glucat

0.8.4

Generated by Doxygen 1.8.18

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

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 subalegbra.
- 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_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 subalegbra.

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 (
            intintfn 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 = 0.693147180559945309417232121458176 6L  [static]
```

Definition at line 70 of file long_double.h.

5.2.4.15 l_pi

```
const long double glucat::l_pi = 3.14159265358979323846264338327950 29L  [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

safe_eig_case	
negative_eig_case	
both_eig_case	

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: sum(x(i,j)*y(i,j))/x.nrows()

Inner product: sum(lhs(i,j)*rhs(i,j))/lhs.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 PyClical._test ( )  [private]
```

Definition at line 1912 of file PyClical.pyx.

5.6.1.2 clifford_hidden_doctests()

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

```
Tests for functions that Doctest cannot see.

For clifford.__cinit__: Construct an object of type clifford.

>>> print(clifford(2))
2
>>> print(clifford(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("+"))
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

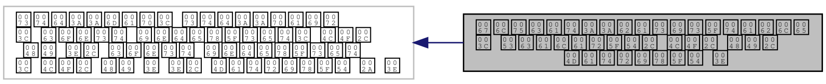
Class Documentation

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

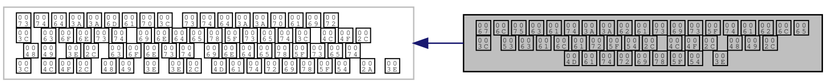
Table of basis elements used as a cache by basis_element()

```
#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	

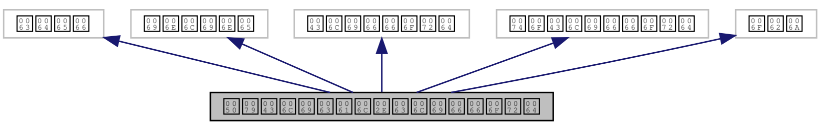
Definition at line 101 of file global.h.

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

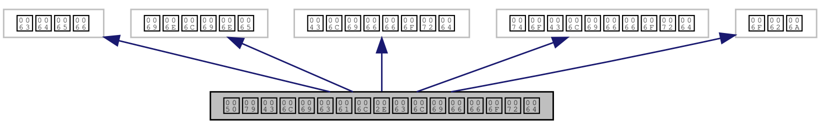
- glucat/global.h

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 PyClical.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 PyClical.pyx.

References glucat.abs().

6.3.2.30 conj()

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

```
Conjugation, reverse o involute == involute o reverse.

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

Definition at line 1138 of file PyClical.pyx.

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

6.3.2.31 even()

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

```
Even part of multivector, sum of even grade terms.

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

Definition at line 1061 of file PyClical.pyx.

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

6.3.2.32 frame()

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

```
Subalgebra generated by all generators of terms of given multivector.

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

```
Pure part.

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

Definition at line 1050 of file PyClical.pyx.

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

6.3.2.42 quad()

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

```
Quadratic form == (rev(x)*x)(0).

>>> print(clifford("1+{1}+{1,2}").quad())
3.0
>>> print(clifford("1+{-1}+{1,2}+{1,2,3}").quad())
2.0
```

Definition at line 1153 of file PyClical.pyx.

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

6.3.2.43 reframe()

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

```
Put self into a larger frame, containing the union of self.frame() and index set ixt.
This can be used to make multiplication faster, by multiplying within a common frame.

>>> clifford("2+3{1}").reframe(index_set({1,2,3}))
clifford("2+3{1}")
>>> s=index_set({1,2,3});t=index_set({-3,-2,-1});x=random_clifford(s); x.reframe(t).frame() == (s|t);
True
```

Definition at line 649 of file PyClical.pyx.

6.3.2.44 reverse()

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

```
Reversion, eg. clifford("{1}")*clifford("{2}") -> clifford("{2}")*clifford("{1}").

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

Definition at line 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} -> -{1}.
- virtual const multivector_t reverse () const =0

  Reversion, eg. {1}{2} -> {2}*{1}.*
- virtual const multivector_t conj () const =0

  Conjugation, reverse o involute == involute o reverse.
- virtual Scalar_T quad () const =0

  *Scalar_T quadratic form == (rev(x)*x)(0)*
- virtual Scalar_T norm () const =0

  Scalar_T norm == sum of norm of coordinates.
- virtual Scalar_T max_abs () const =0

  Maximum of absolute values of components of multivector: multivector infinity norm.
- virtual const multivector_t truncated (const Scalar_T &limit=Scalar_T(DEFAULT_TRUNCATION)) const =0

  Remove all terms with relative size smaller than limit.
- virtual bool isnan () const =0

  Check if a multivector contains any IEEE NaN values.
- virtual void write (const std::string &msg="") const =0

  Write formatted multivector to output.
- virtual void write (std::ofstream &ofile, const std::string &msg="") const =0

  Write formatted multivector to file.

Static Public Member Functions

- static const std::string classname ()

6.4.1 Detailed Description

template<typename Scalar_T, typename Index_Set_T, typename Multivector_T>
class glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >

clifford_algebra<> declares the operations of a Clifford algebra

Definition at line 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

are_same	

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-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:

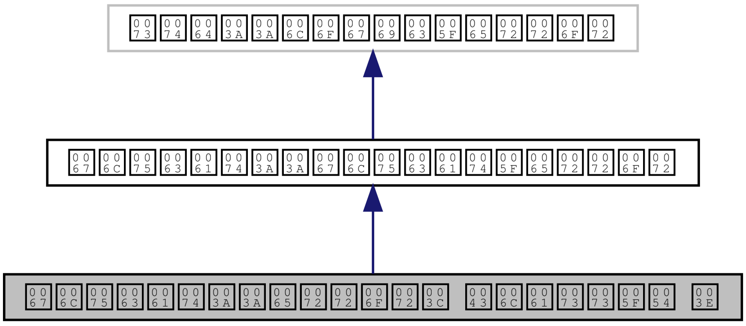
- glucat/matrix.h

6.12 glucat::error< Class_T > Class Template Reference

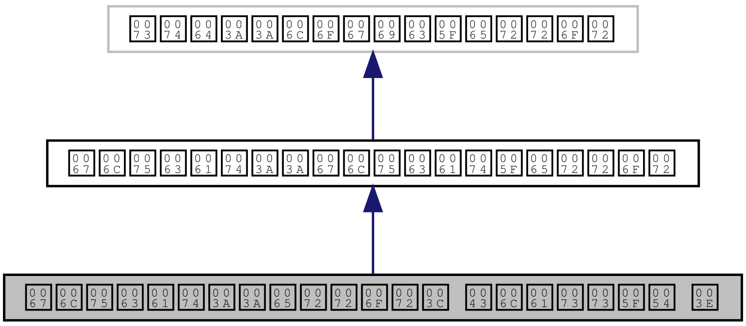
Specific exception class.

`#include <errors.h>`

Inheritance diagram for glucat::error< Class_T >:



Collaboration diagram for glucat::error< Class_T >:



Public Member Functions

- error (const std::string &msg)

  Specific exception class.
- error (const std::string &context, const std::string &msg)
- virtual const std::string heading () const throw ()
- virtual const std::string classname () const throw ()
- virtual void print_error_msg () const

Additional Inherited Members

6.12.1 Detailed Description

template<class Class_T>
class glucat::error< Class_T >

Specific exception class.

Definition at line 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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 >::m_↩ 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 = DEFAU↩
LT_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 = DEFAU↩
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 = DEFAU←
LT_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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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>
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>
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>
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>
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>
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:

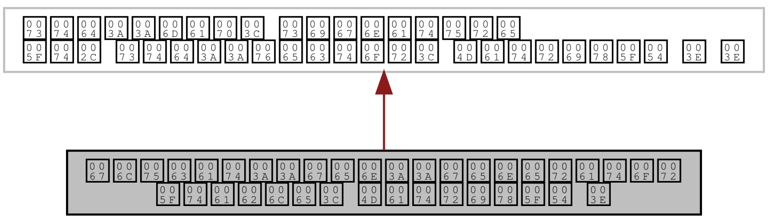
- glucat/framed_multi.h
- glucat/framed_multi_imp.h

6.14 glucat::gen::generator_table< Matrix_T > Class Template Reference

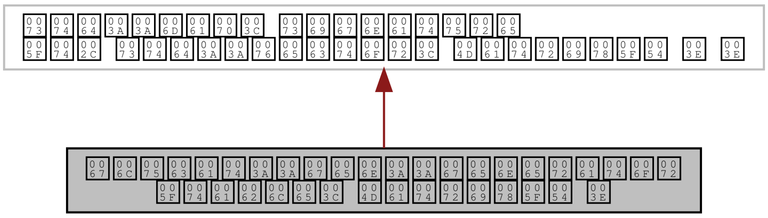
Table of generators for specific signatures.

```
#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:

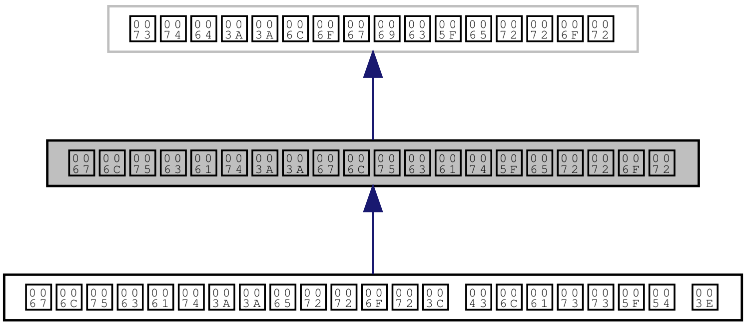
- glucat/generation.h
- glucat/generation_imp.h

6.15 glucat::glucat_error Class Reference

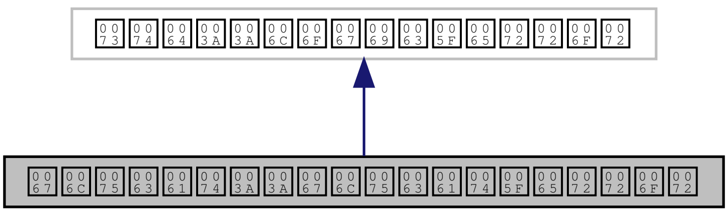
Abstract exception class.

`#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 = DEFAU↩
LT_HI>
glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t::hash_size_t (
            size_t hash_size )  [inline]
```

Definition at line 183 of file framed_multi.h.

6.16.3 Member Function Documentation

6.16.3.1 operator()()

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↩
LT_HI>
size_t glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t::operator() ( ) const  [inline]
```

Definition at line 186 of file framed_multi.h.

References glucat::framed_multi< Scalar_T, LO, HI >::hash_size_t::n.

6.16.4 Member Data Documentation

6.16.4.1 n

```
template<typename Scalar_T = double, const index_t LO = DEFAULT_LO, const index_t HI = DEFAU↩
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_BI↩
TS_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[]() [2/2]

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

Subscripting: Element access.

Definition at line 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]
```

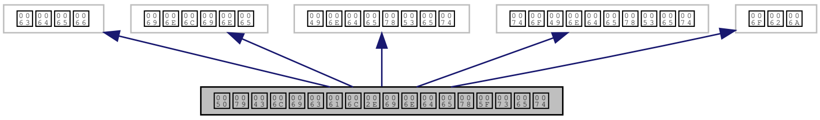
Definition at line 116 of file index_set.h.

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

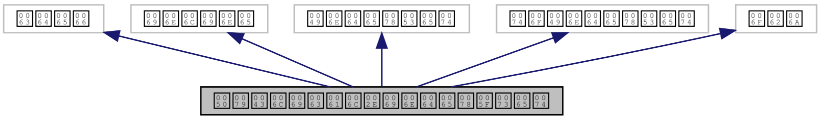
- glucat/index_set.h
- glucat/index_set_imp.h

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:

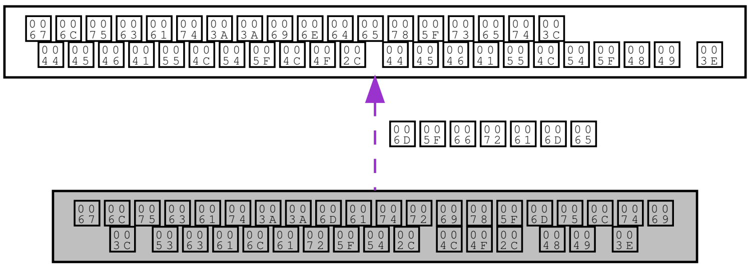
- glucat/framed_multi.h

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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↵
LT_HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
            const char * str )  [inline]
```

Construct a multivector from a char∗: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 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 = DEFAU↵
LT_HI>
glucat::matrix_multi< Scalar_T, LO, HI >::matrix_multi (
            const char * str,
            const index_set_t frm,
            const bool prechecked = false )  [inline]
```

Construct a multivector, within a given frame, from a char∗: eg: "3+2{1,2}-6.1e-2{2,3}".

Definition at line 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_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 = DEFAU←
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 = DEFAU←
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 = DEFAU←
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 = DEFAU←
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 = DEFAU←
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 = DEFAU←
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 = DEFAU←
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 = DEFAU←
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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, H↩
I >::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:

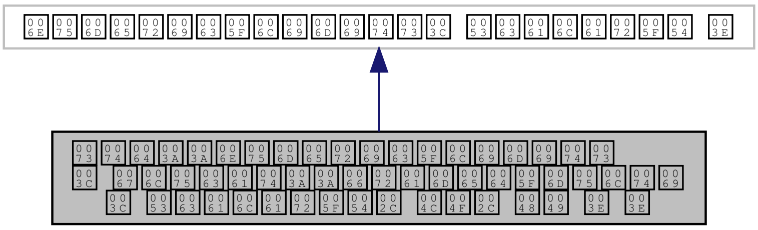
- glucat/framed_multi.h
- glucat/matrix_multi.h
- glucat/matrix_multi_imp.h

6.21 std::numeric_limits< glucat::framed_multi< Scalar_T, LO, HI > > Struct Template Reference

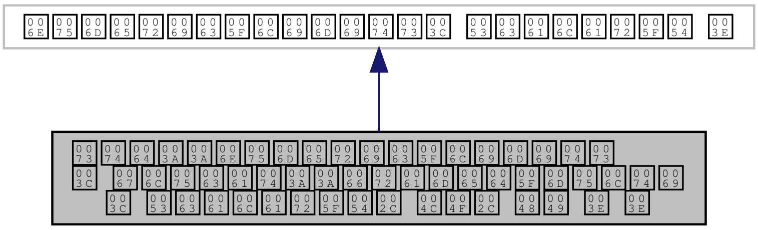
Numeric limits for framed_multi inherit limits for the corresponding scalar type.

```
#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:

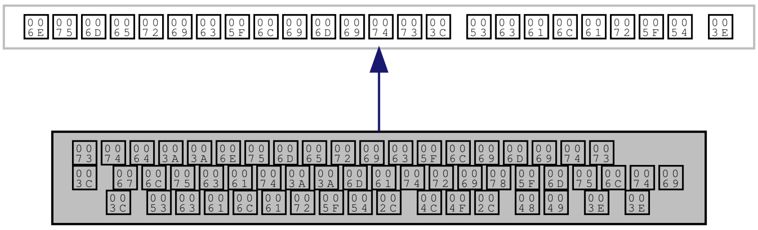
- glucat/framed_multi.h

6.22 std::numeric_limits< glucat::matrix_multi< Scalar_T, LO, HI > > Struct Template Reference

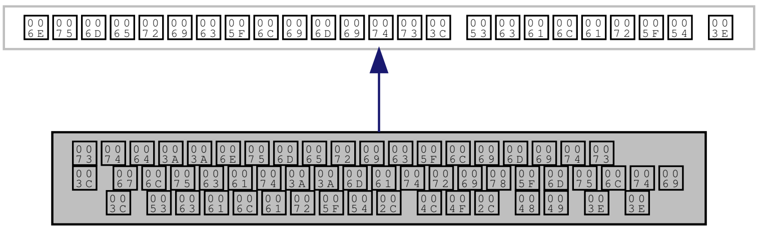
Numeric limits for matrix_multi inherit limits for the corresponding scalar type.

```
#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:

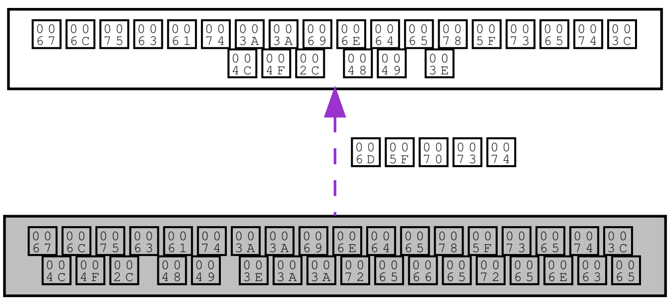
- glucat/random.h

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 Constructor & Destructor Documentation

6.28.3.1 sorted_range()

```
template<typename Sorted_Map_T >
glucat::sorted_range< Sorted_Map_T, Sorted_Map_T >::sorted_range (
            Sorted_Map_T & sorted_val,
            const Sorted_Map_T & val )  [inline]
```

Definition at line 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().

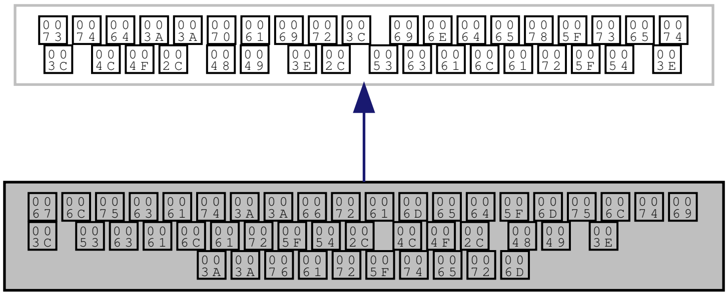
The documentation for this struct was generated from the following file:

- glucat/global.h

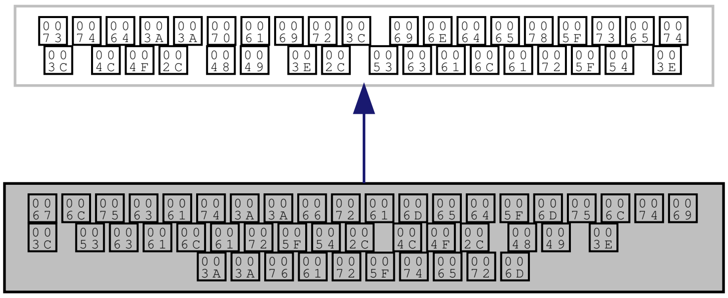
6.30 glucat::framed_multi< Scalar_T, LO, HI >::var_term Class Reference

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 = DEFAU←
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 = DEFAU←
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
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 = DEFAU↩
LT_HI>
var_term_t& glucat::framed_multi< Scalar_T, LO, HI >::var_term::operator*= (
            const term_t & rhs )  [inline]
```

Product of variable term and term.

Definition at line 328 of file framed_multi.h.

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

- glucat/framed_multi.h

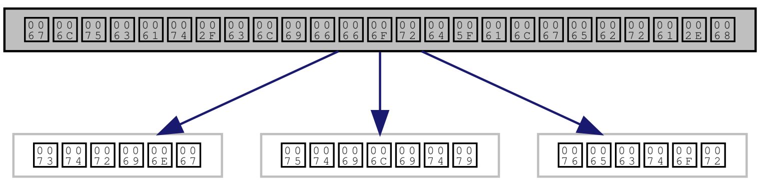
Chapter 7

File Documentation

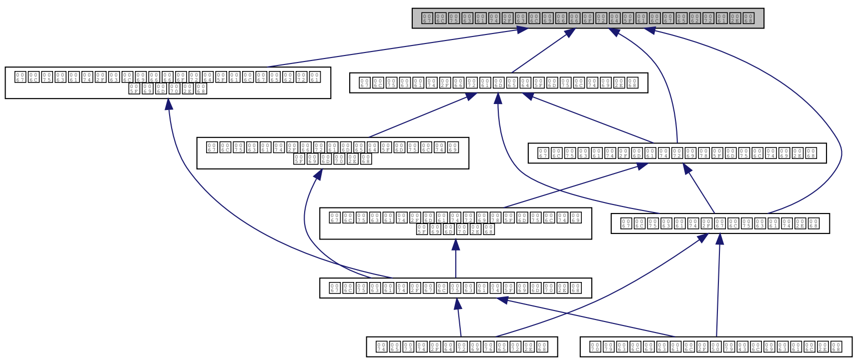
7.1 glucat/clifford_algebra.h File Reference

```
#include <string>
#include <utility>
#include <vector>
```

Include dependency graph for clifford_algebra.h:



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



Classes

- class glucat::clifford_algebra< Scalar_T, Index_Set_T, Multivector_T >

  clifford_algebra<> declares the operations of a Clifford algebra

Namespaces

- glucat

Macros

- #define _GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS

Functions

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

  Test for inequality of multivectors.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  bool glucat::operator!= (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)

  Test for inequality of multivector and scalar.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  bool glucat::operator!= (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)

  Test for inequality of scalar and multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator+ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)

  Geometric sum of multivector and scalar.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator+ (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)

  Geometric sum of scalar and multivector.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator+ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Geometric sum.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator- (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)

  Geometric difference of multivector and scalar.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator- (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)

  Geometric difference of scalar and multivector.

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

  Geometric difference.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator* (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)

  Product of multivector and scalar.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator* (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)

  Product of scalar and multivector.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator* (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Geometric product.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator^ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Outer product.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator& (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Inner product.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator% (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Left contraction.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  Scalar_T glucat::star (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Hestenes scalar product.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const Scalar_T &scr)

  Quotient of multivector and scalar.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Scalar_T &scr, const Multivector< Scalar_T, LO, HI > &rhs)

  Quotient of scalar and multivector.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::operator/ (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Geometric quotient.

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

  Transformation via twisted adjoint action.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::inv (const Multivector< Scalar_T, LO, HI > &val)

  Geometric multiplicative inverse.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)

  Integer power of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, template< typename, const index_t, const index_t > class RHS, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::pow (const Multivector< Scalar_T, LO, HI > &lhs, const RHS< Scalar_T, LO, HI > &rhs)

  Multivector power of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::outer_pow (const Multivector< Scalar_T, LO, HI > &lhs, int rhs)

  Outer product power of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::scalar (const Multivector< Scalar_T, LO, HI > &val)

  Scalar part.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::real (const Multivector< Scalar_T, LO, HI > &val)

  Real part: synonym for scalar part.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::imag (const Multivector< Scalar_T, LO, HI > &val)

  Imaginary part: deprecated (always 0)
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::pure (const Multivector< Scalar_T, LO, HI > &val)

  Pure part.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::even (const Multivector< Scalar_T, LO, HI > &val)

  Even part.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::odd (const Multivector< Scalar_T, LO, HI > &val)

  Odd part.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const std::vector< Scalar_T > glucat::vector_part (const Multivector< Scalar_T, LO, HI > &val)

  Vector part of multivector, as a vector_t with respect to frame()
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
const Multivector< Scalar_T, LO, HI > glucat::involute (const Multivector< Scalar_T, LO, HI > &val)

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

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

  Reversion, eg. {1}{2} -> {2}*{1}.*
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::conj (const Multivector< Scalar_T, LO, HI > &val)

  Conjugation, rev o invo == invo o rev.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  Scalar_T glucat::quad (const Multivector< Scalar_T, LO, HI > &val)

  *Scalar_T quadratic form == (rev(x)*x)(0)*
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  Scalar_T glucat::norm (const Multivector< Scalar_T, LO, HI > &val)

  Scalar_T norm == sum of norm of coordinates.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  Scalar_T glucat::abs (const Multivector< Scalar_T, LO, HI > &val)

  Absolute value == sqrt(norm)
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  Scalar_T glucat::max_abs (const Multivector< Scalar_T, LO, HI > &val)

  Maximum of absolute values of components of multivector: multivector infinity norm.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::complexifier (const Multivector< Scalar_T, LO, HI > &val)

  Square root of -1 which commutes with all members of the frame of the given multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::elliptic (const Multivector< Scalar_T, LO, HI > &val)
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::sqrt (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)

  Square root of multivector with specified complexifier.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::sqrt (const Multivector< Scalar_T, LO, HI > &val)

  Square root of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::clifford_exp (const Multivector< Scalar_T, LO, HI > &val)

  Exponential of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::log (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)

  Natural logarithm of multivector with specified complexifier.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::log (const Multivector< Scalar_T, LO, HI > &val)

  Natural logarithm of multivector.

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

  Cosine of multivector with specified complexifier.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::cos (const Multivector< Scalar_T, LO, HI > &val)

  Cosine of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::acos (const Multivector< Scalar_T, LO, HI > &val, const Multivector< Scalar_T, LO, HI > &i, const bool prechecked=false)

  Inverse cosine of multivector with specified complexifier.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::acos (const Multivector< Scalar_T, LO, HI > &val)

  Inverse cosine of multivector.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::cosh (const Multivector< Scalar_T, LO, HI > &val)

  Hyperbolic cosine of multivector.
- template<template< typename, const 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::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::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::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

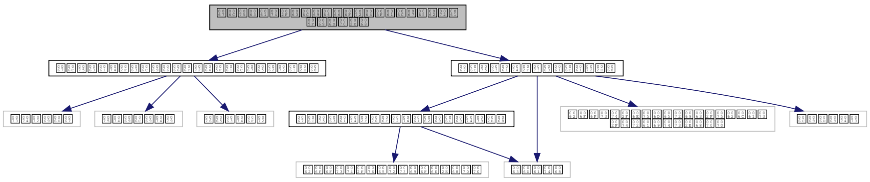
```
#define _GLUCAT_CLIFFORD_ALGEBRA_OPERATIONS
```

Definition at line 164 of file clifford_algebra.h.

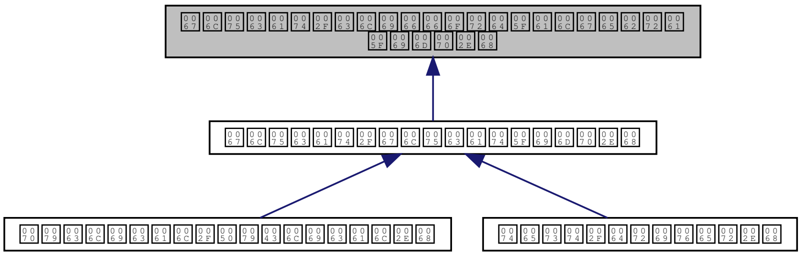
7.2 glucat/clifford_algebra_imp.h File Reference

```
#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)

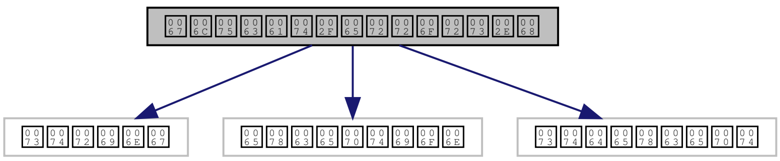
  Inverse tangent of multivector with specified complexifier.
- template<template< typename, const index_t, const index_t > class Multivector, typename Scalar_T , const index_t LO, const index_t HI>
  const Multivector< Scalar_T, LO, HI > glucat::atan (const Multivector< Scalar_T, LO, HI > &val)

  Inverse tangent of multivector.

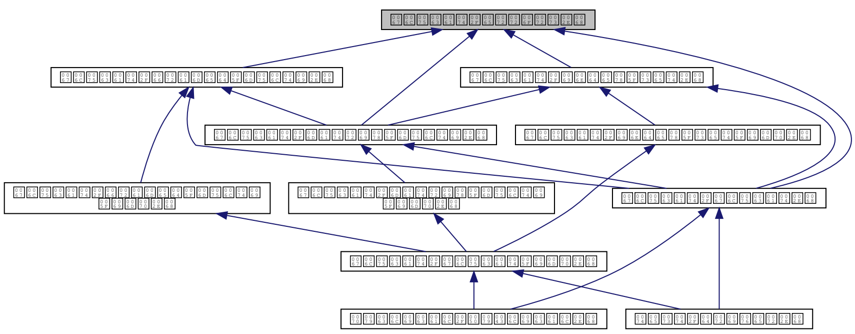
7.3 glucat/errors.h File Reference

```
#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 >

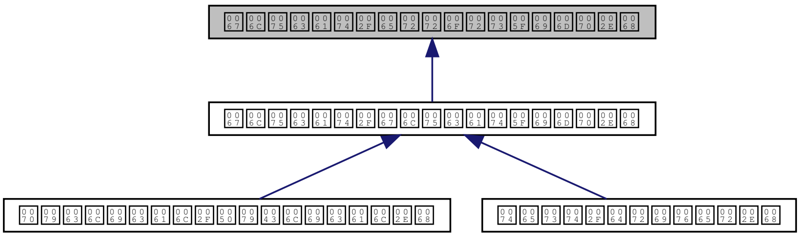
  Specific exception class.

Namespaces

- glucat

7.4 glucat/errors_imp.h File Reference

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



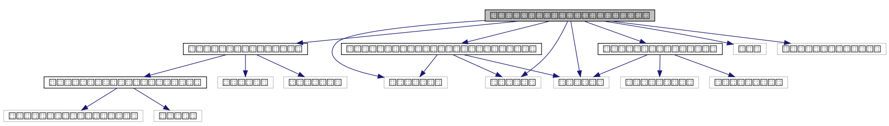
Namespaces

- glucat

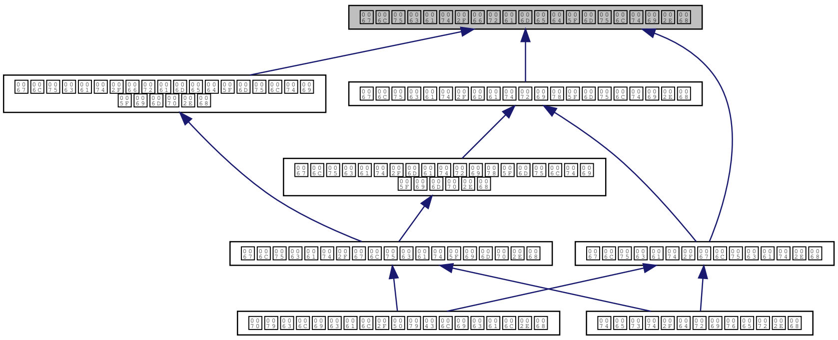
7.5 glucat/framed_multi.h File Reference

```
#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

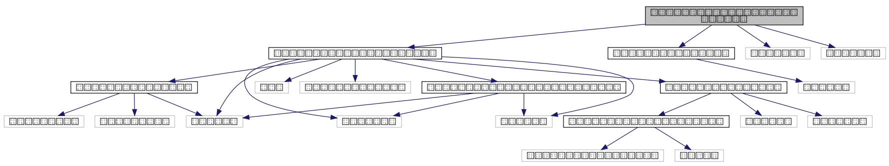
```
#define _GLUCAT_MAP_IS_HASH
```

Definition at line 56 of file framed_multi.h.

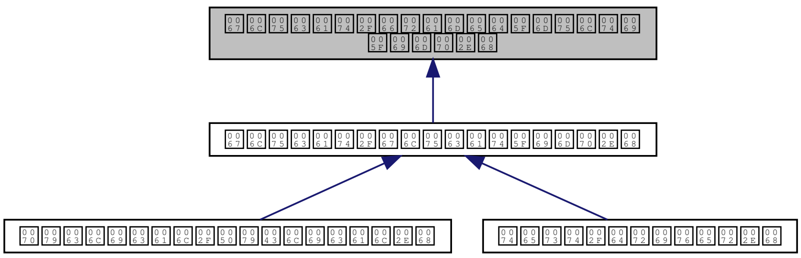
7.6 glucat/framed_multi_imp.h File Reference

```
#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

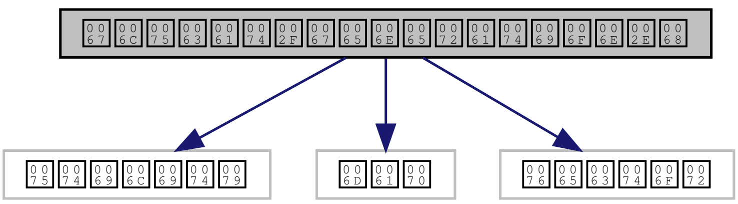
```
#define _GLUCAT_HASH_SIZE_T(
               x )
```

Definition at line 92 of file framed_multi_imp.h.

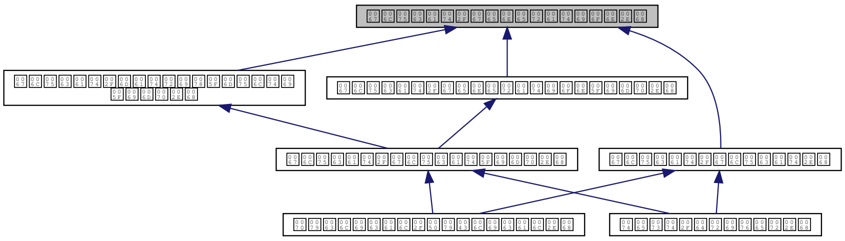
7.7 glucat/generation.h File Reference

```
#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

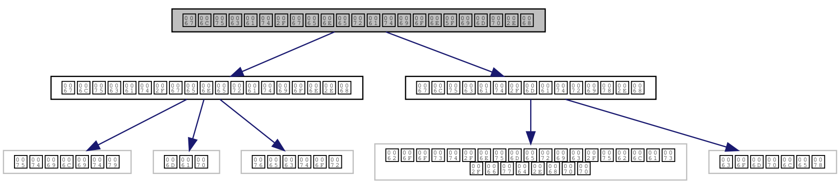
- static const index_t glucat::gen::offset_to_super [ ] = {0,-1, 0,-1,-2, 3, 2, 1}

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

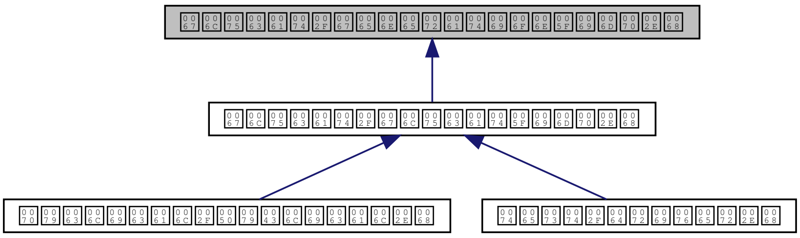
7.8 glucat/generation_imp.h File Reference

```
#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:



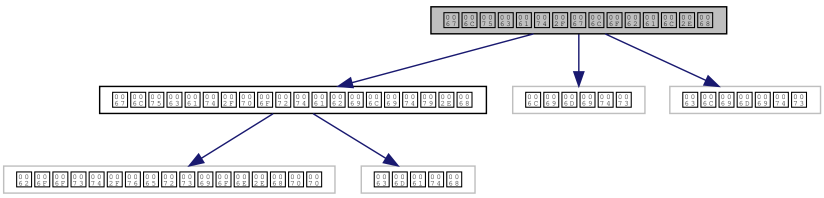
Namespaces

- glucat
- glucat::gen

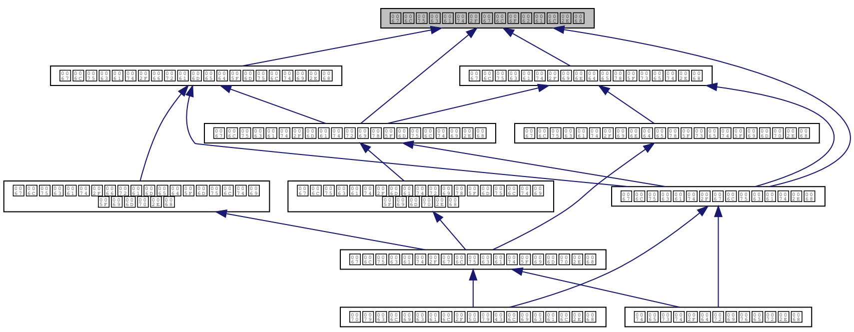
7.9 glucat/global.h File Reference

```
#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_Ste

  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

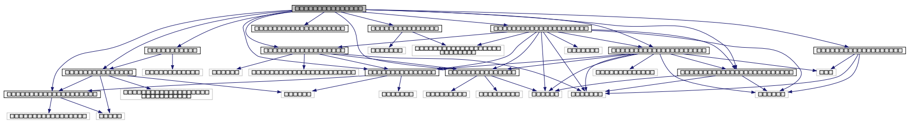
```
#define _GLUCAT_CTAssert(
            expr,
            msg ) namespace { struct msg { glucat::CTAssertion<(expr)> ERROR_##msg; }; }
```

Definition at line 78 of file global.h.

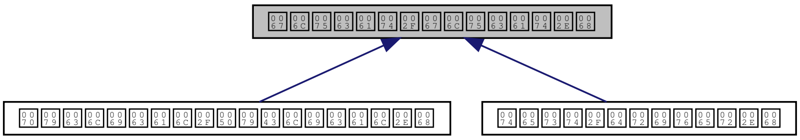
7.10 glucat/glucat.h File Reference

```
#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:

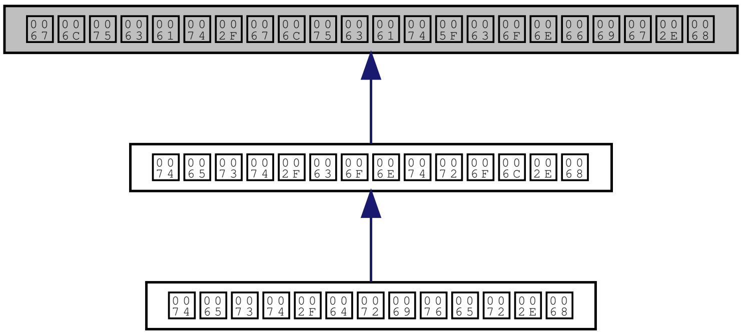


This graph shows which files directly or indirectly include this file:



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

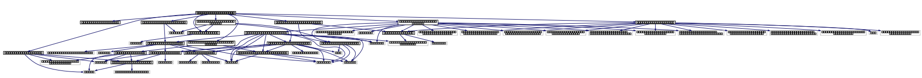
```
#define GLUCAT_VERSION "0.8.4"
```

Definition at line 99 of file glucat_config.h.

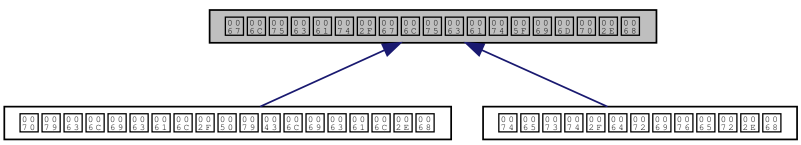
7.12 glucat/glucat_imp.h File Reference

```
#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:



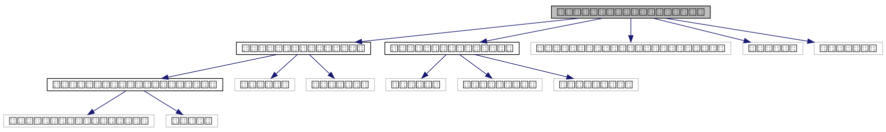
This graph shows which files directly or indirectly include this file:



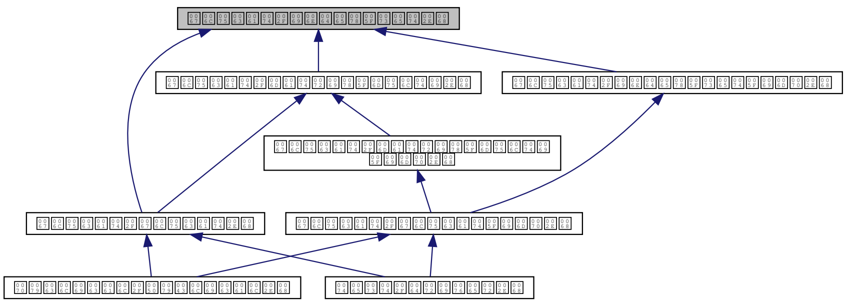
7.13 glucat/index_set.h File Reference

```
#include "glucat/global.h"
#include "glucat/errors.h"
#include <boost/static_assert.hpp>
#include <bitset>
#include <utility>
```

Include dependency graph for index_set.h:



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



Classes

- class glucat::index_set< LO, HI >

  Index set class based on std::bitset<> in Gnu standard C++ library.

- class glucat::index_set< LO, HI >

  Index set class based on std::bitset<> in Gnu standard C++ library.

- class glucat::index_set< LO, HI >::reference

  Index set member reference.

Namespaces

- glucat

Functions

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

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

  Set intersection: and.
- template<const index_t LO, const index_t HI>
  const index_set< LO, HI > glucat::operator| (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)

  Set union: or.
- template<const index_t LO, const index_t HI>
  int glucat::compare (const index_set< LO, HI > &a, const index_set< LO, HI > &b)

  "lexicographic compare" eg. {3,4,5} is less than {3,7,8}
- glucat::_GLUCAT_CTAssert (sizeof(set_value_t) >=sizeof(std::bitset< DEFAULT_HI-DEFAULT_LO >), Default_index_set_too_big_for_value) template< const index_t LO

  Size of set_value_t should be enough to contain bitset<DEFAULT_HI-DEFAULT_LO>
- const index_t HI std::ostream & glucat::operator<< (std::ostream &os, const index_set< LO, HI > &ist)

  Write out index set.
- template<const index_t LO, const index_t HI>
  std::istream & glucat::operator>> (std::istream &s, index_set< LO, HI > &ist)

  Read in index set.
- int glucat::sign_of_square (index_t j)

  Square of generator {j}.
- template<const index_t LO, const index_t HI>
  index_t glucat::min_neg (const index_set< LO, HI > &ist)

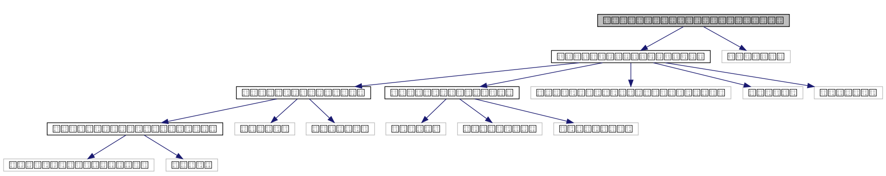
  Minimum negative index, or 0 if none.
- template<const index_t LO, const index_t HI>
  index_t glucat::max_pos (const index_set< LO, HI > &ist)

  Maximum positive index, or 0 if none.

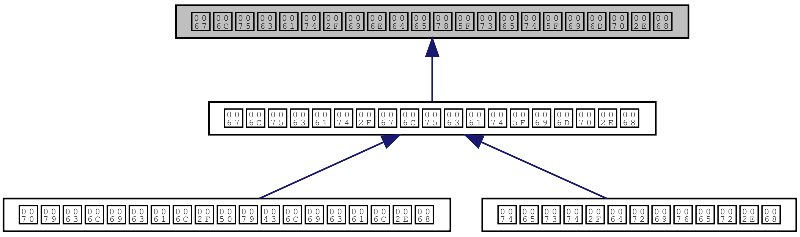
7.14 glucat/index_set_imp.h File Reference

```
#include "glucat/index_set.h"
#include <sstream>
```

Include dependency graph for index_set_imp.h:



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



Namespaces

- glucat

Functions

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

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

  Set intersection: and.
- template<const index_t LO, const index_t HI>
  const index_set< LO, HI > glucat::operator| (const index_set< LO, HI > &lhs, const index_set< LO, HI > &rhs)

  Set union: or.
- template<const index_t LO, const index_t HI>
  int glucat::compare (const index_set< LO, HI > &a, const index_set< LO, HI > &b)

  "lexicographic compare" eg. {3,4,5} is less than {3,7,8}
- const index_t HI std::ostream & glucat::operator<< (std::ostream &os, const index_set< LO, HI > &ist)

  Write out index set.
- template<const index_t LO, const index_t HI>
  std::istream & glucat::operator>> (std::istream &s, index_set< LO, HI > &ist)

  Read in index set.
- static unsigned long glucat::inverse_reversed_gray (unsigned long x)

  Inverse reversed Gray code.
- static unsigned long glucat::inverse_gray (unsigned long x)

  Inverse Gray code.
- int glucat::sign_of_square (index_t j)

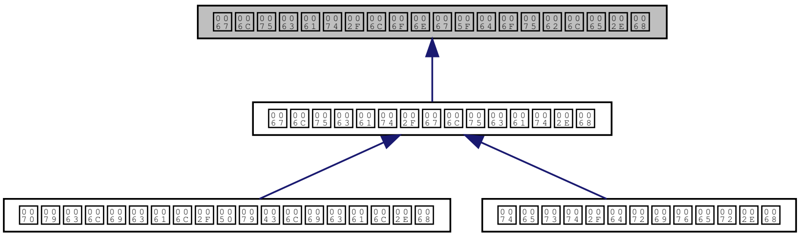
  Square of generator {j}.
- template<const index_t LO, const index_t HI>
  index_t glucat::min_neg (const index_set< LO, HI > &ist)

  Minimum negative index, or 0 if none.
- template<const index_t LO, const index_t HI>
  index_t glucat::max_pos (const index_set< LO, HI > &ist)

  Maximum positive index, or 0 if none.

7.15 glucat/long_double.h File Reference

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



Classes

- struct glucat::numeric_traits< Scalar_T >::demoted

  Demoted type for long double.

Namespaces

- glucat

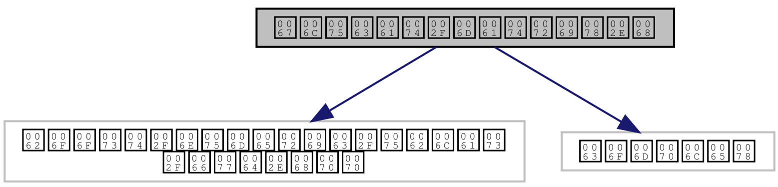
Variables

- static const long double glucat::l_pi = 3.1415926535897932384626433832795029L
- static const long double glucat::l_ln2 = 0.6931471805599453094172321214581766L

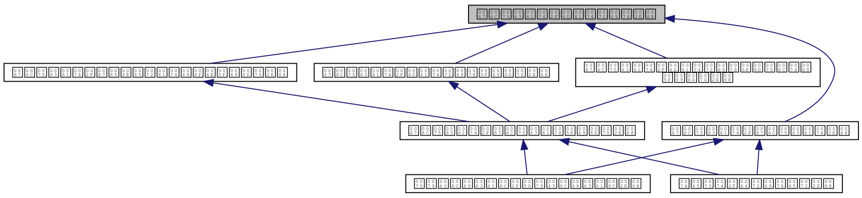
7.16 glucat/matrix.h File Reference

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

Include dependency graph for matrix.h:



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



Classes

- struct glucat::matrix::eig_genus< Matrix_T >

    Structure containing classification of eigenvalues.

Namespaces

- glucat
- glucat::matrix

Enumerations

- enum glucat::matrix::eig_case_t { glucat::matrix::safe_eig_case, glucat::matrix::negative_eig_case, glucat::matrix::both_eig_case }

    Classification of eigenvalues of a matrix.

Functions

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

    Kronecker tensor product of matrices - as per Matlab kron.

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

    Sparse Kronecker tensor product of monomial matrices.

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

    Left inverse of Kronecker product.

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

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

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

    Number of non-zeros.

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

    Not a Number.

- template<typename Matrix_T >
  const Matrix_T glucat::matrix::unit (const typename Matrix_T::size_type n)

Unit matrix - as per Matlab eye.

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

  Product of monomial matrices.
- template< typename LHS_T , typename RHS_T >
  const RHS_T::expression_type glucat::matrix::sparse_prod (const ublas::matrix_expression< LHS_T > &lhs, const ublas::matrix_expression< RHS_T > &rhs)

  Product of sparse matrices.
- template< typename LHS_T , typename RHS_T >
  const RHS_T::expression_type glucat::matrix::prod (const ublas::matrix_expression< LHS_T > &lhs, const ublas::matrix_expression< RHS_T > &rhs)

  Product of matrices.
- template< typename Scalar_T , typename LHS_T , typename RHS_T >
  Scalar_T glucat::matrix::inner (const LHS_T &lhs, const RHS_T &rhs)

  Inner product: sum(x(i,j)∗y(i,j))/x.nrows()
- template< typename Matrix_T >
  Matrix_T::value_type glucat::matrix::norm_frob2 (const Matrix_T &val)

  Square of Frobenius norm.
- template< typename Matrix_T >
  Matrix_T::value_type glucat::matrix::trace (const Matrix_T &val)

  Matrix trace.
- template< typename Matrix_T >
  ublas::vector< std::complex< double > > glucat::matrix::eigenvalues (const Matrix_T &val)

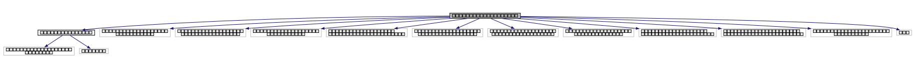
  Eigenvalues of a matrix.
- template< typename Matrix_T >
  eig_genus< Matrix_T > glucat::matrix::classify_eigenvalues (const Matrix_T &val)

  Classify the eigenvalues of a matrix.

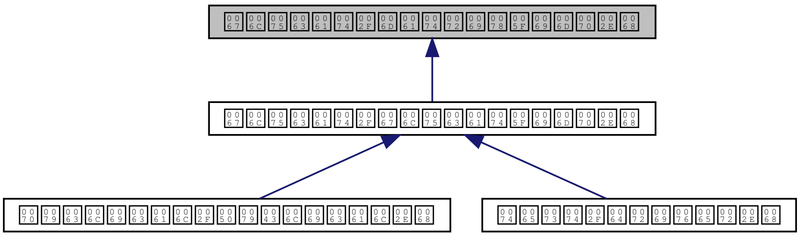
7.17 glucat/matrix_imp.h File Reference

```
#include "glucat/matrix.h"
#include <boost/numeric/ublas/vector.hpp>
#include <boost/numeric/ublas/vector_proxy.hpp>
#include <boost/numeric/ublas/matrix.hpp>
#include <boost/numeric/ublas/matrix_expression.hpp>
#include <boost/numeric/ublas/matrix_proxy.hpp>
#include <boost/numeric/ublas/matrix_sparse.hpp>
#include <boost/numeric/ublas/operation.hpp>
#include <boost/numeric/ublas/operation_sparse.hpp>
#include <boost/numeric/bindings/lapack/driver/gees.hpp>
#include <boost/numeric/bindings/ublas.hpp>
#include <set>
```

Include dependency graph for matrix_imp.h:



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



Namespaces

- glucat
- glucat::matrix

Functions

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

  Kronecker tensor product of matrices - as per Matlab kron.
- template< typename LHS_T , typename RHS_T >
  const RHS_T glucat::matrix::mono_kron (const LHS_T &lhs, const RHS_T &rhs)

  Sparse Kronecker tensor product of monomial matrices.
- template< typename LHS_T , typename RHS_T >
  void glucat::matrix::nork_range (RHS_T &result, const typename LHS_T::const_iterator2 lhs_it2, const RH↩
  S_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)

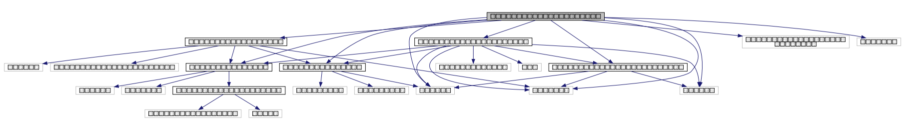
  Eigenvalues of a matrix.
- template< typename Matrix_T >
  eig_genus< Matrix_T > glucat::matrix::classify_eigenvalues (const Matrix_T &val)

  Classify the eigenvalues of a matrix.

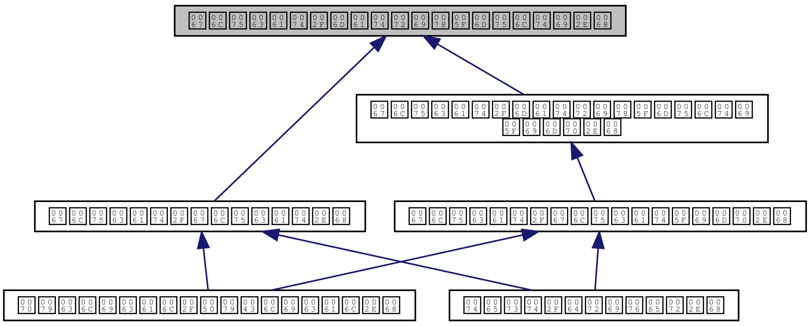
7.18 glucat/matrix_multi.h File Reference

```
#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)

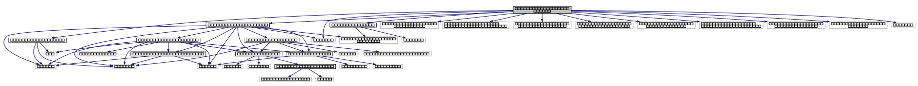
  Exponential of multivector.

7.19 glucat/matrix_multi_imp.h File Reference

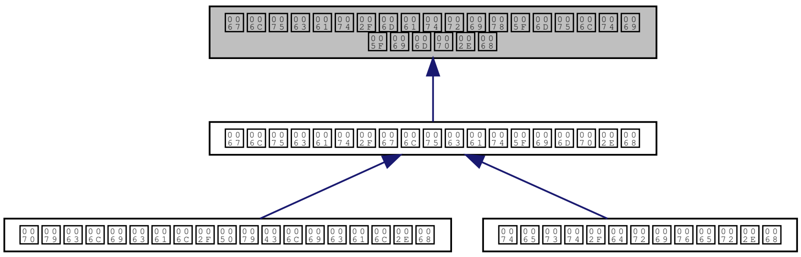
```
#include "glucat/matrix_multi.h"
#include "glucat/matrix.h"
#include "glucat/generation.h"
#include <boost/numeric/ublas/matrix.hpp>
#include <boost/numeric/ublas/matrix_expression.hpp>
#include <boost/numeric/ublas/matrix_proxy.hpp>
#include <boost/numeric/ublas/matrix_sparse.hpp>
```

```
#include <boost/numeric/ublas/operation.hpp>
#include <boost/numeric/ublas/operation_sparse.hpp>
#include <boost/numeric/ublas/triangular.hpp>
#include <boost/numeric/ublas/lu.hpp>
#include <fstream>
#include <iomanip>
```

Include dependency graph for matrix_multi_imp.h:



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



Classes

- class glucat::basis_table< Scalar_T, LO, HI, Matrix_T >

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

Namespaces

- glucat

Functions

- index_t glucat::offset_level (const index_t p, const index_t q)

  Determine the log2 dim corresponding to signature p, q.

- template<typename Matrix_Index_T , const index_t LO, const index_t HI>
  static Matrix_Index_T glucat::folded_dim (const index_set< LO, HI > &sub)

  Determine the matrix dimension of the fold of a subalegbra.

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

  Find a common frame for operands of a binary operator.

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

Geometric product.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator^ (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI > &rhs)

  Outer product.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator& (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI > &rhs)

  Inner product.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator% (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI > &rhs)

  Left contraction.
- template<typename Scalar_T , const index_t LO, const index_t HI>
Scalar_T glucat::star (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI > &rhs)

  Hestenes scalar product.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator/ (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI > &rhs)

  Geometric quotient.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::operator| (const matrix_multi< Scalar_T, LO, HI > &lhs, const matrix_multi< Scalar_T, LO, HI > &rhs)

  Transformation via twisted adjoint action.
- template<typename Scalar_T , const index_t LO, const index_t HI>
std::ostream & glucat::operator<< (std::ostream &os, const matrix_multi< Scalar_T, LO, HI > &val)

  Write multivector to output.
- template<typename Scalar_T , const index_t LO, const index_t HI>
std::istream & glucat::operator>> (std::istream &s, matrix_multi< Scalar_T, LO, HI > &val)

  Read multivector from input.
- template<typename Multivector_T , typename Matrix_T , typename Basis_Matrix_T >
static Multivector_T glucat::fast (const Matrix_T &X, index_t level)

  Inverse generalized Fast Fourier Transform.
- template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix_multi< Scalar_T, LO, HI > glucat::pade_approx (const int array_size, const Scalar_T a[ ], const Scalar_T b[ ], const matrix_multi< Scalar_T, LO, HI > &X)

  Pade' approximation.
- template<typename Scalar_T , const index_t LO, const index_t HI>
static void glucat::db_step (matrix_multi< Scalar_T, LO, HI > &M, matrix_multi< Scalar_T, LO, HI > &Y)

  Single step of product form of Denman-Beavers square root iteration.
- template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix_multi< Scalar_T, LO, HI > glucat::db_sqrt (const matrix_multi< Scalar_T, LO, HI > &val)

  Product form of Denman-Beavers square root iteration.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::sqrt (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i, bool prechecked)

  Square root of multivector with specified complexifier.
- template<typename Scalar_T , const index_t LO, const index_t HI>
const matrix_multi< Scalar_T, LO, HI > glucat::matrix_sqrt (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i)

  Square root of multivector with specified complexifier.
- template<typename Scalar_T , const index_t LO, const index_t HI>
static const matrix_multi< Scalar_T, LO, HI > glucat::pade_log (const matrix_multi< Scalar_T, LO, HI > &val)

Pade' approximation of log.

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

  Incomplete square root cascade and Pade' approximation of log.
- template<typename Scalar_T , const index_t LO, const index_t HI>
  const matrix_multi< Scalar_T, LO, HI > glucat::log (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i, bool prechecked)

  Natural logarithm of multivector with specified complexifier.
- template<typename Scalar_T , const index_t LO, const index_t HI>
  const matrix_multi< Scalar_T, LO, HI > glucat::matrix_log (const matrix_multi< Scalar_T, LO, HI > &val, const matrix_multi< Scalar_T, LO, HI > &i)

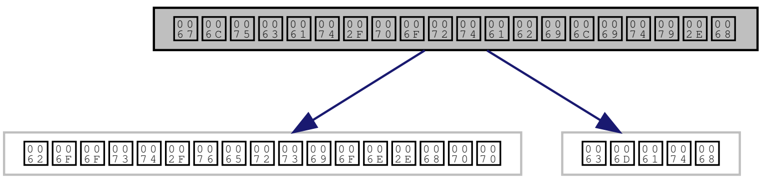
  Natural logarithm of multivector with specified complexifier.
- template<typename Scalar_T , const index_t LO, const index_t HI>
  const matrix_multi< Scalar_T, LO, HI > glucat::exp (const matrix_multi< Scalar_T, LO, HI > &val)

  Exponential of multivector.

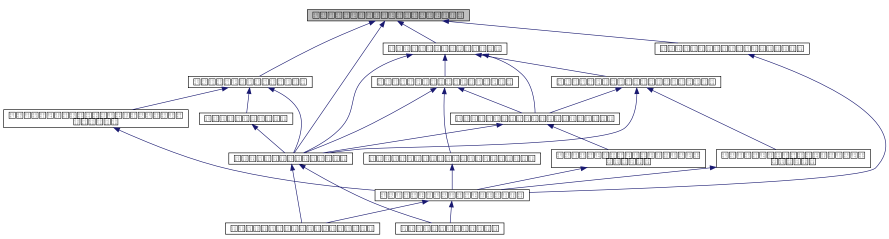
7.20 glucat/portability.h File Reference

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

Include dependency graph for portability.h:



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



Macros

- #define _GLUCAT_ISNAN(x) (x != x)
- #define _GLUCAT_ISINF(x) (!_GLUCAT_ISNAN(x) && _GLUCAT_ISNAN(x-x))
- #define UBLAS_ABS abs
- #define UBLAS_SQRT sqrt

7.20.1 Macro Definition Documentation

7.20.1.1 _GLUCAT_ISINF

```
#define _GLUCAT_ISINF(
             x ) (!_GLUCAT_ISNAN(x) && _GLUCAT_ISNAN(x-x))
```

Definition at line 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

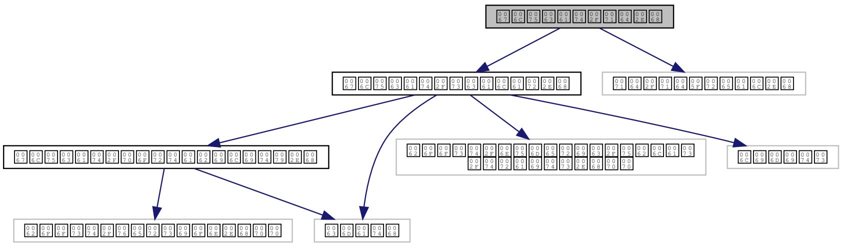
```
#define UBLAS_SQRT sqrt
```

Definition at line 58 of file portability.h.

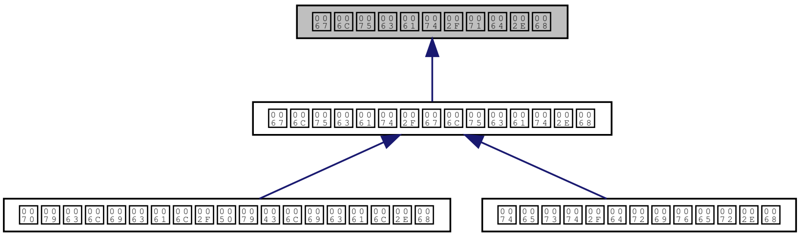
7.21 glucat/qd.h File Reference

```
#include "glucat/scalar.h"
#include <qd/qd_real.h>
```

Include dependency graph for qd.h:



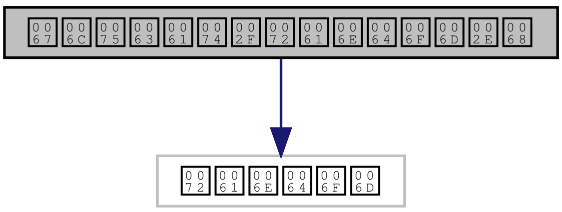
This graph shows which files directly or indirectly include this file:



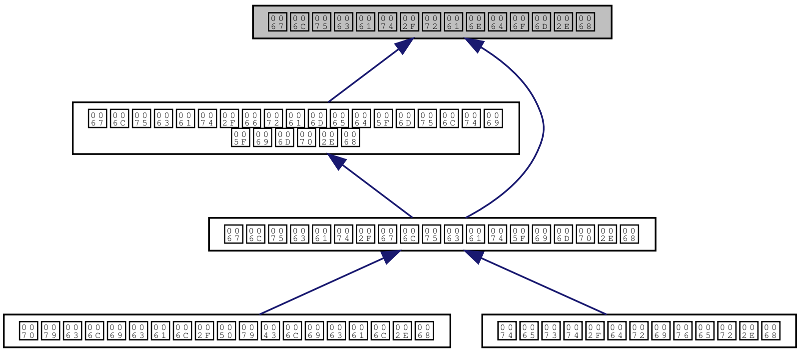
7.22 glucat/random.h File Reference

```
#include <random>
```

Include dependency graph for random.h:



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



Classes

- class glucat::random_generator< Scalar_T >

    Random number generator with single instance per Scalar_T.

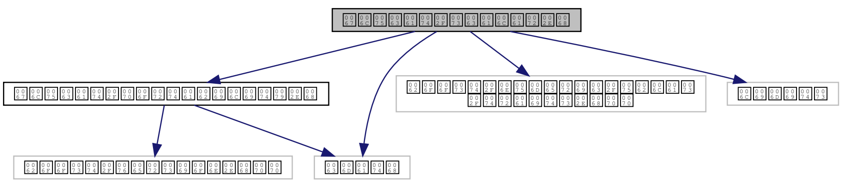
Namespaces

- glucat

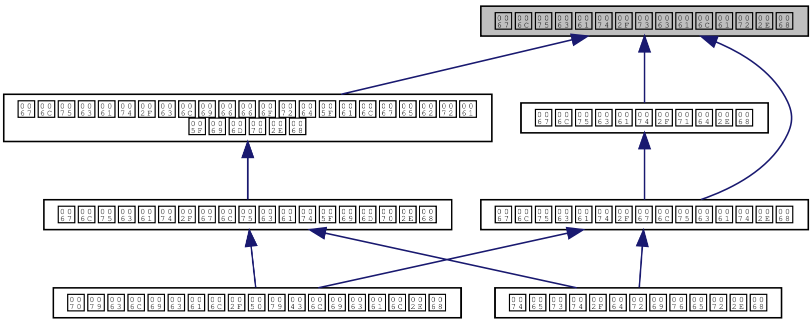
7.23 glucat/scalar.h File Reference

```
#include "glucat/portability.h"
#include <boost/numeric/ublas/traits.hpp>
#include <cmath>
#include <limits>
```

Include dependency graph for scalar.h:



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



Classes

- class glucat::numeric_traits< Scalar_T >

  Extra traits which extend numeric limits.
- struct glucat::numeric_traits< Scalar_T >::promoted

  Promoted type.
- struct glucat::numeric_traits< Scalar_T >::demoted

  Demoted type for long double.

Namespaces

- glucat

Functions

- template<typename Scalar_T >

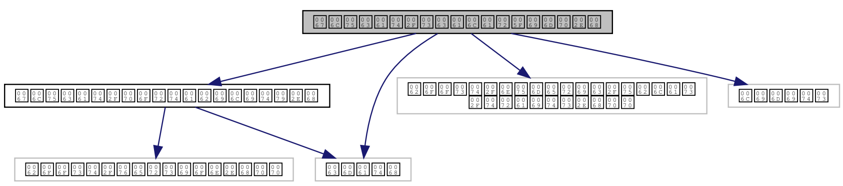
  Scalar_T glucat::log2 (const Scalar_T &x)

  Log base 2 of scalar.

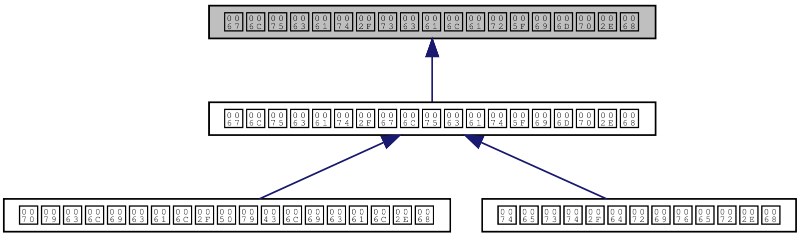
7.24 glucat/scalar_imp.h File Reference

```
#include "glucat/portability.h"
#include <boost/numeric/ublas/traits.hpp>
#include <cmath>
#include <limits>
```

Include dependency graph for scalar_imp.h:



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



Namespaces

- glucat

Functions

- template<typename Scalar_T >
  numeric_traits< Scalar_T >::promoted::type glucat::to_promote (const Scalar_T &val)

  Cast to promote.
- template<typename Scalar_T >
  numeric_traits< Scalar_T >::demoted::type glucat::to_demote (const Scalar_T &val)

  Cast to demote.

7.25 pyclical/glucat.pxd File Reference

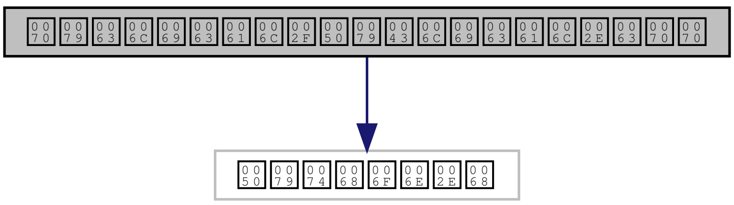
Namespaces

- glucat

7.26 pyclical/PyClical.cpp File Reference

```
#include "Python.h"
```

Include dependency graph for PyClical.cpp:



Macros

- #define PY_SSIZE_T_CLEAN

7.26.1 Macro Definition Documentation

7.26.1.1 PY_SSIZE_T_CLEAN

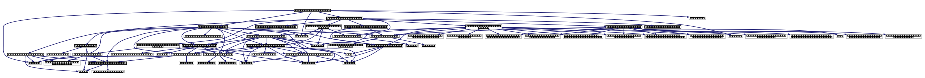
```
#define PY_SSIZE_T_CLEAN
```

Definition at line 70 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

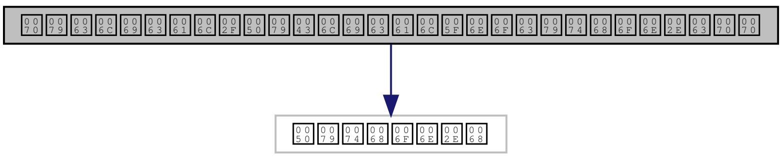
- def PyClical.index_set_hidden_doctests ()
- def PyClical.clifford_hidden_doctests ()
- def PyClical.e (obj)
- def PyClical.istpq (p, q)
- def PyClical._test ()

Variables

- string PyClical.__version__ = "0.8.4"
- PyClical.obj
- PyClical.i
- PyClical.ixt
- PyClical.fill
- PyClical.scalar_epsilon = epsilon
- float PyClical.pi = atan(clifford(1.0)) ∗ 4.0
- float PyClical.tau = atan(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

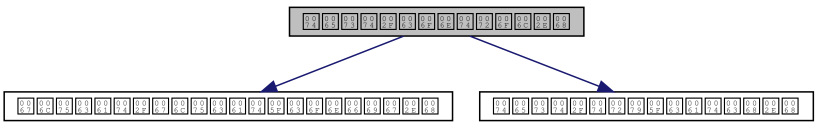
```
#define PY_SSIZE_T_CLEAN
```

Definition at line 67 of file PyClical_nocython.cpp.

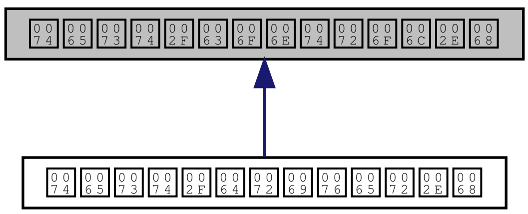
7.31 test/control.h File Reference

```
#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.

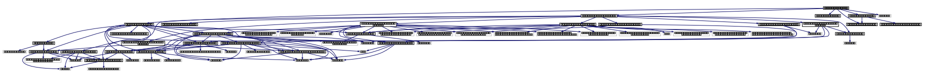
Namespaces

- 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)

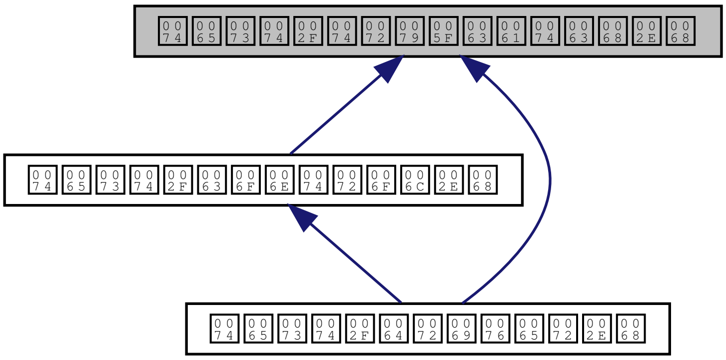
    Timing constant: milliseconds per clock.

- 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)

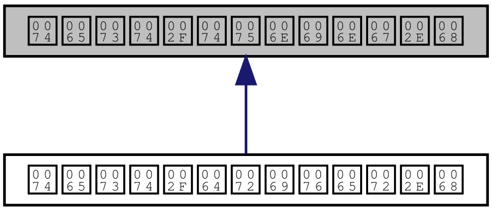
    Exception catching for functions returning int.

- 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