MatrixLibrary

1.0

Generated by Doxygen 1.9.8

Chapter 1

MatrixLibrary

1.1 Introduction

MatrixLibrary is a matrix library with basic linear algebra functionality including eigenvalue decompositions for symmetric matrices.

1.2 Features

- Matrix arithmetic
- Householder tridiagonalization
- QL eigenvalue solver
- Symmetry Check
- Transpose
- HDF5 output

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Class Documentation

5.1 EigsymResult Struct Reference

Final eigenvalues and eigenvectors of matrix.

`#include <matrix.h>`

Collaboration diagram for EigsymResult:

Public Attributes

- vec eigenvalues
- Matrix eigenvectors

5.1.1 Detailed Description

Final eigenvalues and eigenvectors of matrix.

This code is adapted from Numerical Recipes

5.1.2 Member Data Documentation

5.1.2.1 eigenvalues

```
vec EigsymResult::eigenvalues
```

5.1.2.2 eigenvectors

```
Matrix EigsymResult::eigenvectors
```

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

- include/matrix.h

5.2 InvalidMatrixSize Class Reference

Exception thrown this error if the matrix size is invalid for specified operation.

`#include <custom_exception.hpp>`

Inheritance diagram for InvalidMatrixSize:

Collaboration diagram for InvalidMatrixSize:

Public Member Functions

- InvalidMatrixSize (const std::string &what_arg)
- InvalidMatrixSize (const std::string &what_arg)

5.2.1 Detailed Description

Exception thrown this error if the matrix size is invalid for specified operation.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 InvalidMatrixSize() [1/2]

```
InvalidMatrixSize::InvalidMatrixSize (
            const std::string & what_arg )  [inline]
```

5.2.2.2 InvalidMatrixSize() [2/2]

```
InvalidMatrixSize::InvalidMatrixSize (
            const std::string & what_arg )  [inline]
```

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

- include/custom_exception.hpp
- src/custom_exception.hpp

5.3 Matrix Class Reference

Matrix class with basic linear algebra operations.

`#include <matrix.h>`

Public Member Functions

- Matrix (int rows, int cols)

  *Construct a matrix of size (row * cols)*
- Matrix ()
- Matrix (const vec &values, int rows, int cols)

  *Construct a matrix of size (row * cols) with data.*
- Matrix (vec &&values, int rows, int cols)
- int get_num_rows () const
- int get_num_cols () const
- int get_size () const
- const vec & get_data () const
- double & operator() (int x, int y)

  Access a matrix element at (row, col)
- const double & operator() (int x, int y) const

  Const reference to the matrix element at (row, col)
- bool operator== (const Matrix &other) const
- Matrix operator+ (const Matrix &other) const
- Matrix operator- (const Matrix &other) const
- Matrix operator* (const Matrix &other) const
- Matrix operator* (double s) const
- bool is_symmetric (double tol) const
- Matrix transpose () const
- TridiagonalResult householder_tridiagonalize (bool yesvecs=true) const
- QLEigenResult QL (vec d, vec e) const
- EigsymResult eigsym () const

Static Public Member Functions

- static Matrix Ones (int rows, int cols)

  Fill matrix of specified size with 1's.
- static Matrix Zeros (int rows, int cols)

  Fill matrix of specified size with 0's.
- static Matrix Random (int rows, int cols)

  Fill matrix of specific size with random doubles from [0.0, 1.0)
- static Matrix Identity (int n)

  *Create square identity matrix with dimensions n*n.*
- static Matrix diagmat (const vec &vector)
- static Matrix diagmat (const Matrix &mat)
- static void save_hdf5 (const Matrix &data, const std::string &filename, const std::string &dataset_name)
- static void save_hdf5 (const vec &data, const std::string &filename, const std::string &dataset_name)

Private Attributes

- int num_rows
- int num_cols
- int size
- vec matrix

5.3.1 Detailed Description

Matrix class with basic linear algebra operations.

The matrix class stores elements in contiguous memory to improve cache locality This class supports multiplication, addition, subtraction, symmetry checks, transposing matrix, saving to HDF5 file and eigenvalue decompositions for symmetric matrices.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Matrix() [1/4]

```
Matrix::Matrix (
            int rows,
            int cols )
```

Construct a matrix of size (row ∗ cols)

5.3.2.2 Matrix() [2/4]

```
Matrix::Matrix ( )
```

5.3.2.3 Matrix() [3/4]

```
Matrix::Matrix (
            const vec & values,
            int rows,
            int cols )
```

Construct a matrix of size (row ∗ cols) with data.

5.3.2.4 Matrix() [4/4]

```
Matrix::Matrix (
            vec && values,
            int rows,
            int cols )
```

5.3.3 Member Function Documentation

5.3.3.1 diagmat() [1/2]

```
Matrix Matrix::diagmat (
            const Matrix & mat )  [static]
```

5.3.3.2 diagmat() [2/2]

```
Matrix Matrix::diagmat (
            const vec & vector )  [static]
```

5.3.3.3 eigsym()

```
EigsymResult Matrix::eigsym ( ) const
```

5.3.3.4 get_data()

```
const vec & Matrix::get_data ( ) const
```

5.3.3.5 get_num_cols()

```
int Matrix::get_num_cols ( ) const
```

5.3.3.6 get_num_rows()

```
int Matrix::get_num_rows ( ) const
```

5.3.3.7 get_size()

```
int Matrix::get_size ( ) const
```

5.3.3.8 householder_tridiagonalize()

```
TridiagonalResult Matrix::householder_tridiagonalize (
            bool yesvecs = true ) const
```

5.3.3.9 Identity()

```
Matrix Matrix::Identity (
            int n )  [static]
```

Create square identity matrix with dimensions n∗n.

5.3.3.10 is_symmetric()

```
bool Matrix::is_symmetric (
            double tol = 1e-12 ) const
```

5.3.3.11 Ones()

```
Matrix Matrix::Ones (
            int rows,
            int cols )  [static]
```

Fill matrix of specified size with 1's.

5.3.3.12 operator()() [1/2]

```
double & Matrix::operator() (
            int x,
            int y )
```

Access a matrix element at (row, col)

Parameters

x	row index (0-based)
y	column index (0-based)

Returns

reference to the matrix element

Exceptions

OutofBounds	exception if the indices are invalid

5.3.3.13 operator()() [2/2]

```
const double & Matrix::operator() (
            int x,
            int y ) const
```

Const reference to the matrix element at (row, col)

5.3.3.14 operator*() [1/2]

```
Matrix Matrix::operator* (
            const Matrix & other ) const
```

5.3.3.15 operator*() [2/2]

```
Matrix Matrix::operator* (
            double s ) const
```

5.3.3.16 operator+()

```
Matrix Matrix::operator+ (
            const Matrix & other ) const
```

5.3.3.17 operator-()

```
Matrix Matrix::operator- (
            const Matrix & other ) const
```

5.3.3.18 operator==()

```
bool Matrix::operator== (
            const Matrix & other ) const
```

5.3.3.19 QL()

```
QLEigenResult Matrix::QL (
            vec d,
            vec e ) const
```

5.3.3.20 Random()

```
Matrix Matrix::Random (
            int rows,
            int cols )  [static]
```

Fill matrix of specific size with random doubles from [0.0, 1.0)

5.3.3.21 save_hdf5() [1/2]

```
void Matrix::save_hdf5 (
            const Matrix & data,
            const std::string & filename,
            const std::string & dataset_name )  [static]
```

5.3.3.22 save_hdf5() [2/2]

```
void Matrix::save_hdf5 (
            const vec & data,
            const std::string & filename,
            const std::string & dataset_name )  [static]
```

5.3.3.23 transpose()

```
Matrix Matrix::transpose ( ) const
```

5.3.3.24 Zeros()

```
Matrix Matrix::Zeros (
            int rows,
            int cols )  [static]
```

Fill matrix of specified size with 0's.

5.3.4 Member Data Documentation

5.3.4.1 matrix

```
vec Matrix::matrix  [private]
```

5.3.4.2 num_cols

```
int Matrix::num_cols  [private]
```

5.3.4.3 num_rows

```
int Matrix::num_rows  [private]
```

5.3.4.4 size

```
int Matrix::size  [private]
```

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

- include/matrix.h
- src/matrix_basic_func.cpp
- src/matrix_eigendecomp.cpp
- src/matrix_operators.cpp
- src/matrix_utilfuncs.cpp

5.4 QLEigenResult Struct Reference

Result of QL Algorithm.

```
#include <matrix.h>
```

Collaboration diagram for QLEigenResult:

Public Attributes

- vec eigenvalues
- Matrix Q_ql

5.4.1 Detailed Description

Result of QL Algorithm.

This code is adapted from Numerical Recipes

5.4.2 Member Data Documentation

5.4.2.1 eigenvalues

```
vec QLEigenResult::eigenvalues
```

5.4.2.2 Q_ql

```
Matrix QLEigenResult::Q_ql
```

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

- include/matrix.h

5.5 TridiagonalResult Struct Reference

Result of Householder Tridiagonalization.

```
#include <matrix.h>
```

Collaboration diagram for TridiagonalResult:

Public Attributes

- vec d
- vec e
- Matrix Q_house

5.5.1 Detailed Description

Result of Householder Tridiagonalization.

This code is adapted from Numerical Recipes

5.5.2 Member Data Documentation

5.5.2.1 d

```
vec TridiagonalResult::d
```

5.5.2.2 e

```
vec TridiagonalResult::e
```

5.5.2.3 Q_house

```
Matrix TridiagonalResult::Q_house
```

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

- include/matrix.h

Chapter 6

File Documentation

6.1 include/custom_exception.hpp File Reference

`#include <stdexcept>`
Include dependency graph for custom_exception.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class InvalidMatrixSize

  Exception thrown this error if the matrix size is invalid for specified operation.

6.2 custom_exception.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <stdexcept>
00003
00008 class InvalidMatrixSize : public std::domain_error {
00009 public:
00010     InvalidMatrixSize(const std::string& what_arg) : std::domain_error(what_arg)
00011     {}
00012 };
```

6.3 src/custom_exception.hpp File Reference

`#include <stdexcept>`
Include dependency graph for custom_exception.hpp:

Classes

- class InvalidMatrixSize

  Exception thrown this error if the matrix size is invalid for specified operation.

6.4 custom_exception.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <stdexcept>
00003
00004 class InvalidMatrixSize : public std::domain_error {
00005 public:
00006     InvalidMatrixSize(const std::string& what_arg) : std::domain_error(what_arg)
00007     {}
00008 };
```

6.5 include/helper_func.hpp File Reference

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

Functions

- double SIGN (double a, double b)

  Return a value equivalent to $|a| * sign(b)$
- double pythag (double a, double b)

  Computes sqrt(a^2 + b^2) without destructive overflow or underflow.

6.5.1 Function Documentation

6.5.1.1 pythag()

```
double pythag (
            double a,
            double b )
```

Computes sqrt(a^2 + b^2) without destructive overflow or underflow.

Helper function for eigenvalue and eigenvector calculation. Adapted from Numerical Recipes.

6.5.1.2 SIGN()

```
double SIGN (
            double a,
            double b )
```

Return a value equivalent to $|a| * sign(b)$

Helper function for eigenvalue and eigenvector calculation. Adapted from Numerical Recipes.

6.6 helper_func.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002
00009 double SIGN(double a, double b);
00010
00017 double pythag(double a, double b);
```

6.7 include/matrix.h File Reference

```
#include <iostream>
#include <vector>
#include <limits>
#include "custom_exception.hpp"
#include "helper_func.hpp"
#include <cmath>
#include <iomanip>
#include <stdexcept>
#include <random>
```

Include dependency graph for matrix.h: This graph shows which files directly or indirectly include this file:

Classes

- class Matrix

  Matrix class with basic linear algebra operations.
- struct TridiagonalResult

  Result of Householder Tridiagonalization.
- struct QLEigenResult

  Result of QL Algorithm.
- struct EigsymResult

  Final eigenvalues and eigenvectors of matrix.

Typedefs

- typedef std::vector< double > vec

  Alias for vector of doubles, used for matrix storage.

Functions

- std::ostream & operator<< (std::ostream &out, const Matrix &M)

6.7.1 Typedef Documentation

6.7.1.1 vec

```
typedef std::vector<double> vec
```

Alias for vector of doubles, used for matrix storage.

6.7.2 Function Documentation

6.7.2.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & out,
            const Matrix & M )
```

6.8 matrix.h

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include <iostream>
00004 #include <vector>
00005 #include <limits>
00006 #include "custom_exception.hpp"
00007 #include "helper_func.hpp" // numerical recipes helper functions
00008 #include <cmath>
00009 #include <iomanip>
00010 #include <stdexcept>
00011 #include <random>
00012
00033 typedef std::vector<double> vec;
00034
00035 // Forward declarations of global result types
00036 struct TridiagonalResult;
00037 struct EigsymResult;
00038 struct QLEigenResult;
00039
00049 class Matrix {
00050 private:
00051   int num_rows;
00052   int num_cols;
00053   int size;
00054   vec matrix;
00055
00056 public:
00057   // === Constructors ===
00061   Matrix(int rows, int cols);
00062   Matrix();
00066   Matrix(const vec& values, int rows, int cols);
00067   Matrix(vec&& values, int rows, int cols);
00068
00069   // === Factory Methods ===
00073   static Matrix Ones(int rows, int cols);
00074
00078   static Matrix Zeros(int rows, int cols);
00079
00083   static Matrix Random(int rows, int cols);
00084
00088   static Matrix Identity(int n);
00089
00090   // === Accessors ===
00091   int get_num_rows() const;
00092   int get_num_cols() const;
00093   int get_size() const;
00094   const vec& get_data() const;
00095
00096   // === Operators ===
00106   double& operator()(int x, int y);
00107
00111   const double& operator()(int x, int y) const;
00112
00113   bool operator==(const Matrix& other) const;
00114   Matrix operator+(const Matrix& other) const;
00115   Matrix operator-(const Matrix& other) const;
00116   Matrix operator*(const Matrix& other) const;
00117   Matrix operator*(double s) const;
00118
00119   // === Linear Algebra Functionality ===
00120   static Matrix diagmat(const vec& vector);
00121   static Matrix diagmat(const Matrix& mat);
00122   bool is_symmetric(double tol) const;
00123   Matrix transpose() const;
00124   TridiagonalResult householder_tridiagonalize(bool yesvecs = true) const;
00125   QLEigenResult QL(vec d, vec e) const;
00126   EigsymResult eigsym() const;
00127
00128   // === Saving to HDF5 File ===
00129   static void save_hdf5(const Matrix& data, const std::string& filename, const std::string&
      dataset_name);
00130   static void save_hdf5(const vec& data, const std::string& filename, const std::string&
      dataset_name);
00131
00132 };
00133
00139 struct TridiagonalResult {
00140     vec d; // diagonal elements
00141     vec e; // off-diagonal elements
00142     Matrix Q_house; // orthogonal transformation matrix
00143 };
00144
```

```
00150 struct QLEigenResult {
00151     vec eigenvalues;
00152     Matrix Q_ql;
00153 };
00154
00160 struct EigsymResult {
00161     vec eigenvalues;
00162     Matrix eigenvectors;
00163 };
00164
00165 // === Printing Functionality ===
00166 std::ostream& operator«(std::ostream& out, const Matrix & M);
```

6.9 include/print_vec.hpp File Reference

```
#include <iostream>
#include <vector>
#include <iomanip>
```

Include dependency graph for print_vec.hpp:

Functions

- std::ostream & operator<< (std::ostream &out, const std::vector< double > &v)

  Overload print operator for vector.

6.9.1 Function Documentation

6.9.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & out,
            const std::vector< double > & v )  [inline]
```

Overload print operator for vector.

6.10 print_vec.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <iostream>
00003 #include <vector>
00004 #include <iomanip>
00005
00009 inline std::ostream& operator«(std::ostream& out, const std::vector<double>& v) {
00010   out « std::fixed « std::setprecision(4);
00011   for (double x : v) {
00012     out « x « "\n";
00013   }
00014   return out;
00015 }
```

6.11 src/benchmark.hpp File Reference

```
#include <vector>
#include <chrono>
```

Include dependency graph for benchmark.hpp:

6.12 benchmark.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002
00003 #include <vector>
00004 #include <chrono>
00005
00006 template <typename T>
00007 long benchmark_time(T func, int iterations=25) {
00008   std::vector<double> vec_diff;
00009   vec_diff.reserve(iterations);
00010   for (int i = 0; i < iterations; ++i) {
00011     auto start = std::chrono::steady_clock::now();
00012     func();
00013     auto end = std::chrono::steady_clock::now();
00014
00015     // Convert to nanoseconds
00016     double diff = std::chrono::duration<double, std::nano>(end - start).count();
00017     vec_diff.push_back(diff);
00018     }
00019
00020   // get average
00021   double avg = 0.0;
00022   for (double val : vec_diff) {
00023       avg += val;
00024   }
00025   // return time diff in microseconds
00026   return avg / vec_diff.size();
00027 }
00028
00029 // need to implement L2 norm for this benchmarking
00030 template <typename T>
00031 long benchmark_acc(T func1, T func2) {
00032   auto diff = func1() - func2();
00033   return diff;
00034 }
```

6.13 src/helper_func.cpp File Reference

```
#include "helper_func.hpp"
#include <cmath>
```

Include dependency graph for helper_func.cpp: This graph shows which files directly or indirectly include this file:

Functions

- double SIGN (double a, double b)

  Return a value equivalent to $|a| * sign(b)$
- double pythag (const double a, const double b)

  Computes sqrt(a^2 + b^2) without destructive overflow or underflow.

6.13.1 Function Documentation

6.13.1.1 pythag()

```
double pythag (
            double a,
            double b )
```

Computes sqrt(a^2 + b^2) without destructive overflow or underflow.

Helper function for eigenvalue and eigenvector calculation. Adapted from Numerical Recipes.

6.13.1.2 SIGN()

```
double SIGN (
             double a,
             double b )
```

Return a value equivalent to |a| ∗ sign(b)

Helper function for eigenvalue and eigenvector calculation. Adapted from Numerical Recipes.

6.14 helper_func.cpp

Go to the documentation of this file.

```
00001 #include "helper_func.hpp"
00002 #include <cmath>
00003
00004 double SIGN(double a, double b) {
00005   if ( b >= 0.0 ) {
00006     return std::fabs(a);
00007   } else {
00008     return -std::fabs(a);
00009   }
00010 }
00011
00012 double pythag(const double a, const double b) {
00013   // computes sqrt(a^2 + b^2) without destructive underflow or overflow
00014   double absa = std::abs(a);
00015   double absb = std::abs(b);
00016   if (absa > absb) {
00017     return absa * sqrt(1.0 + (absb/absa)*(absb/absa));
00018   } else {
00019     return absb * sqrt(1.0 + (absa/absb)*(absa/absb));
00020   }
00021 }
```

6.15 src/matrix_basic_func.cpp File Reference

```
#include "matrix.h"
#include <algorithm>
```

Include dependency graph for matrix_basic_func.cpp:

6.16 src/matrix_eigendecomp.cpp File Reference

```
#include "matrix.h"
#include "helper_func.cpp"
#include <cmath>
#include <stdexcept>
#include <limits>
#include <algorithm>
#include <numeric>
```

Include dependency graph for matrix_eigendecomp.cpp:

6.17 src/matrix_operators.cpp File Reference

```
#include "matrix.h"
#include <algorithm>
#include <iomanip>
#include <cmath>
#include <limits>
```

Include dependency graph for matrix_operators.cpp:

Functions

- std::ostream & operator<< (std::ostream &out, const Matrix &M)

6.17.1 Function Documentation

6.17.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & out,
            const Matrix & M )
```

6.18 src/matrix_utilfuncs.cpp File Reference

```
#include "matrix.h"
#include <cmath>
#include <highfive/H5File.hpp>
```

Include dependency graph for matrix_utilfuncs.cpp:

Index