LSHBOX

0.8

Generated by Doxygen 1.8.7

Fri Aug 29 2014 22:26:00

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 lshbox Namespace Reference

Classes

- class Benchmark
- class itqLsh
- class Matrix
- class Metric
- class progress_display
- class psdLsh
- class rbsLsh
- class rhpLsh
- class Scanner
- class shLsh
- class Stat
- class thLsh
- class timer
- class Topk

Functions

- bool ascend (const std::pair< unsigned, float > &lhs, const std::pair< unsigned, float > &rhs)
- template<typename DATATYPE >
  DATATYPE sqr (const DATATYPE &x)

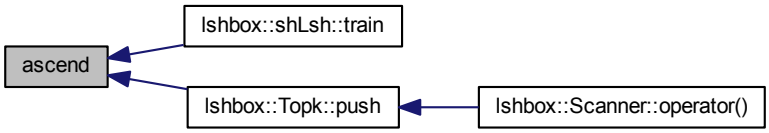
4.1.1 Function Documentation

4.1.1.1 bool lshbox::ascend (const std::pair< unsigned, float > & *lhs*, const std::pair< unsigned, float > & *rhs*)

Sort std::vector<std::pair<unsigned, float> > by the second value.

Definition at line 39 of file basis.h.

Here is the caller graph for this function:

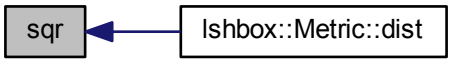


4.1.1.2 DATATYPE lshbox::sqr (const DATATYPE & *x*)

The calculation of square.

Definition at line 39 of file metric.h.

Here is the caller graph for this function:



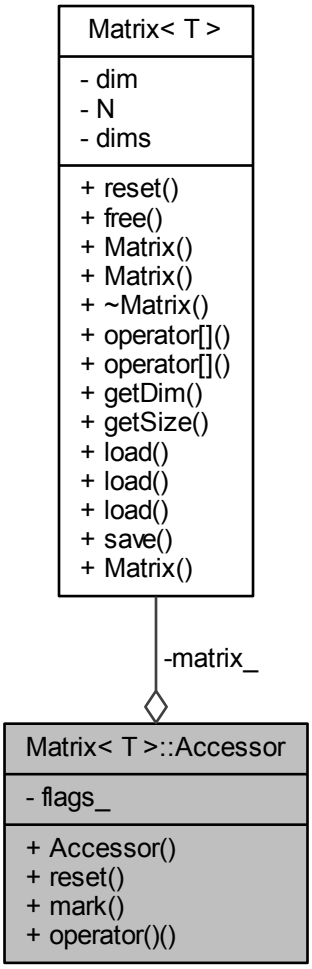
Chapter 5

Class Documentation

5.1 Matrix< T >::Accessor Class Reference

`#include <matrix.h>`

Collaboration diagram for Matrix< T >::Accessor:



Public Types

- typedef unsigned Key
- typedef const T * Value
- typedef T DATATYPE

Public Member Functions

- Accessor (const Matrix &matrix)
- void reset ()
- bool mark (unsigned key)
- const T * operator() (unsigned key)

Private Attributes

- const Matrix & matrix_

- std::vector< bool > flags_

5.1.1 Detailed Description

template<class T>class lshbox::Matrix< T >::Accessor

An accessor class to be used with LSH index.

Definition at line 179 of file matrix.h.

5.1.2 Member Typedef Documentation

5.1.2.1 typedef T DATATYPE

Definition at line 186 of file matrix.h.

5.1.2.2 typedef unsigned Key

Definition at line 184 of file matrix.h.

5.1.2.3 typedef const T* Value

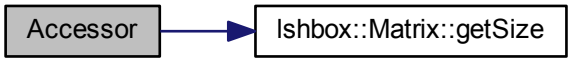
Definition at line 185 of file matrix.h.

5.1.3 Constructor & Destructor Documentation

5.1.3.1 Accessor (const Matrix & *matrix*) [inline]

Definition at line 187 of file matrix.h.

Here is the call graph for this function:



5.1.4 Member Function Documentation

5.1.4.1 bool mark (unsigned *key*) [inline]

Definition at line 196 of file matrix.h.

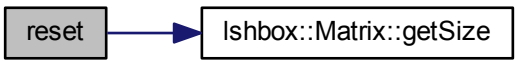
5.1.4.2 const T* operator() (unsigned *key*) [inline]

Definition at line 205 of file matrix.h.

5.1.4.3 void reset ( ) [inline]

Definition at line 191 of file matrix.h.

Here is the call graph for this function:



5.1.5 Member Data Documentation

5.1.5.1 std::vector<bool> flags_ [private]

Definition at line 182 of file matrix.h.

5.1.5.2 const Matrix& matrix_ [private]

Definition at line 181 of file matrix.h.

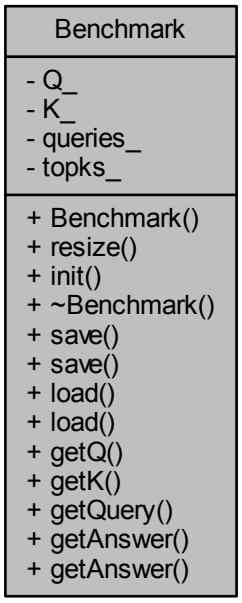
The documentation for this class was generated from the following file:

- include/lshbox/matrix.h

5.2 Benchmark Class Reference

```
#include <eval.h>
```

Collaboration diagram for Benchmark:



Public Member Functions

- Benchmark ()
- void resize (unsigned Q, unsigned K=0)
- void init (unsigned Q, unsigned K, unsigned maxID, unsigned seed=0)
- ~Benchmark ()
- void save (std::ostream &os) const
- void save (const std::string &path) const
- void load (std::istream &is)
- void load (const std::string &path)
- unsigned getQ () const
- unsigned getK () const
- unsigned getQuery (unsigned n) const
- const Topk & getAnswer (unsigned n) const
- Topk & getAnswer (unsigned n)

Private Attributes

- unsigned Q_
- unsigned K_
- std::vector< unsigned > queries_
- std::vector< Topk > topks_

5.2.1 Detailed Description

Use for access a benchmark file.

We assume the feature vectors in the benchmark database are numbered from 0 to N. We sample Q queries as test examples and run K-NN search against the database with linear scan. The results are saved in a benchmark file for evaluation purpose. A benchmark file is made up of Q lines, each line represents a test query and is of the following format:

[query ID] [K] [1st NN's ID] [distance] [2nd NN's ID] [distance] ... [Kth NN's ID] [distance]

For all queries in the benchmark file, the K value should be the same.

Because the query points are also sampled from the database, they should be excluded from scanning when running this particular query.

Definition at line 56 of file eval.h.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 Benchmark ( ) [inline]

Definition at line 59 of file eval.h.

5.2.2.2 ~Benchmark ( ) [inline]

Definition at line 114 of file eval.h.

5.2.3 Member Function Documentation

5.2.3.1 const Topk& getAnswer (unsigned *n*) const [inline]

Get the nearest neighbors of the nth query.

Definition at line 195 of file eval.h.

5.2.3.2 Topk& getAnswer (unsigned *n*) [inline]

Get the KNNs for modification.

Definition at line 202 of file eval.h.

5.2.3.3 unsigned getK ( ) const [inline]

Get the result number for each query.

Definition at line 181 of file eval.h.

5.2.3.4 unsigned getQ ( ) const [inline]

Get the query number for brenchmark.

Definition at line 174 of file eval.h.

5.2.3.5 unsigned getQuery (unsigned *n*) const [inline]

Get the ID of the nth query.

Definition at line 188 of file eval.h.

5.2.3.6 void init (unsigned *Q*, unsigned *K*, unsigned *maxID*, unsigned *seed* = 0) [inline]

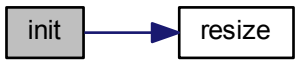
Random initialization.

Parameters

Q	The query number for brenchmark
K	The result number for each query
maxID	The number of vectors in the search library
seed	Seed some value for random to generate different query samples

Definition at line 85 of file eval.h.

Here is the call graph for this function:



5.2.3.7 void load (std::istream & *is*) [inline]

Load the benchmark from byte stream.

Definition at line 144 of file eval.h.

Here is the call graph for this function:

Here is the caller graph for this function:

5.2.3.8 void load (const std::string & *path*) [inline]

Load the benchmark from a text file.

Definition at line 165 of file eval.h.

Here is the call graph for this function:

5.2.3.9 void resize (unsigned *Q*, unsigned *K* = 0) [inline]

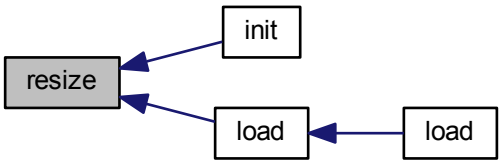
Change the query number for brenchmark and the result number for each query.

Parameters

Q	The new query number for brenchmark
K	The new result number for each query

Definition at line 66 of file eval.h.

Here is the caller graph for this function:



5.2.3.10 void save (std::ostream & *os*) const [inline]

Save the benchmark to byte stream.

Definition at line 118 of file eval.h.

Here is the caller graph for this function:

5.2.3.11 void save (const std::string & *path*) const [inline]

Save the benchmark as a text file.

Definition at line 135 of file eval.h.

Here is the call graph for this function:

5.2.4 Member Data Documentation

5.2.4.1 unsigned K_ [private]

Definition at line 208 of file eval.h.

5.2.4.2 unsigned Q_ [private]

Definition at line 207 of file eval.h.

5.2.4.3 std::vector<unsigned> queries_ [private]

Definition at line 209 of file eval.h.

5.2.4.4 std::vector<Topk> topks_ [private]

Definition at line 210 of file eval.h.

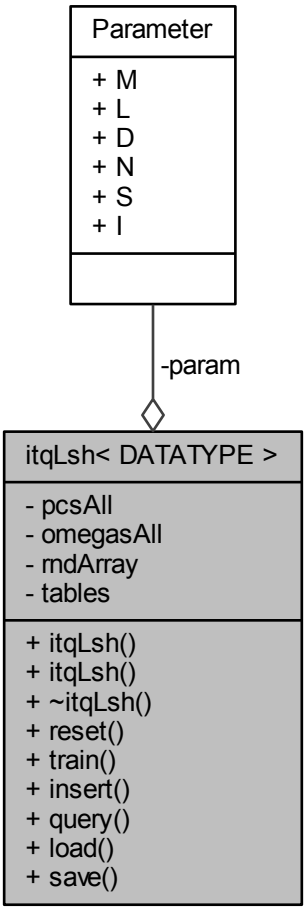
The documentation for this class was generated from the following file:

- include/lshbox/eval.h

5.3 itqLsh< DATATYPE > Class Template Reference

`#include <itqlsh.h>`

Collaboration diagram for itqLsh< DATATYPE >:



Classes

- struct Parameter

Public Member Functions

- itqLsh ()
- itqLsh (const Parameter &param_)
- ~itqLsh ()
- void reset (const Parameter &param_)
- void train (Matrix< DATATYPE > &data)
- void insert (unsigned key, DATATYPE *domin)
- template<typename SCANNER >
  void query (DATATYPE *domin, SCANNER &scanner)

- void load (const std::string &file)
- void save (const std::string &file)

Private Attributes

- Parameter param
- std::vector< std::vector < std::vector< float > > > pcsAll
- std::vector< std::vector < std::vector< float > > > omegasAll
- std::vector< std::vector < unsigned > > rndArray
- std::vector< std::map < unsigned, std::vector < unsigned > > > tables

5.3.1 Detailed Description

template<typename DATATYPE = float>class lshbox::itqLsh< DATATYPE >

Locality-Sensitive Hashing Scheme Based on Iterative Quantization.

For more information on iterative quantization based LSH, see the following reference.

```
Gong Y, Lazebnik S, Gordo A, et al. Iterative quantization: A procrustean
approach to learning binary codes for large-scale image retrieval[J].
Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2013,
35(12): 2916-2929.
```

Definition at line 50 of file itqlsh.h.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 itqLsh ( ) [inline]

Definition at line 68 of file itqlsh.h.

5.3.2.2 itqLsh (const Parameter & *param_*) [inline]

Definition at line 69 of file itqlsh.h.

Here is the call graph for this function:

5.3.2.3 ~itqLsh ( ) [inline]

Definition at line 73 of file itqlsh.h.

5.3.3 Member Function Documentation

5.3.3.1 void insert (unsigned *key*, DATATYPE * *domin*) [inline]

Insert a vector to the index.

Parameters

key	The sequence number of vector
domin	The pointer to the vector

Definition at line 195 of file itqlsh.h.

Here is the caller graph for this function:

5.3.3.2 void load (const std::string & *file*) [inline]

Load the index from binary file.

Parameters

file	The path of binary file.

Definition at line 271 of file itqlsh.h.

5.3.3.3 void query (DATATYPE * *domin*, SCANNER & *scanner*) [inline]

Query the approximate nearest neighborholds.

Parameters

domin	The pointer to the vector
scanner	Top-K scanner, use for scan the approximate nearest neighborholds

Definition at line 231 of file itqlsh.h.

5.3.3.4 void reset (const Parameter & *param_*) [inline]

Reset the parameter setting

Parameters

*param_*	A instance of itqLsh<DATATYPE>::Parametor, which contains the necessary parameters

Definition at line 80 of file itqlsh.h.

Here is the caller graph for this function:

5.3.3.5 void save (const std::string & *file*) [inline]

Save the index as binary file.

Parameters

file	The path of binary file.

Definition at line 315 of file itqlsh.h.

5.3.3.6 void train (Matrix< DATATYPE > & *data*) [inline]

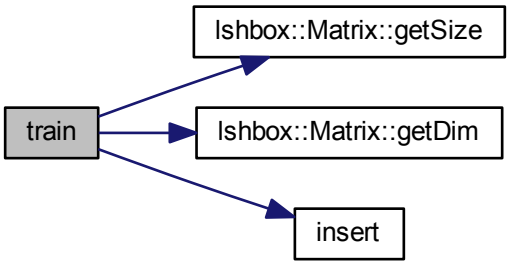
Train the data to get several groups of suitable vector for index.

Parameters

data	A instance of Matrix<DATATYPE>, most of the time, is the search library.

Definition at line 102 of file itqlsh.h.

Here is the call graph for this function:



5.3.4 Member Data Documentation

5.3.4.1 std::vector<std::vector<std::vector<float> > > omegasAll [private]

Definition at line 347 of file itqlsh.h.

5.3.4.2 Parameter param [private]

Definition at line 345 of file itqlsh.h.

5.3.4.3 std::vector<std::vector<std::vector<float> > > pcsAll [private]

Definition at line 346 of file itqlsh.h.

5.3.4.4 std::vector<std::vector<unsigned> > rndArray [private]

Definition at line 348 of file itqlsh.h.

5.3.4.5 std::vector<std::map<unsigned, std::vector<unsigned> > > tables [private]

Definition at line 349 of file itqlsh.h.

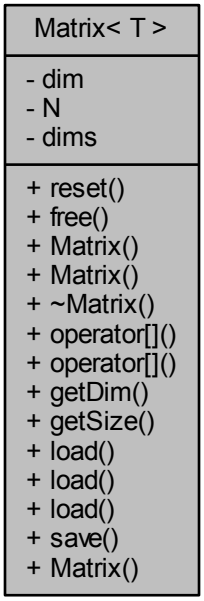
The documentation for this class was generated from the following file:

- include/lshbox/itqlsh.h

5.4 Matrix< T > Class Template Reference

`#include <matrix.h>`

Collaboration diagram for Matrix< T >:



Classes

- class Accessor

Public Member Functions

- void reset (int _dim, int _N)
- void free (void)
- Matrix ()
- Matrix (int _dim, int _N)
- ~Matrix ()
- const T * operator[] (int i) const
- T * operator[] (int i)
- int getDim () const
- int getSize () const
- void load (const std::string &path)
- void load (std::vector< T > &vec, int _N, int _dim)
- void load (T *source, int _N, int _dim)
- void save (const std::string &path)
- Matrix (const std::string &path)

Private Attributes

- int dim
- int N
- T * dims

5.4.1 Detailed Description

template<class T>class lshbox::Matrix< T >

Dataset management class. A dataset is maintained as a matrix in memory.

The file contains N D-dimensional vectors of single precision floating point numbers.

Such binary files can be accessed using lshbox::Matrix<double>.

Definition at line 43 of file matrix.h.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 Matrix ( ) [inline]

Definition at line 74 of file matrix.h.

5.4.2.2 Matrix (int *_dim,* int *_N*) [inline]

Definition at line 75 of file matrix.h.

Here is the call graph for this function:

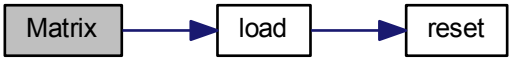
5.4.2.3 ~Matrix ( ) [inline]

Definition at line 79 of file matrix.h.

5.4.2.4 Matrix (const std::string & *path*) [inline]

Definition at line 172 of file matrix.h.

Here is the call graph for this function:



5.4.3 Member Function Documentation

5.4.3.1 void free (void) [inline]

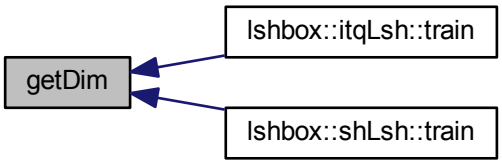
Definition at line 65 of file matrix.h.

5.4.3.2 int getDim ( ) const [inline]

Get the dimension.

Definition at line 103 of file matrix.h.

Here is the caller graph for this function:

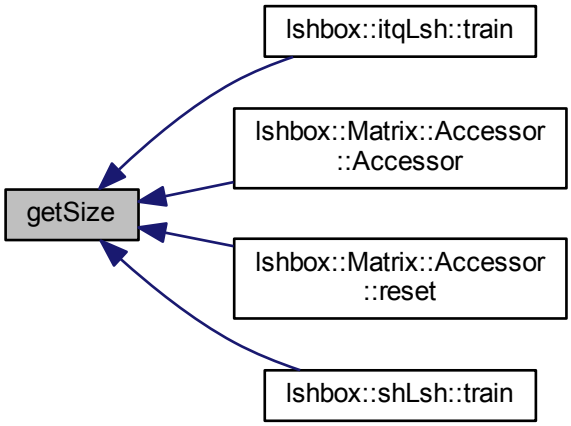


5.4.3.3 int getSize ( ) const [inline]

Get the size.

Definition at line 110 of file matrix.h.

Here is the caller graph for this function:



5.4.3.4 void load (const std::string & *path*) [inline]

Load the Matrix from a binary file.

Definition at line 117 of file matrix.h.

Here is the call graph for this function:

Here is the caller graph for this function:

5.4.3.5 void load (std::vector< T > & *vec*, int *_N*, int *_dim*) [inline]

Load the Matrix from std::vector<T>.

Parameters

vec	The reference of std::vector<T>.
_N	Number of vectors
_dim	Dimension of each vector

Definition at line 134 of file matrix.h.

Here is the call graph for this function:

5.4.3.6 void load (T * *source*, int *_N*, int *_dim*) [inline]

Load the Matrix from T*.

Parameters

source	The pointer to T∗.
_N	Number of vectors
_dim	Dimension of each vector

Definition at line 149 of file matrix.h.

Here is the call graph for this function:

5.4.3.7 const T∗ operator[ ] (int *i*) const [inline]

Access the ith vector.

Definition at line 89 of file matrix.h.

5.4.3.8 T∗ operator[ ] (int *i*) [inline]

Access the ith vector.

Definition at line 96 of file matrix.h.

5.4.3.9 void reset (int *_dim*, int *_N*) [inline]

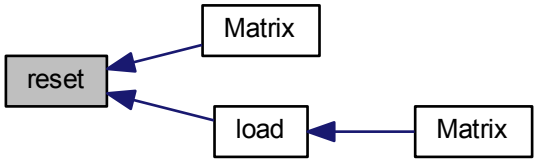
Reset the size.

Parameters

_dim	Dimension of each vector
_N	Number of vectors

Definition at line 55 of file matrix.h.

Here is the caller graph for this function:



5.4.3.10 void save (const std::string & *path*) [inline]

Save the Matrix as a binary file.

Definition at line 160 of file matrix.h.

5.4.4 Member Data Documentation

5.4.4.1 int dim [private]

Definition at line 45 of file matrix.h.

5.4.4.2 T* dims [private]

Definition at line 47 of file matrix.h.

5.4.4.3 int N [private]

Definition at line 46 of file matrix.h.

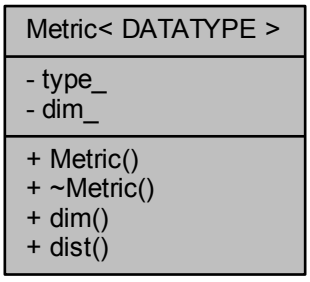
The documentation for this class was generated from the following file:

- include/lshbox/matrix.h

5.5 Metric< DATATYPE > Class Template Reference

`#include <metric.h>`

Collaboration diagram for Metric< DATATYPE >:



Public Member Functions

- Metric (unsigned dim, unsigned type)
- ~Metric ()
- unsigned dim () const
- float dist (const DATATYPE *vec1, const DATATYPE *vec2) const

Private Attributes

- unsigned type_
- unsigned dim_

5.5.1 Detailed Description

template<typename DATATYPE>class lshbox::Metric< DATATYPE >

Use for common distance functions.

Definition at line 47 of file metric.h.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Metric (unsigned *dim,* unsigned *type*) [inline]

Constructor for this class.

Parameters

dim	Dimension of each vector
type	The way to measure the distance, you can choose 1(L1_DIST) or 2(L2_DIST)

Definition at line 58 of file metric.h.

5.5.2.2 ~Metric ( ) [inline]

Definition at line 59 of file metric.h.

5.5.3 Member Function Documentation

5.5.3.1 unsigned dim ( ) const [inline]

Get the dimension of the vectors

Definition at line 63 of file metric.h.

5.5.3.2 float dist (const DATATYPE * *vec1,* const DATATYPE * *vec2*) const [inline]

measure the distance.

Parameters

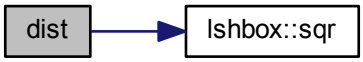
vec1	The first vector
vec2	The second vector

Returns

The distance

Definition at line 74 of file metric.h.

Here is the call graph for this function:



5.5.4 Member Data Documentation

5.5.4.1 unsigned dim_ [private]

Definition at line 50 of file metric.h.

5.5.4.2 unsigned type_ [private]

Definition at line 49 of file metric.h.

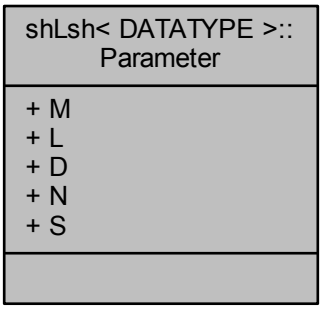
The documentation for this class was generated from the following file:

- include/lshbox/metric.h

5.6 shLsh< DATATYPE >::Parameter Struct Reference

`#include <shlsh.h>`

Collaboration diagram for shLsh< DATATYPE >::Parameter:



Public Attributes

- unsigned M

  Hash table size.
- unsigned L

  Number of hash tables.
- unsigned D

  Dimension of the vector, it can be obtained from the instance of Matrix.
- unsigned N

  Binary code bytes.
- unsigned S

  Size of vectors in train.

5.6.1 Detailed Description

template<typename DATATYPE = float>struct lshbox::shLsh< DATATYPE >::Parameter

Definition at line 51 of file shlsh.h.

5.6.2 Member Data Documentation

5.6.2.1 unsigned D

Dimension of the vector, it can be obtained from the instance of Matrix.

Definition at line 58 of file shlsh.h.

5.6.2.2 unsigned L

Number of hash tables.

Definition at line 56 of file shlsh.h.

5.6.2.3 unsigned M

Hash table size.

Definition at line 54 of file shlsh.h.

5.6.2.4 unsigned N

Binary code bytes.

Definition at line 60 of file shlsh.h.

5.6.2.5 unsigned S

Size of vectors in train.

Definition at line 62 of file shlsh.h.

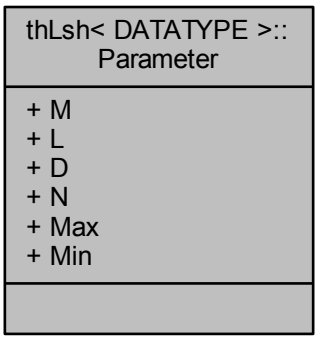
The documentation for this struct was generated from the following file:

- include/lshbox/shlsh.h

5.7 thLsh< DATATYPE >::Parameter Struct Reference

`#include <thlsh.h>`

Collaboration diagram for thLsh< DATATYPE >::Parameter:



Public Attributes

- unsigned M

  Hash table size.
- unsigned L

  Number of hash tables.
- unsigned D

  Dimension of the vector, it can be obtained from the instance of Matrix.
- unsigned N

  Binary code bytes.
- float Max

  Upper bound of each dimension.
- float Min

  Lower bound of each dimension.

5.7.1 Detailed Description

template<typename DATATYPE = float>struct lshbox::thLsh< DATATYPE >::Parameter

Definition at line 57 of file thlsh.h.

5.7.2 Member Data Documentation

5.7.2.1 unsigned D

Dimension of the vector, it can be obtained from the instance of Matrix.

Definition at line 64 of file thlsh.h.

5.7.2.2 unsigned L

Number of hash tables.

Definition at line 62 of file thlsh.h.

5.7.2.3 unsigned M

Hash table size.

Definition at line 60 of file thlsh.h.

5.7.2.4 float Max

Upper bound of each dimension.

Definition at line 68 of file thlsh.h.

5.7.2.5 float Min

Lower bound of each dimension.

Definition at line 70 of file thlsh.h.

5.7.2.6 unsigned N

Binary code bytes.

Definition at line 66 of file thlsh.h.

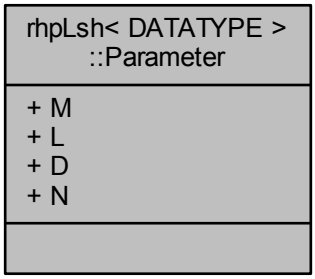
The documentation for this struct was generated from the following file:

- include/lshbox/thlsh.h

5.8 rhpLsh< DATATYPE >::Parameter Struct Reference

`#include <rhplsh.h>`

Collaboration diagram for rhpLsh< DATATYPE >::Parameter:



Public Attributes

- unsigned M

  Hash table size.
- unsigned L

  Number of hash tables.
- unsigned D

  Dimension of the vector, it can be obtained from the instance of Matrix.
- unsigned N

  Binary code bytes.

5.8.1 Detailed Description

template<typename DATATYPE = float>struct lshbox::rhpLsh< DATATYPE >::Parameter

Definition at line 52 of file rhplsh.h.

5.8.2 Member Data Documentation

5.8.2.1 unsigned D

Dimension of the vector, it can be obtained from the instance of Matrix.

Definition at line 59 of file rhplsh.h.

5.8.2.2 unsigned L

Number of hash tables.

Definition at line 57 of file rhplsh.h.

5.8.2.3 unsigned M

Hash table size.

Definition at line 55 of file rhplsh.h.

5.8.2.4 unsigned N

Binary code bytes.

Definition at line 61 of file rhplsh.h.

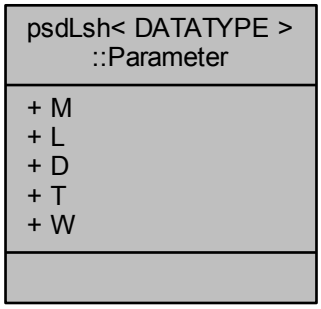
The documentation for this struct was generated from the following file:

- include/lshbox/rhplsh.h

5.9 psdLsh< DATATYPE >::Parameter Struct Reference

```
#include <psdlsh.h>
```

Collaboration diagram for psdLsh< DATATYPE >::Parameter:



Public Attributes

- unsigned M

  Hash table size.
- unsigned L

  Number of hash tables.
- unsigned D

  Dimension of the vector, it can be obtained from the instance of Matrix.
- unsigned T

  Index mode, you can choose 1(CAUCHY) or 2(GAUSSIAN)
- float W

  Window size.

5.9.1 Detailed Description

template<typename DATATYPE = float>struct lshbox::psdLsh< DATATYPE >::Parameter

Definition at line 54 of file psdlsh.h.

5.9.2 Member Data Documentation

5.9.2.1 unsigned D

Dimension of the vector, it can be obtained from the instance of Matrix.

Definition at line 61 of file psdlsh.h.

5.9.2.2 unsigned L

Number of hash tables.

Definition at line 59 of file psdlsh.h.

5.9.2.3 unsigned M

Hash table size.

Definition at line 57 of file psdlsh.h.

5.9.2.4 unsigned T

Index mode, you can choose 1(CAUCHY) or 2(GAUSSIAN)

Definition at line 63 of file psdlsh.h.

5.9.2.5 float W

Window size.

Definition at line 65 of file psdlsh.h.

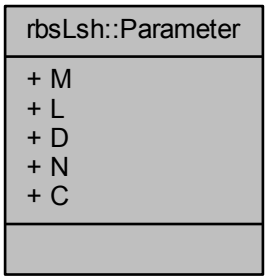
The documentation for this struct was generated from the following file:

- include/lshbox/psdlsh.h

5.10 rbsLsh::Parameter Struct Reference

`#include <rbslsh.h>`

Collaboration diagram for rbsLsh::Parameter:



Public Attributes

- unsigned M

  Hash table size.
- unsigned L

  Number of hash tables.
- unsigned D

  Dimension of the vector, it can be obtained from the instance of Matrix.
- unsigned N

  Binary code bytes.

- unsigned C

  The Difference between upper and lower bound of each dimension.

5.10.1 Detailed Description

Definition at line 54 of file rbslsh.h.

5.10.2 Member Data Documentation

5.10.2.1 unsigned C

The Difference between upper and lower bound of each dimension.

Definition at line 65 of file rbslsh.h.

5.10.2.2 unsigned D

Dimension of the vector, it can be obtained from the instance of Matrix.

Definition at line 61 of file rbslsh.h.

5.10.2.3 unsigned L

Number of hash tables.

Definition at line 59 of file rbslsh.h.

5.10.2.4 unsigned M

Hash table size.

Definition at line 57 of file rbslsh.h.

5.10.2.5 unsigned N

Binary code bytes.

Definition at line 63 of file rbslsh.h.

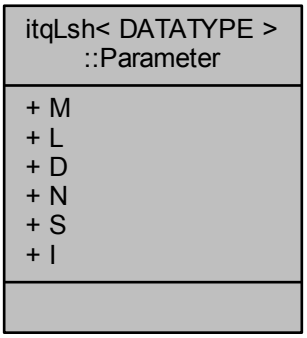
The documentation for this struct was generated from the following file:

- include/lshbox/rbslsh.h

5.11 itqLsh< DATATYPE >::Parameter Struct Reference

```
#include <itqlsh.h>
```

Collaboration diagram for itqLsh< DATATYPE >::Parameter:



Public Attributes

- unsigned M
  Hash table size.
- unsigned L
  Number of hash tables.
- unsigned D
  Dimension of the vector, it can be obtained from the instance of Matrix.
- unsigned N
  Binary code bytes.
- unsigned S
  Size of vectors in train.
- unsigned I
  Training iterations.

5.11.1 Detailed Description

template<typename DATATYPE = float>struct lshbox::itqLsh< DATATYPE >::Parameter

Definition at line 53 of file itqlsh.h.

5.11.2 Member Data Documentation

5.11.2.1 unsigned D

Dimension of the vector, it can be obtained from the instance of Matrix.

Definition at line 60 of file itqlsh.h.

5.11.2.2 unsigned I

Training iterations.

Definition at line 66 of file itqlsh.h.

5.11.2.3 unsigned L

Number of hash tables.

Definition at line 58 of file itqlsh.h.

5.11.2.4 unsigned M

Hash table size.

Definition at line 56 of file itqlsh.h.

5.11.2.5 unsigned N

Binary code bytes.

Definition at line 62 of file itqlsh.h.

5.11.2.6 unsigned S

Size of vectors in train.

Definition at line 64 of file itqlsh.h.

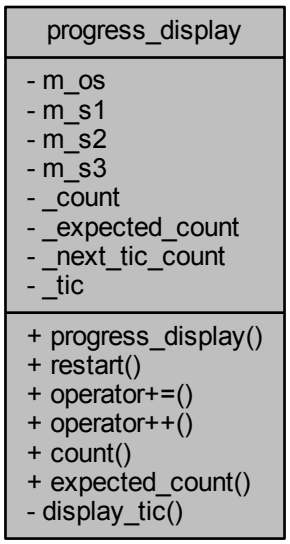
The documentation for this struct was generated from the following file:

- include/lshbox/itqlsh.h

5.12 progress_display Class Reference

```
#include <basis.h>
```

Collaboration diagram for progress_display:



Public Member Functions

- progress_display (unsigned long expected_count, std::ostream &os=std::cout, const std::string &s1="\n", const std::string &s2="", const std::string &s3="")
- void restart (unsigned long expected_count)
- unsigned long operator+= (unsigned long increment)
- unsigned long operator++ ()
- unsigned long count () const
- unsigned long expected_count () const

Private Member Functions

- void display_tic ()

Private Attributes

- std::ostream & m_os
- const std::string m_s1
- const std::string m_s2
- const std::string m_s3
- unsigned long _count
- unsigned long _expected_count
- unsigned long _next_tic_count
- unsigned _tic

5.12.1 Detailed Description

Displays an appropriate indication of progress at an appropriate place in an appropriate form.

If you are familiar with the Boost library, you should be very familiar with this class.

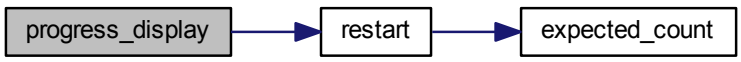
Definition at line 51 of file basis.h.

5.12.2 Constructor & Destructor Documentation

5.12.2.1 progress_display (unsigned long *expected_count*, std::ostream & *os* = `std::cout`, const std::string & *s1* = `"\n"`, const std::string & *s2* = `" "`, const std::string & *s3* = `" "`) [inline],[explicit]

Definition at line 54 of file basis.h.

Here is the call graph for this function:



5.12.3 Member Function Documentation

5.12.3.1 unsigned long count ( ) const [inline]

Definition at line 89 of file basis.h.

5.12.3.2 void display_tic ( ) [inline],[private]

Definition at line 104 of file basis.h.

Here is the caller graph for this function:

5.12.3.3 unsigned long expected_count ( ) const [inline]

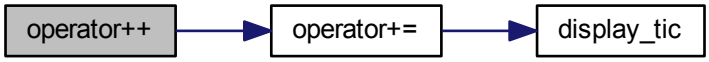
Definition at line 93 of file basis.h.

Here is the caller graph for this function:

5.12.3.4 unsigned long operator++ ( ) [inline]

Definition at line 85 of file basis.h.

Here is the call graph for this function:



5.12.3.5 unsigned long operator+= (unsigned long *increment*) [inline]

Definition at line 77 of file basis.h.

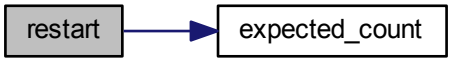
Here is the call graph for this function:

Here is the caller graph for this function:

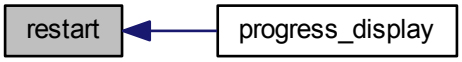
5.12.3.6 void restart (unsigned long *expected_count*) [inline]

Definition at line 64 of file basis.h.

Here is the call graph for this function:



Here is the caller graph for this function:



5.12.4 Member Data Documentation

5.12.4.1 unsigned long _count [private]

Definition at line 102 of file basis.h.

5.12.4.2 unsigned long _expected_count [private]

Definition at line 102 of file basis.h.

5.12.4.3 unsigned long _next_tic_count [private]

Definition at line 102 of file basis.h.

5.12.4.4 unsigned _tic [private]

Definition at line 103 of file basis.h.

5.12.4.5 std::ostream& m_os [private]

Definition at line 98 of file basis.h.

5.12.4.6 const std::string m_s1 [private]

Definition at line 99 of file basis.h.

5.12.4.7 const std::string m_s2 [private]

Definition at line 100 of file basis.h.

5.12.4.8 const std::string m_s3 [private]

Definition at line 101 of file basis.h.

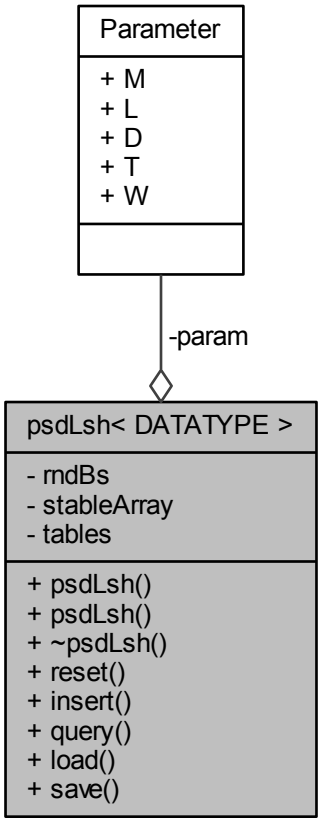
The documentation for this class was generated from the following file:

- include/lshbox/basis.h

5.13 psdLsh< DATATYPE > Class Template Reference

`#include <psdlsh.h>`

Collaboration diagram for psdLsh< DATATYPE >:



Classes

- struct Parameter

Public Member Functions

- psdLsh ()
- psdLsh (const Parameter &param_)
- ~psdLsh ()
- void reset (const Parameter &param_)
- void insert (unsigned key, DATATYPE *domin)
- template<typename SCANNER >
  void query (DATATYPE *domin, SCANNER &scanner)
- void load (const std::string &file)
- void save (const std::string &file)

Private Attributes

- Parameter param
- std::vector< float > rndBs
- std::vector< std::vector< float > > stableArray
- std::vector< std::map
  < unsigned, std::vector
  < unsigned > > > tables

5.13.1 Detailed Description

template<typename DATATYPE = float>class lshbox::psdLsh< DATATYPE >

Locality-Sensitive Hashing Scheme Based on p-Stable Distributions.

For more information on p-stable distribution based LSH, see the following reference.

```
Mayur Datar , Nicole Immorlica , Piotr Indyk , Vahab S. Mirrokni,
Locality-sensitive hashing scheme based on p-stable distributions,
Proceedings of the twentieth annual symposium on Computational geometry, June
08-11, 2004, Brooklyn, New York, USA.
```

Definition at line 49 of file psdlsh.h.

5.13.2 Constructor & Destructor Documentation

5.13.2.1 psdLsh ( ) [inline]

Definition at line 67 of file psdlsh.h.

5.13.2.2 psdLsh (const Parameter & *param_*) [inline]

Definition at line 68 of file psdlsh.h.

Here is the call graph for this function:

5.13.2.3 ~psdLsh ( ) [inline]

Definition at line 72 of file psdlsh.h.

5.13.3 Member Function Documentation

5.13.3.1 void insert (unsigned *key*, DATATYPE * *domin*) [inline]

Insert a vector to the index.

Parameters

key	The sequence number of vector
domin	The pointer to the vector

Definition at line 126 of file psdlsh.h.

5.13.3.2 void load (const std::string & *file*) [inline]

Load the index from binary file.

Parameters

file	The path of binary file.

Definition at line 170 of file psdlsh.h.

5.13.3.3 void query (DATATYPE * *domin*, SCANNER & *scanner*) [inline]

Query the approximate nearest neighborholds.

Parameters

domin	The pointer to the vector
scanner	Top-K scanner, use for scan the approximate nearest neighborholds

Definition at line 146 of file psdlsh.h.

5.13.3.4 void reset (const Parameter & *param_*) [inline]

Reset the parameter setting

Parameters

*param_*	A instance of psdLsh<DATATYPE>::Parametor, which contains the necessary parameters

Definition at line 79 of file psdlsh.h.

Here is the caller graph for this function:

5.13.3.5 void save (const std::string & *file*) [inline]

Save the index as binary file.

Parameters

file	The path of binary file.

Definition at line 204 of file psdlsh.h.

5.13.4 Member Data Documentation

5.13.4.1 Parameter param [private]

Definition at line 229 of file psdlsh.h.

5.13.4.2 std::vector<float> rndBs [private]

Definition at line 230 of file psdlsh.h.

5.13.4.3 std::vector<std::vector<float> > stableArray [private]

Definition at line 231 of file psdlsh.h.

5.13.4.4 std::vector<std::map<unsigned, std::vector<unsigned> > > tables [private]

Definition at line 232 of file psdlsh.h.

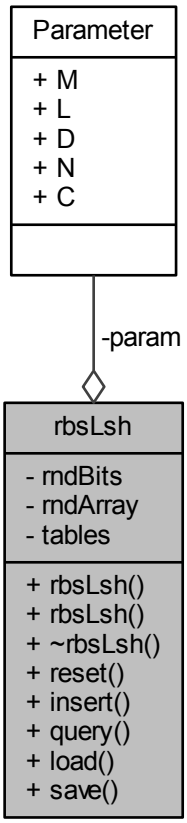
The documentation for this class was generated from the following file:

- include/lshbox/psdlsh.h

5.14 rbsLsh Class Reference

`#include <rbslsh.h>`

Collaboration diagram for rbsLsh:



Classes

- struct Parameter

Public Member Functions

- rbsLsh ()
- rbsLsh (const Parameter &param_)
- ~rbsLsh ()
- void reset (const Parameter &param_)
- void insert (unsigned key, unsigned *domin)
- template<typename SCANNER >
  void query (unsigned *domin, SCANNER &scanner)
- void load (const std::string &file)
- void save (const std::string &file)

Private Attributes

- Parameter param

- std::vector< std::vector < unsigned > > rndBits
- std::vector< std::vector < unsigned > > rndArray
- std::vector< std::map < unsigned, std::vector < unsigned > > > tables

5.14.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on Random Bits Sampling.

For more information on random bits sampling based LSH, see the following reference.

```
P. Indyk and R. Motwani. Approximate Nearest Neighbor - Towards Removing
the Curse of Dimensionality. In Proceedings of the 30th Symposium on Theory
of Computing, 1998, pp. 604-613.

A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions
via hashing. Proceedings of the 25th International Conference on Very Large
Data Bases (VLDB), 1999.
```

Definition at line 51 of file rbslsh.h.

5.14.2 Constructor & Destructor Documentation

5.14.2.1 rbsLsh ( ) [inline]

Definition at line 67 of file rbslsh.h.

5.14.2.2 rbsLsh (const Parameter & *param_*) [inline]

Definition at line 68 of file rbslsh.h.

Here is the call graph for this function:

5.14.2.3 ~rbsLsh ( ) [inline]

Definition at line 72 of file rbslsh.h.

5.14.3 Member Function Documentation

5.14.3.1 void insert (unsigned *key*, unsigned * *domin*) [inline]

Insert a vector to the index.

Parameters

key	The sequence number of vector
domin	The pointer to the vector

Definition at line 114 of file rbslsh.h.

5.14.3.2 void load (const std::string & *file*) [inline]

Load the index from binary file.

Parameters

file	The path of binary file.

Definition at line 162 of file rbslsh.h.

5.14.3.3 void query (unsigned * *domin*, SCANNER & *scanner*) [inline]

Query the approximate nearest neighborholds.

Parameters

domin	The pointer to the vector
scanner	Top-K scanner, use for scan the approximate nearest neighborholds

Definition at line 137 of file rbslsh.h.

5.14.3.4 void reset (const Parameter & *param_*) [inline]

Reset the parameter setting

Parameters

*param_*	A instance of rbsLsh::Parametor, which contains the necessary parameters

Definition at line 79 of file rbslsh.h.

Here is the caller graph for this function:

5.14.3.5 void save (const std::string & *file*) [inline]

Save the index as binary file.

Parameters

file	The path of binary file.

Definition at line 198 of file rbslsh.h.

5.14.4 Member Data Documentation

5.14.4.1 Parameter param [private]

Definition at line 224 of file rbslsh.h.

5.14.4.2 std::vector<std::vector<unsigned> > rndArray [private]

Definition at line 226 of file rbslsh.h.

5.14.4.3 std::vector<std::vector<unsigned> > rndBits [private]

Definition at line 225 of file rbslsh.h.

5.14.4.4 std::vector<std::map<unsigned, std::vector<unsigned> > > tables [private]

Definition at line 227 of file rbslsh.h.

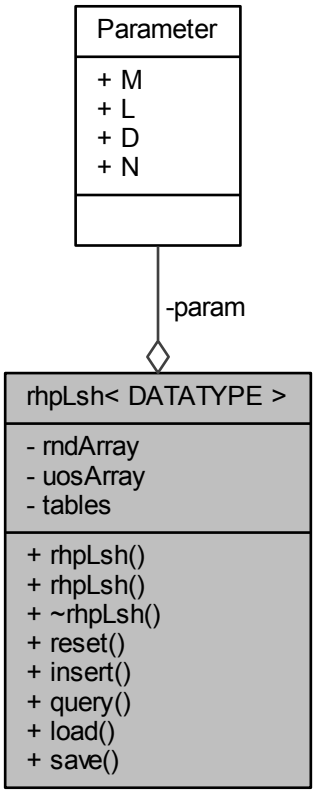
The documentation for this class was generated from the following file:

- include/lshbox/rbslsh.h

5.15 rhpLsh< DATATYPE > Class Template Reference

```
#include <rhplsh.h>
```

Collaboration diagram for rhpLsh< DATATYPE >:



Classes

- struct Parameter

Public Member Functions

- rhpLsh ()
- rhpLsh (const Parameter &param_)
- ~rhpLsh ()
- void reset (const Parameter &param_)
- void insert (unsigned key, DATATYPE *domin)
- template<typename SCANNER >
  void query (DATATYPE *domin, SCANNER &scanner)
- void load (const std::string &file)
- void save (const std::string &file)

Private Attributes

- Parameter param

- std::vector< std::vector < unsigned > > rndArray
- std::vector< std::vector < std::vector< float > > > uosArray
- std::vector< std::map < unsigned, std::vector < unsigned > > > tables

5.15.1 Detailed Description

template<typename DATATYPE = float>class lshbox::rhpLsh< DATATYPE >

Locality-Sensitive Hashing Scheme Based on Random Hyperplane.

For more information on random hyperplane based LSH, see the following reference.

```
Charikar, M. S. 2002. Similarity estimation techniques from rounding
algorithms. In Proceedings of the Thiry-Fourth Annual ACM Symposium on
theory of Computing (Montreal, Quebec, Canada, May 19 - 21, 2002). STOC '02.
ACM, New York, NY, 380-388. DOI= http://doi.acm.org/10.1145/509907.509965
```

Definition at line 49 of file rhplsh.h.

5.15.2 Constructor & Destructor Documentation

5.15.2.1 rhpLsh ( ) [inline]

Definition at line 63 of file rhplsh.h.

5.15.2.2 rhpLsh (const Parameter & *param_*) [inline]

Definition at line 64 of file rhplsh.h.

Here is the call graph for this function:

5.15.2.3 ~rhpLsh ( ) [inline]

Definition at line 68 of file rhplsh.h.

5.15.3 Member Function Documentation

5.15.3.1 void insert (unsigned *key*, DATATYPE * *domin*) [inline]

Insert a vector to the index.

Parameters

key	The sequence number of vector
domin	The pointer to the vector

Definition at line 109 of file rhplsh.h.

5.15.3.2 void load (const std::string & *file*) [inline]

Load the index from binary file.

Parameters

file	The path of binary file.

Definition at line 169 of file rhplsh.h.

5.15.3.3 void query (DATATYPE * *domin,* SCANNER & *scanner*) [inline]

Query the approximate nearest neighborholds.

Parameters

domin	The pointer to the vector
scanner	Top-K scanner, use for scan the approximate nearest neighborholds

Definition at line 137 of file rhplsh.h.

5.15.3.4 void reset (const Parameter & *param_*) [inline]

Reset the parameter setting

Parameters

*param_*	A instance of rhpLsh<DATATYPE>::Parametor, which contains the necessary parameters

Definition at line 75 of file rhplsh.h.

Here is the caller graph for this function:

5.15.3.5 void save (const std::string & *file*) [inline]

Save the index as binary file.

Parameters

file	The path of binary file.

Definition at line 208 of file rhplsh.h.

5.15.4 Member Data Documentation

5.15.4.1 Parameter param [private]

Definition at line 236 of file rhplsh.h.

5.15.4.2 std::vector<std::vector<unsigned> > rndArray [private]

Definition at line 237 of file rhplsh.h.

5.15.4.3 std::vector<std::map<unsigned, std::vector<unsigned> > > tables [private]

Definition at line 239 of file rhplsh.h.

5.15.4.4 std::vector<std::vector<std::vector<float> > > uosArray [private]

Definition at line 238 of file rhplsh.h.

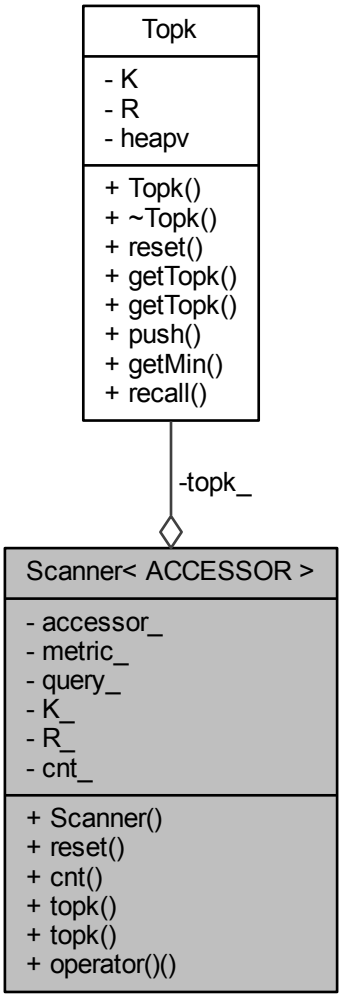
The documentation for this class was generated from the following file:

- include/lshbox/rhplsh.h

5.16 Scanner< ACCESSOR > Class Template Reference

```
#include <topk.h>
```

Collaboration diagram for Scanner< ACCESSOR >:



Public Types

- typedef ACCESSOR::Value Value
- typedef ACCESSOR::DATATYPE DATATYPE

Public Member Functions

- Scanner (const ACCESSOR &accessor, const Metric< DATATYPE > &metric, unsigned K, float R=std↩ ::numeric_limits< float >::max())
- void reset (Value query)
- unsigned cnt () const
- const Topk & topk () const
- Topk & topk ()
- void operator() (unsigned key)

Private Attributes

- ACCESSOR accessor_
- Metric< DATATYPE > metric_
- Topk topk_
- Value query_
- unsigned K_
- float R_
- unsigned cnt_

5.16.1 Detailed Description

template<typename ACCESSOR>class lshbox::Scanner< ACCESSOR >

Top-K scanner.

Scans keys for top-K query, this is the object passed into the LSH query interface.

Definition at line 130 of file topk.h.

5.16.2 Member Typedef Documentation

5.16.2.1 typedef ACCESSOR::DATATYPE DATATYPE

Definition at line 134 of file topk.h.

5.16.2.2 typedef ACCESSOR::Value Value

Definition at line 133 of file topk.h.

5.16.3 Constructor & Destructor Documentation

5.16.3.1 Scanner (const ACCESSOR & *accessor*, const Metric< DATATYPE > & *metric*, unsigned *K*, float *R* = std::numeric_limits<float>::max()) [inline]

Constructor for this class.

Parameters

accessor	The scanner use accessor to retrieva values from keys.
metric	The distance metric.
K	Value used to reset internal Topk class.
R	Value used to reset internal Topk class.

Definition at line 143 of file topk.h.

5.16.4 Member Function Documentation

5.16.4.1 unsigned cnt ( ) const [inline]

Number of points scanned for the current query.

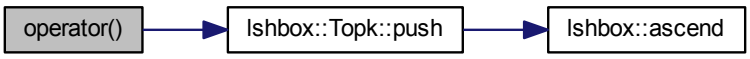
Definition at line 162 of file topk.h.

5.16.4.2 void operator() (unsigned *key*) [inline]

Update the current query by scanning key, this is normally invoked by the LSH index structure.

Definition at line 184 of file topk.h.

Here is the call graph for this function:

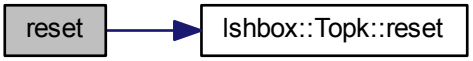


5.16.4.3 void reset (Value *query*) [inline]

Reset the query, this function should be invoked before each query.

Definition at line 152 of file topk.h.

Here is the call graph for this function:



5.16.4.4 const Topk& topk ( ) const [inline]

TopK results.

Definition at line 169 of file topk.h.

5.16.4.5 Topk& topk ( ) [inline]

TopK results.

Definition at line 176 of file topk.h.

5.16.5 Member Data Documentation

5.16.5.1 ACCESSOR accessor_ [private]

Definition at line 193 of file topk.h.

5.16.5.2 unsigned cnt_ [private]

Definition at line 199 of file topk.h.

5.16.5.3 unsigned K_ [private]

Definition at line 197 of file topk.h.

5.16.5.4 Metric<DATATYPE> metric_ [private]

Definition at line 194 of file topk.h.

5.16.5.5 Value query_ [private]

Definition at line 196 of file topk.h.

5.16.5.6 float R_ [private]

Definition at line 198 of file topk.h.

5.16.5.7 Topk topk_ [private]

Definition at line 195 of file topk.h.

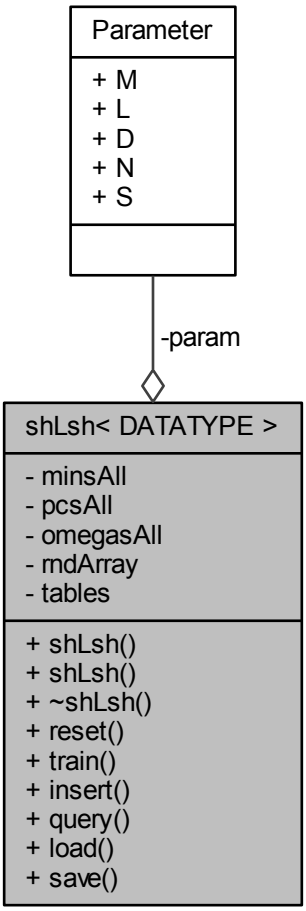
The documentation for this class was generated from the following file:

- include/lshbox/topk.h

5.17 shLsh< DATATYPE > Class Template Reference

```
#include <shlsh.h>
```

Collaboration diagram for shLsh< DATATYPE >:



Classes

- struct Parameter

Public Member Functions

- shLsh ()
- shLsh (const Parameter &param_)
- ~shLsh ()
- void reset (const Parameter &param_)
- void train (Matrix< DATATYPE > &data)
- void insert (unsigned key, DATATYPE *domin)
- template<typename SCANNER >
  void query (const DATATYPE *domin, SCANNER &scanner)
- void load (const std::string &file)
- void save (const std::string &file)

Private Attributes

- Parameter param
- std::vector< std::vector < double > > minsAll
- std::vector< std::vector < std::vector< float > > > pcsAll
- std::vector< std::vector < std::vector< float > > > omegasAll
- std::vector< std::vector < unsigned > > rndArray
- std::vector< std::map < unsigned, std::vector < unsigned > > > tables

5.17.1 Detailed Description

template<typename DATATYPE = float>class lshbox::shLsh< DATATYPE >

Locality-Sensitive Hashing Scheme Based on Spectral Hashing.

For more information on spectral hashing based LSH, see the following reference.

```
Y. Weiss, A. Torralba, R. Fergus. Spectral Hashing.
Advances in Neural Information Processing Systems, 2008.
```

Definition at line 48 of file shlsh.h.

5.17.2 Constructor & Destructor Documentation

5.17.2.1 shLsh ( ) [inline]

Definition at line 64 of file shlsh.h.

5.17.2.2 shLsh (const Parameter & *param_*) [inline]

Definition at line 65 of file shlsh.h.

Here is the call graph for this function:

5.17.2.3 ~shLsh ( ) [inline]

Definition at line 69 of file shlsh.h.

5.17.3 Member Function Documentation

5.17.3.1 void insert (unsigned *key*, DATATYPE * *domin*) [inline]

Insert a vector to the index.

Parameters

key	The sequence number of vector
domin	The pointer to the vector

Definition at line 213 of file shlsh.h.

Here is the caller graph for this function:

5.17.3.2 void load (const std::string & *file*) [inline]

Load the index from binary file.

Parameters

file	The path of binary file.

Definition at line 291 of file shlsh.h.

5.17.3.3 void query (const DATATYPE * *domin,* SCANNER & *scanner*) [inline]

Query the approximate nearest neighborholds.

Parameters

domin	The pointer to the vector
scanner	Top-K scanner, use for scan the approximate nearest neighborholds

Definition at line 250 of file shlsh.h.

5.17.3.4 void reset (const Parameter & *param_*) [inline]

Reset the parameter setting

Parameters

*param_*	A instance of shLsh<DATATYPE>::Parametor, which contains the necessary parameters

Definition at line 76 of file shlsh.h.

Here is the caller graph for this function:

5.17.3.5 void save (const std::string & *file*) [inline]

Save the index as binary file.

Parameters

file	The path of binary file.

Definition at line 338 of file shlsh.h.

5.17.3.6 void train (Matrix< DATATYPE > & *data*) [inline]

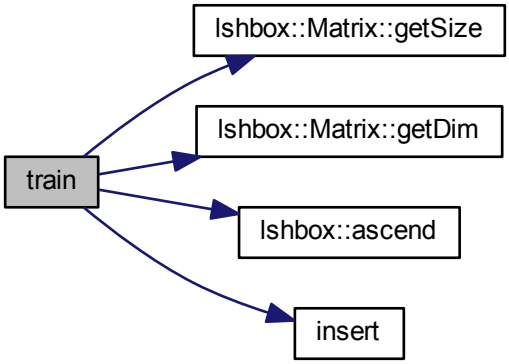
Train the data to get several groups of suitable vector for index.

Parameters

data	A instance of Matrix<DATATYPE>, most of the time, is the search library.

Definition at line 99 of file shlsh.h.

Here is the call graph for this function:



5.17.4 Member Data Documentation

5.17.4.1 std::vector<std::vector<double> > minsAll [private]

Definition at line 370 of file shlsh.h.

5.17.4.2 std::vector<std::vector<std::vector<float> > > omegasAll [private]

Definition at line 372 of file shlsh.h.

5.17.4.3 Parameter param [private]

Definition at line 369 of file shlsh.h.

5.17.4.4 std::vector<std::vector<std::vector<float> > > pcsAll [private]

Definition at line 371 of file shlsh.h.

5.17.4.5 std::vector<std::vector<unsigned> > rndArray [private]

Definition at line 373 of file shlsh.h.

5.17.4.6 std::vector<std::map<unsigned, std::vector<unsigned> > > tables [private]

Definition at line 374 of file shlsh.h.

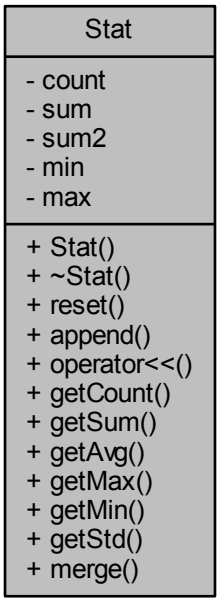
The documentation for this class was generated from the following file:

- include/lshbox/shlsh.h

5.18 Stat Class Reference

```
#include <eval.h>
```

Collaboration diagram for Stat:



Public Member Functions

- Stat ()
- ~Stat ()
- void reset ()
- void append (float r)
- Stat & operator<< (float r)
- int getCount () const
- float getSum () const
- float getAvg () const
- float getMax () const
- float getMin () const
- float getStd () const
- void merge (const Stat &stat)

Private Attributes

- int count
- float sum
- float sum2
- float min
- float max

5.18.1 Detailed Description

Use for basic statistics, and the interface is self-evident.

Usage:

```
Stat stat;

stat << 1.0 << 2.0 << 3.0;

Stat stat2;
stat2 << 3.0 << 5.0 << 6.0;

stat.merge(stat2);

stat.getCount();
stat.getSum();
stat.getMax();
stat.getMin();
stat.getStd();
```

Definition at line 235 of file eval.h.

5.18.2 Constructor & Destructor Documentation

5.18.2.1 Stat ( ) [inline]

Definition at line 243 of file eval.h.

5.18.2.2 ~Stat ( ) [inline]

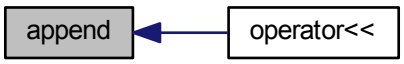
Definition at line 244 of file eval.h.

5.18.3 Member Function Documentation

5.18.3.1 void append (float *r*) [inline]

Definition at line 252 of file eval.h.

Here is the caller graph for this function:



5.18.3.2 float getAvg ( ) const [inline]

Definition at line 273 of file eval.h.

5.18.3.3 int getCount ( ) const [inline]

Definition at line 265 of file eval.h.

5.18.3.4 float getMax ( ) const [inline]

Definition at line 277 of file eval.h.

5.18.3.5 float getMin ( ) const [inline]

Definition at line 281 of file eval.h.

5.18.3.6 float getStd ( ) const [inline]

Definition at line 285 of file eval.h.

5.18.3.7 float getSum ( ) const [inline]

Definition at line 269 of file eval.h.

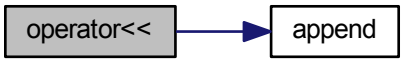
5.18.3.8 void merge (const Stat & *stat*) [inline]

Definition at line 296 of file eval.h.

5.18.3.9 Stat& operator<< (float *r*) [inline]

Definition at line 260 of file eval.h.

Here is the call graph for this function:



5.18.3.10 void reset ( ) [inline]

Definition at line 245 of file eval.h.

5.18.4 Member Data Documentation

5.18.4.1 int count [private]

Definition at line 237 of file eval.h.

5.18.4.2 float max [private]

Definition at line 241 of file eval.h.

5.18.4.3 float min [private]

Definition at line 240 of file eval.h.

5.18.4.4 float sum [private]

Definition at line 238 of file eval.h.

5.18.4.5 float sum2 [private]

Definition at line 239 of file eval.h.

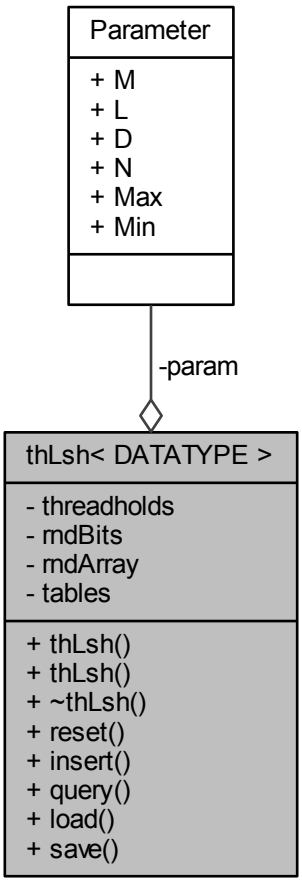
The documentation for this class was generated from the following file:

- include/lshbox/eval.h

5.19 thLsh< DATATYPE > Class Template Reference

```
#include <thlsh.h>
```

Collaboration diagram for thLsh< DATATYPE >:



Classes

- struct Parameter

Public Member Functions

- thLsh ()
- thLsh (const Parameter &param_)
- ~thLsh ()
- void reset (const Parameter &param_)
- void insert (unsigned key, DATATYPE *domin)
- template<typename SCANNER >
  void query (DATATYPE *domin, SCANNER &scanner)
- void load (const std::string &file)
- void save (const std::string &file)

Private Attributes

- Parameter param
- std::vector< float > threadholds
- std::vector< std::vector < unsigned > > rndBits
- std::vector< std::vector < unsigned > > rndArray
- std::vector< std::map < unsigned, std::vector < unsigned > > > tables

5.19.1 Detailed Description

template<typename DATATYPE = float>class lshbox::thLsh< DATATYPE >

Locality-Sensitive Hashing Scheme Based on Thresholding.

For more information on thresholding based LSH, see the following reference.

```
Zhe Wang, Wei Dong, William Josephson, Qin Lv, Moses Charikar, Kai Li.
Sizing Sketches: A Rank-Based Analysis for Similarity Search. In
Proceedings of the 2007 ACM SIGMETRICS International Conference on
Measurement and Modeling of Computer Systems . San Diego, CA, USA. June
2007.

Qin Lv, Moses Charikar, Kai Li. Image Similarity Search with Compact
Data Structures. In Proceedings of ACM 13th Conference on Information
and Knowledge Management (CIKM), Washington D.C., USA. November 2004.
```

Definition at line 54 of file thlsh.h.

5.19.2 Constructor & Destructor Documentation

5.19.2.1 thLsh ( ) [inline]

Definition at line 72 of file thlsh.h.

5.19.2.2 thLsh (const Parameter & *param_*) [inline]

Definition at line 73 of file thlsh.h.

Here is the call graph for this function:

5.19.2.3 ~thLsh ( ) [inline]

Definition at line 77 of file thlsh.h.

5.19.3 Member Function Documentation

5.19.3.1 void insert (unsigned *key,* DATATYPE * *domin*) [inline]

Insert a vector to the index.

Parameters

key	The sequence number of vector
domin	The pointer to the vector

Definition at line 121 of file thlsh.h.

5.19.3.2 void load (const std::string & *file*) [inline]

Load the index from binary file.

Parameters

file	The path of binary file.

Definition at line 173 of file thlsh.h.

5.19.3.3 void query (DATATYPE * *domin,* SCANNER & *scanner*) [inline]

Query the approximate nearest neighborholds.

Parameters

domin	The pointer to the vector
scanner	Top-K scanner, use for scan the approximate nearest neighborholds

Definition at line 145 of file thlsh.h.

5.19.3.4 void reset (const Parameter & *param_*) [inline]

Reset the parameter setting

Parameters

*param_*	A instance of thLsh<DATATYPE>::Parametor, which contains the necessary parameters

Definition at line 84 of file thlsh.h.

Here is the caller graph for this function:

5.19.3.5 void save (const std::string & *file*) [inline]

Save the index as binary file.

Parameters

file	The path of binary file.

Definition at line 212 of file thlsh.h.

5.19.4 Member Data Documentation

5.19.4.1 Parameter param [private]

Definition at line 240 of file thlsh.h.

5.19.4.2 std::vector<std::vector<unsigned> > rndArray [private]

Definition at line 243 of file thlsh.h.

5.19.4.3 std::vector<std::vector<unsigned> > rndBits [private]

Definition at line 242 of file thlsh.h.

5.19.4.4 std::vector<std::map<unsigned, std::vector<unsigned> > > tables [private]

Definition at line 244 of file thlsh.h.

5.19.4.5 std::vector<float> threadholds [private]

Definition at line 241 of file thlsh.h.

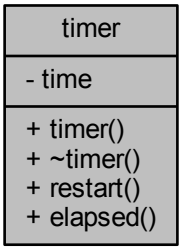
The documentation for this class was generated from the following file:

- include/lshbox/thlsh.h

5.20 timer Class Reference

`#include <basis.h>`

Collaboration diagram for timer:



Public Member Functions

- timer ()
- ~timer ()
- void restart ()
- double elapsed ()

Private Attributes

- double time

5.20.1 Detailed Description

A timer object measures elapsed time, and it is very similar to boost::timer.

Definition at line 123 of file basis.h.

5.20.2 Constructor & Destructor Documentation

5.20.2.1 timer ( ) [inline]

Definition at line 126 of file basis.h.

5.20.2.2 ~timer ( ) [inline]

Definition at line 127 of file basis.h.

5.20.3 Member Function Documentation

5.20.3.1 double elapsed ( ) [inline]

Measures elapsed time.

Returns

The elapsed time

Definition at line 140 of file basis.h.

5.20.3.2 void restart ( ) [inline]

Restart the timer.

Definition at line 131 of file basis.h.

5.20.4 Member Data Documentation

5.20.4.1 double time [private]

Definition at line 145 of file basis.h.

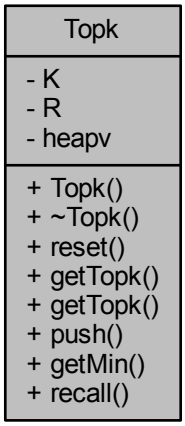
The documentation for this class was generated from the following file:

- include/lshbox/basis.h

5.21 Topk Class Reference

`#include <topk.h>`

Collaboration diagram for Topk:



Public Member Functions

- Topk ()
- ~Topk ()
- void reset (unsigned k, float r=std::numeric_limits< float >::max())
- const std::vector< std::pair < unsigned, float > > & getTopk () const
- std::vector< std::pair < unsigned, float > > & getTopk ()
- void push (unsigned key, float dist)
- float getMin () const
- float recall (const Topk &topk) const

Private Attributes

- unsigned K
- float R
- std::vector< std::pair < unsigned, float > > heapv

5.21.1 Detailed Description

Top-K heap.

At this point topk should contain the nearest k query keys and distances.

Definition at line 40 of file topk.h.

5.21.2 Constructor & Destructor Documentation

5.21.2.1 Topk ( ) [inline]

Definition at line 47 of file topk.h.

5.21.2.2 ~Topk ( ) [inline]

Definition at line 48 of file topk.h.

5.21.3 Member Function Documentation

5.21.3.1 float getMin ( ) const [inline]

Get the most nearest distance in the heap.

Definition at line 100 of file topk.h.

5.21.3.2 const std::vector<std::pair<unsigned, float> >& getTopk ( ) const [inline]

Get the std::vector<std::pair<unsigned, float> > instance which contains the nearest keys and distances.

Definition at line 70 of file topk.h.

Here is the caller graph for this function:

5.21.3.3 std::vector<std::pair<unsigned, float> >& getTopk ( ) [inline]

Get the std::vector<std::pair<unsigned, float> > instance which contains the nearest keys and distances.

Definition at line 78 of file topk.h.

5.21.3.4 void push (unsigned *key*, float *dist*) [inline]

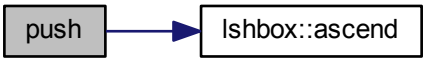
Add an element to the heap and then update it.

Parameters

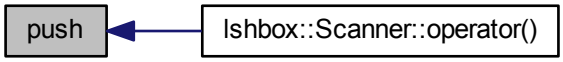
key	The key of the element
dist	The distance of the element

Definition at line 88 of file topk.h.

Here is the call graph for this function:



Here is the caller graph for this function:



5.21.3.5 float recall (const Topk & *topk*) const [inline]

Calculate the recall vale with another heap.

Definition at line 107 of file topk.h.

Here is the call graph for this function:

5.21.3.6 void reset (unsigned *k*, float *r* = std::numeric_limits<float>::max()) [inline]

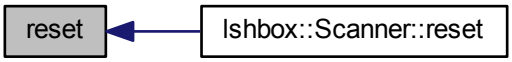
Reset the heap

Parameters

k	The number of nearest query keys
r	The initialization distance

Definition at line 55 of file topk.h.

Here is the caller graph for this function:



5.21.4 Member Data Documentation

5.21.4.1 std::vector<std::pair<unsigned, float> > heapv [private]

Definition at line 45 of file topk.h.

5.21.4.2 unsigned K [private]

Definition at line 43 of file topk.h.

5.21.4.3 float R [private]

Definition at line 44 of file topk.h.

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

- include/lshbox/topk.h

Chapter 6

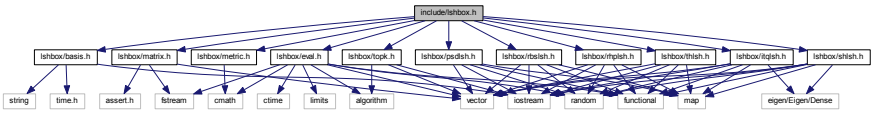
File Documentation

6.1 include/lshbox.h File Reference

The LSHBOX master header file.

```
#include <lshbox/basis.h>
#include <lshbox/matrix.h>
#include <lshbox/metric.h>
#include <lshbox/topk.h>
#include <lshbox/eval.h>
#include <lshbox/rbslsh.h>
#include <lshbox/rhplsh.h>
#include <lshbox/thlsh.h>
#include <lshbox/psdlsh.h>
#include <lshbox/shlsh.h>
#include <lshbox/itqlsh.h>
```

Include dependency graph for lshbox.h:



6.1.1 Detailed Description

The LSHBOX master header file.

This file is part of LSHBOX.

LSHBOX is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or(at your option) any later version.

LSHBOX is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with LSHBOX. If not, see http↩ ://www.gnu.org/licenses/.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

You only need to include this file to use the most functionalities of LSHBOX.

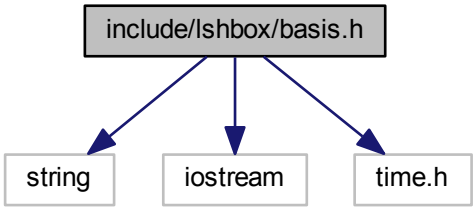
Definition in file lshbox.h.

6.2 include/lshbox/basis.h File Reference

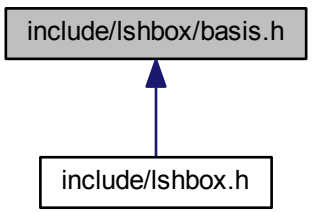
A set of basic tools.

```
#include <string>
#include <iostream>
#include <time.h>
```

Include dependency graph for basis.h:



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



Classes

- class progress_display
- class timer

Namespaces

- lshbox

Macros

- #define M_PI 3.14159265358979323846

Functions

- bool ascend (const std::pair< unsigned, float > &lhs, const std::pair< unsigned, float > &rhs)

6.2.1 Detailed Description

A set of basic tools.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

Definition in file basis.h.

6.2.2 Macro Definition Documentation

6.2.2.1 #define M_PI 3.14159265358979323846

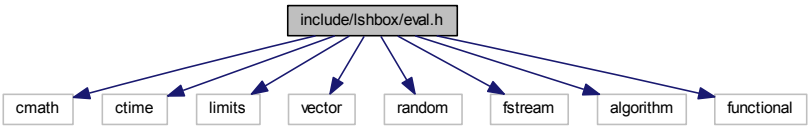
Definition at line 35 of file basis.h.

6.3 include/lshbox/eval.h File Reference

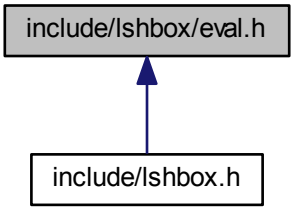
A set of classes for evaluation.

```
#include <cmath>
#include <ctime>
#include <limits>
#include <vector>
#include <random>
#include <fstream>
#include <algorithm>
#include <functional>
```

Include dependency graph for eval.h:



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



Classes

- class Benchmark
- class Stat

Namespaces

- lshbox

6.3.1 Detailed Description

A set of classes for evaluation.

This file is part of LSHBOX.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

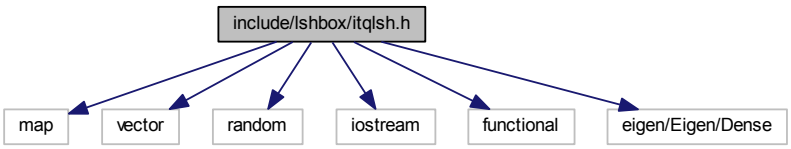
Definition in file eval.h.

6.4 include/lshbox/itqlsh.h File Reference

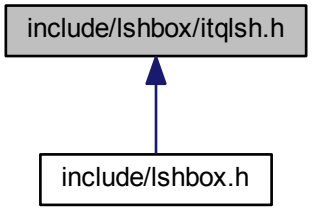
Locality-Sensitive Hashing Scheme Based on Iterative Quantization.

```
#include <map>
#include <vector>
#include <random>
#include <iostream>
#include <functional>
#include <eigen/Eigen/Dense>
```

Include dependency graph for itqlsh.h:



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



Classes

- class itqLsh< DATATYPE >
- struct itqLsh< DATATYPE >::Parameter

Namespaces

- lshbox

6.4.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on Iterative Quantization.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

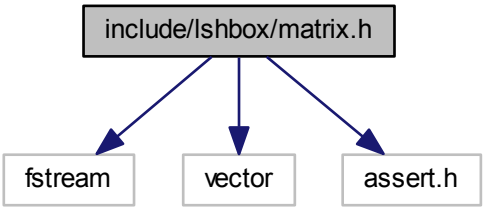
Definition in file itqlsh.h.

6.5 include/lshbox/matrix.h File Reference

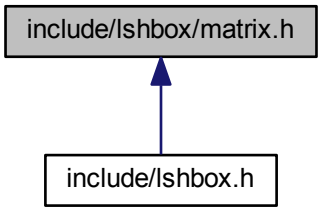
Dataset management class.

```
#include <fstream>
#include <vector>
#include <assert.h>
```

Include dependency graph for matrix.h:



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



Classes

- class Matrix< T >
- class Matrix< T >::Accessor

Namespaces

- lshbox

6.5.1 Detailed Description

Dataset management class.

This file is part of LSHBOX.

LSHBOX is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or(at your option) any later version.

LSHBOX is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with LSHBOX. If not, see http://www.gnu.org/licenses/.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

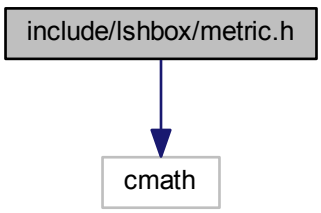
Definition in file matrix.h.

6.6 include/lshbox/metric.h File Reference

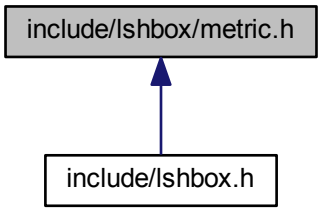
Common distance measures.

```
#include <cmath>
```

Include dependency graph for metric.h:



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



Classes

- class Metric< DATATYPE >

Namespaces

- lshbox

Macros

- #define L1_DIST 1
- #define L2_DIST 2

Functions

- template<typename DATATYPE >
  DATATYPE sqr (const DATATYPE &x)

6.6.1 Detailed Description

Common distance measures.

This file is part of LSHBOX.

LSHBOX is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or(at your option) any later version.

LSHBOX is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with LSHBOX. If not, see http://www.gnu.org/licenses/.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

Definition in file metric.h.

6.6.2 Macro Definition Documentation

6.6.2.1 #define L1_DIST 1

Definition at line 33 of file metric.h.

6.6.2.2 #define L2_DIST 2

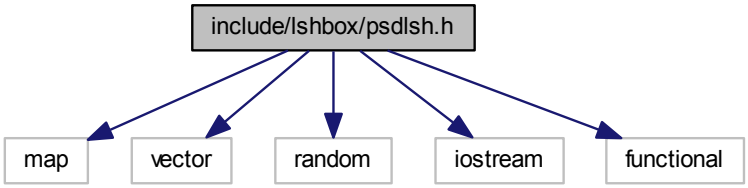
Definition at line 34 of file metric.h.

6.7 include/lshbox/psdlsh.h File Reference

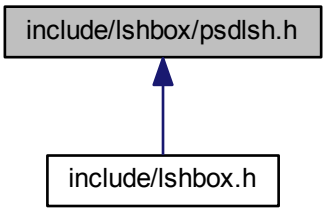
Locality-Sensitive Hashing Scheme Based on p-Stable Distributions.

```
#include <map>
#include <vector>
#include <random>
#include <iostream>
#include <functional>
```

Include dependency graph for psdlsh.h:



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



Classes

- class psdLsh< DATATYPE >
- struct psdLsh< DATATYPE >::Parameter

Namespaces

- lshbox

Macros

- #define CAUCHY 1
- #define GAUSSIAN 2

6.7.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on p-Stable Distributions.

This file is part of LSHBOX.

LSHBOX is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or(at your option) any later version.

LSHBOX is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with LSHBOX. If not, see http://www.gnu.org/licenses/.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

Definition in file psdlsh.h.

6.7.2 Macro Definition Documentation

6.7.2.1 #define CAUCHY 1

Definition at line 51 of file psdlsh.h.

6.7.2.2 #define GAUSSIAN 2

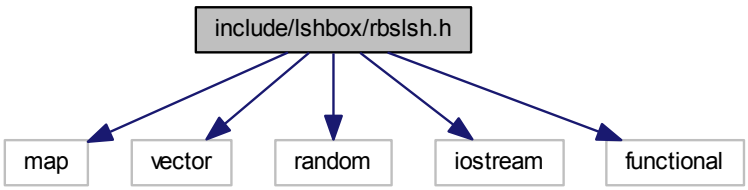
Definition at line 52 of file psdlsh.h.

6.8 include/lshbox/rbslsh.h File Reference

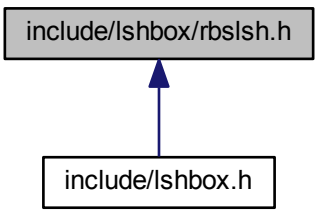
Locality-Sensitive Hashing Scheme Based on Random Bits Sampling.

```
#include <map>
#include <vector>
#include <random>
#include <iostream>
#include <functional>
```

Include dependency graph for rbslsh.h:



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



Classes

- class rbsLsh
- struct rbsLsh::Parameter

Namespaces

- lshbox

6.8.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on Random Bits Sampling.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

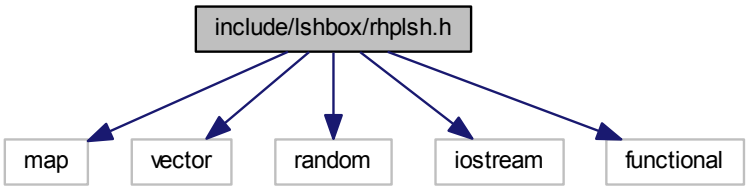
Definition in file rbslsh.h.

6.9 include/lshbox/rhplsh.h File Reference

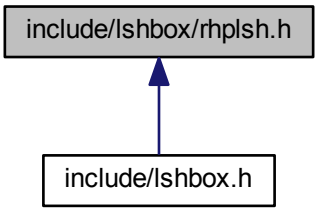
Locality-Sensitive Hashing Scheme Based on Random Hyperplane.

```
#include <map>
#include <vector>
#include <random>
#include <iostream>
#include <functional>
```

Include dependency graph for rhplsh.h:



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



Classes

- class rhpLsh< DATATYPE >
- struct rhpLsh< DATATYPE >::Parameter

Namespaces

- lshbox

6.9.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on Random Hyperplane.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

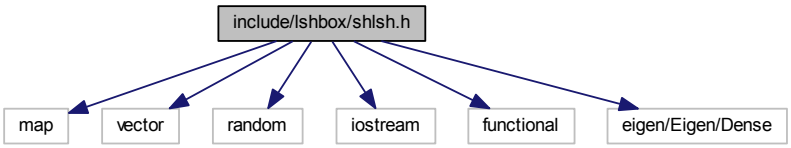
Definition in file rhplsh.h.

6.10 include/lshbox/shlsh.h File Reference

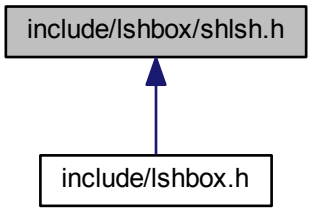
Locality-Sensitive Hashing Scheme Based on Spectral Hashing.

```
#include <map>
#include <vector>
#include <random>
#include <iostream>
#include <functional>
#include <eigen/Eigen/Dense>
```

Include dependency graph for shlsh.h:



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



Classes

- class shLsh< DATATYPE >
- struct shLsh< DATATYPE >::Parameter

Namespaces

- lshbox

6.10.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on Spectral Hashing.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

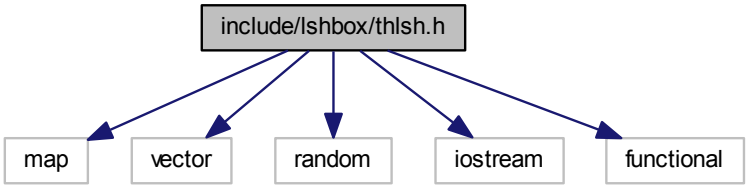
Definition in file shlsh.h.

6.11 include/lshbox/thlsh.h File Reference

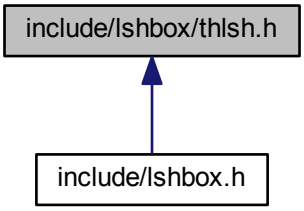
Locality-Sensitive Hashing Scheme Based on Thresholding.

```
#include <map>
#include <vector>
#include <random>
#include <iostream>
#include <functional>
```

Include dependency graph for thlsh.h:



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



Classes

- class thLsh< DATATYPE >
- struct thLsh< DATATYPE >::Parameter

Namespaces

- lshbox

6.11.1 Detailed Description

Locality-Sensitive Hashing Scheme Based on Thresholding.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

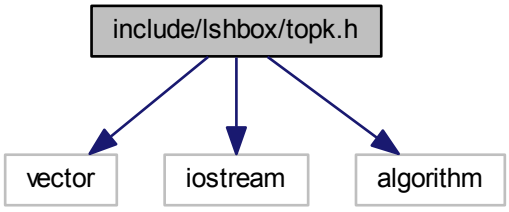
Definition in file thlsh.h.

6.12 include/lshbox/topk.h File Reference

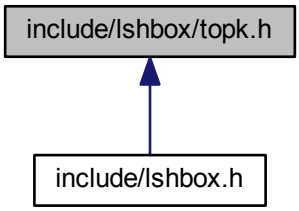
Top-K data structures.

```
#include <vector>
#include <iostream>
#include <algorithm>
```

Include dependency graph for topk.h:



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



Classes

- class Topk
- class Scanner< ACCESSOR >

Namespaces

- lshbox

6.12.1 Detailed Description

Top-K data structures.

Version

0.1

Author

Gefu Tang & Zhifeng Xiao

Date

2014.6.30

Definition in file topk.h.