Project 2 - FYS4150

Generated by Doxygen 1.14.0

Chapter 1

Topic Index

1.1 Topics

Here is a list of all topics 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

Topic Documentation

4.1 Standalone Functions

Various stand-alone functions appearing in the project.

Functions

- Args parse_args (int argc, char ∗argv[ ])

  Function for parsing command-line arguments.
- void jacobi_rotate (arma::mat &A, arma::mat &R, int k, int l)

  Performs a single Jacobi rotation.
- void jacobi_eigensolver (const arma::mat &A, double eps, arma::vec &eigenvalues, arma::mat &eigenvectors, const int maxiter, int &iterations, bool &converged)

  Computes the eigenvalues and eigenvectors of a symmetric matrix using Jacobi's rotation method.
- void analytic_solution (arma::vec &eigenvalues, arma::mat &eigenvectors, double a, double d, int N)

  Gives the analytic solution for the eigenvalues and eigenvectors of $A\vec{v} = \lambda\vec{v}$, where A is a tridiagonal matrix(a,d,a).
- arma::mat create_tridiagonal (int n, double a, double d, double e)
- double max_offdiag_symmetric (const arma::mat &A, int &k, int &l)

  Finds the greatest off-diagonal element in the upper triangular part (in absolute value) of a given symmetric matrix.

4.1.1 Detailed Description

Various stand-alone functions appearing in the project.

4.1.2 Function Documentation

4.1.2.1 analytic_solution()

```
void analytic_solution (
            arma::vec & eigenvalues,
            arma::mat & eigenvectors,
            double a,
            double d,
            int N)
```

Gives the analytic solution for the eigenvalues and eigenvectors of $A\vec{v} = \lambda\vec{v}$, where A is a tridiagonal matrix(a,d,a).

Parameters

eigenvalues	Armadillo vector for eigenvalues.
eigenvectors	Armadillo vector for eigenvectors.
a	Upper and lower diagonal of matrix.
d	Diagonal of matrix.
N	Size of matrix.

4.1.2.2 create_tridiagonal()

```
arma::mat create_tridiagonal (
            int n,
            double a,
            double d,
            double e)
```

Creates a symmetric tridiagonal matrix of size n x n with constant diagonal elements d, sub-diagonal elements a, and super-diagonal elements e.

Parameters

n	size of the matrix
a	sub-diagonal elements
d	diagonal elements
e	super-diagonal elements

Returns

the tridiagonal matrix

4.1.2.3 jacobi_eigensolver()

```
void jacobi_eigensolver (
            const arma::mat & A,
            double eps,
            arma::vec & eigenvalues,
            arma::mat & eigenvectors,
            const int maxiter,
            int & iterations,
            bool & converged)
```

Computes the eigenvalues and eigenvectors of a symmetric matrix using Jacobi's rotation method.

Parameters

A	The symmetric matrix to be diagonalized.
eps	The convergence tolerance for the off-diagonal elements.
eigenvalues	Vector to store the computed eigenvalues (output).
eigenvectors	Matrix to store the computed eigenvectors (output).
maxiter	The maximum number of iterations allowed.
iterations	The number of iterations performed (output).
converged	Boolean flag indicating whether the method converged (output).

4.1.2.4 jacobi_rotate()

```
void jacobi_rotate (
            arma::mat & A,
            arma::mat & R,
            int k,
            int l)
```

Performs a single Jacobi rotation.

Parameters

A	The symmetric matrix to be diagonalized.
R	The matrix of eigenvectors.
k	The row index of the maximal off-diagonal element.
l	The column index of the maximal off-diagonal element.

4.1.2.5 max_offdiag_symmetric()

```
double max_offdiag_symmetric (
            const arma::mat & A,
            int & k,
            int & l)
```

Finds the greatest off-diagonal element in the upper triangular part (in absolute value) of a given symmetric matrix.

Parameters

A	Symmetric matrix.
k	Row index.
l	Column index.

Returns

Greatest off-diagonal element in the upper triangular part (in absolute value) of A.

4.1.2.6 parse_args()

```
Args parse_args (
            int argc,
            char * argv[])
```

Function for parsing command-line arguments.

Parameters

argc	Number of command-line arguments.
argv	Array of command-line argument strings.

Returns

Args struct containing the parsed values.

Chapter 5

Class Documentation

5.1 Args Struct Reference

Struct to hold command-line arguments.

`#include <arg_parser.hpp>`

Public Attributes

- std::string outfile = "build/outfile.csv"
- bool run_tests = false
- bool run_problem_5 = false
- bool run_problem_6 = false
- double tol = 1e-14
- int n_steps = 10
- int N_max = 100
- int maxiter = 10000

5.1.1 Detailed Description

Struct to hold command-line arguments.

5.1.2 Member Data Documentation

5.1.2.1 maxiter

```
int Args::maxiter = 10000
```

5.1.2.2 N_max

```
int Args::N_max = 100
```

5.1.2.3 n_steps

```
int Args::n_steps = 10
```

5.1.2.4 outfile

```
std::string Args::outfile = "build/outfile.csv"
```

5.1.2.5 run_problem_5

```
bool Args::run_problem_5 = false
```

5.1.2.6 run_problem_6

```
bool Args::run_problem_6 = false
```

5.1.2.7 run_tests

```
bool Args::run_tests = false
```

5.1.2.8 tol

```
double Args::tol = 1e-14
```

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

- arg_parser.hpp

5.2 TriDag Class Reference

Class representing a tridiagonal matrix, i.e. an matrix of the form.

```
#include <triDag.hpp>
```

Public Member Functions

- TriDag (double h, int N)

  Creates a new tridiagonal object.
- void compute_eigenvalues ()

  Computes the eigenvalues of the tridiagonal matrix using arma::eig_sym.
- void print (int max=25)

  Prints the elements of the tridiagonal matrix.

Public Attributes

- arma::mat A
- arma::vec eigenvalues
- arma::mat eigenvectors

5.2.1 Detailed Description

Class representing a tridiagonal matrix, i.e. an matrix of the form.

$$A = \begin{pmatrix} a & c & 0 & \cdots & 0 \\ b & a & c & \cdots & 0 \\ 0 & b & a & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & c \\ 0 & \cdots & 0 & b & a \end{pmatrix},$$

where $b = c = -1/h^2$ and $a = 2/h^2$.

Parameters

A	Armadillo matrix.
eigenvalues	Armadillo vector for the eigenvalues of the matrix A.
eigenvectors	Armadillo vector for the eigenvectors of the matrix A.

See also

TriDag

5.2.2 Constructor & Destructor Documentation

5.2.2.1 TriDag()

```
TriDag::TriDag (
            double h,
            int N)
```

Creates a new tridiagonal object.

Parameters

h	Stepsize, defining the diagonals of the tridiagonal matrix, see general description.
N	Size of matrix.

5.2.3 Member Function Documentation

5.2.3.1 compute_eigenvalues()

```
void TriDag::compute_eigenvalues ()
```

Computes the eigenvalues of the tridiagonal matrix using arma::eig_sym.

5.2.3.2 print()

```
void TriDag::print (
            int max = 25)
```

Prints the elements of the tridiagonal matrix.

Parameters

max	Only prints matrix if the size of the matrix is less than or equal to max.

5.2.4 Member Data Documentation

5.2.4.1 A

```
arma::mat TriDag::A
```

5.2.4.2 eigenvalues

```
arma::vec TriDag::eigenvalues
```

5.2.4.3 eigenvectors

```
arma::mat TriDag::eigenvectors
```

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

- triDag.hpp

Chapter 6

File Documentation

6.1 arg_parser.hpp File Reference

```
#include <string>
#include <iostream>
```

Classes

- struct Args

  Struct to hold command-line arguments.

Functions

- Args parse_args (int argc, char *argv[])

  Function for parsing command-line arguments.

6.2 arg_parser.hpp

Go to the documentation of this file.

```
00001 #ifndef ARG_PARSER_CPP
00002 #define ARG_PARSER_CPP
00003
00004 #include <string>
00005 #include <iostream>
00006
00010 struct Args
00011 {
00012     std::string outfile = "build/outfile.csv";
00013     bool run_tests = false;
00014     bool run_problem_5 = false;
00015     bool run_problem_6 = false;
00016     double tol = 1e-14;
00017     int n_steps = 10;
00018     int N_max = 100;
00019     int maxiter = 10000;
00020 };
00021
00025
00032 Args parse_args(int argc, char *argv[]);
00033
00034 #endif
00035
```

6.3 jacobi_eigensolver.hpp File Reference

```
#include <armadillo>
#include "utils.hpp"
```

Functions

- void jacobi_rotate (arma::mat &A, arma::mat &R, int k, int l)

  Performs a single Jacobi rotation.
- void jacobi_eigensolver (const arma::mat &A, double eps, arma::vec &eigenvalues, arma::mat &eigenvectors, const int maxiter, int &iterations, bool &converged)

  Computes the eigenvalues and eigenvectors of a symmetric matrix using Jacobi's rotation method.

6.4 jacobi_eigensolver.hpp

Go to the documentation of this file.

```
00001 #ifndef JACOBI_EIGENSOLVER_HPP
00002 #define JACOBI_EIGENSOLVER_HPP
00003
00004 #include <armadillo>
00005 #include "utils.hpp"
00006
00010
00019 void jacobi_rotate(arma::mat &A, arma::mat &R, int k, int l);
00020
00032 void jacobi_eigensolver(const arma::mat &A, double eps, arma::vec &eigenvalues, arma::mat
      &eigenvectors, const int maxiter, int &iterations, bool &converged);
00033
00034 #endif
00035
```

6.5 triDag.hpp File Reference

```
#include <armadillo>
```

Classes

- class TriDag

  Class representing a tridiagonal matrix, i.e. an matrix of the form.

Functions

- void analytic_solution (arma::vec &eigenvalues, arma::mat &eigenvectors, double a, double d, int N)

  Gives the analytic solution for the eigenvalues and eigenvectors of $A\vec{v} = \lambda\vec{v}$, where A is a tridiagonal matrix(a,d,a).

6.6 triDag.hpp

Go to the documentation of this file.

```
00001 #include <armadillo>
00002
00003
00021 class TriDag{
00022 private:
00023     double a_fill;
00024     double d_fill;
00025     int N;
00026
00027     void create_matrix();  // Creates tridiagonal matrix
00028
00029 public:
00030     arma::mat A;
00031     arma::vec eigenvalues;
00032     arma::mat eigenvectors;
00033
00040     TriDag(double h, int N);
00041
00042
00047     void compute_eigenvalues();
00048
00054     void print(int max=25);
00055 };
00056
00070 void analytic_solution(arma::vec &eigenvalues, arma::mat &eigenvectors, double a, double d, int N);
```

6.7 utils.hpp File Reference

```
#include <armadillo>
```

Functions

- arma::mat create_tridiagonal (int n, double a, double d, double e)
- double max_offdiag_symmetric (const arma::mat &A, int &k, int &l)

  Finds the greatest off-diagonal element in the upper triangular part (in absolute value) of a given symmetric matrix.

6.8 utils.hpp

Go to the documentation of this file.

```
00001 #include <armadillo>
00002
00003
00008
00009
00010
00014
00024 arma::mat create_tridiagonal(int n, double a, double d, double e);
00025
00026
00035 double max_offdiag_symmetric(const arma::mat &A, int &k, int &l);
00036
00037
```

Index