SYMMETRY_TENSOR Reference Manual

1.0

Generated by Doxygen 1.4.7

Thu Sep 5 23:20:32 2013

Contents

Chapter 1

SYMMETRY_TENSOR Class Index

1.1 SYMMETRY_TENSOR Class List

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

Chapter 2

SYMMETRY_TENSOR File Index

2.1 SYMMETRY_TENSOR File List

Here is a list of all documented files with brief descriptions:

Chapter 3

SYMMETRY_TENSOR Class Documentation

3.1 BoxStruct_struct Struct Reference

Use brief, otherwise the index won't have a brief explanation.

`#include <doxygen_c.h>`

Public Attributes

- int **a**
- **int b**
- **double c**

3.1.1 Detailed Description

Use brief, otherwise the index won't have a brief explanation.

Detailed explanation.

3.1.2 Member Data Documentation

3.1.2.1 int BoxStruct_struct::a

Some documentation for the member **BoxStruct::a** (p. 5).

3.1.2.2 int BoxStruct_struct::b

Some documentation for the member **BoxStruct::b** (p. 5).

3.1.2.3 double BoxStruct_struct::c

Etc.

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

- **doxygen_c.h**

3.2 SyTensor_t Class Reference

Class of the symmetry tensor.

`#include <SyTensor.h>`

Public Member Functions

- **SyTensor_t ()**

  How frequent it is used: *.

- **SyTensor_t (const string &fname)**

  To read in a binary file of a tensor which is written out by member function `save()` (p.14).
  How frequent it is used: * * *.

- **SyTensor_t (vector< Bond_t > &_bonds, const string &_name="")**

  To construct a tensor from a given bond array.
  How frequent it is used: * * *.

- **SyTensor_t (vector< Bond_t > &_bonds, vector< int > &labels, const string &_name="")**

  To construct a tensor from a given bond array and a given label array.
  How frequent it is used: * *.

- **SyTensor_t (vector< Bond_t > &_bonds, int *labels, const string &_name="")**

  To construct a tensor from a given bond array and a given label array.
  How frequent it is used: * *.

- **SyTensor_t (const SyTensor_t &SyT)**

  A deep copy constructor.
  How frequent it is used: * * *.

- **SyTensor_t & operator= (const SyTensor_t &SyT)**

  A deep copy assignment.
  How frequent it is used: * *.

- **void addLabel (vector< int > &newLabels)**

  Add labels to the Tensor.
  How frequent it is used: * * *.

- **void addLabel (int *newLabels)**

  Add labels to the Tensor.
  How frequent it is used: * * *.

- **void addRawElem (double *rawElem)**

  Add non-blocked elements to the tensor.
  How frequent it is used: * * *.

- **double at (vector< int >idxs) const**

Get the value of correspoinding array of indices.
*How frequent it is used: * idxs An STL vector of interger array, describing the indices.*

- **vector< Qnum_t > qnums ()**

  Get an array of quantum numbers of the blocks.
  *How frequent it is used: * *.*

- **void save (const string &fname)**

  Write the tensor to an output file of filename `fname`.
  *How frequent it is used: * fname A STL string, describing the filename of the output file.*

- **void reshape (vector< int > &newLabels, int rowBondNum)**

  Reshape the element of the tensor, that is, change the order of bonds to the order of `newLabels` *and also change the element alignment to the corresponding order.*
  *How frequent it is used: * * *.*

- **void reshape (int *newLabels, int rowBondNum)**

  Reshape the element of the tensor, that is, change the order of bonds to the order of `newLabels` *and also change the element alignment to the corresponding order.*
  *How frequent it is used: * * *.*

- **void transpose ()**

  Transpose the tensor.
  *How frequent it is used: * * *.*

- **void randomize ()**

  Randomly give a value(0 ~ 1.0) to each element.
  *How frequent it is used: *.*

- **void setName (const string &_name)**
- **void check ()**
- **void operator *= (SyTensor_t &Tb)**
- **void operator+= (const SyTensor_t &Tb)**
- **void operator *= (double a)**
- **Matrix_t getBlock (Qnum_t qnum, bool diag=false)**
- **void putBlock (const Qnum_t &qnum, Matrix_t &mat)**
- **void orthoRand ()**
- **void orthoRand (const Qnum_t &qnum)**
- **void eye ()**
- **void eye (const Qnum_t &qnum)**
- **void bzero (const Qnum_t &qnum)**
- **void bzero ()**

Friends

- class **Node_t**
- class **Network_t**
- ostream & **operator<<** (ostream &os, **SyTensor_t &SyT)**
- **SyTensor_t operator * (SyTensor_t &Ta, SyTensor_t &Tb)**

- **SyTensor_t operator+ (const SyTensor_t &Ta, const SyTensor_t &Tb)**
- **SyTensor_t operator * (const SyTensor_t &Ta, double a)**
- **SyTensor_t operator * (double a, const SyTensor_t &Ta)**
- **void printRawElem (const SyTensor_t &SyT)**

3.2.1 Detailed Description

Class of the symmetry tensor.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 SyTensor_t::SyTensor_t ()

How frequent it is used: *.

See also:

File demo/SyTensor_basic.cpp

3.2.2.2 SyTensor_t::SyTensor_t (const string & *fname*)

To read in a binary file of a tensor which is written out by member function `save()` (p. 14).

How frequent it is used: * * *.

Parameters:

fname The file name of the tensor being loaded, which is of type STL `string`.

See also:

File demo/SyTensor_basic.cpp

3.2.2.3 SyTensor_t::SyTensor_t (vector< Bond_t > & *_bonds*, const string & *_name* = "")

To construct a tensor from a given bond array.

How frequent it is used: * * *.

Parameters:

_bonds an STL vector of object `Bond_t`.

_name The given name of a tensor, STL string.

See also:

File demo/SyTensor_basic.cpp

Note:

The number of bonds must be larger than one, that is, the library does not support rank 0 tensor.

Warning:

```
assert( _bonds.size() > 0 )
```

3.2.2.4 SyTensor_t::SyTensor_t (vector< Bond_t > & *_bonds*, vector< int > & *labels*, const string & *_name* = "")

To construct a tensor from a given bond array and a given label array.

How frequent it is used: * *.

Parameters:

_bonds An STL vector of object `Bond_t`.

_labels An STL interger vector, describing the labels of bonds.

_name The given name of a tensor, STL string.

See also:

File demo/SyTensor_basic.cpp

Note:

The number of bonds must be larger than one, that is, the library does not support rank 0 tensor.
Each label is 1-1 corresponding to each bond in the order of array.

Warning:

```
assert( _bonds.size() > 0 )
assert( _bonds.size() == _labels.size() )
```

3.2.2.5 SyTensor_t::SyTensor_t (vector< Bond_t > & *_bonds*, int * *labels*, const string & *_name* = "")

To construct a tensor from a given bond array and a given label array.

How frequent it is used: * *.

Parameters:

_bonds An STL vector of object `Bond_t`.

_labels An integer array, describing the labels of bonds.

_name The given name of a tensor, STL string.

See also:

File demo/SyTensor_basic.cpp

Note:

The number of bonds must be larger than one, that is, the library does not support rank 0 tensor.
Each label is 1-1 corresponding to each bond in the order of array.

Warning:

```
assert( _bonds.size() > 0 )
assert( _bonds.size() == _labels.size() )
```

3.2.2.6 SyTensor_t::SyTensor_t (const SyTensor_t & *SyT*)

A deep copy constructor.

How frequent it is used: * * *.

See also:

File demo/SyTensor_basic.cpp

3.2.3 Member Function Documentation

3.2.3.1 void SyTensor_t::addLabel (int * *newLabels*)

Add labels to the Tensor.

How frequent it is used: * * *.

Parameters:

newLabels An interger array, describing the labels of bonds.

See also:

File demo/SyTensor_basic.cpp

Note:

Each added label is 1-1 corresponding to each bond in the order of array.

Warning:

```
assert( _bonds.size() == _labels.size() )
```

3.2.3.2 void SyTensor_t::addLabel (vector< int > & *newLabels*)

Add labels to the Tensor.

How frequent it is used: * * *.

Parameters:

newLabels An STL interger vector, describing the labels of bonds.

See also:

File demo/SyTensor_basic.cpp

Note:

Each added label is 1-1 corresponding to each bond in the order of array.

Warning:

```
assert( _bonds.size() == _labels.size() )
```

3.2.3.3 void SyTensor_t::addRawElem (double * *rawElem*)

Add non-blocked elements to the tensor.

How frequent it is used: * * *.

Parameters:

rawElem An array of element type of size equal to `elemNum`.

See also:

File demo/SyTensor_basic.cpp

Note:

The alignment of the given tensor elements should follow the order of the bonds.

3.2.3.4 double SyTensor_t::at (vector< int > *idxs*) const

Get the value of correspoinding array of indices.

How frequent it is used: * idxs An STL vector of interger array, describing the indices.

See also:

File demo/SyTensor_basic.cpp

3.2.3.5 SyTensor_t& SyTensor_t::operator= (const SyTensor_t & *SyT*)

A deep copy assignment.

How frequent it is used: * *.

See also:

File demo/SyTensor_basic.cpp

3.2.3.6 vector<Qnum_t> SyTensor_t::qnums ()

Get an array of quantum numbers of the blocks.

How frequent it is used: * *.

Returns:

An STL vector of type `Qnum_t`.

See also:

File demo/SyTensor_basic.cpp

3.2.3.7 void SyTensor_t::randomize ()

Randomly give a value($0 \sim 1.0$) to each element.

How frequent it is used: $*$.

See also:

File demo/SyTensor_tool.cpp

3.2.3.8 void SyTensor_t::reshape (int * *newLabels*, int *rowBondNum*)

Reshape the element of the tensor, that is, change the order of bonds to the order of `newLabels` and also change the element alignment to the corresponding order.

How frequent it is used: $* * *$.

Parameters:

newLabels An interger array, describing the labels of bonds after reshape.

rowBondNum An interger, describing the number of row bonds .

See also:

File demo/SyTensor_tool.cpp

Note:

Reshape may cause change of the order of bonds, quantum numbers of blocks of the tensor and the alignment of tensor elements.

Warning:

The only difference between `newLabels` and the original `labels` is the order of the array elements.

3.2.3.9 void SyTensor_t::reshape (vector< int > & *newLabels*, int *rowBondNum*)

Reshape the element of the tensor, that is, change the order of bonds to the order of `newLabels` and also change the element alignment to the corresponding order.

How frequent it is used: $* * *$.

Parameters:

newLabels An STL interger vector, describing the labels of bonds after reshape.

rowBondNum An interger, describing the number of row bonds .

See also:

File demo/SyTensor_tool.cpp

Note:

Reshape may cause change of the order of bonds, quantum numbers of blocks of the tensor and the alignment of tensor elements.

Warning:

The only difference between `newLabels` and the original `labels` is the order of the array elements.

3.2.3.10 void SyTensor_t::save (const string & *fname*)

Write the tensor to an output file of filename `fname`.

How frequent it is used: ∗ fname A STL string, describing the filename of the output file.

See also:

File demo/SyTensor_basic.cpp

3.2.3.11 void SyTensor_t::transpose ()

Transpose the tensor.

How frequent it is used: ∗ ∗ ∗.

See also:

File demo/SyTensor_tool.cpp

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

- **SyTensor.h**

Chapter 4

SYMMETRY_TENSOR File Documentation

4.1 doxygen_c.h File Reference

File containing example of doxygen usage for quick reference.

```
#include <systemheader1.h>
#include <systemheader2.h>
#include <box/header1.h>
#include <box/header2.h>
#include "local_header1.h"
#include "local_header2.h"
```

Classes

- struct **BoxStruct_struct**
  Use brief, otherwise the index won't have a brief explanation.

Typedefs

- typedef enum **BoxEnum_enum BoxEnum**
  Use brief, otherwise the index won't have a brief explanation.
- **typedef BoxStruct_struct BoxStruct**
  Use brief, otherwise the index won't have a brief explanation.

Enumerations

- enum **BoxEnum_enum { BOXENUM_FIRST, BOXENUM_SECOND, BOXENUM_ETC }**

Use brief, otherwise the index won't have a brief explanation.

Functions

- BOXEXPORT **BoxStruct** * **Box_The_Function_Name** (**BoxParamType1 param1, BoxParamType2 param2**)

  Example showing how to document a function with Doxygen.

- **BOXEXPORT void** * **Box_The_Second_Function** (**void**)

  A simple stub function to show how links do work.

- **BOXEXPORT void Box_The_Last_One** (**void**)

4.1.1 Detailed Description

File containing example of doxygen usage for quick reference.

Author:

My Self

Date:

9 Sep 2012

Here typically goes a more extensive explanation of what the header defines. Doxygens tags are words preceeded by either a backslash \ or by an at symbol @.

See also:

`http://www.stack.nl/~dimitri/doxygen/docblocks.html`
`http://www.stack.nl/~dimitri/doxygen/commands.html`

4.1.2 Typedef Documentation

4.1.2.1 typedef enum BoxEnum_enum BoxEnum

Use brief, otherwise the index won't have a brief explanation.

Detailed explanation.

4.1.2.2 typedef struct BoxStruct_struct BoxStruct

Use brief, otherwise the index won't have a brief explanation.

Detailed explanation.

4.1.3 Enumeration Type Documentation

4.1.3.1 enum BoxEnum_enum

Use brief, otherwise the index won't have a brief explanation.

Detailed explanation.

Enumerator:

BOXENUM_FIRST Some documentation for first.

BOXENUM_SECOND Some documentation for second.

BOXENUM_ETC Etc.

```
50                          {
51   BOXENUM_FIRST,
52   BOXENUM_SECOND,
53   BOXENUM_ETC
54 } BoxEnum;
```

4.1.4 Function Documentation

4.1.4.1 BOXEXPORT BoxStruct* Box_The_Function_Name (BoxParamType1 *param1*, BoxParamType2 *param2*)

Example showing how to document a function with Doxygen.

Description of what the function does. This part may refer to the parameters of the function, like `param1` or `param2`. A word of code can also be inserted like `this` which is equivalent to `this` and can be useful to say that the function returns a `void` or an `int`. If you want to have more than one word in typewriter font, then just use <tt>. We can also include text verbatim,

```
like this
```

Sometimes it is also convenient to include an example of usage:

```
BoxStruct *out = Box_The_Function_Name(param1, param2);
printf("something...\n");
```

Or,

```
{.py}
pyval = python_func(arg1, arg2)
print pyval
```

when the language is not the one used in the current source file (but **be careful** as this may be supported only by recent versions of Doxygen). By the way, **this is how you write bold text** or, if it is just one word, then you can just do **this**.

Parameters:

param1 Description of the first parameter of the function.

param2 The second one, which follows `param1`.

Returns:

Describe what the function returns.

See also:

Box_The_Second_Function (p. 18)
Box_The_Last_One (p. 18)
`http://website/`

Note:

Something to note.

Warning:

Warning.

4.1.4.2 BOXEXPORT void Box_The_Last_One (void)

Brief can be omitted. If you configure Doxygen with `JAVADOC_AUTOBRIEF=YES`, then the first Line of the comment is used instead. In this function this would be as if

```
@brief Brief can be omitted.
```

was used instead.

4.1.4.3 BOXEXPORT void* Box_The_Second_Function (void)

A simple stub function to show how links do work.

Links are generated automatically for webpages (like `http://www.google.co.uk`) and for `structures, like BoxStruct_struct` (p. 5). For typedef-ed types use `BoxStruct` (p. 16). For functions, automatic links are generated when the parenthesis `() follow the name of the function, like Box_The_Function_Name()` (p. 17). `Alternatively, you can use Box_The_Function_Name` (p. 17).

`Returns:`

`NULL is always returned.`

4.2 SyTensor.h File Reference

This is the header file for the class of symmetry tensor "SyTensor_t".

```
#include <iostream>
#include <iomanip>
#include <math.h>
#include <vector>
#include <map>
#include <set>
#include <string>
#include <assert.h>
#include <stdint.h>
#include "Block.h"
#include "Bond.h"
#include "myLapack.h"
#include "Matrix.h"
```

Classes

- class **SyTensor_t**

  Class of the symmetry tensor.

Defines

- #define DOUBLE double

Variables

- const int **INIT** = **1**
- **const int HAVELABEL = 2**
- **const int HAVEELEM = 4**

4.2.1 Detailed Description

This is the header file for the class of symmetry tensor "SyTensor_t".

Author:

Yun-Da Hsieh

Date:

28 Aug 2013

See also:

http://www.stack.nl/~dimitri/doxygen/docblocks.html
http://www.stack.nl/~dimitri/doxygen/commands.html

4.2.2 Variable Documentation

4.2.2.1 const int HAVEELEM = 4

A flag for having element assigned

4.2.2.2 const int HAVELABEL = 2

A flag for having labels added

4.2.2.3 const int INIT = 1

A flag for initialization

Index