric\_traits, 199
  - PyClical::clifford, 94
- const\_iterator
  - glucat::framed\_multi, 127
- control
  - glucat::control\_t, 116
- control\_t
  - glucat::control\_t, 115

- cos
  - glucat, [29, 30](#)
  - glucat::numeric\_traits, [199](#)
- cosh
  - glucat, [30](#)
  - glucat::numeric\_traits, [200](#)
- count
  - glucat::index\_set, [156](#)
  - PyClical::index\_set, [172](#)
- count\_neg
  - glucat::index\_set, [156](#)
  - PyClical::index\_set, [172](#)
- count\_pos
  - glucat::index\_set, [156](#)
  - PyClical::index\_set, [173](#)
- crd\_of\_mult
  - glucat, [30, 31](#)
- DEFAULT\_Basis\_Max\_Count
  - glucat, [60](#)
- DEFAULT\_Div\_Max\_Steps
  - glucat, [60](#)
- DEFAULT\_Fast\_Size\_Threshold
  - glucat, [61](#)
- DEFAULT\_Function\_Precision
  - glucat, [61](#)
- DEFAULT\_HI
  - glucat, [61](#)
- DEFAULT\_Inv\_Fast\_Dim\_Threshold
  - glucat, [61](#)
- DEFAULT\_Log\_Max\_Inner\_Steps
  - glucat, [61](#)
- DEFAULT\_Log\_Max\_Outer\_Steps
  - glucat, [61](#)
- DEFAULT\_Mult\_Matrix\_Threshold
  - glucat, [62](#)
- DEFAULT\_Products\_Size\_Threshold
  - glucat, [62](#)
- DEFAULT\_Sqrt\_Max\_Steps
  - glucat, [62](#)
- DEFAULT\_TRUNCATION
  - glucat, [62](#)
- db\_sqrt
  - glucat, [31](#)
- db\_step
  - glucat, [31](#)
- divide
  - glucat::framed\_multi, [136](#)
- e
  - PyClical, [74](#)
- EXTRA\_TRIALS
  - glucat::timing, [71](#)
- eig\_case\_t
  - glucat::matrix, [65](#)
- eigenvalues
  - glucat::matrix, [66](#)
- elapsed
  - glucat::timing, [71](#)
- elliptic
  - glucat, [31](#)
- error
  - glucat::error, [123](#)
- error\_t
  - glucat::framed\_multi, [128](#)
  - glucat::index\_set, [153](#)
  - glucat::matrix\_multi, [180](#)
- even
  - glucat, [32](#)
  - glucat::clifford\_algebra, [104](#)
  - PyClical::clifford, [94](#)
- exp
  - glucat, [32](#)
  - glucat::framed\_multi, [139](#)
  - glucat::numeric\_traits, [200](#)
- fast
  - glucat, [33](#)
  - glucat::framed\_multi, [136](#)
- fast\_framed\_multi
  - glucat::framed\_multi, [137](#)
  - glucat::matrix\_multi, [188](#)
- fast\_matrix\_multi
  - glucat::framed\_multi, [137](#)
  - glucat::matrix\_multi, [189](#)
- fill
  - PyClical, [76](#)
- flip
  - glucat::index\_set, [157](#)
  - glucat::index\_set::reference, [216](#)
- fmod
  - glucat::numeric\_traits, [200](#)
- fold
  - glucat::framed\_multi, [137](#)
  - glucat::index\_set, [157](#)
- folded\_dim
  - glucat, [33](#)
- frame
  - glucat::clifford\_algebra, [105](#)
  - PyClical::clifford, [95](#)
- framed\_multi
  - glucat::framed\_multi, [131–134, 139](#)
  - glucat::matrix\_multi, [190](#)
- framed\_multi.h
  - \_GLUCAT\_MAP\_IS\_HASH, [249](#)
- framed\_multi\_imp.h
  - \_GLUCAT\_HASH\_N, [251](#)
  - \_GLUCAT\_HASH\_SIZE\_T, [252](#)
- framed\_multi\_t
  - glucat::framed\_multi, [128](#)
  - glucat::matrix\_multi, [180](#)
- framed\_pair\_t
  - glucat::framed\_multi, [128](#)
- friend\_for\_private\_destructor
  - glucat::basis\_table, [81](#)
  - glucat::control\_t, [117](#)
  - glucat::gen::generator\_table, [146](#)
  - glucat::random\_generator, [212](#)

- function\_precision
  - glucat::tuning, 227
- GLUCAT\_HAVE\_INTTYPES\_H
  - glucat\_config.h, 259
- GLUCAT\_HAVE\_MEMORY\_H
  - glucat\_config.h, 259
- GLUCAT\_HAVE\_STDINT\_H
  - glucat\_config.h, 259
- GLUCAT\_HAVE\_STDLIB\_H
  - glucat\_config.h, 259
- GLUCAT\_HAVE\_STRING\_H
  - glucat\_config.h, 260
- GLUCAT\_HAVE\_STRINGS\_H
  - glucat\_config.h, 260
- GLUCAT\_HAVE\_SYS\_STAT\_H
  - glucat\_config.h, 260
- GLUCAT\_HAVE\_SYS\_TYPES\_H
  - glucat\_config.h, 260
- GLUCAT\_HAVE\_UNISTD\_H
  - glucat\_config.h, 260
- GLUCAT\_PACKAGE\_BUGREPORT
  - glucat\_config.h, 261
- GLUCAT\_PACKAGE\_NAME
  - glucat\_config.h, 261
- GLUCAT\_PACKAGE\_STRING
  - glucat\_config.h, 261
- GLUCAT\_PACKAGE\_TARNAME
  - glucat\_config.h, 261
- GLUCAT\_PACKAGE\_URL
  - glucat\_config.h, 261
- GLUCAT\_PACKAGE\_VERSION
  - glucat\_config.h, 261
- GLUCAT\_PACKAGE
  - glucat\_config.h, 260
- GLUCAT\_STDC\_HEADERS
  - glucat\_config.h, 262
- GLUCAT\_VERSION
  - glucat\_config.h, 262
- gen\_from\_pm1\_qm1
  - glucat::gen::generator\_table, 144
- gen\_from\_pm4\_qp4
  - glucat::gen::generator\_table, 144
- gen\_from\_pp4\_qm4
  - glucat::gen::generator\_table, 144
- gen\_from\_qp1\_pm1
  - glucat::gen::generator\_table, 144
- gen\_vector
  - glucat::gen::generator\_table, 145
- generator
  - glucat::gen::generator\_table, 145
  - glucat::random\_generator, 211
- generator\_table
  - glucat::gen::generator\_table, 143
- global.h
  - \_GLUCAT\_CTAssert, 257
- glucat, 10
  - \_GLUCAT\_CTAssert, 23
  - abs, 23
  - acos, 23, 24
  - acosh, 24
  - asin, 25
  - asinh, 25, 26
  - atan, 26
  - atanh, 27
  - BITS\_PER\_SET\_VALUE, 60
  - cascade\_log, 27
  - check\_complex, 28
  - clifford\_exp, 28
  - compare, 28
  - complexifier, 29
  - conj, 29
  - cos, 29, 30
  - cosh, 30
  - crd\_of\_mult, 30, 31
  - DEFAULT\_Basis\_Max\_Count, 60
  - DEFAULT\_Div\_Max\_Steps, 60
  - DEFAULT\_Fast\_Size\_Threshold, 61
  - DEFAULT\_Function\_Precision, 61
  - DEFAULT\_HI, 61
  - DEFAULT\_Inv\_Fast\_Dim\_Threshold, 61
  - DEFAULT\_Log\_Max\_Inner\_Steps, 61
  - DEFAULT\_Log\_Max\_Outer\_Steps, 61
  - DEFAULT\_Mult\_Matrix\_Threshold, 62
  - DEFAULT\_Products\_Size\_Threshold, 62
  - DEFAULT\_Sqrt\_Max\_Steps, 62
  - DEFAULT\_TRUNCATION, 62
  - db\_sqrt, 31
  - db\_step, 31
  - elliptic, 31
  - even, 32
  - exp, 32
  - fast, 33
  - folded\_dim, 33
  - imag, 33
  - index\_t, 21
  - intfn, 21
  - intintfn, 22
  - inv, 34
  - inverse\_gray, 34
  - inverse\_reversed\_gray, 34
  - involute, 34
  - I\_ln2, 62
  - I\_pi, 62
  - log, 35, 36
  - log2, 36
  - MS\_PER\_S, 63
  - matrix\_log, 36
  - matrix\_sqrt, 37
  - max\_abs, 37
  - max\_pos, 37
  - min\_neg, 38
  - norm, 38
  - odd, 38
  - offset\_level, 38
  - operator &, 39, 40
  - operator!=", 41

- operator<<, [47](#), [48](#)
- operator>>, [48](#), [49](#)
- operator\*, [42–44](#)
- operator<sup>^</sup>, [49](#), [50](#)
- operator+, [44](#), [45](#)
- operator-, [45](#)
- operator/, [46](#), [47](#)
- operator%, [41](#), [42](#)
- operator|, [50](#), [51](#)
- outer\_pow, [51](#)
- pade\_approx, [51](#)
- pade\_log, [51](#)
- pos\_mod, [52](#)
- pow, [52](#)
- precision\_t, [22](#)
- pure, [53](#)
- quad, [53](#)
- real, [53](#)
- reframe, [53](#)
- reverse, [54](#)
- scalar, [54](#)
- set\_value\_t, [22](#)
- sign\_of\_square, [54](#)
- sin, [55](#)
- sinh, [55](#)
- sqrt, [56](#), [57](#)
- star, [57](#)
- tan, [58](#)
- tanh, [58](#)
- to\_demote, [59](#)
- to\_promote, [59](#)
- try\_catch, [59](#)
- vector\_part, [60](#)
- glucat/clifford\_algebra.h, [231](#)
- glucat/clifford\_algebra\_imp.h, [238](#)
- glucat/errors.h, [245](#)
- glucat/errors\_imp.h, [246](#)
- glucat/framed\_multi.h, [246](#)
- glucat/framed\_multi\_imp.h, [249](#)
- glucat/generation.h, [252](#)
- glucat/generation\_imp.h, [254](#)
- glucat/global.h, [254](#)
- glucat/glucat.h, [257](#)
- glucat/glucat\_config.h, [258](#)
- glucat/glucat\_imp.h, [262](#)
- glucat/index\_set.h, [263](#)
- glucat/index\_set\_imp.h, [265](#)
- glucat/long\_double.h, [266](#)
- glucat/matrix.h, [267](#)
- glucat/matrix\_imp.h, [269](#)
- glucat/matrix\_multi.h, [271](#)
- glucat/matrix\_multi\_imp.h, [274](#)
- glucat/portability.h, [276](#)
- glucat/qd.h, [278](#)
- glucat/random.h, [279](#)
- glucat/scalar.h, [280](#)
- glucat/scalar\_imp.h, [282](#)
- glucat::CTAssertion< bool >, [118](#)
- glucat::CTAssertion< true >, [119](#)
- glucat::basis\_table
  - ~basis\_table, [80](#)
  - basis, [81](#)
  - basis\_table, [80](#)
  - friend\_for\_private\_destructor, [81](#)
  - operator=, [81](#)
- glucat::basis\_table< Scalar\_T, LO, HI, Matrix\_T >, [79](#)
- glucat::bool\_to\_type< truth\_value >, [82](#)
- glucat::clifford\_algebra
  - ~clifford\_algebra, [104](#)
  - classname, [104](#)
  - conj, [104](#)
  - even, [104](#)
  - frame, [105](#)
  - grade, [105](#)
  - index\_set\_t, [103](#)
  - inv, [105](#)
  - involute, [105](#)
  - isnan, [105](#)
  - max\_abs, [106](#)
  - multivector\_t, [103](#)
  - norm, [106](#)
  - odd, [106](#)
  - operator &=, [106](#)
  - operator\*=, [107](#)
  - operator<sup>^</sup>=, [109](#)
  - operator(), [107](#)
  - operator+=, [107](#)
  - operator-, [108](#)
  - operator-=, [108](#)
  - operator/=: [108](#)
  - operator==, [108](#), [109](#)
  - operator%=: [106](#)
  - operator[], [109](#)
  - operator|=: [109](#)
  - outer\_pow, [109](#)
  - pair\_t, [103](#)
  - pow, [110](#)
  - pure, [110](#)
  - quad, [110](#)
  - reverse, [110](#)
  - scalar, [110](#)
  - scalar\_t, [103](#)
  - truncated, [111](#)
  - vector\_part, [111](#)
  - vector\_t, [103](#)
  - write, [111](#)
- glucat::clifford\_algebra< Scalar\_T, Index\_Set\_↵  
T, Multivector\_T >, [101](#)
- glucat::compare\_types< LHS\_T, RHS\_T >, [112](#)
- glucat::compare\_types< T, T >, [113](#)
- glucat::control\_t, [113](#)
  - ~control\_t, [115](#)
  - call, [115](#), [116](#)
  - catch\_exceptions, [116](#)
  - control, [116](#)
  - control\_t, [115](#)

- friend\_for\_private\_destructor, 117
- m\_catch\_exceptions, 117
- m\_valid, 118
- m\_verbose\_output, 118
- operator=, 116
- valid, 117
- verbose, 117
- glucat::error
  - classname, 123
  - error, 123
  - heading, 123
  - print\_error\_msg, 124
- glucat::error< Class\_T >, 122
- glucat::framed\_multi
  - ~framed\_multi, 131
  - centre\_pm4\_qp4, 135
  - centre\_pp4\_qm4, 135
  - centre\_qp1\_pm1, 135
  - classname, 136
  - const\_iterator, 127
  - divide, 136
  - error\_t, 128
  - exp, 139
  - fast, 136
  - fast\_framed\_multi, 137
  - fast\_matrix\_multi, 137
  - fold, 137
  - framed\_multi, 131–134, 139
  - framed\_multi\_t, 128
  - framed\_pair\_t, 128
  - index\_set\_t, 128
  - iterator, 128
  - map\_t, 129
  - matrix\_multi, 139
  - matrix\_multi\_t, 129
  - matrix\_t, 129
  - multivector\_t, 129
  - nbr\_terms, 138
  - operator &, 139
  - operator<<, 140
  - operator>>, 141
  - operator\*, 140
  - operator^, 141
  - operator+=, 138
  - operator/, 140
  - operator%, 140
  - operator|, 141
  - random, 138
  - scalar\_t, 129
  - size\_type, 130
  - sorted\_map\_t, 130
  - star, 141
  - term\_t, 130
  - unfold, 138
  - var\_term\_t, 130
  - vector\_t, 130
- glucat::framed\_multi< Scalar\_T, LO, HI >, 124
- glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size←\_t, 149
- glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, 227
- glucat::framed\_multi::hash\_size\_t
  - hash\_size\_t, 149
  - n, 150
  - operator(), 149
- glucat::framed\_multi::var\_term
  - ~var\_term, 229
  - classname, 230
  - operator\*=, 230
  - var\_pair\_t, 229
  - var\_term, 229
- glucat::gen, 63
  - offset\_to\_super, 64
  - signature\_t, 63
- glucat::gen::generator\_table
  - ~generator\_table, 143
  - friend\_for\_private\_destructor, 146
  - gen\_from\_pm1\_qm1, 144
  - gen\_from\_pm4\_qp4, 144
  - gen\_from\_pp4\_qm4, 144
  - gen\_from\_qp1\_pm1, 144
  - gen\_vector, 145
  - generator, 145
  - generator\_table, 143
  - operator(), 145
  - operator=, 145
- glucat::gen::generator\_table< Matrix\_T >, 142
- glucat::glucat\_error, 146
  - ~glucat\_error, 147
  - classname, 148
  - glucat\_error, 147
  - heading, 148
  - name, 148
  - print\_error\_msg, 148
- glucat::index\_set
  - BOOST\_STATIC\_ASSERT, 156
  - bitset\_t, 153
  - classname, 156
  - compare, 164
  - count, 156
  - count\_neg, 156
  - count\_pos, 156
  - error\_t, 153
  - flip, 157
  - fold, 157
  - hash\_fn, 158
  - index\_pair\_t, 153
  - index\_set, 154, 155
  - index\_set\_t, 154
  - is\_contiguous, 158
  - lex\_less\_than, 158
  - max, 158
  - min, 159
  - operator &, 164
  - operator &=, 159



- operator!=, 159
- operator<, 159
- operator~, 161
- operator<sup>^</sup>, 164
- operator<sup>^</sup>==, 160
- operator==, 160
- operator[], 160
- operator|, 164
- operator|=, 161
- reference, 164
- reset, 161
- set, 162
- sign\_of\_mult, 162
- sign\_of\_square, 163
- test, 163
- unfold, 163
- v\_hi, 165
- v\_lo, 165
- value\_of\_fold, 163
- glucat::index\_set< LO, HI >, 150
- glucat::index\_set< LO, HI >::reference, 214
- glucat::index\_set::reference
  - ~reference, 215
  - flip, 216
  - index\_set, 217
  - m\_idx, 217
  - m\_pst, 217
  - operator bool, 216
  - operator~, 217
  - operator=, 216
  - reference, 215
- glucat::index\_set\_hash
  - index\_set\_t, 176
  - operator(), 176
- glucat::index\_set\_hash< LO, HI >, 175
- glucat::matrix, 64
  - classify\_eigenvalues, 66
  - eig\_case\_t, 65
  - eigenvalues, 66
  - inner, 66
  - isnan, 66
  - kron, 67
  - mono\_kron, 67
  - mono\_prod, 67
  - nnz, 67
  - nork, 68
  - nork\_range, 68
  - norm\_frob2, 68
  - prod, 69
  - signed\_perm\_nork, 69
  - sparse\_prod, 69
  - to\_lapack, 69
  - trace, 70
  - unit, 70
- glucat::matrix::eig\_genus
  - m\_eig\_case, 121
  - m\_safe\_arg, 121
  - Scalar\_T, 121
- glucat::matrix::eig\_genus< Matrix\_T >, 120
- glucat::matrix\_multi
  - ~matrix\_multi, 183
  - basis\_element, 188
  - basis\_matrix\_t, 180
  - classname, 188
  - error\_t, 180
  - fast\_framed\_multi, 188
  - fast\_matrix\_multi, 189
  - framed\_multi, 190
  - framed\_multi\_t, 180
  - index\_set\_t, 181
  - m\_frame, 193
  - m\_matrix, 193
  - matrix\_index\_t, 181
  - matrix\_log, 190
  - matrix\_multi, 183–187, 190
  - matrix\_multi\_t, 181
  - matrix\_sqrt, 190
  - matrix\_t, 181
  - multivector\_t, 181
  - operator &, 191
  - operator<<, 191, 192
  - operator>>, 192
  - operator\*, 191
  - operator<sup>^</sup>, 192
  - operator+=, 189
  - operator/, 191
  - operator=, 189
  - operator%, 191
  - operator|, 192
  - orientation\_t, 182
  - random, 189
  - reframe, 192
  - scalar\_t, 182
  - star, 193
  - term\_t, 182
  - vector\_t, 182
- glucat::matrix\_multi< Scalar\_T, LO, HI >, 177
- glucat::numeric\_traits
  - abs, 198
  - acos, 198
  - asin, 199
  - atan, 199
  - conj, 199
  - cos, 199
  - cosh, 200
  - exp, 200
  - fmod, 200
  - imag, 200
  - isInf, 201
  - isNaN\_or\_isInf, 202
  - isNaN, 201, 202
  - ln\_2, 202, 203
  - log, 203
  - log2, 203
  - NaN, 203
  - pi, 204

- pow, [204](#)
- real, [204](#)
- sin, [205](#)
- sinh, [205](#)
- sqrt, [205](#)
- tan, [205](#)
- tanh, [206](#)
- to\_double, [206](#)
- to\_int, [206](#)
- to\_scalar\_t, [206–208](#)
- glucat::numeric\_traits< Scalar\_T >, [196](#)
- glucat::numeric\_traits< Scalar\_T >::demoted<>, [119](#)
- glucat::numeric\_traits< Scalar\_T >::promoted, [209](#)
- glucat::numeric\_traits::demoted
  - type, [120](#)
- glucat::numeric\_traits::promoted
  - type, [209](#)
- glucat::random\_generator
  - ~random\_generator, [211](#)
  - friend\_for\_private\_destructor, [212](#)
  - generator, [211](#)
  - normal, [211](#)
  - normal\_dist, [212](#)
  - operator=, [212](#)
  - random\_generator, [211](#)
  - seed, [213](#)
  - uint\_gen, [213](#)
  - uniform, [212](#)
  - uniform\_dist, [213](#)
- glucat::random\_generator< Scalar\_T >, [210](#)
- glucat::sorted\_range
  - map\_t, [218](#)
  - sorted\_begin, [219](#)
  - sorted\_end, [220](#)
  - sorted\_iterator, [219](#)
  - sorted\_map\_t, [219](#)
  - sorted\_range, [219](#)
- glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, [218](#)
- glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_↵
  - T >, [220](#)
  - map\_t, [221](#)
  - sorted\_begin, [221](#)
  - sorted\_end, [222](#)
  - sorted\_iterator, [221](#)
  - sorted\_map\_t, [221](#)
  - sorted\_range, [221](#)
- glucat::timing, [70](#)
  - EXTRA\_TRIALS, [71](#)
  - elapsed, [71](#)
  - MS\_PER\_CLOCK, [71](#)
  - MS\_PER\_SEC, [71](#)
- glucat::tuning
  - function\_precision, [227](#)
- glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_↵
  - Steps, Sqrt\_Max\_Steps, Log\_Max\_Outer\_↵
  - Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_↵
  - \_Count, Fast\_Size\_Threshold, Inv\_Fast\_↵
  - Dim\_Threshold, Products\_Size\_Threshold,
- Function\_Precision >, [222](#)
- glucat\_config.h
  - GLUCAT\_HAVE\_INTTYPES\_H, [259](#)
  - GLUCAT\_HAVE\_MEMORY\_H, [259](#)
  - GLUCAT\_HAVE\_STDINT\_H, [259](#)
  - GLUCAT\_HAVE\_STDLIB\_H, [259](#)
  - GLUCAT\_HAVE\_STRING\_H, [260](#)
  - GLUCAT\_HAVE\_STRINGS\_H, [260](#)
  - GLUCAT\_HAVE\_SYS\_STAT\_H, [260](#)
  - GLUCAT\_HAVE\_SYS\_TYPES\_H, [260](#)
  - GLUCAT\_HAVE\_UNISTD\_H, [260](#)
  - GLUCAT\_PACKAGE\_BUGREPORT, [261](#)
  - GLUCAT\_PACKAGE\_NAME, [261](#)
  - GLUCAT\_PACKAGE\_STRING, [261](#)
  - GLUCAT\_PACKAGE\_TARNAME, [261](#)
  - GLUCAT\_PACKAGE\_URL, [261](#)
  - GLUCAT\_PACKAGE\_VERSION, [261](#)
  - GLUCAT\_PACKAGE, [260](#)
  - GLUCAT\_STDC\_HEADERS, [262](#)
  - GLUCAT\_VERSION, [262](#)
- glucat\_error
  - glucat::glucat\_error, [147](#)
- grade
  - glucat::clifford\_algebra, [105](#)
- hash\_fn
  - glucat::index\_set, [158](#)
  - PyClical::index\_set, [173](#)
- hash\_size\_t
  - glucat::framed\_multi::hash\_size\_t, [149](#)
- heading
  - glucat::error, [123](#)
  - glucat::glucat\_error, [148](#)
- hi\_ndx
  - PyClical.h, [287](#)
- i
  - PyClical, [76](#)
- imag
  - glucat, [33](#)
  - glucat::numeric\_traits, [200](#)
- index\_pair\_t
  - glucat::index\_set, [153](#)
- index\_set
  - glucat::index\_set, [154, 155](#)
  - glucat::index\_set::reference, [217](#)
- index\_set\_hidden\_doctests
  - PyClical, [74](#)
- index\_set\_t
  - glucat::clifford\_algebra, [103](#)
  - glucat::framed\_multi, [128](#)
  - glucat::index\_set, [154](#)
  - glucat::index\_set\_hash, [176](#)
  - glucat::matrix\_multi, [181](#)
- index\_set\_to\_repr
  - PyClical.h, [286](#)
- index\_set\_to\_str
  - PyClical.h, [287](#)
- index\_t

- glucat, 21
- IndexSet
  - PyClical.h, 285
- inner
  - glucat::matrix, 66
- instance
  - PyClical::clifford, 100
  - PyClical::index\_set, 175
- intfn
  - glucat, 21
- intintfn
  - glucat, 22
- inv
  - glucat, 34
  - glucat::clifford\_algebra, 105
  - PyClical::clifford, 95
- inverse\_gray
  - glucat, 34
- inverse\_reversed\_gray
  - glucat, 34
- involute
  - glucat, 34
  - glucat::clifford\_algebra, 105
  - PyClical::clifford, 95
- is\_contiguous
  - glucat::index\_set, 158
- isInf
  - glucat::numeric\_traits, 201
- isNaN\_or\_isInf
  - glucat::numeric\_traits, 202
- isNaN
  - glucat::numeric\_traits, 201, 202
- isnan
  - glucat::clifford\_algebra, 105
  - glucat::matrix, 66
  - PyClical::clifford, 96
- ist
  - PyClical, 77
- istpq
  - PyClical, 75
- iterator
  - glucat::framed\_multi, 128
- ixt
  - PyClical, 77
- kron
  - glucat::matrix, 67
- l\_in2
  - glucat, 62
- l\_pi
  - glucat, 62
- lex\_less\_than
  - glucat::index\_set, 158
- ln\_2
  - glucat::numeric\_traits, 202, 203
- lo\_ndx
  - PyClical.h, 287
- log
  - glucat, 35, 36
  - glucat::numeric\_traits, 203
- log2
  - glucat, 36
  - glucat::numeric\_traits, 203
- m\_catch\_exceptions
  - glucat::control\_t, 117
- m\_eig\_case
  - glucat::matrix::eig\_genus, 121
- m\_frame
  - glucat::matrix\_multi, 193
- m\_idx
  - glucat::index\_set::reference, 217
- m\_matrix
  - glucat::matrix\_multi, 193
- m\_pst
  - glucat::index\_set::reference, 217
- m\_safe\_arg
  - glucat::matrix::eig\_genus, 121
- m\_valid
  - glucat::control\_t, 118
- m\_verbose\_output
  - glucat::control\_t, 118
- MS\_PER\_CLOCK
  - glucat::timing, 71
- MS\_PER\_SEC
  - glucat::timing, 71
- MS\_PER\_S
  - glucat, 63
- map\_t
  - glucat::framed\_multi, 129
  - glucat::sorted\_range, 218
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_↔  
Map\_T >, 221
- matrix\_index\_t
  - glucat::matrix\_multi, 181
- matrix\_log
  - glucat, 36
  - glucat::matrix\_multi, 190
- matrix\_multi
  - glucat::framed\_multi, 139
  - glucat::matrix\_multi, 183–187, 190
- matrix\_multi\_t
  - glucat::framed\_multi, 129
  - glucat::matrix\_multi, 181
- matrix\_sqrt
  - glucat, 37
  - glucat::matrix\_multi, 190
- matrix\_t
  - glucat::framed\_multi, 129
  - glucat::matrix\_multi, 181
- max
  - glucat::index\_set, 158
  - PyClical::index\_set, 173
- max\_abs
  - glucat, 37
  - glucat::clifford\_algebra, 106
  - PyClical::clifford, 96

- max\_pos
  - glucat, 37
- min
  - glucat::index\_set, 159
  - PyClical::index\_set, 174
- min\_neg
  - glucat, 38
- mono\_kron
  - glucat::matrix, 67
- mono\_prod
  - glucat::matrix, 67
- multivector\_t
  - glucat::clifford\_algebra, 103
  - glucat::framed\_multi, 129
  - glucat::matrix\_multi, 181
- n
  - glucat::framed\_multi::hash\_size\_t, 150
- name
  - glucat::glucat\_error, 148
- NaN
  - glucat::numeric\_traits, 203
- nbar3
  - PyClical, 77
- nbr\_terms
  - glucat::framed\_multi, 138
- ninf3
  - PyClical, 77
- nnz
  - glucat::matrix, 67
- nork
  - glucat::matrix, 68
- nork\_range
  - glucat::matrix, 68
- norm
  - glucat, 38
  - glucat::clifford\_algebra, 106
  - PyClical::clifford, 96
- norm\_frob2
  - glucat::matrix, 68
- normal
  - glucat::random\_generator, 211
- normal\_dist
  - glucat::random\_generator, 212
- obj
  - PyClical, 77
- odd
  - glucat, 38
  - glucat::clifford\_algebra, 106
  - PyClical::clifford, 97
- offset\_level
  - glucat, 38
- offset\_to\_super
  - glucat::gen, 64
- operator &
  - glucat, 39, 40
  - glucat::framed\_multi, 139
  - glucat::index\_set, 164
  - glucat::matrix\_multi, 191
- operator &=
  - glucat::clifford\_algebra, 106
  - glucat::index\_set, 159
- operator bool
  - glucat::index\_set::reference, 216
- operator!=
  - glucat, 41
  - glucat::index\_set, 159
- operator<
  - glucat::index\_set, 159
- operator<<
  - glucat, 47, 48
  - glucat::framed\_multi, 140
  - glucat::matrix\_multi, 191, 192
- operator>>
  - glucat, 48, 49
  - glucat::framed\_multi, 141
  - glucat::matrix\_multi, 192
- operator\*
  - glucat, 42–44
  - glucat::framed\_multi, 140
  - glucat::matrix\_multi, 191
- operator\*=
  - glucat::clifford\_algebra, 107
  - glucat::framed\_multi::var\_term, 230
- operator~
  - glucat::index\_set, 161
  - glucat::index\_set::reference, 217
- operator^
  - glucat, 49, 50
  - glucat::framed\_multi, 141
  - glucat::index\_set, 164
  - glucat::matrix\_multi, 192
- operator^=
  - glucat::clifford\_algebra, 109
  - glucat::index\_set, 160
- operator()
  - glucat::clifford\_algebra, 107
  - glucat::framed\_multi::hash\_size\_t, 149
  - glucat::gen::generator\_table, 145
  - glucat::index\_set\_hash, 176
- operator+
  - glucat, 44, 45
- operator+=
  - glucat::clifford\_algebra, 107
  - glucat::framed\_multi, 138
  - glucat::matrix\_multi, 189
- operator-
  - glucat, 45
  - glucat::clifford\_algebra, 108
- operator-=
  - glucat::clifford\_algebra, 108
- operator/
  - glucat, 46, 47
  - glucat::framed\_multi, 140
  - glucat::matrix\_multi, 191
- operator/=
  - glucat::matrix\_multi, 191

- glucat::clifford\_algebra, 108
- operator=
  - glucat::basis\_table, 81
  - glucat::control\_t, 116
  - glucat::gen::generator\_table, 145
  - glucat::index\_set::reference, 216
  - glucat::matrix\_multi, 189
  - glucat::random\_generator, 212
- operator==
  - glucat::clifford\_algebra, 108, 109
  - glucat::index\_set, 160
- operator%
  - glucat, 41, 42
  - glucat::framed\_multi, 140
  - glucat::matrix\_multi, 191
- operator%=
  - glucat::clifford\_algebra, 106
- operator[]
  - glucat::clifford\_algebra, 109
  - glucat::index\_set, 160
- operator |
  - glucat, 50, 51
  - glucat::framed\_multi, 141
  - glucat::index\_set, 164
  - glucat::matrix\_multi, 192
- operator | =
  - glucat::clifford\_algebra, 109
  - glucat::index\_set, 161
- orientation\_t
  - glucat::matrix\_multi, 182
- outer\_pow
  - glucat, 51
  - glucat::clifford\_algebra, 109
  - PyClical::clifford, 97
- PY\_SSIZE\_T\_CLEAN
  - PyClical.cpp, 283
  - PyClical\_nocython.cpp, 289
- pade\_approx
  - glucat, 51
- pade\_log
  - glucat, 51
- pair\_t
  - glucat::clifford\_algebra, 103
- pi
  - glucat::numeric\_traits, 204
  - PyClical, 78
- portability.h
  - \_GLUCAT\_ISINF, 277
  - \_GLUCAT\_ISNAN, 277
  - UBLAS\_ABS, 277
  - UBLAS\_SQRT, 278
- pos\_mod
  - glucat, 52
- pow
  - glucat, 52
  - glucat::clifford\_algebra, 110
  - glucat::numeric\_traits, 204
  - PyClical::clifford, 97
- precision\_t
  - glucat, 22
  - tuning.h, 293
- print\_error\_msg
  - glucat::error, 124
  - glucat::glucat\_error, 148
- prod
  - glucat::matrix, 69
- pure
  - glucat, 53
  - glucat::clifford\_algebra, 110
  - PyClical::clifford, 98
- PyClical, 72
  - \_\_version\_\_, 76
  - \_test, 72
  - cl, 76
  - clifford\_hidden\_doctests, 72
  - e, 74
  - fill, 76
  - i, 76
  - index\_set\_hidden\_doctests, 74
  - ist, 77
  - istpq, 75
  - ixt, 77
  - nbar3, 77
  - ninf3, 77
  - obj, 77
  - pi, 78
  - tau, 78
- PyClical.clifford, 83
- PyClical.cpp
  - PY\_SSIZE\_T\_CLEAN, 283
- PyClical.h
  - Clifford, 285
  - clifford\_to\_repr, 286
  - clifford\_to\_str, 286
  - hi\_ndx, 287
  - index\_set\_to\_repr, 286
  - index\_set\_to\_str, 287
  - IndexSet, 285
  - lo\_ndx, 287
  - PyFloat\_FromDouble, 287
  - scalar\_t, 285
  - String, 285
  - Tune\_P, 286
- PyClical.index\_set, 165
- PyClical::clifford
  - \_\_add\_\_, 84
  - \_\_and\_\_, 85
  - \_\_call\_\_, 85
  - \_\_cinit\_\_, 85
  - \_\_contains\_\_, 86
  - \_\_dealloc\_\_, 86
  - \_\_div\_\_, 86
  - \_\_getitem\_\_, 87
  - \_\_iadd\_\_, 87
  - \_\_iand\_\_, 87
  - \_\_idiv\_\_, 88

- `__imod__`, 88
- `__imul__`, 88
- `__ior__`, 89
- `__isub__`, 89
- `__iter__`, 89
- `__ixor__`, 90
- `__mod__`, 90
- `__mul__`, 90
- `__neg__`, 91
- `__or__`, 91
- `__pos__`, 91
- `__pow__`, 92
- `__repr__`, 92
- `__richcmp__`, 92
- `__str__`, 93
- `__sub__`, 93
- `__xor__`, 93
- `abs`, 94
- `conj`, 94
- `even`, 94
- `frame`, 95
- `instance`, 100
- `inv`, 95
- `involute`, 95
- `isnan`, 96
- `max_abs`, 96
- `norm`, 96
- `odd`, 97
- `outer_pow`, 97
- `pow`, 97
- `pure`, 98
- `quad`, 98
- `reframe`, 98
- `reverse`, 99
- `scalar`, 99
- `truncated`, 99
- `vector_part`, 100
- `PyClical::index_set`
  - `__and__`, 167
  - `__cinit__`, 167
  - `__contains__`, 167
  - `__dealloc__`, 168
  - `__getitem__`, 168
  - `__iand__`, 168
  - `__invert__`, 169
  - `__ior__`, 169
  - `__iter__`, 169
  - `__ixor__`, 170
  - `__or__`, 170
  - `__repr__`, 170
  - `__richcmp__`, 171
  - `__setitem__`, 171
  - `__str__`, 171
  - `__xor__`, 172
- `count`, 172
- `count_neg`, 172
- `count_pos`, 173
- `hash_fn`, 173
- `instance`, 175
- `max`, 173
- `min`, 174
- `sign_of_mult`, 174
- `sign_of_square`, 174
- `PyClical_nocython.cpp`
  - `PY_SSIZE_T_CLEAN`, 289
- `PyFloat_FromDouble`
  - `PyClical.h`, 287
- `pyclical/PyClical.cpp`, 283
- `pyclical/PyClical.h`, 284
- `pyclical/PyClical.pxd`, 288
- `pyclical/PyClical.pyx`, 288
- `pyclical/PyClical_nocython.cpp`, 289
- `pyclical/glucat.pxd`, 283
- `quad`
  - `glucat`, 53
  - `glucat::clifford_algebra`, 110
  - `PyClical::clifford`, 98
- `random`
  - `glucat::framed_multi`, 138
  - `glucat::matrix_multi`, 189
- `random_generator`
  - `glucat::random_generator`, 211
- `real`
  - `glucat`, 53
  - `glucat::numeric_traits`, 204
- `reference`
  - `glucat::index_set`, 164
  - `glucat::index_set::reference`, 215
- `reframe`
  - `glucat`, 53
  - `glucat::matrix_multi`, 192
  - `PyClical::clifford`, 98
- `reset`
  - `glucat::index_set`, 161
- `reverse`
  - `glucat`, 54
  - `glucat::clifford_algebra`, 110
  - `PyClical::clifford`, 99
- `scalar`
  - `glucat`, 54
  - `glucat::clifford_algebra`, 110
  - `PyClical::clifford`, 99
- `Scalar_T`
  - `glucat::matrix::eig_genus`, 121
- `scalar_t`
  - `glucat::clifford_algebra`, 103
  - `glucat::framed_multi`, 129
  - `glucat::matrix_multi`, 182
  - `PyClical.h`, 285
- `seed`
  - `glucat::random_generator`, 213
- `set`
  - `glucat::index_set`, 162
- `set_value_t`

- glucat, [22](#)
- sign\_of\_mult
  - glucat::index\_set, [162](#)
  - PyClical::index\_set, [174](#)
- sign\_of\_square
  - glucat, [54](#)
  - glucat::index\_set, [163](#)
  - PyClical::index\_set, [174](#)
- signature\_t
  - glucat::gen, [63](#)
- signed\_perm\_nork
  - glucat::matrix, [69](#)
- sin
  - glucat, [55](#)
  - glucat::numeric\_traits, [205](#)
- sinh
  - glucat, [55](#)
  - glucat::numeric\_traits, [205](#)
- size\_type
  - glucat::framed\_multi, [130](#)
- sorted\_begin
  - glucat::sorted\_range, [219](#)
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, [221](#)
- sorted\_end
  - glucat::sorted\_range, [220](#)
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, [222](#)
- sorted\_iterator
  - glucat::sorted\_range, [219](#)
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, [221](#)
- sorted\_map\_t
  - glucat::framed\_multi, [130](#)
  - glucat::sorted\_range, [219](#)
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, [221](#)
- sorted\_range
  - glucat::sorted\_range, [219](#)
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, [221](#)
- sparse\_prod
  - glucat::matrix, [69](#)
- sqrt
  - glucat, [56](#), [57](#)
  - glucat::numeric\_traits, [205](#)
- star
  - glucat, [57](#)
  - glucat::framed\_multi, [141](#)
  - glucat::matrix\_multi, [193](#)
- std, [78](#)
- std::numeric\_limits< glucat::framed\_multi< Scalar\_T, LO, HI > >, [194](#)
- std::numeric\_limits< glucat::matrix\_multi< Scalar\_T, LO, HI > >, [195](#)
- String
  - PyClical.h, [285](#)
- tan
  - glucat, [58](#)
  - glucat::numeric\_traits, [205](#)
- tanh
  - glucat, [58](#)
  - glucat::numeric\_traits, [206](#)
- tau
  - PyClical, [78](#)
- term\_t
  - glucat::framed\_multi, [130](#)
  - glucat::matrix\_multi, [182](#)
- test
  - glucat::index\_set, [163](#)
- test/control.h, [289](#)
- test/driver.h, [290](#)
- test/timing.h, [290](#)
- test/try\_catch.h, [291](#)
- test/tuning.h, [292](#)
- test/undefine.h, [295](#)
- Test\_Tuning\_Function\_Precision
  - tuning.h, [295](#)
- Test\_Tuning\_Max\_Threshold
  - tuning.h, [295](#)
- to\_demote
  - glucat, [59](#)
- to\_double
  - glucat::numeric\_traits, [206](#)
- to\_int
  - glucat::numeric\_traits, [206](#)
- to\_lapack
  - glucat::matrix, [69](#)
- to\_promote
  - glucat, [59](#)
- to\_scalar\_t
  - glucat::numeric\_traits, [206–208](#)
- trace
  - glucat::matrix, [70](#)
- truncated
  - glucat::clifford\_algebra, [111](#)
  - PyClical::clifford, [99](#)
- try\_catch
  - glucat, [59](#)
- Tune\_P
  - PyClical.h, [286](#)
  - tuning.h, [293](#)
- tuning.h
  - \_GLUCAT\_CTAssert, [295](#)
  - \_\_TEST\_TUNING\_DEFAULT\_CONSTANT, [293–295](#)
  - precision\_t, [293](#)
  - Test\_Tuning\_Function\_Precision, [295](#)
  - Test\_Tuning\_Max\_Threshold, [295](#)
  - Tune\_P, [293](#)
- type
  - glucat::numeric\_traits::demoted, [120](#)
  - glucat::numeric\_traits::promoted, [209](#)
- UBLAS\_ABS
  - portability.h, [277](#)
- UBLAS\_SQRT

- portability.h, [278](#)
- uint\_gen
  - glucat::random\_generator, [213](#)
- unfold
  - glucat::framed\_multi, [138](#)
  - glucat::index\_set, [163](#)
- uniform
  - glucat::random\_generator, [212](#)
- uniform\_dist
  - glucat::random\_generator, [213](#)
- unit
  - glucat::matrix, [70](#)
- v\_hi
  - glucat::index\_set, [165](#)
- v\_lo
  - glucat::index\_set, [165](#)
- valid
  - glucat::control\_t, [117](#)
- value\_of\_fold
  - glucat::index\_set, [163](#)
- var\_pair\_t
  - glucat::framed\_multi::var\_term, [229](#)
- var\_term
  - glucat::framed\_multi::var\_term, [229](#)
- var\_term\_t
  - glucat::framed\_multi, [130](#)
- vector\_part
  - glucat, [60](#)
  - glucat::clifford\_algebra, [111](#)
  - PyClical::clifford, [100](#)
- vector\_t
  - glucat::clifford\_algebra, [103](#)
  - glucat::framed\_multi, [130](#)
  - glucat::matrix\_multi, [182](#)
- verbose
  - glucat::control\_t, [117](#)
- write
  - glucat::clifford\_algebra, [111](#)