

glucat

0.8.4

Generated by Doxygen 1.8.18



<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List	1
<b>2 Hierarchical Index</b>	<b>3</b>
2.1 Class Hierarchy	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List	5
<b>4 File Index</b>	<b>7</b>
4.1 File List	7
<b>5 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	21
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	22
5.2.3.1 _GLUCAT_CTAssert() [1/3]	22
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]	23
5.2.3.6 acos() [2/2]	24
5.2.3.7 acosh() [1/2]	24
5.2.3.8 acosh() [2/2]	24
5.2.3.9 asin() [1/2]	25
5.2.3.10 asin() [2/2]	25
5.2.3.11 asinh() [1/2]	25
5.2.3.12 asinh() [2/2]	26
5.2.3.13 atan() [1/2]	26
5.2.3.14 atan() [2/2]	26
5.2.3.15 atanh() [1/2]	26
5.2.3.16 atanh() [2/2]	27

5.2.3.17 cascade_log()	27
5.2.3.18 check_complex()	27
5.2.3.19 clifford_exp()	28
5.2.3.20 compare()	28
5.2.3.21 complexifier()	28
5.2.3.22 conj()	28
5.2.3.23 cos() $[1/2]$	29
5.2.3.24 cos() $[2/2]$	29
5.2.3.25 cosh()	29
5.2.3.26 crd_of_mult() $[1/2]$	30
5.2.3.27 crd_of_mult() $[2/2]$	30
5.2.3.28 db_sqrt()	30
5.2.3.29 db_step()	30
5.2.3.30 elliptic()	31
5.2.3.31 even()	31
5.2.3.32 exp() $[1/2]$	31
5.2.3.33 exp() $[2/2]$	31
5.2.3.34 fast()	32
5.2.3.35 folded_dim()	32
5.2.3.36 imag()	32
5.2.3.37 inv()	32
5.2.3.38 inverse_gray()	33
5.2.3.39 inverse_reversed_gray()	33
5.2.3.40 involute()	33
5.2.3.41 log() $[1/4]$	33
5.2.3.42 log() $[2/4]$	34
5.2.3.43 log() $[3/4]$	34
5.2.3.44 log() $[4/4]$	34
5.2.3.45 log2()	35
5.2.3.46 matrix_log()	35
5.2.3.47 matrix_sqrt()	35
5.2.3.48 max_abs()	36
5.2.3.49 max_pos()	36
5.2.3.50 min_neg()	36
5.2.3.51 norm()	36
5.2.3.52 odd()	37
5.2.3.53 offset_level()	37
5.2.3.54 operator"!=() $[1/3]$	37
5.2.3.55 operator"!=() $[2/3]$	37
5.2.3.56 operator"!=() $[3/3]$	38
5.2.3.57 operator%() $[1/3]$	38
5.2.3.58 operator%() $[2/3]$	38

5.2.3.59 operator%() [3/3] . . . . .	38
5.2.3.60 operator&() [1/4] . . . . .	39
5.2.3.61 operator&() [2/4] . . . . .	39
5.2.3.62 operator&() [3/4] . . . . .	39
5.2.3.63 operator&() [4/4] . . . . .	39
5.2.3.64 operator*() [1/6] . . . . .	40
5.2.3.65 operator*() [2/6] . . . . .	40
5.2.3.66 operator*() [3/6] . . . . .	40
5.2.3.67 operator*() [4/6] . . . . .	40
5.2.3.68 operator*() [5/6] . . . . .	41
5.2.3.69 operator*() [6/6] . . . . .	41
5.2.3.70 operator+() [1/3] . . . . .	41
5.2.3.71 operator+() [2/3] . . . . .	41
5.2.3.72 operator+() [3/3] . . . . .	42
5.2.3.73 operator-() [1/3] . . . . .	42
5.2.3.74 operator-() [2/3] . . . . .	42
5.2.3.75 operator-() [3/3] . . . . .	42
5.2.3.76 operator/() [1/5] . . . . .	43
5.2.3.77 operator/() [2/5] . . . . .	43
5.2.3.78 operator/() [3/5] . . . . .	43
5.2.3.79 operator/() [4/5] . . . . .	43
5.2.3.80 operator/() [5/5] . . . . .	44
5.2.3.81 operator<<() [1/4] . . . . .	44
5.2.3.82 operator<<() [2/4] . . . . .	44
5.2.3.83 operator<<() [3/4] . . . . .	44
5.2.3.84 operator<<() [4/4] . . . . .	45
5.2.3.85 operator>>() [1/3] . . . . .	45
5.2.3.86 operator>>() [2/3] . . . . .	45
5.2.3.87 operator>>() [3/3] . . . . .	45
5.2.3.88 operator^() [1/4] . . . . .	46
5.2.3.89 operator^() [2/4] . . . . .	46
5.2.3.90 operator^() [3/4] . . . . .	46
5.2.3.91 operator^() [4/4] . . . . .	46
5.2.3.92 operator"   () [1/4] . . . . .	47
5.2.3.93 operator"   () [2/4] . . . . .	47
5.2.3.94 operator"   () [3/4] . . . . .	47
5.2.3.95 operator"   () [4/4] . . . . .	47
5.2.3.96 outer_pow() . . . . .	48
5.2.3.97 pade_approx() . . . . .	48
5.2.3.98 pade_log() . . . . .	48
5.2.3.99 pos_mod() . . . . .	48
5.2.3.100 pow() [1/2] . . . . .	49

5.2.3.101 pow() [2/2]	49
5.2.3.102 pure()	49
5.2.3.103 quad()	49
5.2.3.104 real()	50
5.2.3.105 reframe()	50
5.2.3.106 reverse()	50
5.2.3.107 scalar()	50
5.2.3.108 sign_of_square()	51
5.2.3.109 sin() [1/2]	51
5.2.3.110 sin() [2/2]	51
5.2.3.111 sinh()	52
5.2.3.112 sqrt() [1/4]	52
5.2.3.113 sqrt() [2/4]	52
5.2.3.114 sqrt() [3/4]	53
5.2.3.115 sqrt() [4/4]	53
5.2.3.116 star() [1/3]	53
5.2.3.117 star() [2/3]	53
5.2.3.118 star() [3/3]	54
5.2.3.119 tan() [1/2]	54
5.2.3.120 tan() [2/2]	54
5.2.3.121 tanh()	55
5.2.3.122 to_demote()	55
5.2.3.123 to_promote()	55
5.2.3.124 try_catch() [1/2]	55
5.2.3.125 try_catch() [2/2]	56
5.2.3.126 vector_part()	56
5.2.4 Variable Documentation	56
5.2.4.1 BITS_PER_SET_VALUE	56
5.2.4.2 DEFAULT_Basis_Max_Count	56
5.2.4.3 DEFAULT_Div_Max_Steps	56
5.2.4.4 DEFAULT_Fast_Size_Threshold	57
5.2.4.5 DEFAULT_Function_Precision	57
5.2.4.6 DEFAULT_HI	57
5.2.4.7 DEFAULT_Inv_Fast_Dim_Threshold	57
5.2.4.8 DEFAULT_Log_Max_Inner_Steps	57
5.2.4.9 DEFAULT_Log_Max_Outer_Steps	57
5.2.4.10 DEFAULT_Mult_Matrix_Threshold	58
5.2.4.11 DEFAULT_Products_Size_Threshold	58
5.2.4.12 DEFAULT_Sqrt_Max_Steps	58
5.2.4.13 DEFAULT_TRUNCATION	58
5.2.4.14 I_ln2	58
5.2.4.15 I_pi	58

5.2.4.16 MS_PER_S	59
5.3 glucat::gen Namespace Reference	59
5.3.1 Typedef Documentation	59
5.3.1.1 signature_t	59
5.3.2 Variable Documentation	59
5.3.2.1 offset_to_super	60
5.4 glucat::matrix Namespace Reference	60
5.4.1 Enumeration Type Documentation	61
5.4.1.1 eig_case_t	61
5.4.2 Function Documentation	62
5.4.2.1 classify_eigenvalues()	62
5.4.2.2 eigenvalues()	62
5.4.2.3 inner()	62
5.4.2.4 isnan()	63
5.4.2.5 kron()	63
5.4.2.6 mono_kron()	63
5.4.2.7 mono_prod()	63
5.4.2.8 nnz()	64
5.4.2.9 nork()	64
5.4.2.10 nork_range()	64
5.4.2.11 norm_frob2()	64
5.4.2.12 prod()	65
5.4.2.13 signed_perm_nork()	65
5.4.2.14 sparse_prod()	65
5.4.2.15 to_lapack()	65
5.4.2.16 trace()	66
5.4.2.17 unit()	66
5.5 glucat::timing Namespace Reference	66
5.5.1 Function Documentation	66
5.5.1.1 elapsed()	66
5.5.2 Variable Documentation	67
5.5.2.1 EXTRA_TRIALS	67
5.5.2.2 MS_PER_CLOCK	67
5.5.2.3 MS_PER_SEC	67
5.6 PyClical Namespace Reference	67
5.6.1 Function Documentation	68
5.6.1.1 _test()	68
5.6.1.2 clifford_hidden_doctests()	68
5.6.1.3 e()	69
5.6.1.4 index_set_hidden_doctests()	70
5.6.1.5 istpq()	71
5.6.2 Variable Documentation	71

5.6.2.1 <code>__version__</code>	71
5.6.2.2 <code>cl</code>	72
5.6.2.3 <code>fill</code>	72
5.6.2.4 <code>i</code>	72
5.6.2.5 <code>ist</code>	72
5.6.2.6 <code>ixt</code>	72
5.6.2.7 <code>nbar3</code>	73
5.6.2.8 <code>ninf3</code>	73
5.6.2.9 <code>obj</code>	73
5.6.2.10 <code>pi</code>	73
5.6.2.11 <code>scalar_epsilon</code>	73
5.6.2.12 <code>tau</code>	73
5.7 std Namespace Reference	73
<b>6 Class Documentation</b>	<b>75</b>
6.1 <code>glucat::basis_table&lt; Scalar_T, LO, HI, Matrix_T &gt;</code> Class Template Reference	75
6.1.1 Detailed Description	76
6.1.2 Constructor & Destructor Documentation	76
6.1.2.1 <code>basis_table()</code> [1/2]	76
6.1.2.2 <code>~basis_table()</code>	76
6.1.2.3 <code>basis_table()</code> [2/2]	76
6.1.3 Member Function Documentation	76
6.1.3.1 <code>basis()</code>	77
6.1.3.2 <code>operator=()</code>	77
6.1.4 Friends And Related Function Documentation	77
6.1.4.1 <code>friend_for_private_destructor</code>	77
6.2 <code>glucat::bool_to_type&lt; truth_value &gt;</code> Class Template Reference	77
6.2.1 Detailed Description	78
6.2.2 Member Enumeration Documentation	78
6.2.2.1 anonymous enum	78
6.3 <code>PyClical.clifford</code> Class Reference	78
6.3.1 Detailed Description	80
6.3.2 Member Function Documentation	80
6.3.2.1 <code>__add__()</code>	80
6.3.2.2 <code>__and__()</code>	80
6.3.2.3 <code>__call__()</code>	81
6.3.2.4 <code>__cinit__()</code>	81
6.3.2.5 <code>__contains__()</code>	82
6.3.2.6 <code>__dealloc__()</code>	82
6.3.2.7 <code>__getitem__()</code>	82
6.3.2.8 <code>__iadd__()</code>	83
6.3.2.9 <code>__iand__()</code>	83



6.3.2.10	<a href="#">__idiv__()</a>	83
6.3.2.11	<a href="#">__imod__()</a>	84
6.3.2.12	<a href="#">__imul__()</a>	84
6.3.2.13	<a href="#">__ior__()</a>	84
6.3.2.14	<a href="#">__isub__()</a>	85
6.3.2.15	<a href="#">__iter__()</a>	85
6.3.2.16	<a href="#">__ixor__()</a>	85
6.3.2.17	<a href="#">__mod__()</a>	86
6.3.2.18	<a href="#">__mul__()</a>	86
6.3.2.19	<a href="#">__neg__()</a>	86
6.3.2.20	<a href="#">__or__()</a>	87
6.3.2.21	<a href="#">__pos__()</a>	87
6.3.2.22	<a href="#">__pow__()</a>	87
6.3.2.23	<a href="#">__repr__()</a>	88
6.3.2.24	<a href="#">__richcmp__()</a>	88
6.3.2.25	<a href="#">__str__()</a>	88
6.3.2.26	<a href="#">__sub__()</a>	89
6.3.2.27	<a href="#">__truediv__()</a>	89
6.3.2.28	<a href="#">__xor__()</a>	89
6.3.2.29	<a href="#">abs()</a>	90
6.3.2.30	<a href="#">conj()</a>	90
6.3.2.31	<a href="#">even()</a>	90
6.3.2.32	<a href="#">frame()</a>	91
6.3.2.33	<a href="#">inv()</a>	91
6.3.2.34	<a href="#">involute()</a>	91
6.3.2.35	<a href="#">isnan()</a>	92
6.3.2.36	<a href="#">max_abs()</a>	92
6.3.2.37	<a href="#">norm()</a>	92
6.3.2.38	<a href="#">odd()</a>	93
6.3.2.39	<a href="#">outer_pow()</a>	93
6.3.2.40	<a href="#">pow()</a>	93
6.3.2.41	<a href="#">pure()</a>	94
6.3.2.42	<a href="#">quad()</a>	94
6.3.2.43	<a href="#">reframe()</a>	94
6.3.2.44	<a href="#">reverse()</a>	95
6.3.2.45	<a href="#">scalar()</a>	95
6.3.2.46	<a href="#">truncated()</a>	95
6.3.2.47	<a href="#">vector_part()</a>	96
6.3.3	<a href="#">Member Data Documentation</a>	96
6.3.3.1	<a href="#">instance</a>	96
6.4	<a href="#">glucat::clifford_algebra&lt; Scalar_T, Index_Set_T, Multivector_T &gt; Class Template Reference</a>	96
6.4.1	<a href="#">Detailed Description</a>	98

6.4.2 Member Typedef Documentation	98
6.4.2.1 index_set_t	99
6.4.2.2 multivector_t	99
6.4.2.3 pair_t	99
6.4.2.4 scalar_t	99
6.4.2.5 vector_t	99
6.4.3 Constructor & Destructor Documentation	99
6.4.3.1 ~clifford_algebra()	100
6.4.4 Member Function Documentation	100
6.4.4.1 classname()	100
6.4.4.2 conj()	100
6.4.4.3 even()	100
6.4.4.4 frame()	100
6.4.4.5 grade()	101
6.4.4.6 inv()	101
6.4.4.7 involute()	101
6.4.4.8 isnan()	101
6.4.4.9 max_abs()	101
6.4.4.10 norm()	102
6.4.4.11 odd()	102
6.4.4.12 operator%=( )	102
6.4.4.13 operator&=( )	102
6.4.4.14 operator()( )	102
6.4.4.15 operator*=( ) [1/2]	103
6.4.4.16 operator*=( ) [2/2]	103
6.4.4.17 operator+=( ) [1/2]	103
6.4.4.18 operator+=( ) [2/2]	103
6.4.4.19 operator-( )	103
6.4.4.20 operator-=( )	104
6.4.4.21 operator/=( ) [1/2]	104
6.4.4.22 operator/=( ) [2/2]	104
6.4.4.23 operator==( ) [1/2]	104
6.4.4.24 operator==( ) [2/2]	104
6.4.4.25 operator[]( )	105
6.4.4.26 operator^=( )	105
6.4.4.27 operator"   =( )	105
6.4.4.28 outer_pow()	105
6.4.4.29 pow()	105
6.4.4.30 pure()	106
6.4.4.31 quad()	106
6.4.4.32 reverse()	106
6.4.4.33 scalar()	106

6.4.4.34 truncated()	106
6.4.4.35 vector_part() [1/2]	107
6.4.4.36 vector_part() [2/2]	107
6.4.4.37 write() [1/2]	107
6.4.4.38 write() [2/2]	107
6.5 glucat::compare_types< LHS_T, RHS_T > Class Template Reference	107
6.5.1 Detailed Description	108
6.5.2 Member Enumeration Documentation	108
6.5.2.1 anonymous enum	108
6.6 glucat::compare_types< T, T > Class Template Reference	108
6.6.1 Detailed Description	108
6.6.2 Member Enumeration Documentation	109
6.6.2.1 anonymous enum	109
6.7 glucat::control_t Class Reference	109
6.7.1 Detailed Description	110
6.7.2 Constructor & Destructor Documentation	110
6.7.2.1 control_t() [1/3]	110
6.7.2.2 control_t() [2/3]	110
6.7.2.3 ~control_t()	111
6.7.2.4 control_t() [3/3]	111
6.7.3 Member Function Documentation	111
6.7.3.1 call() [1/2]	111
6.7.3.2 call() [2/2]	111
6.7.3.3 catch_exceptions()	111
6.7.3.4 control()	112
6.7.3.5 operator=()	112
6.7.3.6 valid()	112
6.7.3.7 verbose()	112
6.7.4 Friends And Related Function Documentation	112
6.7.4.1 friend_for_private_destructor	113
6.7.5 Member Data Documentation	113
6.7.5.1 m_catch_exceptions	113
6.7.5.2 m_valid	113
6.7.5.3 m_verbose_output	113
6.8 glucat::CTAssertion< bool > Struct Template Reference	114
6.8.1 Detailed Description	114
6.9 glucat::CTAssertion< true > Struct Reference	114
6.9.1 Detailed Description	114
6.10 glucat::numeric_traits< Scalar_T >::demoted Struct Reference	114
6.10.1 Detailed Description	115
6.10.2 Member Typedef Documentation	115
6.10.2.1 type [1/2]	115

6.10.2.2 type [2/2]	115
6.11 glucat::matrix::eig_genus< Matrix_T > Struct Template Reference	115
6.11.1 Detailed Description	116
6.11.2 Member Typedef Documentation	116
6.11.2.1 Scalar_T	116
6.11.3 Member Data Documentation	116
6.11.3.1 m_eig_case	116
6.11.3.2 m_safe_arg	117
6.12 glucat::error< Class_T > Class Template Reference	117
6.12.1 Detailed Description	118
6.12.2 Constructor & Destructor Documentation	118
6.12.2.1 error() [1/2]	118
6.12.2.2 error() [2/2]	118
6.12.3 Member Function Documentation	118
6.12.3.1 classname()	119
6.12.3.2 heading()	119
6.12.3.3 print_error_msg()	119
6.13 glucat::framed_multi< Scalar_T, LO, HI > Class Template Reference	119
6.13.1 Detailed Description	122
6.13.2 Member Typedef Documentation	122
6.13.2.1 const_iterator	122
6.13.2.2 error_t	122
6.13.2.3 framed_multi_t	122
6.13.2.4 framed_pair_t	122
6.13.2.5 index_set_t	123
6.13.2.6 iterator	123
6.13.2.7 map_t	123
6.13.2.8 matrix_multi_t	123
6.13.2.9 matrix_t	123
6.13.2.10 multivector_t	124
6.13.2.11 scalar_t	124
6.13.2.12 size_type	124
6.13.2.13 sorted_map_t	124
6.13.2.14 term_t	124
6.13.2.15 var_term_t	125
6.13.2.16 vector_t	125
6.13.3 Constructor & Destructor Documentation	125
6.13.3.1 ~framed_multi()	125
6.13.3.2 framed_multi() [1/15]	125
6.13.3.3 framed_multi() [2/15]	126
6.13.3.4 framed_multi() [3/15]	126
6.13.3.5 framed_multi() [4/15]	126

6.13.3.6 framed_multi() [5/15]	126
6.13.3.7 framed_multi() [6/15]	127
6.13.3.8 framed_multi() [7/15]	127
6.13.3.9 framed_multi() [8/15]	127
6.13.3.10 framed_multi() [9/15]	127
6.13.3.11 framed_multi() [10/15]	128
6.13.3.12 framed_multi() [11/15]	128
6.13.3.13 framed_multi() [12/15]	128
6.13.3.14 framed_multi() [13/15]	128
6.13.3.15 framed_multi() [14/15]	129
6.13.3.16 framed_multi() [15/15]	129
6.13.4 Member Function Documentation	129
6.13.4.1 centre_pm4_qp4()	129
6.13.4.2 centre_pp4_qm4()	129
6.13.4.3 centre_qp1_pm1()	130
6.13.4.4 classname()	130
6.13.4.5 divide()	130
6.13.4.6 fast()	130
6.13.4.7 fast_framed_multi()	131
6.13.4.8 fast_matrix_multi()	131
6.13.4.9 fold()	131
6.13.4.10 nbr_terms()	131
6.13.4.11 operator+=()	132
6.13.4.12 random()	132
6.13.4.13 unfold()	132
6.13.5 Friends And Related Function Documentation	132
6.13.5.1 exp	132
6.13.5.2 framed_multi	133
6.13.5.3 matrix_multi	133
6.13.5.4 operator%	133
6.13.5.5 operator&	133
6.13.5.6 operator*	133
6.13.5.7 operator/	134
6.13.5.8 operator<< [1/2]	134
6.13.5.9 operator<< [2/2]	134
6.13.5.10 operator>>	134
6.13.5.11 operator^	134
6.13.5.12 operator"	135
6.13.5.13 star	135
6.14 glucat::gen::generator_table< Matrix_T > Class Template Reference	135
6.14.1 Detailed Description	137
6.14.2 Constructor & Destructor Documentation	137

6.14.2.1 generator_table() [1/2]	137
6.14.2.2 ~generator_table()	137
6.14.2.3 generator_table() [2/2]	137
6.14.3 Member Function Documentation	137
6.14.3.1 gen_from_pm1_qm1()	138
6.14.3.2 gen_from_pm4_qp4()	138
6.14.3.3 gen_from_pp4_qm4()	138
6.14.3.4 gen_from_qp1_pm1()	138
6.14.3.5 gen_vector()	139
6.14.3.6 generator()	139
6.14.3.7 operator>()	139
6.14.3.8 operator=()	139
6.14.4 Friends And Related Function Documentation	139
6.14.4.1 friend_for_private_destructor	140
6.15 glucat::glucat_error Class Reference	140
6.15.1 Detailed Description	141
6.15.2 Constructor & Destructor Documentation	141
6.15.2.1 glucat_error()	141
6.15.2.2 ~glucat_error()	141
6.15.3 Member Function Documentation	141
6.15.3.1 classname()	142
6.15.3.2 heading()	142
6.15.3.3 print_error_msg()	142
6.15.4 Member Data Documentation	142
6.15.4.1 name	142
6.16 glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t Class Reference	142
6.16.1 Detailed Description	143
6.16.2 Constructor & Destructor Documentation	143
6.16.2.1 hash_size_t()	143
6.16.3 Member Function Documentation	143
6.16.3.1 operator>()	143
6.16.4 Member Data Documentation	143
6.16.4.1 n	144
6.17 glucat::index_set< LO, HI > Class Template Reference	144
6.17.1 Detailed Description	146
6.17.2 Member Typedef Documentation	146
6.17.2.1 bitset_t	147
6.17.2.2 error_t	147
6.17.2.3 index_pair_t	147
6.17.2.4 index_set_t	147
6.17.3 Constructor & Destructor Documentation	147
6.17.3.1 index_set() [1/6]	147

6.17.3.2 <code>index_set()</code> [2/6]	148
6.17.3.3 <code>index_set()</code> [3/6]	148
6.17.3.4 <code>index_set()</code> [4/6]	148
6.17.3.5 <code>index_set()</code> [5/6]	148
6.17.3.6 <code>index_set()</code> [6/6]	149
6.17.4 Member Function Documentation	149
6.17.4.1 <code>BOOST_STATIC_ASSERT()</code>	149
6.17.4.2 <code>classname()</code>	149
6.17.4.3 <code>count()</code>	149
6.17.4.4 <code>count_neg()</code>	150
6.17.4.5 <code>count_pos()</code>	150
6.17.4.6 <code>flip()</code> [1/2]	150
6.17.4.7 <code>flip()</code> [2/2]	150
6.17.4.8 <code>fold()</code> [1/2]	151
6.17.4.9 <code>fold()</code> [2/2]	151
6.17.4.10 <code>hash_fn()</code>	151
6.17.4.11 <code>is_contiguous()</code>	151
6.17.4.12 <code>lex_less_than()</code>	152
6.17.4.13 <code>max()</code>	152
6.17.4.14 <code>min()</code>	152
6.17.4.15 <code>operator!=()</code>	152
6.17.4.16 <code>operator&amp;=()</code>	153
6.17.4.17 <code>operator&lt;()</code>	153
6.17.4.18 <code>operator==()</code>	153
6.17.4.19 <code>operator[]()</code> [1/2]	153
6.17.4.20 <code>operator[]()</code> [2/2]	154
6.17.4.21 <code>operator^=()</code>	154
6.17.4.22 <code>operator"  =()</code>	154
6.17.4.23 <code>operator~()</code>	154
6.17.4.24 <code>reset()</code> [1/2]	155
6.17.4.25 <code>reset()</code> [2/2]	155
6.17.4.26 <code>set()</code> [1/3]	155
6.17.4.27 <code>set()</code> [2/3]	155
6.17.4.28 <code>set()</code> [3/3]	156
6.17.4.29 <code>sign_of_mult()</code>	156
6.17.4.30 <code>sign_of_square()</code>	156
6.17.4.31 <code>test()</code>	156
6.17.4.32 <code>unfold()</code>	157
6.17.4.33 <code>value_of_fold()</code>	157
6.17.5 Friends And Related Function Documentation	157
6.17.5.1 <code>compare</code>	157
6.17.5.2 <code>operator&amp;</code>	157

6.17.5.3 operator^	158
6.17.5.4 operator"	158
6.17.5.5 reference	158
6.17.6 Member Data Documentation	158
6.17.6.1 v_hi	158
6.17.6.2 v_lo	158
6.18 PyClical.index_set Class Reference	159
6.18.1 Detailed Description	160
6.18.2 Member Function Documentation	160
6.18.2.1 __and__()	160
6.18.2.2 __cinit__()	160
6.18.2.3 __contains__()	161
6.18.2.4 __dealloc__()	161
6.18.2.5 __getitem__()	161
6.18.2.6 __iand__()	162
6.18.2.7 __invert__()	162
6.18.2.8 __ior__()	162
6.18.2.9 __iter__()	163
6.18.2.10 __ixor__()	163
6.18.2.11 __or__()	163
6.18.2.12 __repr__()	164
6.18.2.13 __richcmp__()	164
6.18.2.14 __setitem__()	164
6.18.2.15 __str__()	165
6.18.2.16 __xor__()	165
6.18.2.17 count()	165
6.18.2.18 count_neg()	166
6.18.2.19 count_pos()	166
6.18.2.20 hash_fn()	166
6.18.2.21 max()	167
6.18.2.22 min()	167
6.18.2.23 sign_of_mult()	167
6.18.2.24 sign_of_square()	168
6.18.3 Member Data Documentation	168
6.18.3.1 instance	168
6.19 glucat::index_set_hash< LO, HI > Class Template Reference	168
6.19.1 Detailed Description	169
6.19.2 Member Typedef Documentation	169
6.19.2.1 index_set_t	169
6.19.3 Member Function Documentation	169
6.19.3.1 operator()()	169
6.20 glucat::matrix_multi< Scalar_T, LO, HI > Class Template Reference	170



6.20.1 Detailed Description	172
6.20.2 Member Typedef Documentation	173
6.20.2.1 basis_matrix_t	173
6.20.2.2 error_t	173
6.20.2.3 framed_multi_t	173
6.20.2.4 index_set_t	173
6.20.2.5 matrix_index_t	174
6.20.2.6 matrix_multi_t	174
6.20.2.7 matrix_t	174
6.20.2.8 multivector_t	174
6.20.2.9 orientation_t	174
6.20.2.10 scalar_t	175
6.20.2.11 term_t	175
6.20.2.12 vector_t	175
6.20.3 Constructor & Destructor Documentation	175
6.20.3.1 ~matrix_multi()	175
6.20.3.2 matrix_multi() [1/17]	176
6.20.3.3 matrix_multi() [2/17]	176
6.20.3.4 matrix_multi() [3/17]	176
6.20.3.5 matrix_multi() [4/17]	177
6.20.3.6 matrix_multi() [5/17]	177
6.20.3.7 matrix_multi() [6/17]	177
6.20.3.8 matrix_multi() [7/17]	177
6.20.3.9 matrix_multi() [8/17]	178
6.20.3.10 matrix_multi() [9/17]	178
6.20.3.11 matrix_multi() [10/17]	178
6.20.3.12 matrix_multi() [11/17]	178
6.20.3.13 matrix_multi() [12/17]	179
6.20.3.14 matrix_multi() [13/17]	179
6.20.3.15 matrix_multi() [14/17]	179
6.20.3.16 matrix_multi() [15/17]	179
6.20.3.17 matrix_multi() [16/17]	180
6.20.3.18 matrix_multi() [17/17]	180
6.20.4 Member Function Documentation	180
6.20.4.1 basis_element()	180
6.20.4.2 classname()	180
6.20.4.3 fast_framed_multi()	181
6.20.4.4 fast_matrix_multi()	181
6.20.4.5 operator+=()	181
6.20.4.6 operator=()	181
6.20.4.7 random()	182
6.20.5 Friends And Related Function Documentation	182

6.20.5.1 framed_multi . . . . .	182
6.20.5.2 matrix_log . . . . .	182
6.20.5.3 matrix_multi . . . . .	182
6.20.5.4 matrix_sqrt . . . . .	183
6.20.5.5 operator% . . . . .	183
6.20.5.6 operator& . . . . .	183
6.20.5.7 operator* . . . . .	183
6.20.5.8 operator/ . . . . .	183
6.20.5.9 operator<< [1/2] . . . . .	184
6.20.5.10 operator<< [2/2] . . . . .	184
6.20.5.11 operator>> . . . . .	184
6.20.5.12 operator^ . . . . .	184
6.20.5.13 operator"   . . . . .	184
6.20.5.14 reframe . . . . .	185
6.20.5.15 star . . . . .	185
6.20.6 Member Data Documentation . . . . .	185
6.20.6.1 m_frame . . . . .	185
6.20.6.2 m_matrix . . . . .	185
6.21 std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > > Struct Template Reference . . . . .	186
6.21.1 Detailed Description . . . . .	186
6.22 std::numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > > Struct Template Reference . . . . .	187
6.22.1 Detailed Description . . . . .	187
6.23 glucat::numeric_traits< Scalar_T > Class Template Reference . . . . .	188
6.23.1 Detailed Description . . . . .	190
6.23.2 Member Function Documentation . . . . .	190
6.23.2.1 abs() . . . . .	190
6.23.2.2 acos() . . . . .	190
6.23.2.3 asin() . . . . .	191
6.23.2.4 atan() . . . . .	191
6.23.2.5 conj() . . . . .	191
6.23.2.6 cos() . . . . .	191
6.23.2.7 cosh() . . . . .	192
6.23.2.8 exp() . . . . .	192
6.23.2.9 fmod() . . . . .	192
6.23.2.10 imag() . . . . .	192
6.23.2.11 isInf() [1/3] . . . . .	193
6.23.2.12 isInf() [2/3] . . . . .	193
6.23.2.13 isInf() [3/3] . . . . .	193
6.23.2.14 isNaN() [1/3] . . . . .	193
6.23.2.15 isNaN() [2/3] . . . . .	194
6.23.2.16 isNaN() [3/3] . . . . .	194
6.23.2.17 isNaN_or_isInf() . . . . .	194

6.23.2.18 <a href="#">ln_2()</a> [1/2]	194
6.23.2.19 <a href="#">ln_2()</a> [2/2]	195
6.23.2.20 <a href="#">log()</a>	195
6.23.2.21 <a href="#">log2()</a>	195
6.23.2.22 <a href="#">NaN()</a>	195
6.23.2.23 <a href="#">pi()</a> [1/2]	196
6.23.2.24 <a href="#">pi()</a> [2/2]	196
6.23.2.25 <a href="#">pow()</a>	196
6.23.2.26 <a href="#">real()</a>	196
6.23.2.27 <a href="#">sin()</a>	197
6.23.2.28 <a href="#">sinh()</a>	197
6.23.2.29 <a href="#">sqrt()</a>	197
6.23.2.30 <a href="#">tan()</a>	197
6.23.2.31 <a href="#">tanh()</a>	198
6.23.2.32 <a href="#">to_double()</a>	198
6.23.2.33 <a href="#">to_int()</a>	198
6.23.2.34 <a href="#">to_scalar_t()</a> [1/9]	198
6.23.2.35 <a href="#">to_scalar_t()</a> [2/9]	199
6.23.2.36 <a href="#">to_scalar_t()</a> [3/9]	199
6.23.2.37 <a href="#">to_scalar_t()</a> [4/9]	199
6.23.2.38 <a href="#">to_scalar_t()</a> [5/9]	199
6.23.2.39 <a href="#">to_scalar_t()</a> [6/9]	200
6.23.2.40 <a href="#">to_scalar_t()</a> [7/9]	200
6.23.2.41 <a href="#">to_scalar_t()</a> [8/9]	200
6.23.2.42 <a href="#">to_scalar_t()</a> [9/9]	200
6.24 <a href="#">glucat::numeric_traits&lt; Scalar_T &gt;::promoted Struct Reference</a>	201
6.24.1 Detailed Description	201
6.24.2 Member Typedef Documentation	201
6.24.2.1 type	201
6.25 <a href="#">glucat::random_generator&lt; Scalar_T &gt; Class Template Reference</a>	201
6.25.1 Detailed Description	202
6.25.2 Constructor & Destructor Documentation	202
6.25.2.1 <a href="#">random_generator()</a> [1/2]	202
6.25.2.2 <a href="#">random_generator()</a> [2/2]	203
6.25.2.3 <a href="#">~random_generator()</a>	203
6.25.3 Member Function Documentation	203
6.25.3.1 <a href="#">generator()</a>	203
6.25.3.2 <a href="#">normal()</a>	203
6.25.3.3 <a href="#">operator=()</a>	203
6.25.3.4 <a href="#">uniform()</a>	204
6.25.4 Friends And Related Function Documentation	204
6.25.4.1 <a href="#">friend_for_private_destructor</a>	204

6.25.5 Member Data Documentation	204
6.25.5.1 normal_dist	204
6.25.5.2 seed	204
6.25.5.3 uint_gen	205
6.25.5.4 uniform_dist	205
6.26 glucat::index_set< LO, HI >::reference Class Reference	205
6.26.1 Detailed Description	206
6.26.2 Constructor & Destructor Documentation	206
6.26.2.1 reference() [1/2]	207
6.26.2.2 reference() [2/2]	207
6.26.2.3 ~reference()	207
6.26.3 Member Function Documentation	207
6.26.3.1 flip()	207
6.26.3.2 operator bool()	208
6.26.3.3 operator=() [1/2]	208
6.26.3.4 operator=() [2/2]	208
6.26.3.5 operator~()	208
6.26.4 Friends And Related Function Documentation	208
6.26.4.1 index_set	209
6.26.5 Member Data Documentation	209
6.26.5.1 m_idx	209
6.26.5.2 m_pst	209
6.27 glucat::sorted_range< Map_T, Sorted_Map_T > Class Template Reference	209
6.27.1 Detailed Description	210
6.27.2 Member Typedef Documentation	210
6.27.2.1 map_t	210
6.27.2.2 sorted_iterator	210
6.27.2.3 sorted_map_t	210
6.27.3 Constructor & Destructor Documentation	211
6.27.3.1 sorted_range()	211
6.27.4 Member Data Documentation	211
6.27.4.1 sorted_begin	211
6.27.4.2 sorted_end	211
6.28 glucat::sorted_range< Sorted_Map_T, Sorted_Map_T > Class Template Reference	211
6.28.1 Detailed Description	212
6.28.2 Member Typedef Documentation	212
6.28.2.1 map_t	212
6.28.2.2 sorted_iterator	212
6.28.2.3 sorted_map_t	212
6.28.3 Constructor & Destructor Documentation	213
6.28.3.1 sorted_range()	213
6.28.4 Member Data Documentation	213

6.28.4.1 sorted_begin . . . . .	213
6.28.4.2 sorted_end . . . . .	213
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 . . . . .	213
6.29.1 Detailed Description . . . . .	214
6.29.2 Member Enumeration Documentation . . . . .	214
6.29.2.1 anonymous enum . . . . .	215
6.29.2.2 anonymous enum . . . . .	216
6.29.2.3 anonymous enum . . . . .	216
6.29.2.4 anonymous enum . . . . .	217
6.29.2.5 anonymous enum . . . . .	217
6.29.2.6 anonymous enum . . . . .	217
6.29.2.7 anonymous enum . . . . .	219
6.29.2.8 anonymous enum . . . . .	219
6.29.2.9 anonymous enum . . . . .	220
6.29.3 Member Data Documentation . . . . .	220
6.29.3.1 function_precision . . . . .	220
6.30 glucat::framed_multi< Scalar_T, LO, HI >::var_term Class Reference . . . . .	221
6.30.1 Detailed Description . . . . .	222
6.30.2 Member Typedef Documentation . . . . .	222
6.30.2.1 var_pair_t . . . . .	222
6.30.3 Constructor & Destructor Documentation . . . . .	222
6.30.3.1 ~var_term() . . . . .	222
6.30.3.2 var_term() [1/2] . . . . .	223
6.30.3.3 var_term() [2/2] . . . . .	223
6.30.4 Member Function Documentation . . . . .	223
6.30.4.1 classname() . . . . .	223
6.30.4.2 operator*=( ) . . . . .	223
<b>7 File Documentation</b>	<b>225</b>
7.1 glucat/clifford_algebra.h File Reference . . . . .	225
7.1.1 Macro Definition Documentation . . . . .	231
7.1.1.1 _GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS . . . . .	231
7.2 glucat/clifford_algebra_imp.h File Reference . . . . .	232
7.3 glucat/errors.h File Reference . . . . .	238
7.4 glucat/errors_imp.h File Reference . . . . .	239
7.5 glucat/framed_multi.h File Reference . . . . .	239
7.5.1 Macro Definition Documentation . . . . .	241
7.5.1.1 _GLUCAT_MAP_IS_HASH . . . . .	242
7.6 glucat/framed_multi_imp.h File Reference . . . . .	242
7.6.1 Macro Definition Documentation . . . . .	244
7.6.1.1 _GLUCAT_HASH_N . . . . .	244

7.6.1.2 <code>_GLUCAT_HASH_SIZE_T</code>	244
7.7 <code>glucat/generation.h</code> File Reference	244
7.8 <code>glucat/generation_imp.h</code> File Reference	245
7.9 <code>glucat/global.h</code> File Reference	246
7.9.1 Macro Definition Documentation	248
7.9.1.1 <code>_GLUCAT_CTAssert</code>	248
7.10 <code>glucat/glucat.h</code> File Reference	248
7.11 <code>glucat/glucat_config.h</code> File Reference	249
7.11.1 Macro Definition Documentation	250
7.11.1.1 <code>GLUCAT_HAVE_INTTYPES_H</code>	250
7.11.1.2 <code>GLUCAT_HAVE_MEMORY_H</code>	250
7.11.1.3 <code>GLUCAT_HAVE_STDINT_H</code>	250
7.11.1.4 <code>GLUCAT_HAVE_STDLIB_H</code>	250
7.11.1.5 <code>GLUCAT_HAVE_STRING_H</code>	251
7.11.1.6 <code>GLUCAT_HAVE_STRINGS_H</code>	251
7.11.1.7 <code>GLUCAT_HAVE_SYS_STAT_H</code>	251
7.11.1.8 <code>GLUCAT_HAVE_SYS_TYPES_H</code>	251
7.11.1.9 <code>GLUCAT_HAVE_UNISTD_H</code>	251
7.11.1.10 <code>GLUCAT_PACKAGE</code>	251
7.11.1.11 <code>GLUCAT_PACKAGE_BUGREPORT</code>	252
7.11.1.12 <code>GLUCAT_PACKAGE_NAME</code>	252
7.11.1.13 <code>GLUCAT_PACKAGE_STRING</code>	252
7.11.1.14 <code>GLUCAT_PACKAGE_TARNAME</code>	252
7.11.1.15 <code>GLUCAT_PACKAGE_URL</code>	252
7.11.1.16 <code>GLUCAT_PACKAGE_VERSION</code>	252
7.11.1.17 <code>GLUCAT_STDC_HEADERS</code>	253
7.11.1.18 <code>GLUCAT_VERSION</code>	253
7.12 <code>glucat/glucat_imp.h</code> File Reference	253
7.13 <code>glucat/index_set.h</code> File Reference	254
7.14 <code>glucat/index_set_imp.h</code> File Reference	255
7.15 <code>glucat/long_double.h</code> File Reference	257
7.16 <code>glucat/matrix.h</code> File Reference	257
7.17 <code>glucat/matrix_imp.h</code> File Reference	259
7.18 <code>glucat/matrix_multi.h</code> File Reference	261
7.19 <code>glucat/matrix_multi_imp.h</code> File Reference	263
7.20 <code>glucat/portability.h</code> File Reference	266
7.20.1 Macro Definition Documentation	267
7.20.1.1 <code>_GLUCAT_ISINF</code>	267
7.20.1.2 <code>_GLUCAT_ISNAN</code>	267
7.20.1.3 <code>UBLAS_ABS</code>	267
7.20.1.4 <code>UBLAS_SQRT</code>	267
7.21 <code>glucat/qd.h</code> File Reference	268

7.22 glucat/random.h File Reference . . . . .	268
7.23 glucat/scalar.h File Reference . . . . .	269
7.24 glucat/scalar_imp.h File Reference . . . . .	270
7.25 pyclical/glucat.pxd File Reference . . . . .	271
7.26 pyclical/PyClical.cpp File Reference . . . . .	271
7.26.1 Macro Definition Documentation . . . . .	272
7.26.1.1 PY_SSIZE_T_CLEAN . . . . .	272
7.27 pyclical/PyClical.h File Reference . . . . .	272
7.27.1 Typedef Documentation . . . . .	273
7.27.1.1 Clifford . . . . .	273
7.27.1.2 IndexSet . . . . .	273
7.27.1.3 scalar_t . . . . .	274
7.27.1.4 String . . . . .	274
7.27.1.5 Tune_P . . . . .	274
7.27.2 Function Documentation . . . . .	274
7.27.2.1 clifford_to_repr() . . . . .	274
7.27.2.2 clifford_to_str() . . . . .	275
7.27.2.3 index_set_to_repr() . . . . .	275
7.27.2.4 index_set_to_str() . . . . .	275
7.27.2.5 PyFloat_FromDouble() . . . . .	275
7.27.3 Variable Documentation . . . . .	276
7.27.3.1 epsilon . . . . .	276
7.27.3.2 hi_ndx . . . . .	276
7.27.3.3 lo_ndx . . . . .	276
7.28 pyclical/PyClical.pxd File Reference . . . . .	276
7.29 pyclical/PyClical.pyx File Reference . . . . .	276
7.30 pyclical/PyClical_nocython.cpp File Reference . . . . .	277
7.30.1 Macro Definition Documentation . . . . .	277
7.30.1.1 PY_SSIZE_T_CLEAN . . . . .	278
7.31 test/control.h File Reference . . . . .	278
7.32 test/driver.h File Reference . . . . .	279
7.33 test/timing.h File Reference . . . . .	279
7.34 test/try_catch.h File Reference . . . . .	280
7.35 test/tuning.h File Reference . . . . .	281
7.35.1 Macro Definition Documentation . . . . .	282
7.35.1.1 __TEST_TUNING_DEFAULT_CONSTANT . . . . .	282
7.35.2 Typedef Documentation . . . . .	282
7.35.2.1 precision_t . . . . .	282
7.35.2.2 Tune_P . . . . .	282
7.35.3 Function Documentation . . . . .	282
7.35.3.1 __TEST_TUNING_DEFAULT_CONSTANT() [1/9] . . . . .	282
7.35.3.2 __TEST_TUNING_DEFAULT_CONSTANT() [2/9] . . . . .	283

---

7.35.3.3 __TEST_TUNING_DEFAULT_CONSTANT() [3/9]	283
7.35.3.4 __TEST_TUNING_DEFAULT_CONSTANT() [4/9]	283
7.35.3.5 __TEST_TUNING_DEFAULT_CONSTANT() [5/9]	283
7.35.3.6 __TEST_TUNING_DEFAULT_CONSTANT() [6/9]	283
7.35.3.7 __TEST_TUNING_DEFAULT_CONSTANT() [7/9]	283
7.35.3.8 __TEST_TUNING_DEFAULT_CONSTANT() [8/9]	283
7.35.3.9 __TEST_TUNING_DEFAULT_CONSTANT() [9/9]	284
7.35.3.10 _GLUCAT_CTAssert()	284
7.35.4 Variable Documentation	284
7.35.4.1 Test_Tuning_Function_Precision	284
7.35.4.2 Test_Tuning_Max_Threshold	284
7.36 test/undefine.h File Reference	284

<b>Index</b>	<b>285</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>		59
<a href="#">glucat::matrix</a>		60
<a href="#">glucat::timing</a>		66
<a href="#">PyClical</a>		67
<a href="#">std</a>		73



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

glucat::bool_to_type< truth_value > . . . . .	77
cdef	
PyClical.clifford . . . . .	78
PyClical.index_set . . . . .	159
Clifford	
PyClical.clifford . . . . .	78
glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T > . . . . .	96
glucat::compare_types< LHS_T, RHS_T > . . . . .	107
glucat::compare_types< T, T > . . . . .	108
glucat::control_t . . . . .	109
glucat::CTAssertion< bool > . . . . .	114
glucat::CTAssertion< true > . . . . .	114
glucat::numeric_traits< Scalar_T >::demoted . . . . .	114
glucat::matrix::eig_genus< Matrix_T > . . . . .	115
glucat::framed_multi< Scalar_T, LO, HI > . . . . .	119
glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t . . . . .	142
glucat::index_set< LO, HI > . . . . .	144
glucat::index_set< DEFAULT_LO, DEFAULT_HI > . . . . .	144
glucat::index_set_hash< LO, HI > . . . . .	168
IndexSet	
PyClical.index_set . . . . .	159
inline	
PyClical.clifford . . . . .	78
PyClical.index_set . . . . .	159
logic_error	
glucat::glucat_error . . . . .	140
glucat::error< Class_T > . . . . .	117
map	
glucat::basis_table< Scalar_T, LO, HI, Matrix_T > . . . . .	75
glucat::gen::generator_table< Matrix_T > . . . . .	135
glucat::matrix_multi< Scalar_T, LO, HI > . . . . .	170
numeric_limits	
std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > > . . . . .	186
std::numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > > . . . . .	187
glucat::numeric_traits< Scalar_T > . . . . .	188

obj	
PyClical.clifford	78
PyClical.index_set	159
pair	
glucat::framed_multi< Scalar_T, LO, HI >::var_term	221
glucat::numeric_traits< Scalar_T >::promoted	201
glucat::random_generator< Scalar_T >	201
glucat::index_set< LO, HI >::reference	205
glucat::sorted_range< Map_T, Sorted_Map_T >	209
glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >	211
toClifford	
PyClical.clifford	78
toIndexSet	
PyClical.index_set	159
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 >	213

## 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>	75
Table of basis elements used as a cache by basis_element()	
<a href="#">glucat::bool_to_type&lt; truth_value &gt;</a>	77
Bool to type	
<a href="#">PyClical.clifford</a>	78
<a href="#">glucat::clifford_algebra&lt; Scalar_T, Index_Set_T, Multivector_T &gt;</a>	
Clifford_algebra<> declares the operations of a Clifford algebra	96
<a href="#">glucat::compare_types&lt; LHS_T, RHS_T &gt;</a>	
Type comparison	107
<a href="#">glucat::compare_types&lt; T, T &gt;</a>	108
<a href="#">glucat::control_t</a>	
Parameters to control tests	109
<a href="#">glucat::CTAssertion&lt; bool &gt;</a>	
Compile time assertion	114
<a href="#">glucat::CTAssertion&lt; true &gt;</a>	114
<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;::demoted</a>	
Demoted type for long double	114
<a href="#">glucat::matrix::eig_genus&lt; Matrix_T &gt;</a>	
Structure containing classification of eigenvalues	115
<a href="#">glucat::error&lt; Class_T &gt;</a>	
Specific exception class	117
<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	119
<a href="#">glucat::gen::generator_table&lt; Matrix_T &gt;</a>	
Table of generators for specific signatures	135
<a href="#">glucat::glucat_error</a>	
Abstract exception class	140
<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;::hash_size_t</a>	142
<a href="#">glucat::index_set&lt; LO, HI &gt;</a>	
Index set class based on std::bitset<> in Gnu standard C++ library	144
<a href="#">PyClical.index_set</a>	159
<a href="#">glucat::index_set_hash&lt; LO, HI &gt;</a>	168
<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	170
<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	186

<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 . . . . .	187
<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;</a>	
Extra traits which extend numeric limits . . . . .	188
<a href="#">glucat::numeric_traits&lt; Scalar_T &gt;::promoted</a>	
Promoted type . . . . .	201
<a href="#">glucat::random_generator&lt; Scalar_T &gt;</a>	
Random number generator with single instance per Scalar_T . . . . .	201
<a href="#">glucat::index_set&lt; LO, HI &gt;::reference</a>	
Index set member reference . . . . .	205
<a href="#">glucat::sorted_range&lt; Map_T, Sorted_Map_T &gt;</a>	
Sorted range for use with output . . . . .	209
<a href="#">glucat::sorted_range&lt; Sorted_Map_T, Sorted_Map_T &gt;</a>	211
<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 . . . . .	213
<a href="#">glucat::framed_multi&lt; Scalar_T, LO, HI &gt;::var_term</a>	
Variable term . . . . .	221

## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

glucat/clifford_algebra.h	225
glucat/clifford_algebra_imp.h	232
glucat/errors.h	238
glucat/errors_imp.h	239
glucat/framed_multi.h	239
glucat/framed_multi_imp.h	242
glucat/generation.h	244
glucat/generation_imp.h	245
glucat/global.h	246
glucat/glucat.h	248
glucat/glucat_config.h	249
glucat/glucat_imp.h	253
glucat/index_set.h	254
glucat/index_set_imp.h	255
glucat/long_double.h	257
glucat/matrix.h	257
glucat/matrix_imp.h	259
glucat/matrix_multi.h	261
glucat/matrix_multi_imp.h	263
glucat/portability.h	266
glucat/qd.h	268
glucat/random.h	268
glucat/scalar.h	269
glucat/scalar_imp.h	270
pyclical/glucat.pxd	271
pyclical/PyClical.cpp	271
pyclical/PyClical.h	272
pyclical/PyClical.pxd	276
pyclical/PyClical.pyx	276
pyclical/PyClical_nocython.cpp	277
test/control.h	278
test/driver.h	279
test/timing.h	279
test/try_catch.h	280
test/tuning.h	281
test/undefine.h	284





## 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 139 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 116 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 127 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, 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>  
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, BitsPerULongDoes↵  
NotMatchSetValueT) 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.*
- 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 subalgebra.*

- `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 [l\\_pi](#) = 3.1415926535897932384626433832795029L
- static const long double [l\\_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 106 of file [global.h](#).

### 5.2.1.2 [intfn](#)

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

For exception catching: pointer to function returning int.

Definition at line 66 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 69 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 108 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 146 of file global.h.

## 5.2.3 Function Documentation

### 5.2.3.1 \_GLUCAT\_CTAssert() [1/3]

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

Default lowest index in an index set.

**5.2.3.2 \_GLUCAT\_CTAssert() [2/3]**

```
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

**5.2.3.3 \_GLUCAT\_CTAssert() [3/3]**

```
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

**5.2.3.4 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 520 of file clifford\_algebra\_imp.h.

Referenced by PyClical.clifford::abs(), clifford\_to\_str(), cos(), matrix\_log(), matrix\_sqrt(), and sin().

**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 ) [inline]
```

Inverse cosine of multivector.

Definition at line 847 of file clifford\_algebra\_imp.h.

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse cosine of multivector with specified complexifier.

Definition at line 827 of file clifford\_algebra\_imp.h.

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

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

**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 ) [inline]
```

Inverse hyperbolic cosine of multivector.

Definition at line 787 of file clifford\_algebra\_imp.h.

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

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse hyperbolic cosine of multivector with specified complexifier.

Definition at line 767 of file clifford\_algebra\_imp.h.

Referenced by `cos()`.



**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 ) [inline]
```

Inverse sine of multivector.

Definition at line 954 of file clifford\_algebra\_imp.h.

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse sine of multivector with specified complexifier.

Definition at line 934 of file clifford\_algebra\_imp.h.

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

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

**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 ) [inline]
```

Inverse hyperbolic sine of multivector.

Definition at line 894 of file clifford\_algebra\_imp.h.

References complexifier(), and sin().

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse hyperbolic sine of multivector with specified complexifier.

Definition at line 874 of file clifford\_algebra\_imp.h.

Referenced by sin().

**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 ) [inline]
```

Inverse tangent of multivector.

Definition at line 1054 of file clifford\_algebra\_imp.h.

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse tangent of multivector with specified complexifier.

Definition at line 1034 of file clifford\_algebra\_imp.h.

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

**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 ) [inline]
```

Inverse hyperbolic tangent of multivector.

Definition at line 998 of file clifford\_algebra\_imp.h.

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Inverse hyperbolic tangent of multivector with specified complexifier.

Definition at line 981 of file clifford\_algebra\_imp.h.

Referenced by tan(), and tanh().

**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 1979 of file matrix\_multi\_imp.h.

References db\_step(), epsilon, 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 595 of file clifford\_algebra\_imp.h.

References complexifier(), and sqrt().

Referenced by 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 633 of file clifford\_algebra\_imp.h.

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 602 of file index\_set\_imp.h.

### 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 535 of file clifford\_algebra\_imp.h.

Referenced by acosh(), asinh(), check\_complex(), cos(), sin(), tan(), and tanh().

### 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 o invo == invo o rev.

Definition at line 496 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 ) [inline]
```

Cosine of multivector.

Definition at line 818 of file clifford\_algebra\_imp.h.

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

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false )
```

Cosine of multivector with specified complexifier.

Definition at line 794 of file clifford\_algebra\_imp.h.

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

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

**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 749 of file clifford\_algebra\_imp.h.

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

**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&().

**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 1936 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 1407 of file matrix\_multi\_imp.h.

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 1394 of file matrix\_multi\_imp.h.

References norm().

Referenced by cascade\_log().

**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 586 of file clifford\_algebra\_imp.h.

References PyClical::i, and sqrt().

**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 456 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 1977 of file framed\_multi\_imp.h.

Referenced by acos(), exp(), glucat::numeric\_traits< Scalar\_T >::fmod(), matrix\_log(), matrix\_sqrt(), pow(), and PyClical.clifford::pow().

**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 2143 of file matrix\_multi\_imp.h.

References clifford\_exp(), exp(), glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_Max\_Outer\_Steps, Log\_Max\_Inner\_Steps, Basis\_Max\_Count, Fast\_Size\_Threshold, Inv\_Fast\_Dim\_Threshold, Products\_Size\_Threshold, Function\_Precision >::function\_precision, 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 1115 of file matrix\_multi\_imp.h.

**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 123 of file matrix\_multi\_imp.h.

**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 440 of file clifford\_algebra\_imp.h.

**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 350 of file clifford\_algebra\_imp.h.

Referenced by 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 890 of file index\_set\_imp.h.

**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 873 of file index\_set\_imp.h.

**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 480 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 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 2027 of file framed\_multi\_imp.h.

**5.2.3.42 log() [2/4]**

```
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 2026 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, matrix\_log(), precision\_demoted, and precision\_promoted.

**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 ) [inline]
```

Natural logarithm of multivector.

Definition at line 741 of file clifford\_algebra\_imp.h.

**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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Natural logarithm of multivector with specified complexifier.

Definition at line 733 of file clifford\_algebra\_imp.h.

Referenced by glucat::numeric\_traits< Scalar\_T >::conj(), matrix\_log(), pow(), and PyClical.clifford::pow().

**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 331 of file scalar.h.

**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 2067 of file matrix\_multi\_imp.h.

References `abs()`, `glucat::matrix::both_eig_case`, `cascade_log()`, `glucat::matrix::classify_eigenvalues()`, `exp()`, `PyClical::i`, `inv()`, `glucat::matrix::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()`, and `PyClical::pi`.

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 1648 of file matrix\_multi\_imp.h.

References `abs()`, `glucat::matrix::both_eig_case`, `glucat::matrix::classify_eigenvalues()`, `db_sqrt()`, `exp()`, `PyClical::i`, `inv()`, `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()`, 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 528 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 1003 of file index\_set\_imp.h.

**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 996 of file index\_set\_imp.h.

**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 512 of file clifford\_algebra\_imp.h.

Referenced by cascade\_log(), db\_step(), 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 464 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 108 of file matrix\_multi\_imp.h.

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

**5.2.3.54 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 107 of file clifford\_algebra\_imp.h.

**5.2.3.55 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 115 of file clifford\_algebra\_imp.h.

**5.2.3.56 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 123 of file clifford\_algebra\_imp.h.

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

```
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 749 of file framed\_multi\_imp.h.

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

```
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 667 of file matrix\_multi\_imp.h.

**5.2.3.59 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 272 of file clifford\_algebra\_imp.h.

**5.2.3.60 operator&() [1/4]**

```
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 )
```

Inner product.

Definition at line 631 of file framed\_multi\_imp.h.

References `crd_of_mult()`.

**5.2.3.61 operator&() [2/4]**

```
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 intersection: `and`.

Definition at line 215 of file index\_set\_imp.h.

**5.2.3.62 operator&() [3/4]**

```
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]
```

Inner product.

Definition at line 648 of file matrix\_multi\_imp.h.

**5.2.3.63 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]
```

Inner product.

Definition at line 257 of file clifford\_algebra\_imp.h.

References `star()`.

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

```
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 430 of file framed\_multi\_imp.h.

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

```
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 579 of file matrix\_multi\_imp.h.

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

```
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 227 of file clifford\_algebra\_imp.h.

**5.2.3.67 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 Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Product of multivector and scalar.

Definition at line 201 of file clifford\_algebra\_imp.h.



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

```
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 212 of file clifford\_algebra\_imp.h.

**5.2.3.69 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 1944 of file framed\_multi\_imp.h.

**5.2.3.70 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>
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 156 of file clifford\_algebra\_imp.h.

**5.2.3.71 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 Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Geometric sum of multivector and scalar.

Definition at line 131 of file clifford\_algebra\_imp.h.

**5.2.3.72 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>
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 142 of file clifford\_algebra\_imp.h.

**5.2.3.73 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>
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 190 of file clifford\_algebra\_imp.h.

**5.2.3.74 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 Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Geometric difference of multivector and scalar.

Definition at line 167 of file clifford\_algebra\_imp.h.

**5.2.3.75 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>
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 178 of file clifford\_algebra\_imp.h.

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

```
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 944 of file framed\_multi\_imp.h.

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

```
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 700 of file matrix\_multi\_imp.h.

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

```
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 324 of file clifford\_algebra\_imp.h.

**5.2.3.79 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 Multivector< Scalar_T, LO, HI > & lhs,
    const Scalar_T & scr ) [inline]
```

Quotient of multivector and scalar.

Definition at line 298 of file clifford\_algebra\_imp.h.

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

```
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 309 of file clifford\_algebra\_imp.h.

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

```
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 1395 of file framed\_multi\_imp.h.

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

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

Write out index set.

Definition at line 639 of file index\_set\_imp.h.

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

```
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 1056 of file matrix\_multi\_imp.h.

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

```
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 1427 of file framed\_multi\_imp.h.

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

```
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 1466 of file framed\_multi\_imp.h.

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

```
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 662 of file index\_set\_imp.h.

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

```
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 1067 of file matrix\_multi\_imp.h.

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

```
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 531 of file framed\_multi\_imp.h.

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

```
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 190 of file index\_set\_imp.h.

**5.2.3.90 operator^() [3/4]**

```
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 629 of file matrix\_multi\_imp.h.

**5.2.3.91 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 242 of file clifford\_algebra\_imp.h.

**5.2.3.92 operator" | () [1/4]**

```
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 970 of file framed\_multi\_imp.h.

References pow().

**5.2.3.93 operator" | () [2/4]**

```
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 240 of file index\_set\_imp.h.

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

```
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 809 of file matrix\_multi\_imp.h.

**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]
```

Transformation via twisted adjoint action.

Definition at line 339 of file clifford\_algebra\_imp.h.

### 5.2.3.96 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 413 of file clifford\_algebra\_imp.h.

### 5.2.3.97 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 1334 of file matrix\_multi\_imp.h.

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

### 5.2.3.98 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 1958 of file matrix\_multi\_imp.h.

References pade\_approx().

Referenced by cascade\_log().

### 5.2.3.99 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 216 of file global.h.



**5.2.3.100 pow()** [1/2]

```
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 390 of file clifford\_algebra\_imp.h.

**5.2.3.101 pow()** [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::pow (
    const Multivector< Scalar_T, LO, HI > & lhs,
    int rhs )
```

Integer power of multivector.

Definition at line 357 of file clifford\_algebra\_imp.h.

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

Referenced by `cascade_log()`, and `operator|()`.

**5.2.3.102 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 448 of file clifford\_algebra\_imp.h.

**5.2.3.103 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 504 of file clifford\_algebra\_imp.h.

**5.2.3.104 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 429 of file clifford\_algebra\_imp.h.

**5.2.3.105 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 382 of file matrix\_multi\_imp.h.

Referenced by glucat::matrix\_multi< Scalar\_T, LO, HI >::operator+=(.).

**5.2.3.106 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 488 of file clifford\_algebra\_imp.h.

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

**5.2.3.107 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 421 of file clifford\_algebra\_imp.h.

Referenced by acos(), asin(), exp(), glucat::framed\_multi< Scalar\_T, LO, HI >::fast(), and tan().

**5.2.3.108 sign\_of\_square()**

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

Square of generator {j}.

Square of generator index j.

Definition at line 989 of file index\_set\_imp.h.

**5.2.3.109 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 ) [inline]
```

Sine of multivector.

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

References asin(), and complexifier().

**5.2.3.110 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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false )
```

Sine of multivector with specified complexifier.

Definition at line 901 of file clifford\_algebra\_imp.h.

References abs(), asin(), asinh(), check\_complex(), and PyClical::i.

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

**5.2.3.111 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 855 of file clifford\_algebra\_imp.h.

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

**5.2.3.112 sqrt() [1/4]**

```
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 1954 of file framed\_multi\_imp.h.

**5.2.3.113 sqrt() [2/4]**

```
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 1603 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`, `matrix_sqrt()`, `precision_demoted`, and `precision_promoted`.

**5.2.3.114 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 ) [inline]
```

Square root of multivector.

Definition at line 626 of file clifford\_algebra\_imp.h.

**5.2.3.115 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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Square root of multivector with specified complexifier.

Definition at line 618 of file clifford\_algebra\_imp.h.

Referenced by check\_complex(), elliptic(), and matrix\_sqrt().

**5.2.3.116 star()** [1/3]

```
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 885 of file framed\_multi\_imp.h.

References glucat::matrix::isnan().

**5.2.3.117 star()** [2/3]

```
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 686 of file matrix\_multi\_imp.h.

**5.2.3.118 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 287 of file clifford\_algebra\_imp.h.

Referenced by operator&().

**5.2.3.119 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 ) [inline]
```

Tangent of multivector.

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

References atan(), and complexifier().

**5.2.3.120 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,
    const Multivector< Scalar_T, LO, HI > & i,
    const bool prechecked = false ) [inline]
```

Tangent of multivector with specified complexifier.

Definition at line 1006 of file clifford\_algebra\_imp.h.

References atan(), atanh(), check\_complex(), PyClical::i, and scalar().

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

**5.2.3.121 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 962 of file clifford\_algebra\_imp.h.

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

Referenced by `asin()`, and `glucat::numeric_traits< Scalar_T >::log2()`.

**5.2.3.122 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 163 of file scalar\_imp.h.

**5.2.3.123 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 153 of file scalar\_imp.h.

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

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

Exception catching for functions returning int.

Definition at line 78 of file try\_catch.h.

Referenced by `glucat::control_t::verbose()`.

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

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

Exception catching for functions of int returning int.

Definition at line 93 of file try\_catch.h.

**5.2.3.126 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 472 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 132 of file global.h.

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

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

Definition at line 159 of file global.h.

**5.2.4.3 DEFAULT\_Div\_Max\_Steps**

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

Definition at line 155 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 160 of file global.h.

#### 5.2.4.5 DEFAULT\_Function\_Precision

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

Definition at line 163 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 140 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 161 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 158 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 157 of file global.h.

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

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

Definition at line 154 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 162 of file global.h.

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

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

Definition at line 156 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 143 of file global.h.

#### 5.2.4.14 l\_ln2

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

Definition at line 70 of file long\_double.h.

#### 5.2.4.15 l\_pi

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

Definition at line 69 of file long\_double.h.

#### 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 112 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 101 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 139 of file generation.h.

## 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

<a href="#">safe_eig_case</a>	
<a href="#">negative_eig_case</a>	
<a href="#">both_eig_case</a>	

Definition at line 187 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 588 of file `matrix_imp.h`.

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 555 of file `matrix_imp.h`.

### 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:  $\text{sum}(x(i,j)*y(i,j))/x.\text{nrows}()$

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

Definition at line 453 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 354 of file matrix\_imp.h.

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

#### 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 135 of file matrix\_imp.h.

#### 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 178 of file matrix\_imp.h.

#### 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 388 of file matrix\_imp.h.

Referenced by glucat::gen::generator\_table< Matrix\_T >::gen\_from\_pm4\_qp4().

#### 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 331 of file matrix\_imp.h.

#### 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 250 of file matrix\_imp.h.

#### 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 217 of file matrix\_imp.h.

#### 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 475 of file matrix\_imp.h.



#### 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 435 of file matrix\_imp.h.

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

#### 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 299 of file matrix\_imp.h.

#### 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 424 of file matrix\_imp.h.

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

#### 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 523 of file matrix\_imp.h.

#### 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 499 of file matrix\_imp.h.

#### 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 379 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 109 of file timing.h.

## 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 103 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 100 of file timing.h.

### 5.5.2.3 MS\_PER\_SEC

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

Timing constant: milliseconds per second.

Definition at line 97 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__` = "0.8.4"
- `obj`
- `i`
- `ixt`
- `fill`
- `scalar_epsilon` = `epsilon`
- float `pi` = `atan(clifford(1.0)) * 4.0`
- float `tau` = `atan(clifford(1.0)) * 8.0`
- `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 PyClicl._test ( ) [private]
```

Definition at line 1912 of file PyClicl.pyx.

### 5.6.1.2 `clifford_hidden_doctests()`

```
def PyClicl.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(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 <class 'list'>.
```

```

>>> print(clifford(None))
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from <class 'NoneType'>.
>>> print(clifford(None,[1,2]))
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from (<class 'NoneType'>, <class 'list'>).
>>> print(clifford([1,2],[1,2]))
Traceback (most recent call last):
...
TypeError: Cannot initialize clifford object from (<class 'list'>, <class '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("{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 1244 of file PyClical.pyx.

### 5.6.1.3 e()

```
def PyClical.e (
    obj )
```

Abbreviation for clifford(index\_set(obj)).

```

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

```

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

Definition at line 1886 of file PyClical.pyx.

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

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

```
def PyClical.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 {1, 2, 100}.
>>> print(index_set([1,2]))
Traceback (most recent call last):
...
TypeError: Cannot initialize index_set object from <class '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 406 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 1899 of file PyClical.pyx.

## 5.6.2 Variable Documentation

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

```

string PyClical.__version__ = "0.8.4" [private]

```

Definition at line 33 of file PyClical.pyx.

### 5.6.2.2 cl

```
PyClical.cl = clifford
```

Definition at line 1860 of file PyClical.pyx.

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

### 5.6.2.3 fill

```
PyClical.fill
```

Definition at line 1814 of file PyClical.pyx.

### 5.6.2.4 i

```
PyClical.i
```

Definition at line 1541 of file PyClical.pyx.

Referenced by `glucat.cos()`, `glucat.elliptic()`, `glucat.log()`, `glucat.matrix_log()`, `glucat.matrix_sqrt()`, `glucat.sin()`, `glucat.sqrt()`, and `glucat.tan()`.

### 5.6.2.5 ist

```
PyClical.ist = index_set
```

Definition at line 1878 of file PyClical.pyx.

Referenced by `cga3.agc3()`, `cga3.cga3()`, `cga3.cga3std()`, `glucat::framed_multi< Scalar_T, LO, HI >.framed_multi()`, `index_set_to_repr()`, and `index_set_to_str()`.

### 5.6.2.6 ixt

```
PyClical.ixt
```

Definition at line 1814 of file PyClical.pyx.



### 5.6.2.7 nbar3

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

Definition at line 1909 of file PyClical.pyx.

### 5.6.2.8 ninf3

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

Definition at line 1908 of file PyClical.pyx.

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

### 5.6.2.9 obj

```
PyClical.obj
```

Definition at line 1541 of file PyClical.pyx.

### 5.6.2.10 pi

```
float PyClical.pi = atan(clifford(1.0)) * 4.0
```

Definition at line 1857 of file PyClical.pyx.

Referenced by `glucat.matrix_log()`.

### 5.6.2.11 scalar\_epsilon

```
PyClical.scalar_epsilon = epsilon
```

Definition at line 1855 of file PyClical.pyx.

### 5.6.2.12 tau

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

Definition at line 1858 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

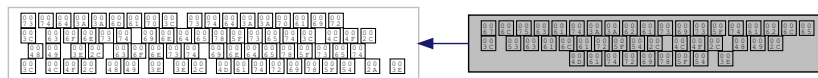
Table of basis elements used as a cache by basis\_element()

```
#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 1250 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 1260 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 1261 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 1256 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 1268 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 98 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	
-------	--

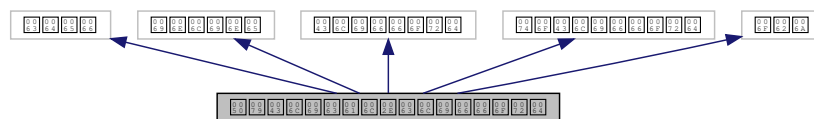
Definition at line 101 of file global.h.

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

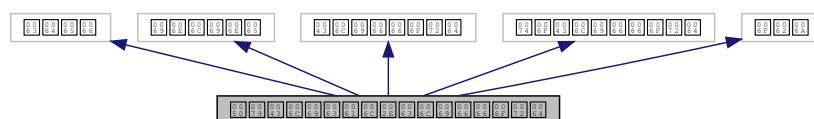
- glucat/[global.h](#)

## 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 __truediv__ (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 740 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 836 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 1020 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(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 565 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 627 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 621 of file PyClical.pyx.

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

### 6.3.2.7 `__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 707 of file PyClical.pyx.

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

**6.3.2.8 `__iadd__()`**

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

Geometric sum.

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

Definition at line 751 of file PyClical.pyx.

**6.3.2.9 `__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 851 of file PyClical.pyx.

**6.3.2.10 `__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 911 of file PyClical.pyx.

**6.3.2.11 `__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 821 of file PyClical.pyx.

**6.3.2.12 `__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 793 of file PyClical.pyx.

**6.3.2.13 `__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 950 of file PyClical.pyx.

**6.3.2.14 `__isub__()`**

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

Geometric difference.

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

Definition at line 771 of file PyClical.pyx.

**6.3.2.15 `__iter__()`**

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

Not applicable.

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

Definition at line 638 of file PyClical.pyx.

**6.3.2.16 `__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 881 of file PyClical.pyx.

**6.3.2.17 `__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 806 of file PyClical.pyx.

**6.3.2.18 `__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 780 of file PyClical.pyx.

**6.3.2.19 `__neg__()`**

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

Unary -.

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

Definition at line 722 of file PyClical.pyx.

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

**6.3.2.20 `__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 939 of file PyClical.pyx.

**6.3.2.21 `__pos__()`**

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

Unary +.

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

Definition at line 731 of file PyClical.pyx.

**6.3.2.22 `__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 961 of file PyClical.pyx.

References `PyClical.clifford.pow()`.

### 6.3.2.23 `__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 1226 of file PyClical.pyx.

References `clifford_to_repr()`.

### 6.3.2.24 `__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 672 of file PyClical.pyx.

### 6.3.2.25 `__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 1235 of file PyClical.pyx.

References `clifford_to_str()`.



**6.3.2.26 `__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 760 of file PyClical.pyx.

**6.3.2.27 `__truediv__()`**

```
def PyClical.clifford.__truediv__ (
    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 896 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 866 of file PyClical.pyx.

### 6.3.2.29 abs()

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

Absolute value: square root of norm.

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

Definition at line 1175 of file PyClicl.pyx.

References `glucat.abs()`.

### 6.3.2.30 conj()

```
def PyClicl.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 1138 of file PyClicl.pyx.

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

### 6.3.2.31 even()

```
def PyClicl.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 1061 of file PyClicl.pyx.

References `PyClicl.index_set.instance`, and `PyClicl.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)
<class 'PyClical.index_set'>
```

Definition at line 1215 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 926 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 1107 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 1206 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 1184 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 1164 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 1070 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 1004 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 980 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 PyClicl.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 1050 of file PyClicl.pyx.

References PyClicl.index\_set.instance, and PyClicl.clifford.instance.

**6.3.2.42 quad()**

```
def PyClicl.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 1153 of file PyClicl.pyx.

References PyClicl.index\_set.instance, and PyClicl.clifford.instance.

**6.3.2.43 reframe()**

```
def PyClicl.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 649 of file PyClicl.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 1123 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 1039 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 1195 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 1079 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.  $\{1\} \rightarrow -\{1\}$ .*
- virtual const `multivector_t reverse` () const =0  
*Reversion, eg.  $\{1\}*\{2\} \rightarrow \{2\}*\{1\}$ .*
- 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 71 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 104 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 105 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 106 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 103 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 107 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 111 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 94 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.

### 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.

**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]
```

Contraction.

**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]
```

Inner product.

**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 multivector_t & rhs ) [pure virtual]
```

Geometric product.

**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 Scalar_T & scr ) [pure virtual]
```

Product of multivector and scalar.

**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 multivector_t & rhs ) [pure virtual]
```

Geometric quotient.

**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 Scalar_T & scr ) [pure virtual]
```

Quotient of multivector and scalar.

**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 83 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

<a href="#">are_same</a>	
--------------------------	--

Definition at line 86 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 89 of file global.h.

## 6.6.2 Member Enumeration Documentation

### 6.6.2.1 anonymous enum

```
template<typename T >
anonymous enum
```

#### Enumerator

are_same	
----------	--

Definition at line 92 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 68 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 117 of file control.h.

#### 6.7.2.2 `control_t()` [2/3]

```
glucat::control_t::control_t ( ) [inline], [private]
```

Definition at line 117 of file control.h.

### 6.7.2.3 ~control\_t()

```
glucat::control_t::~~control_t ( ) [inline], [private]
```

Definition at line 118 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 165 of file control.h.

### 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 179 of file control.h.

### 6.7.3.3 catch\_exceptions()

```
bool glucat::control_t::catch_exceptions ( ) const [inline], [private]
```

Definition at line 107 of file control.h.

Referenced by verbose().

#### 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 129 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 102 of file control.h.

References `m_valid`.

Referenced by `verbose()`.

#### 6.7.3.7 verbose()

```
static bool glucat::control_t::verbose ( ) [inline], [static]
```

Produce more detailed output from tests.

Definition at line 138 of file control.h.

References `catch_exceptions()`, `glucat::try_catch()`, and `valid()`.

### 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 125 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 106 of file control.h.

#### 6.7.5.2 m\_valid

```
bool glucat::control_t::m_valid [private]
```

Test parameters are valid.

Definition at line 101 of file control.h.

Referenced by 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 111 of file control.h.

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 75 of file `global.h`.

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

- [glucat/global.h](#)

## 6.9 `glucat::CTAssertion< true >` Struct Reference

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

### 6.9.1 Detailed Description

Definition at line 76 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 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>
struct glucat::numeric_traits< Scalar_T >::demoted
```

Demoted type for long double.

Demoted type.

Definition at line 76 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 78 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 205 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 191 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 193 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 195 of file matrix.h.

Referenced by `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 197 of file matrix.h.

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

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

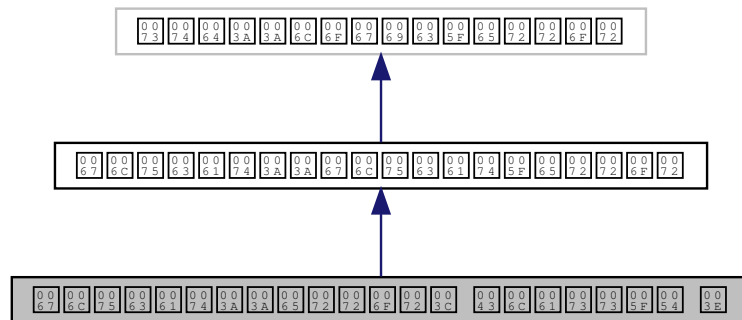
- glucat/matrix.h

## 6.12 glucat::error&lt; Class\_T &gt; Class Template Reference

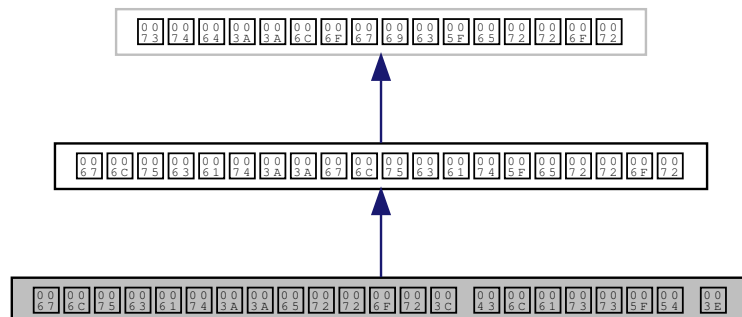
Specific exception class.

```
#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 86 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 67 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 73 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 throw ( )    [virtual]
```

Implements [glucat::glucat\\_error](#).

Definition at line 86 of file `errors_imp.h`.

### 6.12.3.2 `heading()`

```
template<class Class_T >
const std::string glucat::error< Class_T >::heading throw ( )    [virtual]
```

Implements [glucat::glucat\\_error](#).

Definition at line 80 of file `errors_imp.h`.

### 6.12.3.3 `print_error_msg()`

```
template<class Class_T >
void glucat::error< Class_T >::print_error_msg    [virtual]
```

Implements [glucat::glucat\\_error](#).

Definition at line 92 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>
```

### 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 = DEFAU↵
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 = DEFAULT_HI>
LT_HI>
typedef class var_term glucat::framed_multi< Scalar_T, LO, HI >::var_term_t [private]
```

Definition at line 160 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 & 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 96 of file framed\_multi\_imp.h.

**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 103 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 111 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 126 of file framed\_multi\_imp.h.

References PyClical::ist.

**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 141 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 149 of file framed\_multi\_imp.h.

**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 159 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().

**6.13.3.9 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 172 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 182 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 192 of file framed\_multi\_imp.h.

**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 215 of file framed\_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 >::inv\_fast\_dim\_threshold, and glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix.

**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 231 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>
LT_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 244 of file framed\_multi\_imp.h.

## 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 1684 of file framed\_multi\_imp.h.

### 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 1729 of file framed\_multi\_imp.h.

#### 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 1774 of file framed\_multi\_imp.h.

#### 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 82 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 1810 of file framed\_multi\_imp.h.

#### 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 1829 of file framed\_multi\_imp.h.

References `glucat::odd()`, and `glucat::scalar()`.

#### 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 [inline]
```

Use inverse generalized FFT to construct a framed\_multi\_t.

Definition at line 1927 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 1895 of file framed\_multi\_imp.h.

#### 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 1643 of file framed\_multi\_imp.h.

#### 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
```

Number of terms.

Definition at line 1574 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 358 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 1302 of file framed\_multi\_imp.h.

#### 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 1663 of file framed\_multi\_imp.h.

### 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>
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 160 of file framed\_multi.h.

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

### 6.13.5.3 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 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<< [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>>**

```
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 = DEFAU↵
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 = DEFAU↵
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:

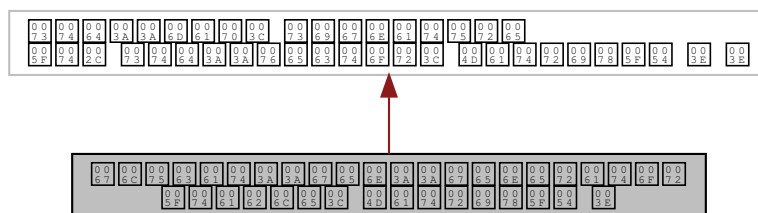
- [glucat/framed\\_multi.h](#)
- [glucat/framed\\_multi\\_imp.h](#)

## 6.14 glucat::gen::generator\_table< Matrix\_T > Class Template Reference

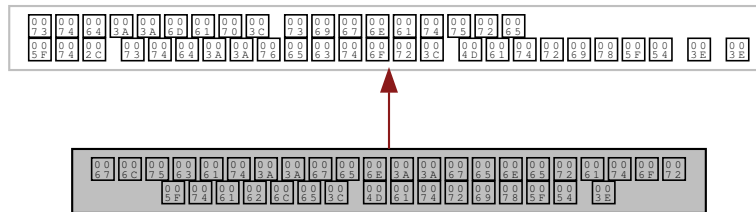
Table of generators for specific signatures.

```
#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 105 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 127 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 128 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 184 of file generation\_imp.h.

#### 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 221 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 252 of file generation\_imp.h.

#### 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 282 of file generation\_imp.h.

### 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 137 of file generation\_imp.h.

### 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 107 of file generation\_imp.h.

### 6.14.3.7 operator()()

```
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 116 of file generation\_imp.h.

### 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 135 of file generation.h.

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

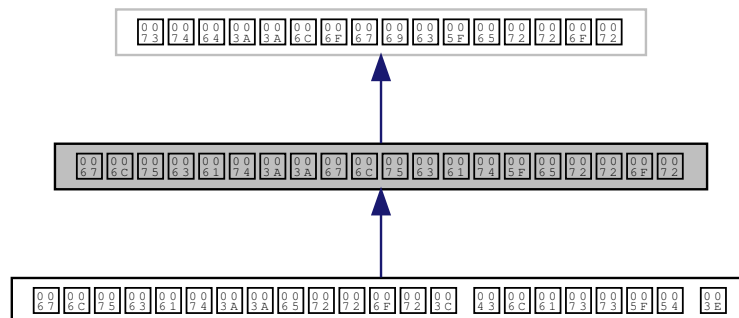
- [glucat/generation.h](#)
- [glucat/generation\\_imp.h](#)

### 6.15 glucat::glucat\_error Class Reference

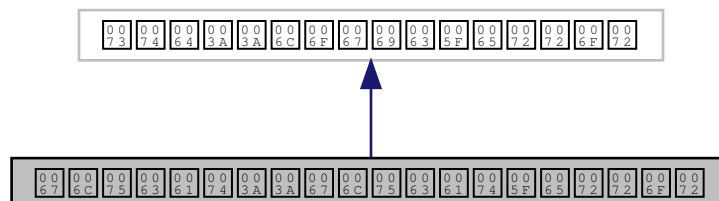
Abstract exception class.

```
#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 70 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 102 of file errors.h.

#### 6.15.2.2 ~glucat\_error()

```
glucat::glucat_error::~~glucat_error ( ) throw ( ) [inline]
```

Definition at line 105 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 110 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>
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>
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>
LT_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>
```

### 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 74 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 110 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 111 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 114 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 113 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 121 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 89 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 83 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 96 of file index\_set\_imp.h.

**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 110 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 130 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 77 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 [inline]
```

Cardinality: Number of indices included in set.

Definition at line 372 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::count().

Referenced by glucat::index\_set< LO, HI >::count(), glucat::index\_set< LO, HI >::flip(), glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi(), and glucat::matrix\_multi< Scalar\_T, LO, HI >::matrix\_multi().

#### 6.17.4.4 count\_neg()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::count_neg [inline]
```

Number of negative indices included in set.

Definition at line 392 of file index\_set\_imp.h.

#### 6.17.4.5 count\_pos()

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::count_pos [inline]
```

Number of positive indices included in set.

Definition at line 404 of file index\_set\_imp.h.

#### 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 347 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 358 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::count().

**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 [inline]
```

Fold this index set within itself as a frame.

Definition at line 776 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::set().

Referenced by glucat::index\_set< LO, HI >::unfold().

**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 784 of file index\_set\_imp.h.

**6.17.4.10 hash\_fn()**

```
template<const index_t LO, const index_t HI>
size_t glucat::index_set< LO, HI >::hash_fn [inline]
```

Hash function.

Definition at line 976 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 [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 760 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::set().

**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 616 of file index\_set\_imp.h.

**6.17.4.13 max()**

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::max
```

Maximum member.

Maximum member, or 0 if none.

Definition at line 578 of file index\_set\_imp.h.

Referenced by PyClical.index\_set::\_\_iter\_\_(), and glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi().

**6.17.4.14 min()**

```
template<const index_t LO, const index_t HI>
index_t glucat::index_set< LO, HI >::min
```

Minimum member.

Minimum member, or 0 if none.

Definition at line 489 of file index\_set\_imp.h.

Referenced by PyClical.index\_set::\_\_iter\_\_(), glucat::framed\_multi< Scalar\_T, LO, HI >::framed\_multi(), and glucat::index\_set< LO, HI >::unfold().

**6.17.4.15 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 158 of file index\_set\_imp.h.



#### 6.17.4.16 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 intersection: and.

Definition at line 202 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 624 of file index\_set\_imp.h.

#### 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 147 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 260 of file index\_set\_imp.h.

#### 6.17.4.20 operator[]()

```
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 252 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 177 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 227 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~ [inline]
```

Set complement: not.

Definition at line 169 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 322 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 333 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 283 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::fold(), and glucat::index\_set< LO, HI >::is\_contiguous().

**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 294 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 308 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 907 of file index\_set\_imp.h.

**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 [inline]
```

Sign of geometric square of a Clifford basis element.

Definition at line 956 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 268 of file index\_set\_imp.h.

### 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 822 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::fold(), and glucat::index\_set< LO, HI >::min().

### 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 856 of file index\_set\_imp.h.

## 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 203 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 117 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]
```

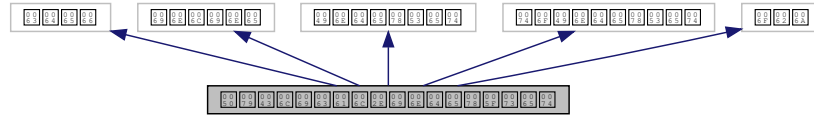
Definition at line 116 of file index\_set.h.

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

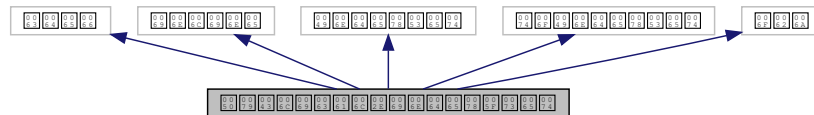
- glucat/index\_set.h
- glucat/index\_set\_imp.h

## 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 38 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 271 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 74 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 210 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 116 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 191 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 282 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 240 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 304 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, end=", ")
-3,4,7,
```

Definition at line 229 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 260 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 293 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 384 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 122 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 179 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 395 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 249 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 315 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 324 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 333 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 360 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 351 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 342 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 366 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 375 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 95 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< LO, HI > 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:

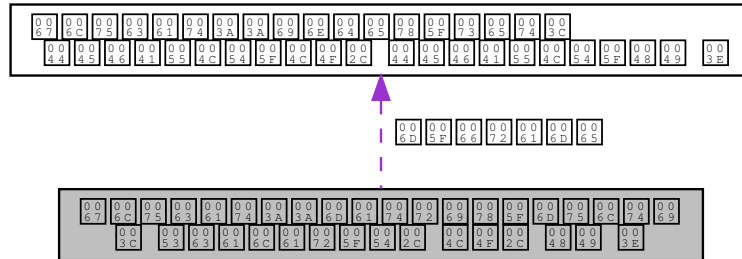
- `glucat/framed_multi.h`

## 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>
```

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 = DEFAULT_HI>
LT_HI>
typedef ublas::compressed_matrix< int, orientation_t > glucat::matrix_multi< Scalar_T, LO, HI
>::basis_matrix_t [private]
```

Definition at line 181 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 = DEFAULT_HI>
LT_HI>
typedef error<multivector_t> glucat::matrix_multi< Scalar_T, LO, HI >::error_t
```

Definition at line 171 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 172 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 168 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 188 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 166 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 186 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 165 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 179 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 167 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 169 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 170 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>
LT_HI>
glucat::matrix_multi< Scalar_T, LO, HI >::~~matrix_multi ( ) [inline]
```

Destructor.

Definition at line 194 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 128 of file matrix\_multi\_imp.h.

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

**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 137 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix, and glucat::numeric\_traits< Scalar\_T >::to\_↔ scalar\_t().

**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 159 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 187 of file matrix\_multi\_imp.h.

**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 199 of file matrix\_multi\_imp.h.

**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 211 of file matrix\_multi\_imp.h.

References glucat::index\_set< LO, HI >::count(), 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 225 of file matrix\_multi\_imp.h.

**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 237 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 243 of file matrix\_multi\_imp.h.

**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 270 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 276 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 = DEFAULT_HI>
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 225 of file matrix\_multi.h.

**6.20.3.14 matrix\_multi()** [13/17]

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAULT_HI>
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 228 of file matrix\_multi.h.

**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 283 of file matrix\_multi\_imp.h.

References PyClical::e(), and glucat::tuning< Mult\_Matrix\_Threshold, Div\_Max\_Steps, Sqrt\_Max\_Steps, Log\_Max\_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.

**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 310 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_matrix, and glucat::numeric\_traits< Scalar\_T >::to\_scalar\_t().

### 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 338 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.

### 6.20.3.18 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 359 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 1274 of file matrix\_multi\_imp.h.

### 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 100 of file matrix\_multi\_imp.h.

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

### 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
```

Use inverse generalized FFT to construct a framed\_multi\_t.

Definition at line 1197 of file matrix\_multi\_imp.h.

### 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 1184 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 501 of file matrix\_multi\_imp.h.

References glucat::matrix\_multi< Scalar\_T, LO, HI >::m\_frame, glucat::matrix\_multi< Scalar\_T, LO, HI >::m↔  
matrix, and glucat::reframe().

### 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 367 of file matrix\_multi\_imp.h.

#### 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 1027 of file matrix\_multi\_imp.h.

### 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 174 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 176 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%

```
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 304 of file `matrix_multi.h`.

Referenced by `glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi()`, and `glucat::matrix_multi< Scalar_T, LO, HI >::operator+=(())`.

### 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 306 of file `matrix_multi.h`.

Referenced by `glucat::matrix_multi< Scalar_T, LO, HI >::classname()`, `glucat::framed_multi< Scalar_T, LO, HI >::framed_multi()`, `glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi()`, `glucat::offset_level()`, and `glucat::matrix_multi< Scalar_T, LO, HI >::operator+=(())`.

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

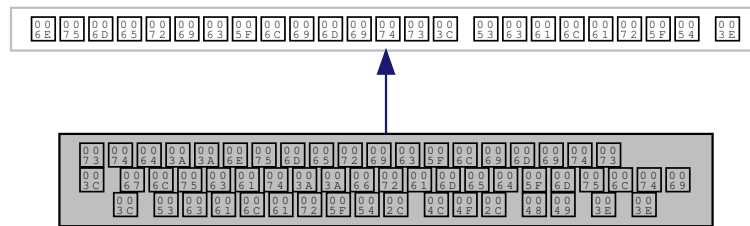
- [glucat/framed\\_multi.h](#)
- [glucat/matrix\\_multi.h](#)
- [glucat/matrix\\_multi\\_imp.h](#)

## 6.21 std::numeric\_limits< glucat::framed\_multi< Scalar\_T, LO, HI > > Struct Template Reference

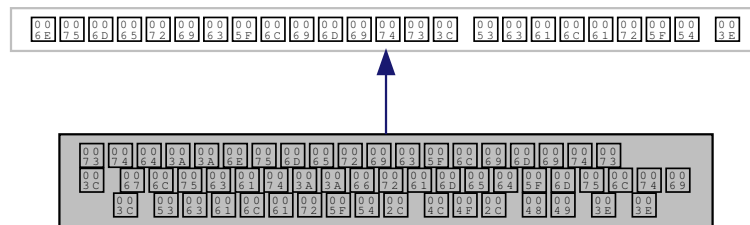
Numeric limits for framed\_multi inherit limits for the corresponding scalar type.

```
#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:

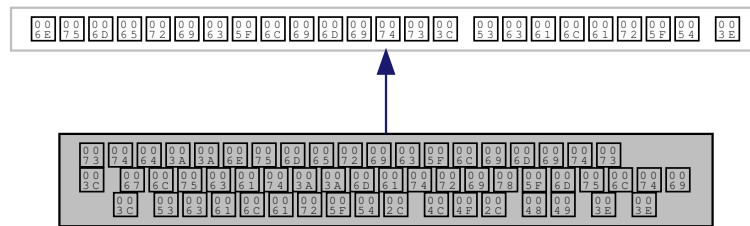
- [glucat/framed\\_multi.h](#)

## 6.22 std::numeric\_limits< glucat::matrix\_multi< Scalar\_T, LO, HI > > Struct Template Reference

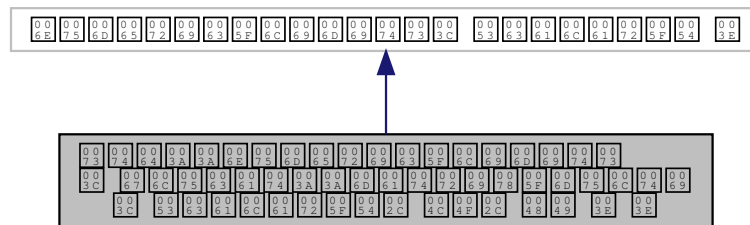
Numeric limits for matrix\_multi inherit limits for the corresponding scalar type.

```
#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

- long double [pi](#) ()  
*Pi for long double.*
- long double [ln\\_2](#) ()  
*log(2) for long double*
- float [to\\_scalar\\_t](#) (const Other\_Scalar\_T &val)  
*Extra traits which extend numeric limits.*
- double [to\\_scalar\\_t](#) (const Other\_Scalar\_T &val)  
*Cast to double.*
- long double [to\\_scalar\\_t](#) (const dd\_real &val)  
*Cast to long double.*
- long double [to\\_scalar\\_t](#) (const qd\_real &val)  
*Cast to long double.*
- dd\_real [to\\_scalar\\_t](#) (const long double &val)  
*Cast to dd\_real.*
- dd\_real [to\\_scalar\\_t](#) (const qd\_real &val)  
*Cast to dd\_real.*
- qd\_real [to\\_scalar\\_t](#) (const long double &val)  
*Cast to qd\_real.*
- 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 75 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 239 of file scalar.h.

References `glucat::acos()`.

#### 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 302 of file scalar.h.

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

### 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 323 of file scalar.h.

### 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 344 of file scalar.h.

### 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 218 of file scalar.h.

References glucat::log().

### 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 295 of file scalar.h.

### 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 309 of file scalar.h.

### 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 274 of file scalar.h.

References glucat::tan().

### 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 211 of file scalar.h.

References glucat::exp().

### 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 232 of file scalar.h.

References glucat::cos().



#### 6.23.2.11 isInf() [1/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isInf (
    const Scalar_T & val ) [inline], [static]
```

Smart isinf.

Definition at line 140 of file scalar.h.

#### 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< false > ) [inline], [static], [private]
```

Smart isinf specialised for Scalar\_T without infinity.

Definition at line 111 of file scalar.h.

#### 6.23.2.13 isInf() [3/3]

```
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 118 of file scalar.h.

#### 6.23.2.14 isNaN() [1/3]

```
template<typename Scalar_T >
static bool glucat::numeric_traits< Scalar_T >::isNaN (
    const Scalar_T & val ) [inline], [static]
```

Smart isnan.

Definition at line 150 of file scalar.h.

#### 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< false > ) [inline], [static], [private]
```

Smart isnan specialised for Scalar\_T without quiet NaN.

Definition at line 125 of file scalar.h.

#### 6.23.2.16 isNaN() [3/3]

```
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 132 of file scalar.h.

#### 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 160 of file scalar.h.

#### 6.23.2.18 ln\_2() [1/2]

```
long double glucat::numeric_traits< long double >::ln_2 ( ) [inline]
```

log(2) for long double

Definition at line 111 of file long\_double.h.

### 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 253 of file scalar.h.

References glucat::sin().

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

### 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 281 of file scalar.h.

References glucat::atan().

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

### 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 288 of file scalar.h.

References glucat::tanh().

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

### 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 172 of file scalar.h.

Referenced by glucat::matrix::prod(), and glucat::matrix::sparse\_prod().

**6.23.2.23 pi()** [1/2]

```
long double glucat::numeric_traits< long double >::pi ( ) [inline]
```

Pi for long double.

Definition at line 103 of file long\_double.h.

**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 246 of file scalar.h.

References glucat::cosh().

**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 260 of file scalar.h.

References glucat::asin().

**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 225 of file scalar.h.

References glucat::numeric\_traits< Scalar\_T >::ln\_2(), and glucat::numeric\_traits< Scalar\_T >::log().

### 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 316 of file scalar.h.

### 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 330 of file scalar.h.

### 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 267 of file scalar.h.

References glucat::sinh().

Referenced by glucat::reverse().

### 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 337 of file scalar.h.

### 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 351 of file scalar.h.

### 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 190 of file scalar.h.

### 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 183 of file scalar.h.

### 6.23.2.34 to\_scalar\_t() [1/9]

```
long double glucat::numeric_traits< long double >::to_scalar_t (
    const dd_real & val ) [inline]
```

Cast to long double.

Definition at line 99 of file scalar\_imp.h.

**6.23.2.35 to\_scalar\_t()** [2/9]

```
qd_real glucat::numeric_traits< qd_real >::to_scalar_t (
    const dd_real & val ) [inline]
```

Cast to qd\_real.

Definition at line 144 of file scalar\_imp.h.

**6.23.2.36 to\_scalar\_t()** [3/9]

```
dd_real glucat::numeric_traits< dd_real >::to_scalar_t (
    const long double & val ) [inline]
```

Cast to dd\_real.

Definition at line 117 of file scalar\_imp.h.

**6.23.2.37 to\_scalar\_t()** [4/9]

```
qd_real glucat::numeric_traits< qd_real >::to_scalar_t (
    const long double & val ) [inline]
```

Cast to qd\_real.

Definition at line 135 of file scalar\_imp.h.

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

**6.23.2.38 to\_scalar\_t()** [5/9]

```
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 80 of file scalar\_imp.h.

**6.23.2.39 to\_scalar\_t()** [6/9]

```
double glucat::numeric_traits< double >::to_scalar_t (
    const Other_Scalar_T & val ) [inline]
```

Cast to double.

Definition at line 89 of file scalar\_imp.h.

**6.23.2.40 to\_scalar\_t()** [7/9]

```
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 198 of file scalar.h.

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

**6.23.2.41 to\_scalar\_t()** [8/9]

```
long double glucat::numeric_traits< long double >::to_scalar_t (
    const qd_real & val ) [inline]
```

Cast to long double.

Definition at line 108 of file scalar\_imp.h.

**6.23.2.42 to\_scalar\_t()** [9/9]

```
dd_real glucat::numeric_traits< dd_real >::to_scalar_t (
    const qd_real & val ) [inline]
```

Cast to dd\_real.

Definition at line 126 of file scalar\_imp.h.

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 202 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 202 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 76 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 141 of file random.h.

### 6.25.2.3 ~random\_generator()

```
template<typename Scalar_T >  
glucat::random_generator< Scalar_T >::~~random_generator ( ) [inline], [private]
```

Definition at line 145 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 109 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 151 of file random.h.

### 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 149 of file random.h.

### 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 114 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 139 of file random.h.

#### 6.25.5.2 seed

```
template<typename Scalar_T >
const unsigned long glucat::random_generator< Scalar_T >::seed = 19590921UL [static], [private]
```

Definition at line 117 of file random.h.

### 6.25.5.3 uint\_gen

```
template<typename Scalar_T >
std::mt19937 glucat::random_generator< Scalar_T >::uint_gen [private]
```

Definition at line 137 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 138 of file random.h.

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

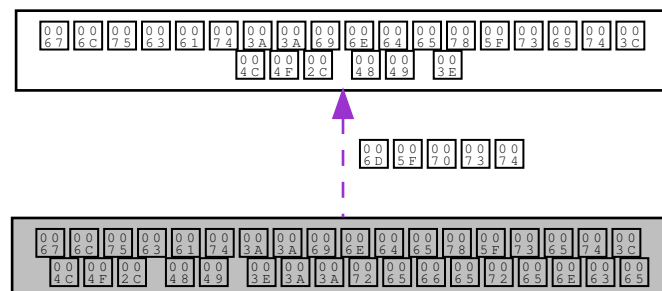
- glucat/[random.h](#)

## 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 206 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 1011 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 213 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 1064 of file index\_set\_imp.h.

Referenced by glucat::index\_set< LO, HI >::reference::operator=().

### 6.26.3.2 operator bool()

```
template<const index_t LO, const index_t HI>
glucat::index_set< LO, HI >::reference::operator bool  [inline]
```

for x = b[i];

Definition at line 1056 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 1021 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 1035 of file index\_set\_imp.h.

References glucat::index\_set< LO, HI >::reference::flip().

### 6.26.3.5 operator~()

```
template<const index_t LO, const index_t HI>
bool glucat::index_set< LO, HI >::reference::operator~  [inline]
```

Flips a bit.

flips the bit

Definition at line 1049 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 207 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 227 of file index\_set.h.

#### 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 226 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

## 6.27 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 1356 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 1359 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 1361 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 1360 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 1363 of file framed\_multi\_imp.h.

### 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 1373 of file framed\_multi\_imp.h.

#### 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 1374 of file framed\_multi\_imp.h.

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 1378 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 1381 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 1383 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 1382 of file framed\_multi\_imp.h.

#### 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 1385 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 1389 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 1390 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 Public Attributes

- 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 180 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 203 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 206 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 184 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 187 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 190 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 193 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 195 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 198 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 201 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 209 of file global.h.

Referenced by glucat::exp(), glucat::log(), and glucat::sqrt().

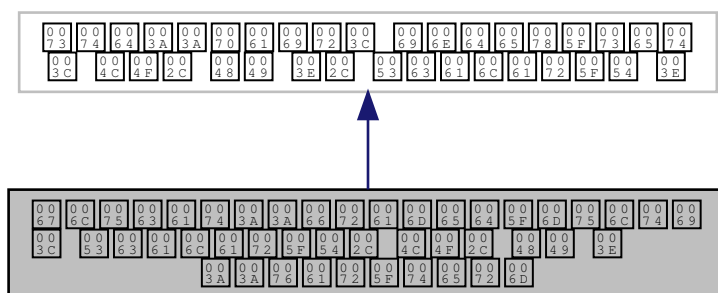
The documentation for this struct was generated from the following file:

- [glucat/global.h](#)

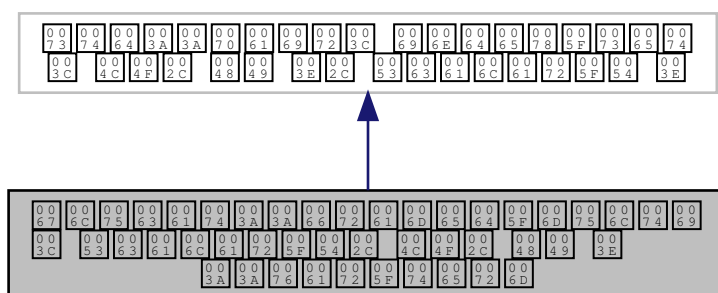
## 6.30 glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term Class Reference

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](#)





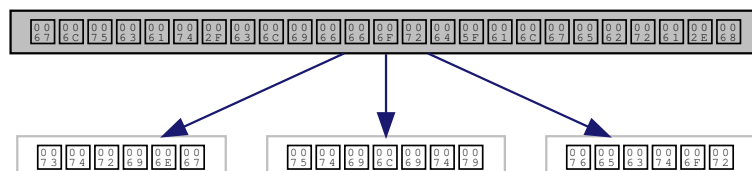
## Chapter 7

# File Documentation

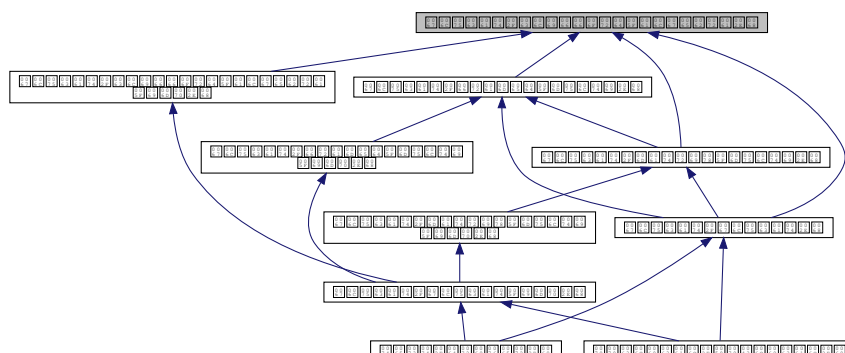
### 7.1 glucat/clifford\_algebra.h File Reference

```
#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.  $\{1\}*\{2\} \rightarrow \{2\}*\{1\}$ .*
- `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::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 > glucat::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 > glucat::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 > glucat::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 > glucat::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 > 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
```

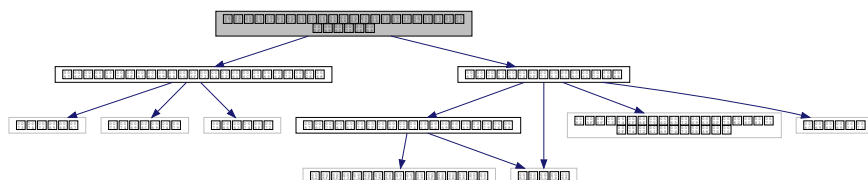
Definition at line 164 of file clifford\_algebra.h.

## 7.2 glucat/clifford\_algebra\_imp.h File Reference

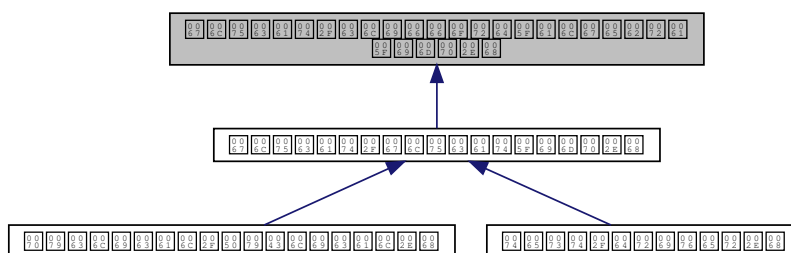
```
#include <glucat/clifford_algebra.h>
```

```
#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)`

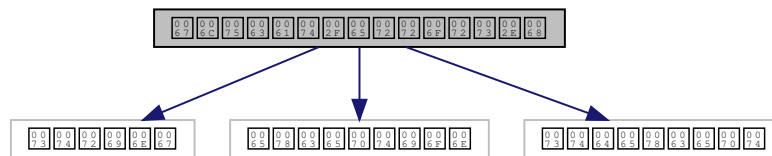
*Inverse cosine of multivector.*

- `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)`  
*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 > glucat::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 > 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.*
- `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)`  
*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::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 > glucat::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 > 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.*
- `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)`  
*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::tan (const Multivector< Scalar_T, LO, HI > &val)`  
*Tangent of multivector.*

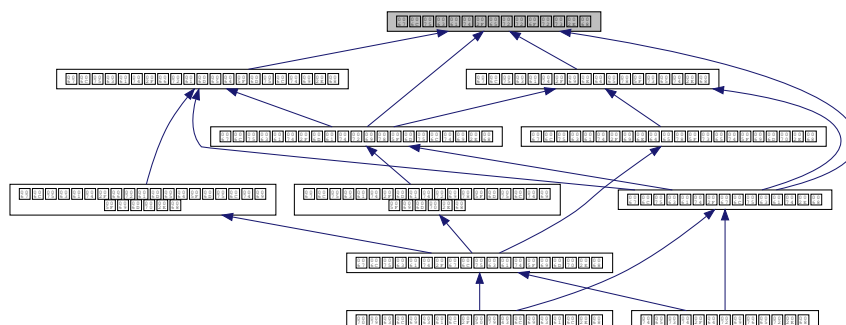
- `template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>`  
`const Multivector< Scalar_T, LO, HI > glucat::atan (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)`  
*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 > glucat::atan (const Multivector< Scalar_T, LO, HI > &val)`  
*Inverse tangent of multivector.*

## 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

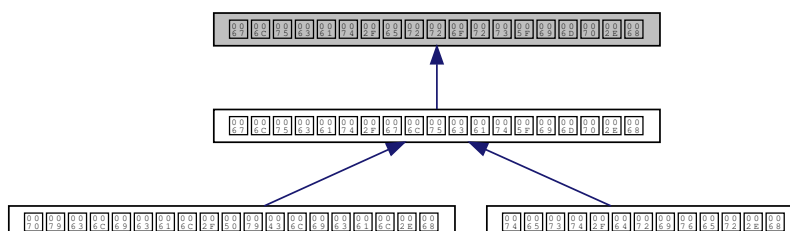
- class [glucat::glucat\\_error](#)  
*Abstract exception class.*
- class [glucat::error< Class\\_T >](#)  
*Specific exception class.*

## Namespaces

- [glucat](#)

## 7.4 glucat/errors\_imp.h File Reference

This graph shows which files directly or indirectly include this file:



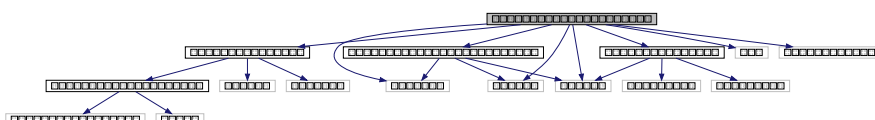
## Namespaces

- [glucat](#)

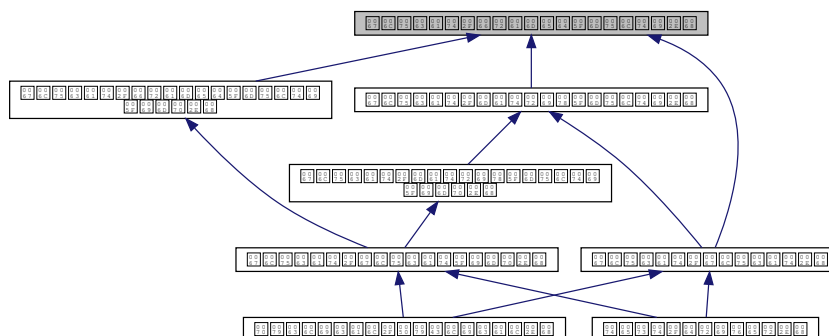
## 7.5 glucat/framed\_multi.h File Reference

```
#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

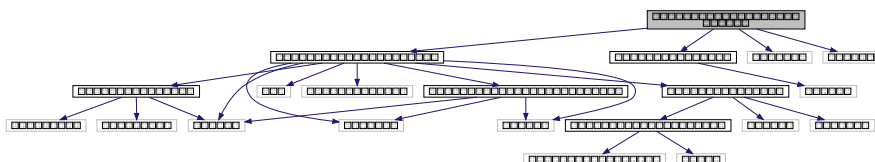
### 7.5.1.1 \_GLUCAT\_MAP\_IS\_HASH

```
#define _GLUCAT_MAP_IS_HASH
```

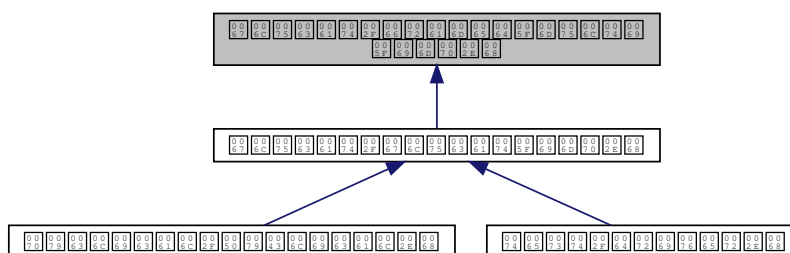
Definition at line 56 of file framed\_multi.h.

## 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 91 of file framed\_multi\_imp.h.

### 7.6.1.2 \_GLUCAT\_HASH\_SIZE\_T

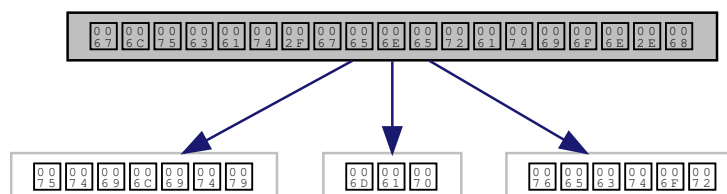
```
#define _GLUCAT_HASH_SIZE_T(  
    x )
```

Definition at line 92 of file framed\_multi\_imp.h.

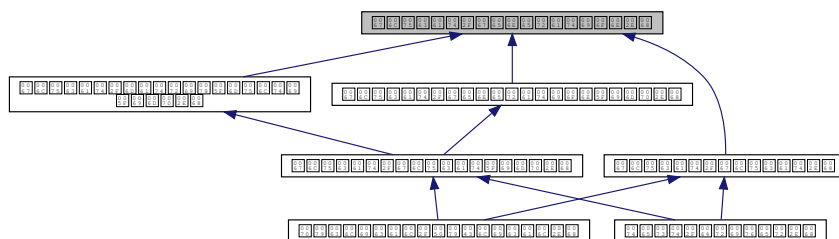
## 7.7 glucat/generation.h File Reference

```
#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

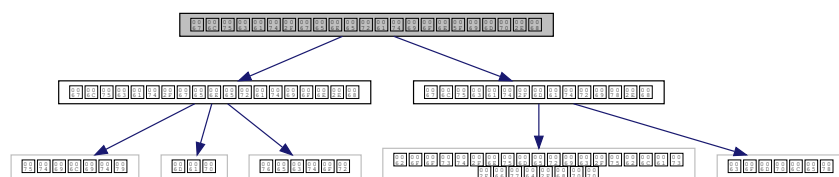
- `typedef std::pair< index_t, index_t > glucat::gen::signature_t`  
*A signature is a pair of indices, p, q, with p == frame.max(), q == -frame.min()*

## Variables

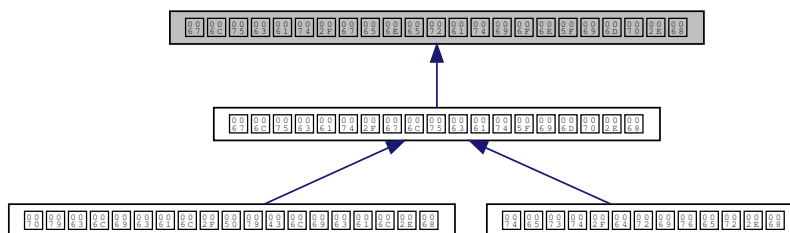
- static const index\_t **glucat::gen::offset\_to\_super** [] = {0,-1, 0,-1,-2, 3, 2, 1}
- Offsets between the current signature and that of the real superalgebra.*

## 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:

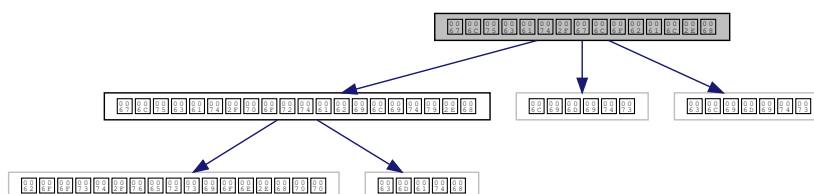


## Namespaces

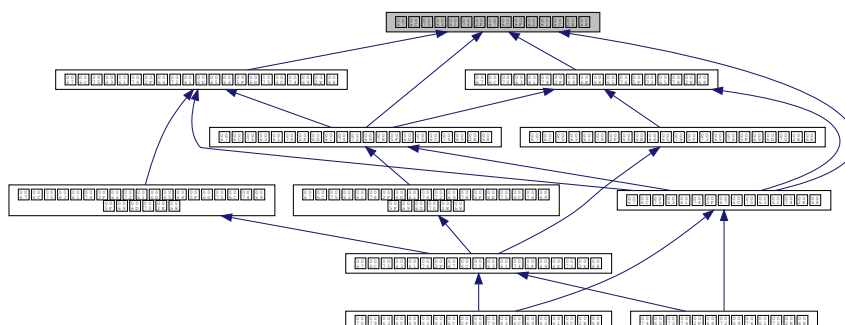
- [glucat](#)
- [glucat::gen](#)

## 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 78 of file global.h.

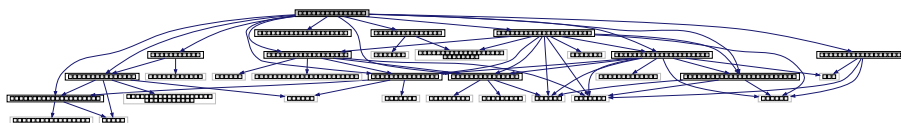
## 7.10 glucat/glucat.h File Reference

```
#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"
```

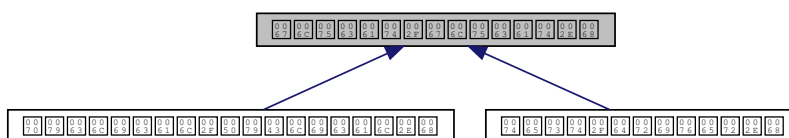


```
#include "glucat/matrix_multi.h"
```

Include dependency graph for glucat.h:

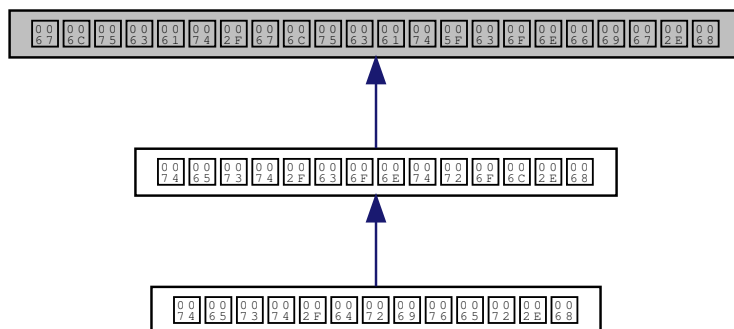


This graph shows which files directly or indirectly include this file:



## 7.11 glucat/glucat\_config.h File Reference

This graph shows which files directly or indirectly include this file:



### 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 0.8.4"`
- `#define GLUCAT_PACKAGE_TARNAME "glucat"`
- `#define GLUCAT_PACKAGE_URL ""`
- `#define GLUCAT_PACKAGE_VERSION "0.8.4"`
- `#define GLUCAT_STDC_HEADERS 1`
- `#define GLUCAT_VERSION "0.8.4"`

### 7.11.1 Macro Definition Documentation

#### 7.11.1.1 GLUCAT\_HAVE\_INTTYPES\_H

```
#define GLUCAT_HAVE_INTTYPES_H 1
```

Definition at line 11 of file glucat\_config.h.

#### 7.11.1.2 GLUCAT\_HAVE\_MEMORY\_H

```
#define GLUCAT_HAVE_MEMORY_H 1
```

Definition at line 19 of file glucat\_config.h.

#### 7.11.1.3 GLUCAT\_HAVE\_STDINT\_H

```
#define GLUCAT_HAVE_STDINT_H 1
```

Definition at line 24 of file glucat\_config.h.

#### 7.11.1.4 GLUCAT\_HAVE\_STDLIB\_H

```
#define GLUCAT_HAVE_STDLIB_H 1
```

Definition at line 29 of file glucat\_config.h.

#### 7.11.1.5 GLUCAT\_HAVE\_STRING\_H

```
#define GLUCAT_HAVE_STRING_H 1
```

Definition at line 39 of file glucat\_config.h.

#### 7.11.1.6 GLUCAT\_HAVE\_STRINGS\_H

```
#define GLUCAT_HAVE_STRINGS_H 1
```

Definition at line 34 of file glucat\_config.h.

#### 7.11.1.7 GLUCAT\_HAVE\_SYS\_STAT\_H

```
#define GLUCAT_HAVE_SYS_STAT_H 1
```

Definition at line 44 of file glucat\_config.h.

#### 7.11.1.8 GLUCAT\_HAVE\_SYS\_TYPES\_H

```
#define GLUCAT_HAVE_SYS_TYPES_H 1
```

Definition at line 49 of file glucat\_config.h.

#### 7.11.1.9 GLUCAT\_HAVE\_UNISTD\_H

```
#define GLUCAT_HAVE_UNISTD_H 1
```

Definition at line 54 of file glucat\_config.h.

#### 7.11.1.10 GLUCAT\_PACKAGE

```
#define GLUCAT_PACKAGE "glucat"
```

Definition at line 59 of file glucat\_config.h.

#### 7.11.1.11 GLUCAT\_PACKAGE\_BUGREPORT

```
#define GLUCAT_PACKAGE_BUGREPORT ""
```

Definition at line 64 of file glucat\_config.h.

#### 7.11.1.12 GLUCAT\_PACKAGE\_NAME

```
#define GLUCAT_PACKAGE_NAME "glucat"
```

Definition at line 69 of file glucat\_config.h.

#### 7.11.1.13 GLUCAT\_PACKAGE\_STRING

```
#define GLUCAT_PACKAGE_STRING "glucat 0.8.4"
```

Definition at line 74 of file glucat\_config.h.

#### 7.11.1.14 GLUCAT\_PACKAGE\_TARNAME

```
#define GLUCAT_PACKAGE_TARNAME "glucat"
```

Definition at line 79 of file glucat\_config.h.

#### 7.11.1.15 GLUCAT\_PACKAGE\_URL

```
#define GLUCAT_PACKAGE_URL ""
```

Definition at line 84 of file glucat\_config.h.

#### 7.11.1.16 GLUCAT\_PACKAGE\_VERSION

```
#define GLUCAT_PACKAGE_VERSION "0.8.4"
```

Definition at line 89 of file glucat\_config.h.

## 7.11.1.17 GLUCAT\_STDC\_HEADERS

```
#define GLUCAT_STDC_HEADERS 1
```

Definition at line 94 of file glucat\_config.h.

## 7.11.1.18 GLUCAT\_VERSION

```
#define GLUCAT_VERSION "0.8.4"
```

Definition at line 99 of file glucat\_config.h.

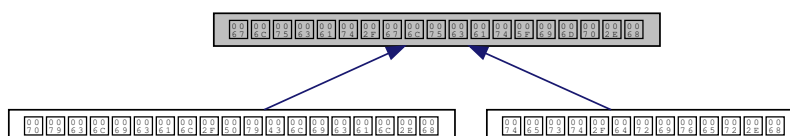
## 7.12 glucat/glucat\_imp.h File Reference

```
#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:



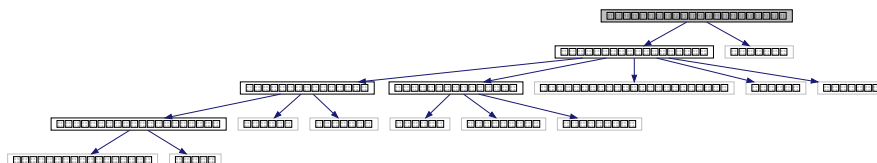


## Functions

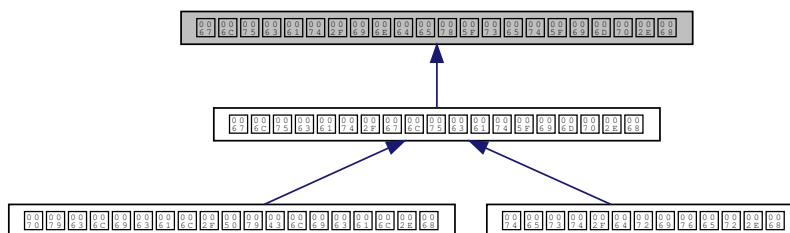
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator^ (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)`  
*Symmetric set difference: exclusive or.*
- `template<const index_t LO, const index_t HI>`  
`const index_set< LO, HI > glucat::operator& (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](#)

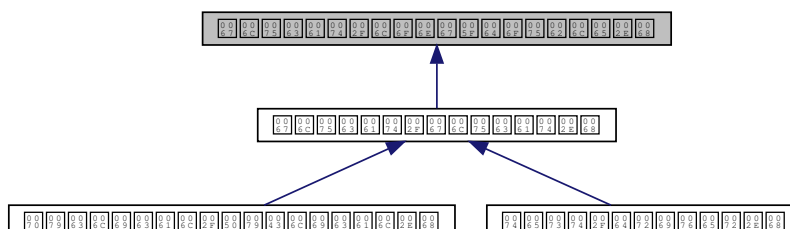
## Functions

- `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

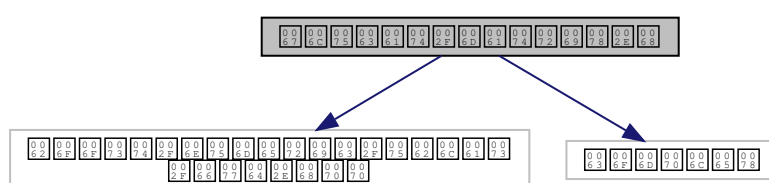
- static const long double [glucat::l\\_pi](#) = 3.1415926535897932384626433832795029L
- static const long double [glucat::l\\_ln2](#) = 0.6931471805599453094172321214581766L

## 7.16 glucat/matrix.h File Reference

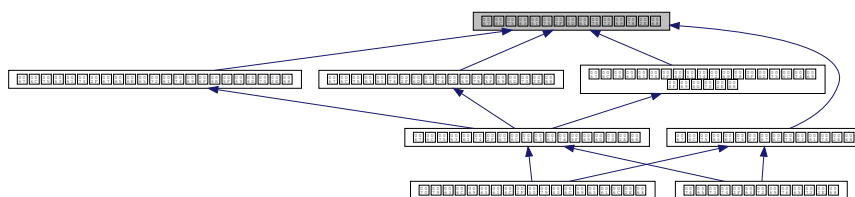
```
#include <boost/numeric/ublas/fwd.hpp>
```

```
#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.*

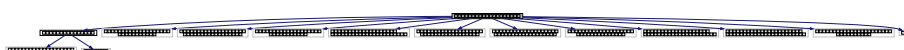
- template<typename Matrix\_T >  
eig\_genus< Matrix\_T > [glucat::matrix::classify\\_eigenvalues](#) (const Matrix\_T &val)

*Classify the eigenvalues of a matrix.*

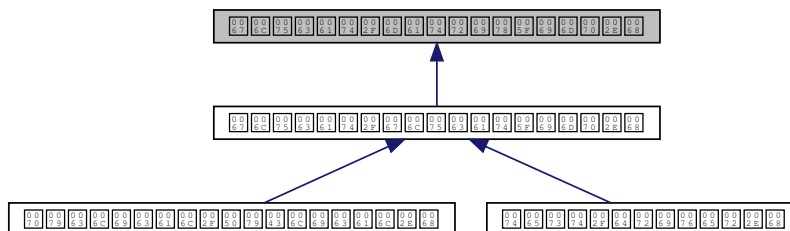
## 7.17 glucat/matrix\_imp.h File Reference

```
#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

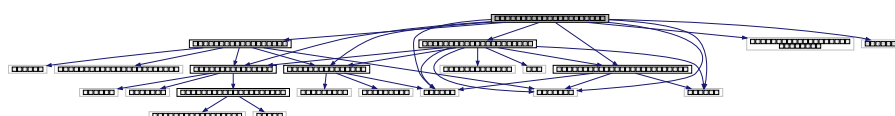
- `template<typename LHS_T, typename RHS_T >`  
`const RHS_T glucat::matrix::kron (const LHS_T &lhs, const RHS_T &rhs)`  
*Kronecker tensor product of matrices - as per Matlab kron.*
- `template<typename LHS_T, typename RHS_T >`  
`const RHS_T glucat::matrix::mono\_kron (const LHS_T &lhs, const RHS_T &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::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 >`  
`static ublas::matrix< double, ublas::column_major > glucat::matrix::to_lapack (const Matrix_T &val)`  
*Convert matrix to LAPACK format.*
- `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.*

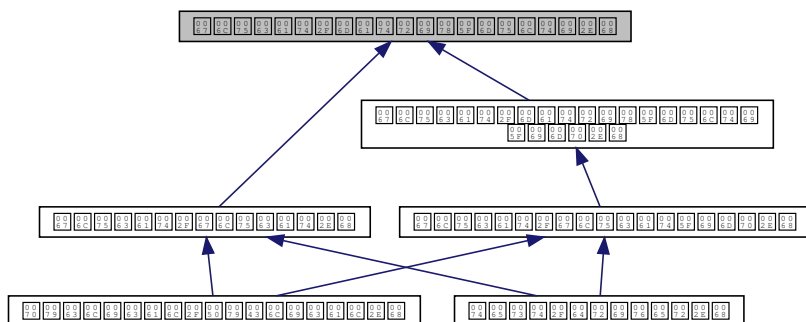
## 7.18 glucat/matrix\_multi.h File Reference

```
#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>
```

Include dependency graph for matrix\_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::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>
```





*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)`

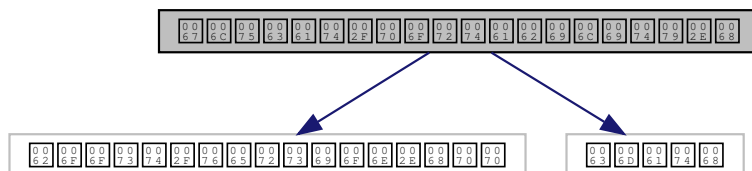
*Exponential of multivector.*

## 7.20 glucat/portability.h File Reference

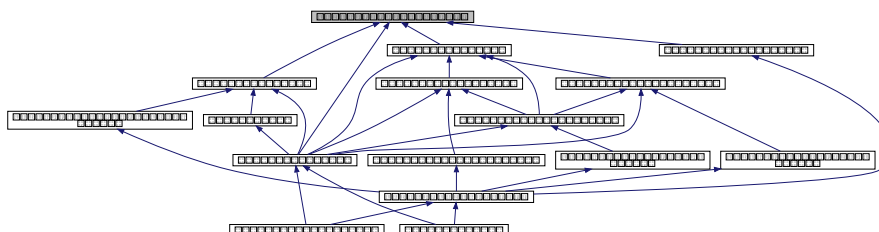
```
#include <boost/version.hpp>
```

```
#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 49 of file portability.h.

### 7.20.1.2 `_GLUCAT_ISNAN`

```
#define _GLUCAT_ISNAN(  
    x ) ( x != x )
```

Definition at line 48 of file portability.h.

### 7.20.1.3 `UBLAS_ABS`

```
#define UBLAS_ABS abs
```

Definition at line 57 of file portability.h.

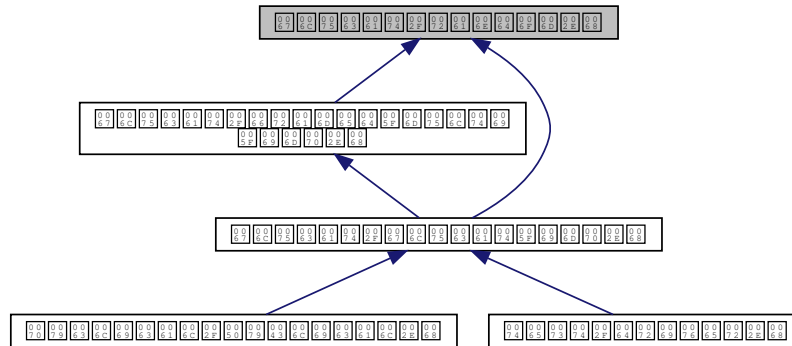
### 7.20.1.4 `UBLAS_SQRT`

```
#define UBLAS_SQRT sqrt
```

Definition at line 58 of file portability.h.



This graph shows which files directly or indirectly include this file:



## Classes

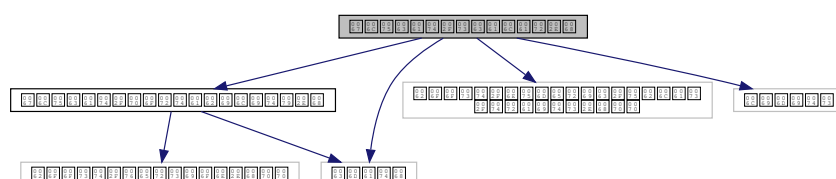
- class `glucat::random_generator< Scalar_T >`  
Random number generator with single instance per `Scalar_T`.

## Namespaces

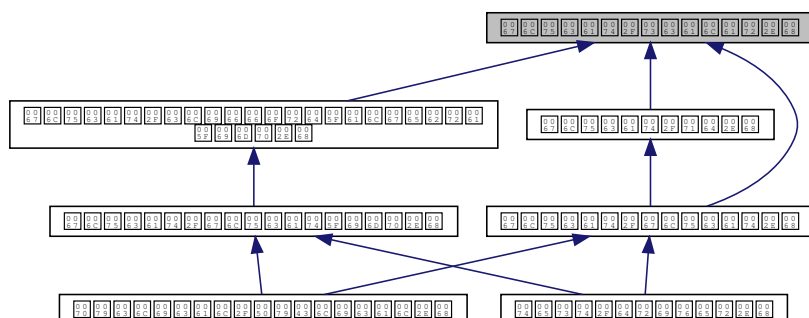
- `glucat`

## 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](#)

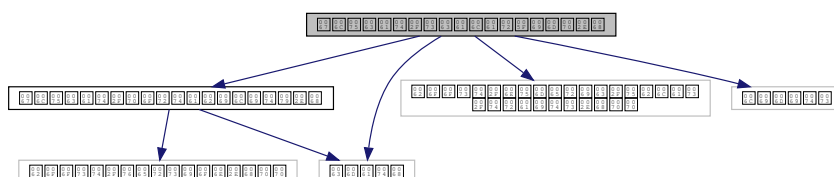
## Functions

- template<typename Scalar\_T >  
Scalar\_T [glucat::log2](#) (const Scalar\_T &x)  
*Log base 2 of scalar.*

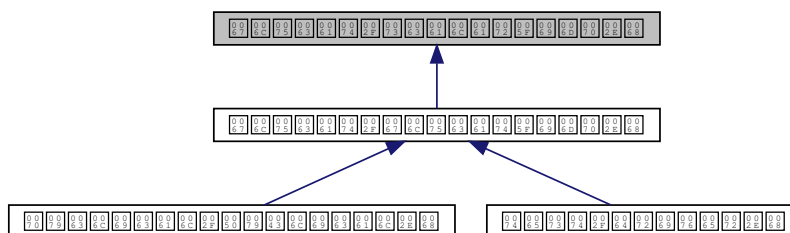
## 7.24 glucat/scalar\_imp.h File Reference

```
#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

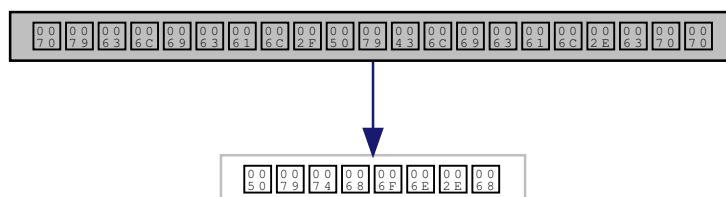
## Namespaces

- 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 69 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 <limits>
```

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`
- `const scalar_t epsilon = std::numeric_limits<scalar_t>::epsilon()`

### 7.27.1 Typedef Documentation

#### 7.27.1.1 Clifford

```
typedef matrix_multi<scalar_t> Clifford
```

Definition at line 161 of file PyClical.h.

#### 7.27.1.2 IndexSet

```
typedef index_set<lo_ndx, hi_ndx> IndexSet
```

Definition at line 158 of file PyClical.h.

### 7.27.1.3 scalar\_t

```
typedef double scalar_t
```

Definition at line 160 of file PyClical.h.

### 7.27.1.4 String

```
typedef std::string String
```

Definition at line 66 of file PyClical.h.

### 7.27.1.5 Tune\_P

```
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
```

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 88 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 99 of file PyClical.h.

References `glucat::abs()`, `PyClical::e()`, and `epsilon`.

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 70 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 79 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 60 of file PyClical.h.

## 7.27.3 Variable Documentation

### 7.27.3.1 epsilon

```
const scalar_t epsilon = std::numeric_limits<scalar_t>::epsilon()
```

Definition at line 163 of file PyClical.h.

Referenced by `glucat::cascade_log()`, and `clifford_to_str()`.

### 7.27.3.2 hi\_ndx

```
const index_t hi_ndx = DEFAULT_HI
```

Definition at line 157 of file PyClical.h.

### 7.27.3.3 lo\_ndx

```
const index_t lo_ndx = DEFAULT_LO
```

Definition at line 156 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](#) ()

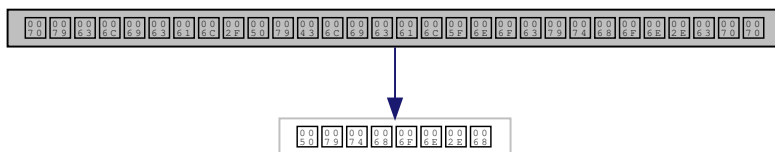
## Variables

- string [PyClical.\\_\\_version\\_\\_](#) = "0.8.4"
- [PyClical.obj](#)
- [PyClical.i](#)
- [PyClical.ixt](#)
- [PyClical.fill](#)
- [PyClical.scalar\\_epsilon](#) = [epsilon](#)
- float [PyClical.pi](#) =  $\text{atan}(\text{clifford}(1.0)) * 4.0$
- float [PyClical.tau](#) =  $\text{atan}(\text{clifford}(1.0)) * 8.0$
- [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
```

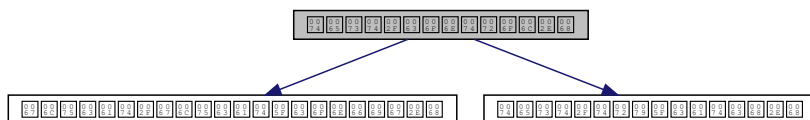
Definition at line 67 of file PyClical\_nocython.cpp.

## 7.31 test/control.h File Reference

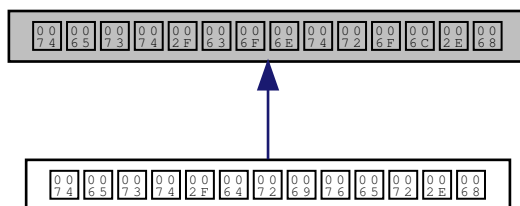
```
#include "glucat/glucat_config.h"
```

```
#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)

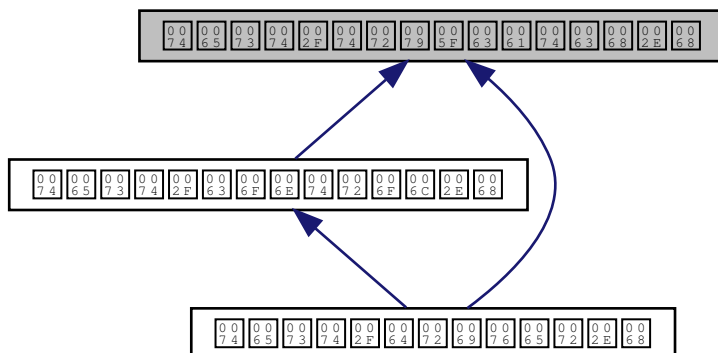
*Timing constant: milliseconds per clock.*

- 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.*

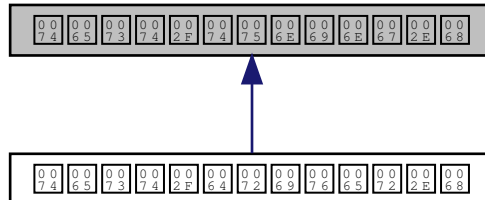
## Functions

- 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 42 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 (  
    Basis_Max_Count )
```

**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 (
    Fast_Size_Threshold )
```

**7.35.3.4 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT() [4/9]**

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Inv_Fast_Dim_Threshold )
```

**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 (
    Log_Max_Outer_Steps )
```

**7.35.3.7 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT() [7/9]**

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Mult_Matrix_Threshold )
```

**7.35.3.8 \_\_TEST\_TUNING\_DEFAULT\_CONSTANT() [8/9]**

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Products_Size_Threshold )
```

### 7.35.3.9 `__TEST_TUNING_DEFAULT_CONSTANT()` [9/9]

```
__TEST_TUNING_DEFAULT_CONSTANT (
    Sqrt_Max_Steps )
```

### 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`, 231

`_GLUCAT_CTAssert`  
    `global.h`, 248  
    `glucat`, 22, 23  
    `tuning.h`, 284

`_GLUCAT_HASH_N`  
    `framed_multi_imp.h`, 244

`_GLUCAT_HASH_SIZE_T`  
    `framed_multi_imp.h`, 244

`_GLUCAT_ISINF`  
    `portability.h`, 267

`_GLUCAT_ISNAN`  
    `portability.h`, 267

`_GLUCAT_MAP_IS_HASH`  
    `framed_multi.h`, 241

`_TEST_TUNING_DEFAULT_CONSTANT`  
    `tuning.h`, 282, 283

`__add__`  
    `PyClical.clifford`, 80

`__and__`  
    `PyClical.clifford`, 80  
    `PyClical.index_set`, 160

`__call__`  
    `PyClical.clifford`, 80

`__cinit__`  
    `PyClical.clifford`, 81  
    `PyClical.index_set`, 160

`__contains__`  
    `PyClical.clifford`, 81  
    `PyClical.index_set`, 160

`__dealloc__`  
    `PyClical.clifford`, 82  
    `PyClical.index_set`, 161

`__getitem__`  
    `PyClical.clifford`, 82  
    `PyClical.index_set`, 161

`__iadd__`  
    `PyClical.clifford`, 82

`__iand__`  
    `PyClical.clifford`, 83  
    `PyClical.index_set`, 161

`__idiv__`  
    `PyClical.clifford`, 83

`__imod__`  
    `PyClical.clifford`, 83

`__imul__`  
    `PyClical.clifford`, 84

`__invert__`  
    `PyClical.index_set`, 162

`__ior__`  
    `PyClical.clifford`, 84  
    `PyClical.index_set`, 162

`__isub__`  
    `PyClical.clifford`, 84

`__iter__`  
    `PyClical.clifford`, 85  
    `PyClical.index_set`, 162

`__ixor__`  
    `PyClical.clifford`, 85  
    `PyClical.index_set`, 163

`__mod__`  
    `PyClical.clifford`, 85

`__mul__`  
    `PyClical.clifford`, 86

`__neg__`  
    `PyClical.clifford`, 86

`__or__`  
    `PyClical.clifford`, 86  
    `PyClical.index_set`, 163

`__pos__`  
    `PyClical.clifford`, 87

`__pow__`  
    `PyClical.clifford`, 87

`__repr__`  
    `PyClical.clifford`, 87  
    `PyClical.index_set`, 163

`__richcmp__`  
    `PyClical.clifford`, 88  
    `PyClical.index_set`, 164

`__setitem__`  
    `PyClical.index_set`, 164

`__str__`  
    `PyClical.clifford`, 88  
    `PyClical.index_set`, 164

`__sub__`  
    `PyClical.clifford`, 88

`__truediv__`  
    `PyClical.clifford`, 89

`__version__`  
    `PyClical`, 71

`__xor__`  
    `PyClical.clifford`, 89  
    `PyClical.index_set`, 165

`_test`  
    `PyClical`, 68

`~basis_table`  
    `glucat::basis_table< Scalar_T, LO, HI, Matrix_T >`,

- 76
- ~clifford\_algebra
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 99
- ~control\_t
  - glucat::control\_t, 110
- ~framed\_multi
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 125
- ~generator\_table
  - glucat::gen::generator\_table< Matrix\_T >, 137
- ~glucat\_error
  - glucat::glucat\_error, 141
- ~matrix\_multi
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 175
- ~random\_generator
  - glucat::random\_generator< Scalar\_T >, 203
- ~reference
  - glucat::index\_set< LO, HI >::reference, 207
- ~var\_term
  - glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, 222
- abs
  - glucat, 23
  - glucat::numeric\_traits< Scalar\_T >, 190
  - PyClical.clifford, 89
- acos
  - glucat, 23
  - glucat::numeric\_traits< Scalar\_T >, 190
- acosh
  - glucat, 24
- agc3
  - cga3, 9
- are\_same
  - glucat::compare\_types< LHS\_T, RHS\_T >, 108
  - glucat::compare\_types< T, T >, 109
- asin
  - glucat, 24, 25
  - glucat::numeric\_traits< Scalar\_T >, 190
- asinh
  - glucat, 25
- atan
  - glucat, 26
  - glucat::numeric\_traits< Scalar\_T >, 191
- atanh
  - glucat, 26
- basis
  - glucat::basis\_table< Scalar\_T, LO, HI, Matrix\_T >, 76
- basis\_element
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 180
- basis\_matrix\_t
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 173
- basis\_max\_count
  - 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 >, 219
- basis\_table
  - glucat::basis\_table< Scalar\_T, LO, HI, Matrix\_T >, 76
- BITS\_PER\_SET\_VALUE
  - glucat, 56
- bitset\_t
  - glucat::index\_set< LO, HI >, 146
- BOOST\_STATIC\_ASSERT
  - glucat::index\_set< LO, HI >, 149
- both\_eig\_case
  - glucat::matrix, 61
- call
  - glucat::control\_t, 111
- cascade\_log
  - glucat, 27
- catch\_exceptions
  - glucat::control\_t, 111
- centre\_pm4\_qp4
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 129
- centre\_pp4\_qm4
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 129
- centre\_qp1\_pm1
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 129
- cga3, 9
  - agc3, 9
  - cga3, 9
  - cga3std, 10
- cga3std
  - cga3, 10
- check\_complex
  - glucat, 27
- cl
  - PyClical, 71
- classify\_eigenvalues
  - glucat::matrix, 62
- classname
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 100
  - glucat::error< Class\_T >, 118
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 130
  - glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, 223
  - glucat::glucat\_error, 141
  - glucat::index\_set< LO, HI >, 149
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 180
- Clifford
  - PyClical.h, 273
- clifford\_algebra.h
  - \_GLUCAT\_CLIFFORD\_ALGEBRA\_OPERATIONS, 231
- clifford\_exp
  - glucat, 27
- clifford\_hidden\_doctests
  - PyClical, 68
- clifford\_to\_repr
  - PyClical.h, 274

- clifford\_to\_str
  - PyClical.h, [274](#)
- compare
  - glucat, [28](#)
  - glucat::index\_set< LO, HI >, [157](#)
- complexifier
  - glucat, [28](#)
- conj
  - glucat, [28](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [100](#)
  - glucat::numeric\_traits< Scalar\_T >, [191](#)
  - PyClical.clifford, [90](#)
- const\_iterator
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [122](#)
- control
  - glucat::control\_t, [111](#)
- control\_t
  - glucat::control\_t, [110](#), [111](#)
- cos
  - glucat, [28](#), [29](#)
  - glucat::numeric\_traits< Scalar\_T >, [191](#)
- cosh
  - glucat, [29](#)
  - glucat::numeric\_traits< Scalar\_T >, [191](#)
- count
  - glucat::index\_set< LO, HI >, [149](#)
  - PyClical.index\_set, [165](#)
- count\_neg
  - glucat::index\_set< LO, HI >, [149](#)
  - PyClical.index\_set, [165](#)
- count\_pos
  - glucat::index\_set< LO, HI >, [150](#)
  - PyClical.index\_set, [166](#)
- crd\_of\_mult
  - glucat, [29](#), [30](#)
- db\_sqrt
  - glucat, [30](#)
- db\_step
  - glucat, [30](#)
- DEFAULT\_Basis\_Max\_Count
  - glucat, [56](#)
- DEFAULT\_Div\_Max\_Steps
  - glucat, [56](#)
- DEFAULT\_Fast\_Size\_Threshold
  - glucat, [56](#)
- DEFAULT\_Function\_Precision
  - glucat, [57](#)
- DEFAULT\_HI
  - glucat, [57](#)
- DEFAULT\_Inv\_Fast\_Dim\_Threshold
  - glucat, [57](#)
- DEFAULT\_Log\_Max\_Inner\_Steps
  - glucat, [57](#)
- DEFAULT\_Log\_Max\_Outer\_Steps
  - glucat, [57](#)
- DEFAULT\_Mult\_Matrix\_Threshold
  - glucat, [57](#)
- DEFAULT\_Products\_Size\_Threshold
  - glucat, [58](#)
- DEFAULT\_Sqrt\_Max\_Steps
  - glucat, [58](#)
- DEFAULT\_TRUNCATION
  - glucat, [58](#)
- div\_max\_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 >, [217](#)
- divide
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [130](#)
- e
  - PyClical, [69](#)
- eig\_case\_t
  - glucat::matrix, [61](#)
- eigenvalues
  - glucat::matrix, [62](#)
- elapsed
  - glucat::timing, [66](#)
- elliptic
  - glucat, [30](#)
- epsilon
  - PyClical.h, [276](#)
- error
  - glucat::error< Class\_T >, [118](#)
- error\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [122](#)
  - glucat::index\_set< LO, HI >, [147](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [173](#)
- even
  - glucat, [31](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [100](#)
  - PyClical.clifford, [90](#)
- exp
  - glucat, [31](#)
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [132](#)
  - glucat::numeric\_traits< Scalar\_T >, [192](#)
- EXTRA\_TRIALS
  - glucat::timing, [67](#)
- fast
  - glucat, [31](#)
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [130](#)
- fast\_framed\_multi
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [130](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [180](#)
- fast\_matrix\_multi
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [131](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [181](#)
- fast\_size\_threshold
  - 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 >, 220
- fill
  - PyClical, 72
- flip
  - glucat::index\_set< LO, HI >, 150
  - glucat::index\_set< LO, HI >::reference, 207
- fmod
  - glucat::numeric\_traits< Scalar\_T >, 192
- fold
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 131
  - glucat::index\_set< LO, HI >, 150, 151
- folded\_dim
  - glucat, 32
- frame
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 100
  - PyClical.clifford, 90
- framed\_multi
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 125–129, 132
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 182
- framed\_multi.h
  - \_GLUCAT\_MAP\_IS\_HASH, 241
- framed\_multi\_imp.h
  - \_GLUCAT\_HASH\_N, 244
  - \_GLUCAT\_HASH\_SIZE\_T, 244
- framed\_multi\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 122
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 173
- framed\_pair\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 122
- friend\_for\_private\_destructor
  - glucat::basis\_table< Scalar\_T, LO, HI, Matrix\_T >, 77
  - glucat::control\_t, 112
  - glucat::gen::generator\_table< Matrix\_T >, 139
  - glucat::random\_generator< Scalar\_T >, 204
- function\_precision
  - 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 >, 220
- gen\_from\_pm1\_qm1
  - glucat::gen::generator\_table< Matrix\_T >, 137
- gen\_from\_pm4\_qp4
  - glucat::gen::generator\_table< Matrix\_T >, 138
- gen\_from\_pp4\_qm4
  - glucat::gen::generator\_table< Matrix\_T >, 138
- gen\_from\_qp1\_pm1
  - glucat::gen::generator\_table< Matrix\_T >, 138
- gen\_vector
  - glucat::gen::generator\_table< Matrix\_T >, 138
- generator
  - glucat::gen::generator\_table< Matrix\_T >, 139
- glucat::random\_generator< Scalar\_T >, 203
- generator\_table
  - glucat::gen::generator\_table< Matrix\_T >, 137
- global.h
  - \_GLUCAT\_CTAssert, 248
- glucat, 10
  - \_GLUCAT\_CTAssert, 22, 23
  - abs, 23
  - acos, 23
  - acosh, 24
  - asin, 24, 25
  - asinh, 25
  - atan, 26
  - atanh, 26
  - BITS\_PER\_SET\_VALUE, 56
  - cascade\_log, 27
  - check\_complex, 27
  - clifford\_exp, 27
  - compare, 28
  - complexifier, 28
  - conj, 28
  - cos, 28, 29
  - cosh, 29
  - crd\_of\_mult, 29, 30
  - db\_sqrt, 30
  - db\_step, 30
  - DEFAULT\_Basis\_Max\_Count, 56
  - DEFAULT\_Div\_Max\_Steps, 56
  - DEFAULT\_Fast\_Size\_Threshold, 56
  - DEFAULT\_Function\_Precision, 57
  - DEFAULT\_HI, 57
  - DEFAULT\_Inv\_Fast\_Dim\_Threshold, 57
  - DEFAULT\_Log\_Max\_Inner\_Steps, 57
  - DEFAULT\_Log\_Max\_Outer\_Steps, 57
  - DEFAULT\_Mult\_Matrix\_Threshold, 57
  - DEFAULT\_Products\_Size\_Threshold, 58
  - DEFAULT\_Sqrt\_Max\_Steps, 58
  - DEFAULT\_TRUNCATION, 58
  - elliptic, 30
  - even, 31
  - exp, 31
  - fast, 31
  - folded\_dim, 32
  - imag, 32
  - index\_t, 21
  - intfn, 21
  - intintfn, 21
  - inv, 32
  - inverse\_gray, 32
  - inverse\_reversed\_gray, 33
  - involute, 33
  - I\_ln2, 58
  - I\_pi, 58
  - log, 33, 34
  - log2, 34
  - matrix\_log, 35
  - matrix\_sqrt, 35
  - max\_abs, 35



- max\_pos, 36
- min\_neg, 36
- MS\_PER\_S, 58
- norm, 36
- odd, 36
- offset\_level, 37
- operator!=, 37
- operator<<, 44
- operator>>, 45
- operator\*, 39–41
- operator<sup>^</sup>, 45, 46
- operator+, 41
- operator-, 42
- operator/, 42, 43
- operator%, 38
- operator&, 38, 39
- operator|, 46, 47
- outer\_pow, 47
- pade\_approx, 48
- pade\_log, 48
- pos\_mod, 48
- pow, 48, 49
- precision\_demoted, 22
- precision\_promoted, 22
- precision\_same, 22
- precision\_t, 22
- pure, 49
- quad, 49
- real, 49
- reframe, 50
- reverse, 50
- scalar, 50
- set\_value\_t, 22
- sign\_of\_square, 50
- sin, 51
- sinh, 51
- sqrt, 52, 53
- star, 53
- tan, 54
- tanh, 54
- to\_demote, 55
- to\_promote, 55
- try\_catch, 55
- vector\_part, 56
- glucat/clifford\_algebra.h, 225
- glucat/clifford\_algebra\_imp.h, 232
- glucat/errors.h, 238
- glucat/errors\_imp.h, 239
- glucat/framed\_multi.h, 239
- glucat/framed\_multi\_imp.h, 242
- glucat/generation.h, 244
- glucat/generation\_imp.h, 245
- glucat/global.h, 246
- glucat/glucat.h, 248
- glucat/glucat\_config.h, 249
- glucat/glucat\_imp.h, 253
- glucat/index\_set.h, 254
- glucat/index\_set\_imp.h, 255
- glucat/long\_double.h, 257
- glucat/matrix.h, 257
- glucat/matrix\_imp.h, 259
- glucat/matrix\_multi.h, 261
- glucat/matrix\_multi\_imp.h, 263
- glucat/portability.h, 266
- glucat/qd.h, 268
- glucat/random.h, 268
- glucat/scalar.h, 269
- glucat/scalar\_imp.h, 270
- glucat::basis\_table< Scalar\_T, LO, HI, Matrix\_T >, 75
  - ~basis\_table, 76
  - basis, 76
  - basis\_table, 76
  - friend\_for\_private\_destructor, 77
  - operator=, 77
- glucat::bool\_to\_type< truth\_value >, 77
  - value, 78
- glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multi-
  - vector\_T >, 96
  - ~clifford\_algebra, 99
  - classname, 100
  - conj, 100
  - even, 100
  - frame, 100
  - grade, 100
  - index\_set\_t, 98
  - inv, 101
  - involute, 101
  - isnan, 101
  - max\_abs, 101
  - multivector\_t, 99
  - norm, 101
  - odd, 102
  - operator\*=, 102, 103
  - operator<sup>^</sup>=, 105
  - operator(), 102
  - operator+=, 103
  - operator-, 103
  - operator=, 103
  - operator/=: 104
  - operator==, 104
  - operator%=: 102
  - operator&=: 102
  - operator[], 104
  - operator|=, 105
  - outer\_pow, 105
  - pair\_t, 99
  - pow, 105
  - pure, 105
  - quad, 106
  - reverse, 106
  - scalar, 106
  - scalar\_t, 99
  - truncated, 106
  - vector\_part, 106, 107
  - vector\_t, 99
  - write, 107

- glucat::compare\_types< LHS\_T, RHS\_T >, 107
  - are\_same, 108
- glucat::compare\_types< T, T >, 108
  - are\_same, 109
- glucat::control\_t, 109
  - ~control\_t, 110
  - call, 111
  - catch\_exceptions, 111
  - control, 111
  - control\_t, 110, 111
  - friend\_for\_private\_destructor, 112
  - m\_catch\_exceptions, 113
  - m\_valid, 113
  - m\_verbose\_output, 113
  - operator=, 112
  - valid, 112
  - verbose, 112
- glucat::CTAssertion< bool >, 114
- glucat::CTAssertion< true >, 114
- glucat::error< Class\_T >, 117
  - classname, 118
  - error, 118
  - heading, 119
  - print\_error\_msg, 119
- glucat::framed\_multi< Scalar\_T, LO, HI >, 119
  - ~framed\_multi, 125
  - centre\_pm4\_qp4, 129
  - centre\_pp4\_qm4, 129
  - centre\_qp1\_pm1, 129
  - classname, 130
  - const\_iterator, 122
  - divide, 130
  - error\_t, 122
  - exp, 132
  - fast, 130
  - fast\_framed\_multi, 130
  - fast\_matrix\_multi, 131
  - fold, 131
  - framed\_multi, 125–129, 132
  - framed\_multi\_t, 122
  - framed\_pair\_t, 122
  - index\_set\_t, 122
  - iterator, 123
  - map\_t, 123
  - matrix\_multi, 133
  - matrix\_multi\_t, 123
  - matrix\_t, 123
  - multivector\_t, 123
  - nbr\_terms, 131
  - operator<<, 134
  - operator>>, 134
  - operator\*, 133
  - operator^, 134
  - operator+=, 131
  - operator/, 133
  - operator%, 133
  - operator&, 133
  - operator|, 134
  - random, 132
  - scalar\_t, 124
  - size\_type, 124
  - sorted\_map\_t, 124
  - star, 135
  - term\_t, 124
  - unfold, 132
  - var\_term\_t, 124
  - vector\_t, 125
- glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size\_t, 142
  - hash\_size\_t, 143
  - n, 143
  - operator(), 143
- glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, 221
  - ~var\_term, 222
  - classname, 223
  - operator\*=, 223
  - var\_pair\_t, 222
  - var\_term, 222, 223
- glucat::gen, 59
  - offset\_to\_super, 59
  - signature\_t, 59
- glucat::gen::generator\_table< Matrix\_T >, 135
  - ~generator\_table, 137
  - friend\_for\_private\_destructor, 139
  - gen\_from\_pm1\_qm1, 137
  - gen\_from\_pm4\_qp4, 138
  - gen\_from\_pp4\_qm4, 138
  - gen\_from\_qp1\_pm1, 138
  - gen\_vector, 138
  - generator, 139
  - generator\_table, 137
  - operator(), 139
  - operator=, 139
- glucat::glucat\_error, 140
  - ~glucat\_error, 141
  - classname, 141
  - glucat\_error, 141
  - heading, 142
  - name, 142
  - print\_error\_msg, 142
- glucat::index\_set< LO, HI >, 144
  - bitset\_t, 146
  - BOOST\_STATIC\_ASSERT, 149
  - classname, 149
  - compare, 157
  - count, 149
  - count\_neg, 149
  - count\_pos, 150
  - error\_t, 147
  - flip, 150
  - fold, 150, 151
  - hash\_fn, 151
  - index\_pair\_t, 147
  - index\_set, 147, 148
  - index\_set\_t, 147

- is\_contiguous, 151
- lex\_less\_than, 151
- max, 152
- min, 152
- operator!=, 152
- operator<, 153
- operator~, 154
- operator<sup>^</sup>, 157
- operator<sup>^</sup>=, 154
- operator==, 153
- operator&, 157
- operator&=, 152
- operator[], 153
- operator|, 158
- operator|=, 154
- reference, 158
- reset, 154, 155
- set, 155
- sign\_of\_mult, 156
- sign\_of\_square, 156
- test, 156
- unfold, 156
- v\_hi, 158
- v\_lo, 158
- value\_of\_fold, 157
- glucat::index\_set< LO, HI >::reference, 205
  - ~reference, 207
  - flip, 207
  - index\_set, 208
  - m\_idx, 209
  - m\_pst, 209
  - operator bool, 207
  - operator~, 208
  - operator=, 208
  - reference, 206, 207
- glucat::index\_set\_hash< LO, HI >, 168
  - index\_set\_t, 169
  - operator(), 169
- glucat::matrix, 60
  - both\_eig\_case, 61
  - classify\_eigenvalues, 62
  - eig\_case\_t, 61
  - eigenvalues, 62
  - inner, 62
  - isnan, 62
  - kron, 63
  - mono\_kron, 63
  - mono\_prod, 63
  - negative\_eig\_case, 61
  - nnz, 63
  - nork, 64
  - nork\_range, 64
  - norm\_frob2, 64
  - prod, 64
  - safe\_eig\_case, 61
  - signed\_perm\_nork, 65
  - sparse\_prod, 65
  - to\_lapack, 65
  - trace, 65
  - unit, 66
- glucat::matrix::eig\_genus< Matrix\_T >, 115
  - m\_eig\_case, 116
  - m\_safe\_arg, 116
  - Scalar\_T, 116
- glucat::matrix\_multi< Scalar\_T, LO, HI >, 170
  - ~matrix\_multi, 175
  - basis\_element, 180
  - basis\_matrix\_t, 173
  - classname, 180
  - error\_t, 173
  - fast\_framed\_multi, 180
  - fast\_matrix\_multi, 181
  - framed\_multi, 182
  - framed\_multi\_t, 173
  - index\_set\_t, 173
  - m\_frame, 185
  - m\_matrix, 185
  - matrix\_index\_t, 173
  - matrix\_log, 182
  - matrix\_multi, 175–180, 182
  - matrix\_multi\_t, 174
  - matrix\_sqrt, 182
  - matrix\_t, 174
  - multivector\_t, 174
  - operator<<, 183, 184
  - operator>>, 184
  - operator\*, 183
  - operator<sup>^</sup>, 184
  - operator+=, 181
  - operator/, 183
  - operator=, 181
  - operator%, 183
  - operator&, 183
  - operator|, 184
  - orientation\_t, 174
  - random, 181
  - reframe, 184
  - scalar\_t, 174
  - star, 185
  - term\_t, 175
  - vector\_t, 175
- glucat::numeric\_traits< Scalar\_T >, 188
  - abs, 190
  - acos, 190
  - asin, 190
  - atan, 191
  - conj, 191
  - cos, 191
  - cosh, 191
  - exp, 192
  - fmod, 192
  - imag, 192
  - isInf, 192, 193
  - isNaN, 193, 194
  - isNaN\_or\_isInf, 194
  - ln\_2, 194

- log, 195
- log2, 195
- NaN, 195
- pi, 195, 196
- pow, 196
- real, 196
- sin, 196
- sinh, 197
- sqrt, 197
- tan, 197
- tanh, 197
- to\_double, 198
- to\_int, 198
- to\_scalar\_t, 198–200
- glucat::numeric\_traits< Scalar\_T >::demoted, 114
- type, 115
- glucat::numeric\_traits< Scalar\_T >::promoted, 201
- type, 201
- glucat::random\_generator< Scalar\_T >, 201
  - ~random\_generator, 203
  - friend\_for\_private\_destructor, 204
  - generator, 203
  - normal, 203
  - normal\_dist, 204
  - operator=, 203
  - random\_generator, 202
  - seed, 204
  - uint\_gen, 204
  - uniform, 203
  - uniform\_dist, 205
- glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, 209
  - map\_t, 210
  - sorted\_begin, 211
  - sorted\_end, 211
  - sorted\_iterator, 210
  - sorted\_map\_t, 210
  - sorted\_range, 211
- glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, 211
  - map\_t, 212
  - sorted\_begin, 213
  - sorted\_end, 213
  - sorted\_iterator, 212
  - sorted\_map\_t, 212
  - sorted\_range, 213
- glucat::timing, 66
  - elapsed, 66
  - EXTRA\_TRIALS, 67
  - MS\_PER\_CLOCK, 67
  - MS\_PER\_SEC, 67
- 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 >, 213
  - basis\_max\_count, 219
  - div\_max\_steps, 217
  - fast\_size\_threshold, 220
  - function\_precision, 220
  - inv\_fast\_dim\_threshold, 216
  - log\_max\_inner\_steps, 219
  - log\_max\_outer\_steps, 219
  - mult\_matrix\_threshold, 216
  - products\_size\_threshold, 216
  - sqrt\_max\_steps, 217
- glucat\_config.h
  - GLUCAT\_HAVE\_INTTYPES\_H, 250
  - GLUCAT\_HAVE\_MEMORY\_H, 250
  - GLUCAT\_HAVE\_STDINT\_H, 250
  - GLUCAT\_HAVE\_STDLIB\_H, 250
  - GLUCAT\_HAVE\_STRING\_H, 250
  - GLUCAT\_HAVE\_STRINGS\_H, 251
  - GLUCAT\_HAVE\_SYS\_STAT\_H, 251
  - GLUCAT\_HAVE\_SYS\_TYPES\_H, 251
  - GLUCAT\_HAVE\_UNISTD\_H, 251
  - GLUCAT\_PACKAGE, 251
  - GLUCAT\_PACKAGE\_BUGREPORT, 251
  - GLUCAT\_PACKAGE\_NAME, 252
  - GLUCAT\_PACKAGE\_STRING, 252
  - GLUCAT\_PACKAGE\_TARNAME, 252
  - GLUCAT\_PACKAGE\_URL, 252
  - GLUCAT\_PACKAGE\_VERSION, 252
  - GLUCAT\_STDC\_HEADERS, 252
  - GLUCAT\_VERSION, 253
- glucat\_error
  - glucat::glucat\_error, 141
- GLUCAT\_HAVE\_INTTYPES\_H
  - glucat\_config.h, 250
- GLUCAT\_HAVE\_MEMORY\_H
  - glucat\_config.h, 250
- GLUCAT\_HAVE\_STDINT\_H
  - glucat\_config.h, 250
- GLUCAT\_HAVE\_STDLIB\_H
  - glucat\_config.h, 250
- GLUCAT\_HAVE\_STRING\_H
  - glucat\_config.h, 250
- GLUCAT\_HAVE\_STRINGS\_H
  - glucat\_config.h, 251
- GLUCAT\_HAVE\_SYS\_STAT\_H
  - glucat\_config.h, 251
- GLUCAT\_HAVE\_SYS\_TYPES\_H
  - glucat\_config.h, 251
- GLUCAT\_HAVE\_UNISTD\_H
  - glucat\_config.h, 251
- GLUCAT\_PACKAGE
  - glucat\_config.h, 251
- GLUCAT\_PACKAGE\_BUGREPORT
  - glucat\_config.h, 251
- GLUCAT\_PACKAGE\_NAME
  - glucat\_config.h, 252
- GLUCAT\_PACKAGE\_STRING
  - glucat\_config.h, 252
- GLUCAT\_PACKAGE\_TARNAME
  - glucat\_config.h, 252
- GLUCAT\_PACKAGE\_URL
  - glucat\_config.h, 252

- glucat\_config.h, [252](#)
- GLUCAT\_PACKAGE\_VERSION
  - glucat\_config.h, [252](#)
- GLUCAT\_STDC\_HEADERS
  - glucat\_config.h, [252](#)
- GLUCAT\_VERSION
  - glucat\_config.h, [253](#)
- grade
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [100](#)
- hash\_fn
  - glucat::index\_set< LO, HI >, [151](#)
  - PyClical.index\_set, [166](#)
- hash\_size\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size\_t, [143](#)
- heading
  - glucat::error< Class\_T >, [119](#)
  - glucat::glucat\_error, [142](#)
- hi\_ndx
  - PyClical.h, [276](#)
- i
  - PyClical, [72](#)
- imag
  - glucat, [32](#)
  - glucat::numeric\_traits< Scalar\_T >, [192](#)
- index\_pair\_t
  - glucat::index\_set< LO, HI >, [147](#)
- index\_set
  - glucat::index\_set< LO, HI >, [147](#), [148](#)
  - glucat::index\_set< LO, HI >::reference, [208](#)
- index\_set\_hidden\_doctests
  - PyClical, [70](#)
- index\_set\_t
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [98](#)
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [122](#)
  - glucat::index\_set< LO, HI >, [147](#)
  - glucat::index\_set\_hash< LO, HI >, [169](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [173](#)
- index\_set\_to\_repr
  - PyClical.h, [275](#)
- index\_set\_to\_str
  - PyClical.h, [275](#)
- index\_t
  - glucat, [21](#)
- IndexSet
  - PyClical.h, [273](#)
- inner
  - glucat::matrix, [62](#)
- instance
  - PyClical.clifford, [96](#)
  - PyClical.index\_set, [168](#)
- intfn
  - glucat, [21](#)
- intintfn
  - glucat, [21](#)
- inv
  - glucat, [32](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [101](#)
  - PyClical.clifford, [91](#)
- inv\_fast\_dim\_threshold
  - 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 >, [216](#)
- inverse\_gray
  - glucat, [32](#)
- inverse\_reversed\_gray
  - glucat, [33](#)
- involute
  - glucat, [33](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [101](#)
  - PyClical.clifford, [91](#)
- is\_contiguous
  - glucat::index\_set< LO, HI >, [151](#)
- isInf
  - glucat::numeric\_traits< Scalar\_T >, [192](#), [193](#)
- isNaN
  - glucat::numeric\_traits< Scalar\_T >, [193](#), [194](#)
- isnan
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [101](#)
  - glucat::matrix, [62](#)
  - PyClical.clifford, [91](#)
- isNaN\_or\_isInf
  - glucat::numeric\_traits< Scalar\_T >, [194](#)
- ist
  - PyClical, [72](#)
- istpq
  - PyClical, [71](#)
- iterator
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [123](#)
- ixt
  - PyClical, [72](#)
- kron
  - glucat::matrix, [63](#)
- l\_In2
  - glucat, [58](#)
- l\_pi
  - glucat, [58](#)
- lex\_less\_than
  - glucat::index\_set< LO, HI >, [151](#)
- ln\_2
  - glucat::numeric\_traits< Scalar\_T >, [194](#)
- lo\_ndx
  - PyClical.h, [276](#)
- log
  - glucat, [33](#), [34](#)
  - glucat::numeric\_traits< Scalar\_T >, [195](#)

- log2
  - glucat, [34](#)
  - glucat::numeric\_traits< Scalar\_T >, [195](#)
- 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 >, [219](#)
- log\_max\_outer\_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 >, [219](#)
- m\_catch\_exceptions
  - glucat::control\_t, [113](#)
- m\_eig\_case
  - glucat::matrix::eig\_genus< Matrix\_T >, [116](#)
- m\_frame
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [185](#)
- m\_idx
  - glucat::index\_set< LO, HI >::reference, [209](#)
- m\_matrix
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [185](#)
- m\_pst
  - glucat::index\_set< LO, HI >::reference, [209](#)
- m\_safe\_arg
  - glucat::matrix::eig\_genus< Matrix\_T >, [116](#)
- m\_valid
  - glucat::control\_t, [113](#)
- m\_verbose\_output
  - glucat::control\_t, [113](#)
- map\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [123](#)
  - glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, [210](#)
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, [212](#)
- matrix\_index\_t
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [173](#)
- matrix\_log
  - glucat, [35](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [182](#)
- matrix\_multi
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [133](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [175](#)–[180](#), [182](#)
- matrix\_multi\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [123](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [174](#)
- matrix\_sqrt
  - glucat, [35](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [182](#)
- matrix\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [123](#)
- glucat::matrix\_multi< Scalar\_T, LO, HI >, [174](#)
- max
  - glucat::index\_set< LO, HI >, [152](#)
  - PyClical.index\_set, [166](#)
- max\_abs
  - glucat, [35](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [101](#)
  - PyClical.clifford, [92](#)
- max\_pos
  - glucat, [36](#)
- glucat::index\_set< LO, HI >, [152](#)
- PyClical.index\_set, [167](#)
- min\_neg
  - glucat, [36](#)
- mono\_kron
  - glucat::matrix, [63](#)
- mono\_prod
  - glucat::matrix, [63](#)
- MS\_PER\_CLOCK
  - glucat::timing, [67](#)
- MS\_PER\_S
  - glucat, [58](#)
- MS\_PER\_SEC
  - glucat::timing, [67](#)
- mult\_matrix\_threshold
  - 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 >, [216](#)
- multivector\_t
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [99](#)
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [123](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [174](#)
- n
  - glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size\_t, [143](#)
- name
  - glucat::glucat\_error, [142](#)
- NaN
  - glucat::numeric\_traits< Scalar\_T >, [195](#)
- nbar3
  - PyClical, [72](#)
- nbr\_terms
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [131](#)
- negative\_eig\_case
  - glucat::matrix, [61](#)
- ninf3
  - PyClical, [73](#)
- nnz
  - glucat::matrix, [63](#)
- nork
  - glucat::matrix, [64](#)
- nork\_range

- glucat::matrix, 64
- norm
  - glucat, 36
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 101
  - PyClical.clifford, 92
- norm\_frob2
  - glucat::matrix, 64
- normal
  - glucat::random\_generator< Scalar\_T >, 203
- normal\_dist
  - glucat::random\_generator< Scalar\_T >, 204
- obj
  - PyClical, 73
- odd
  - glucat, 36
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 102
  - PyClical.clifford, 92
- offset\_level
  - glucat, 37
- offset\_to\_super
  - glucat::gen, 59
- operator bool
  - glucat::index\_set< LO, HI >::reference, 207
- operator!=
  - glucat, 37
  - glucat::index\_set< LO, HI >, 152
- operator<
  - glucat::index\_set< LO, HI >, 153
- operator<<
  - glucat, 44
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 134
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 183, 184
- operator>>
  - glucat, 45
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 134
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 184
- operator\*
  - glucat, 39–41
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 133
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 183
- operator\*=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 102, 103
  - glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, 223
- operator~
  - glucat::index\_set< LO, HI >, 154
  - glucat::index\_set< LO, HI >::reference, 208
- operator^
  - glucat, 45, 46
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 134
  - glucat::index\_set< LO, HI >, 157
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 184
- operator^=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 105
- glucat::index\_set< LO, HI >, 154
- operator()
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 102
  - glucat::framed\_multi< Scalar\_T, LO, HI >::hash\_size\_t, 143
  - glucat::gen::generator\_table< Matrix\_T >, 139
  - glucat::index\_set\_hash< LO, HI >, 169
- operator+
  - glucat, 41
- operator+=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 103
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 131
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 181
- operator-
  - glucat, 42
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 103
- operator-=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 103
- operator/
  - glucat, 42, 43
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 133
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 183
- operator/=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 104
- operator=
  - glucat::basis\_table< Scalar\_T, LO, HI, Matrix\_T >, 77
  - glucat::control\_t, 112
  - glucat::gen::generator\_table< Matrix\_T >, 139
  - glucat::index\_set< LO, HI >::reference, 208
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 181
  - glucat::random\_generator< Scalar\_T >, 203
- operator==
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 104
  - glucat::index\_set< LO, HI >, 153
- operator%
  - glucat, 38
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 133
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 183
- operator%=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 102
- operator&
  - glucat, 38, 39
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 133
  - glucat::index\_set< LO, HI >, 157
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 183
- operator&=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 102
  - glucat::index\_set< LO, HI >, 152
- operator[]

- glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [104](#)
- glucat::index\_set< LO, HI >, [153](#)
- operator |
  - glucat, [46, 47](#)
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [134](#)
  - glucat::index\_set< LO, HI >, [158](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [184](#)
- operator |=
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [105](#)
  - glucat::index\_set< LO, HI >, [154](#)
- orientation\_t
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [174](#)
- outer\_pow
  - glucat, [47](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [105](#)
  - PyClical.clifford, [93](#)
- pade\_approx
  - glucat, [48](#)
- pade\_log
  - glucat, [48](#)
- pair\_t
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [99](#)
- pi
  - glucat::numeric\_traits< Scalar\_T >, [195, 196](#)
  - PyClical, [73](#)
- portability.h
  - \_GLUCAT\_ISINF, [267](#)
  - \_GLUCAT\_ISNAN, [267](#)
  - UBLAS\_ABS, [267](#)
  - UBLAS\_SQRT, [267](#)
- pos\_mod
  - glucat, [48](#)
- pow
  - glucat, [48, 49](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [105](#)
  - glucat::numeric\_traits< Scalar\_T >, [196](#)
  - PyClical.clifford, [93](#)
- precision\_demoted
  - glucat, [22](#)
- precision\_promoted
  - glucat, [22](#)
- precision\_same
  - glucat, [22](#)
- precision\_t
  - glucat, [22](#)
  - tuning.h, [282](#)
- print\_error\_msg
  - glucat::error< Class\_T >, [119](#)
  - glucat::glucat\_error, [142](#)
- prod
  - glucat::matrix, [64](#)
- products\_size\_threshold
  - 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 >, [216](#)
- pure
  - glucat, [49](#)
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [105](#)
  - PyClical.clifford, [93](#)
- PY\_SSIZE\_T\_CLEAN
  - PyClical.cpp, [272](#)
  - PyClical\_nocython.cpp, [277](#)
- PyClical, [67](#)
  - \_\_version\_\_, [71](#)
  - \_test, [68](#)
  - cl, [71](#)
  - clifford\_hidden\_doctests, [68](#)
  - e, [69](#)
  - fill, [72](#)
  - i, [72](#)
  - index\_set\_hidden\_doctests, [70](#)
  - ist, [72](#)
  - istpq, [71](#)
  - ixt, [72](#)
  - nbar3, [72](#)
  - ninf3, [73](#)
  - obj, [73](#)
  - pi, [73](#)
  - scalar\_epsilon, [73](#)
  - tau, [73](#)
- PyClical.clifford, [78](#)
  - \_\_add\_\_, [80](#)
  - \_\_and\_\_, [80](#)
  - \_\_call\_\_, [80](#)
  - \_\_cinit\_\_, [81](#)
  - \_\_contains\_\_, [81](#)
  - \_\_dealloc\_\_, [82](#)
  - \_\_getitem\_\_, [82](#)
  - \_\_iadd\_\_, [82](#)
  - \_\_iand\_\_, [83](#)
  - \_\_idiv\_\_, [83](#)
  - \_\_imod\_\_, [83](#)
  - \_\_imul\_\_, [84](#)
  - \_\_ior\_\_, [84](#)
  - \_\_isub\_\_, [84](#)
  - \_\_iter\_\_, [85](#)
  - \_\_ixor\_\_, [85](#)
  - \_\_mod\_\_, [85](#)
  - \_\_mul\_\_, [86](#)
  - \_\_neg\_\_, [86](#)
  - \_\_or\_\_, [86](#)
  - \_\_pos\_\_, [87](#)
  - \_\_pow\_\_, [87](#)
  - \_\_repr\_\_, [87](#)
  - \_\_richcmp\_\_, [88](#)
  - \_\_str\_\_, [88](#)



- `__sub__`, 88
- `__truediv__`, 89
- `__xor__`, 89
- `abs`, 89
- `conj`, 90
- `even`, 90
- `frame`, 90
- `instance`, 96
- `inv`, 91
- `involute`, 91
- `isnan`, 91
- `max_abs`, 92
- `norm`, 92
- `odd`, 92
- `outer_pow`, 93
- `pow`, 93
- `pure`, 93
- `quad`, 94
- `reframe`, 94
- `reverse`, 94
- `scalar`, 95
- `truncated`, 95
- `vector_part`, 95
- `PyClical.cpp`
  - `PY_SSIZE_T_CLEAN`, 272
- `PyClical.h`
  - `Clifford`, 273
  - `clifford_to_repr`, 274
  - `clifford_to_str`, 274
  - `epsilon`, 276
  - `hi_ndx`, 276
  - `index_set_to_repr`, 275
  - `index_set_to_str`, 275
  - `IndexSet`, 273
  - `lo_ndx`, 276
  - `PyFloat_FromDouble`, 275
  - `scalar_t`, 273
  - `String`, 274
  - `Tune_P`, 274
- `PyClical.index_set`, 159
  - `__and__`, 160
  - `__cinit__`, 160
  - `__contains__`, 160
  - `__dealloc__`, 161
  - `__getitem__`, 161
  - `__iand__`, 161
  - `__invert__`, 162
  - `__ior__`, 162
  - `__iter__`, 162
  - `__ixor__`, 163
  - `__or__`, 163
  - `__repr__`, 163
  - `__richcmp__`, 164
  - `__setitem__`, 164
  - `__str__`, 164
  - `__xor__`, 165
  - `count`, 165
  - `count_neg`, 165
  - `count_pos`, 166
  - `hash_fn`, 166
  - `instance`, 168
  - `max`, 166
  - `min`, 167
  - `sign_of_mult`, 167
  - `sign_of_square`, 167
- `pyclical/glucat.pxd`, 271
- `pyclical/PyClical.cpp`, 271
- `pyclical/PyClical.h`, 272
- `pyclical/PyClical.pxd`, 276
- `pyclical/PyClical.pyx`, 276
- `pyclical/PyClical_nocython.cpp`, 277
- `PyClical_nocython.cpp`
  - `PY_SSIZE_T_CLEAN`, 277
- `PyFloat_FromDouble`
  - `PyClical.h`, 275
- `quad`
  - `glucat`, 49
  - `glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >`, 106
  - `PyClical.clifford`, 94
- `random`
  - `glucat::framed_multi< Scalar_T, LO, HI >`, 132
  - `glucat::matrix_multi< Scalar_T, LO, HI >`, 181
- `random_generator`
  - `glucat::random_generator< Scalar_T >`, 202
- `real`
  - `glucat`, 49
  - `glucat::numeric_traits< Scalar_T >`, 196
- `reference`
  - `glucat::index_set< LO, HI >`, 158
  - `glucat::index_set< LO, HI >::reference`, 206, 207
- `reframe`
  - `glucat`, 50
  - `glucat::matrix_multi< Scalar_T, LO, HI >`, 184
  - `PyClical.clifford`, 94
- `reset`
  - `glucat::index_set< LO, HI >`, 154, 155
- `reverse`
  - `glucat`, 50
  - `glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >`, 106
  - `PyClical.clifford`, 94
- `safe_eig_case`
  - `glucat::matrix`, 61
- `scalar`
  - `glucat`, 50
  - `glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >`, 106
  - `PyClical.clifford`, 95
- `scalar_epsilon`
  - `PyClical`, 73
- `Scalar_T`
  - `glucat::matrix::eig_genus< Matrix_T >`, 116
- `scalar_t`

- glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, 99
- glucat::framed\_multi< Scalar\_T, LO, HI >, 124
- glucat::matrix\_multi< Scalar\_T, LO, HI >, 174
- PyClical.h, 273
- seed
  - glucat::random\_generator< Scalar\_T >, 204
- set
  - glucat::index\_set< LO, HI >, 155
- set\_value\_t
  - glucat, 22
- sign\_of\_mult
  - glucat::index\_set< LO, HI >, 156
  - PyClical.index\_set, 167
- sign\_of\_square
  - glucat, 50
  - glucat::index\_set< LO, HI >, 156
  - PyClical.index\_set, 167
- signature\_t
  - glucat::gen, 59
- signed\_perm\_nork
  - glucat::matrix, 65
- sin
  - glucat, 51
  - glucat::numeric\_traits< Scalar\_T >, 196
- sinh
  - glucat, 51
  - glucat::numeric\_traits< Scalar\_T >, 197
- size\_type
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 124
- sorted\_begin
  - glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, 211
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, 213
- sorted\_end
  - glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, 211
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, 213
- sorted\_iterator
  - glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, 210
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, 212
- sorted\_map\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 124
  - glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, 210
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, 212
- sorted\_range
  - glucat::sorted\_range< Map\_T, Sorted\_Map\_T >, 211
  - glucat::sorted\_range< Sorted\_Map\_T, Sorted\_Map\_T >, 213
- sparse\_prod
  - glucat::matrix, 65
- sqrt
  - glucat, 52, 53
  - glucat::numeric\_traits< Scalar\_T >, 197
- sqrt\_max\_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 >, 217
- star
  - glucat, 53
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 135
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 185
- std, 73
  - std::numeric\_limits< glucat::framed\_multi< Scalar\_T, LO, HI >, 186
  - std::numeric\_limits< glucat::matrix\_multi< Scalar\_T, LO, HI >, 187
- String
  - PyClical.h, 274
- tan
  - glucat, 54
  - glucat::numeric\_traits< Scalar\_T >, 197
- tanh
  - glucat, 54
  - glucat::numeric\_traits< Scalar\_T >, 197
- tau
  - PyClical, 73
- term\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, 124
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, 175
- test
  - glucat::index\_set< LO, HI >, 156
  - test/control.h, 278
  - test/driver.h, 279
  - test/timing.h, 279
  - test/try\_catch.h, 280
  - test/tuning.h, 281
  - test/undefine.h, 284
  - Test\_Tuning\_Function\_Precision
    - tuning.h, 284
  - Test\_Tuning\_Max\_Threshold
    - tuning.h, 284
- to\_demote
  - glucat, 55
- to\_double
  - glucat::numeric\_traits< Scalar\_T >, 198
- to\_int
  - glucat::numeric\_traits< Scalar\_T >, 198
- to\_lapack
  - glucat::matrix, 65
- to\_promote
  - glucat, 55
- to\_scalar\_t
  - glucat::numeric\_traits< Scalar\_T >, 198–200
- trace
  - glucat::matrix, 65

- truncated
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [106](#)
  - PyClical.clifford, [95](#)
- try\_catch
  - glucat, [55](#)
- Tune\_P
  - PyClical.h, [274](#)
  - tuning.h, [282](#)
- tuning.h
  - \_GLUCAT\_CTAssert, [284](#)
  - \_\_TEST\_TUNING\_DEFAULT\_CONSTANT, [282](#), [283](#)
  - precision\_t, [282](#)
  - Test\_Tuning\_Function\_Precision, [284](#)
  - Test\_Tuning\_Max\_Threshold, [284](#)
  - Tune\_P, [282](#)
- type
  - glucat::numeric\_traits< Scalar\_T >::demoted, [115](#)
  - glucat::numeric\_traits< Scalar\_T >::promoted, [201](#)
- UBLAS\_ABS
  - portability.h, [267](#)
- UBLAS\_SQRT
  - portability.h, [267](#)
- uint\_gen
  - glucat::random\_generator< Scalar\_T >, [204](#)
- unfold
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [132](#)
  - glucat::index\_set< LO, HI >, [156](#)
- uniform
  - glucat::random\_generator< Scalar\_T >, [203](#)
- uniform\_dist
  - glucat::random\_generator< Scalar\_T >, [205](#)
- unit
  - glucat::matrix, [66](#)
- v\_hi
  - glucat::index\_set< LO, HI >, [158](#)
- v\_lo
  - glucat::index\_set< LO, HI >, [158](#)
- valid
  - glucat::control\_t, [112](#)
- value
  - glucat::bool\_to\_type< truth\_value >, [78](#)
- value\_of\_fold
  - glucat::index\_set< LO, HI >, [157](#)
- var\_pair\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, [222](#)
- var\_term
  - glucat::framed\_multi< Scalar\_T, LO, HI >::var\_term, [222](#), [223](#)
- var\_term\_t
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [124](#)
- vector\_part
  - glucat, [56](#)
- glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [106](#), [107](#)
- PyClical.clifford, [95](#)
- vector\_t
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [99](#)
  - glucat::framed\_multi< Scalar\_T, LO, HI >, [125](#)
  - glucat::matrix\_multi< Scalar\_T, LO, HI >, [175](#)
- verbose
  - glucat::control\_t, [112](#)
- write
  - glucat::clifford\_algebra< Scalar\_T, Index\_Set\_T, Multivector\_T >, [107](#)