# Systems of linear equations

Wygenerowano przez Doxygen 1.8.17

# Rozdział 1

# Indeks hierarchiczny

## 1.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

# Rozdział 2

# Indeks klas

## 2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

# Rozdział 3

# Indeks plików

## 3.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

# Rozdział 4

# Dokumentacja klas

## 4.1 Dokumentacja klasy Matrix

< Dynamic allocated Matrix

`#include <Matrix.h>`

### Metody publiczne

- Matrix ()
- Matrix (int a, int b)
- Matrix (Matrix &A)
- ~Matrix ()
- double * operator[ ] (int a) const
- Matrix & operator= (const Matrix &A)
- Matrix & operator= (double **a)
- Matrix operator+ (const Matrix &A) const
- Matrix & operator+= (Matrix &A)
- Matrix operator* (const Matrix &A) const
- Matrix operator* (const double a) const
- Matrix operator*= (double a)
- void swap_x (int x1, int x2)
- void swap_y (int y1, int y2)
- void add_d_x (int x)
- void set_d (int y, int x)
- void add_d_y (int y)
- const void print ()
- const void print (int precision, int width)
- const void print (int precision)
- const int size_y () const
- const int size_x () const
- const int capacity_x () const
- const int capacity_y () const

### Przyjaciele

- Matrix operator* (double a, Matrix A)
- std::ostream & operator<< (std::ostream &os, Matrix &A)

### 4.1.1 Opis szczegółowy

< Dynamic allocated Matrix

### 4.1.2 Dokumentacja konstruktora i destruktora

#### 4.1.2.1 Matrix() [1/3]

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

Konstruktor domyslny macierzy Tworzy obiekt ale nie rezerwuje na niego zadnej pamieci

#### 4.1.2.2 Matrix() [2/3]

```
Matrix::Matrix (
            int a,
            int b )
```

Konstruktor - tworzy macierz o rozmiarach AxB gdzie A to liczba wierszy, a B to liczba kolumn

**Parametry**

| | | |
|---|---|---|
| in | *a* | liczba wierszy macierzy |
| in | *b* | liczba kolumn macierzy |

#### 4.1.2.3 Matrix() [3/3]

```
Matrix::Matrix (
            Matrix & A )
```

Konstruktor kopiujacy

**Parametry**

| | | |
|---|---|---|
| in | *A* | macierz za pomoca ktorej nalezy zainicjowac dana macierz |

#### 4.1.2.4 ~Matrix()

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

Destruktor macierzy

### 4.1.3 Dokumentacja funkcji składowych

#### 4.1.3.1 add_d_x()

```
void Matrix::add_d_x (
            int x )
```

Dodaje x kolumn do macierzy

**Parametry**

| | | |
|---|---|---|
| in | *x* | Liczba kolumn ktore chcemy dodac |

#### 4.1.3.2 add_d_y()

```
void Matrix::add_d_y (
            int y )
```

Dodaje y wierszy do macierzy

**Parametry**

| | | |
|---|---|---|
| in | *y* | Liczba wierszy ktore chcemy dodac |

#### 4.1.3.3 capacity_x()

```
const int Matrix::capacity_x ( ) const
```

Zwraca wartosc x pojemnosci macierzy

**Zwraca**

pojemnosc x macierzy

#### 4.1.3.4 capacity_y()

```
const int Matrix::capacity_y ( ) const
```

Zwraca wartosc y pojemnosci macierzy

**Zwraca**

pojemnosc y macierzy

#### 4.1.3.5 operator∗() [1/2]

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

Operator mnozenia macierzy przez skalar

**Parametry**

| | | |
|---|---|---|
| in | *a* | skalar przez ktory ma byc pomnozona macierz |

**Zwraca**

macierz iloczynu

#### 4.1.3.6 operator∗() [2/2]

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

Operator mnozenia macierzy

**Parametry**

| | | |
|---|---|---|
| in | *A* | prawostronny czynnik iloczynu macierzy |

**Zwraca**

macierz iloczynu

#### 4.1.3.7 operator∗=()

```
Matrix Matrix::operator*= (
            double a )
```

Operator mnozenia macierzy pezez skalar i przypisania

**Parametry**

| | | |
|---|---|---|
| in | *a* | skalar przez ktory ma byc pomnozona macierz |

**Zwraca**

macierz iloczynu

#### 4.1.3.8 operator+()

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

Operator dodawania macierzy

**Parametry**

| | | |
|---|---|---|
| in | *skladnik* | sumy |

**Zwraca**

macierz sumy

#### 4.1.3.9 operator+=()

```
Matrix & Matrix::operator+= (
            Matrix & A )
```

Operator dodawania i przypisania macierzy

**Parametry**

| | | |
|---|---|---|
| in | *A* | skladnik sumy |

**Zwraca**

∗this

#### 4.1.3.10 operator=() [1/2]

```
Matrix & Matrix::operator= (
            const Matrix & A )
```

Operator przepisania macierzy

**Parametry**

| | | |
|---|---|---|
| in | *A* | macierz ktora chcemy przepisac |

**Zwraca**

*this

#### 4.1.3.11 operator=() [2/2]

```
Matrix & Matrix::operator= (
            double ** a )
```

Operator przepisania tablicy do macierzy. Zakladam, ze tablica ma rozmiar taki jak aktualnie ma macierz

**Parametry**

| | | |
|---|---|---|
| in | *a* | tablica zmiennych typu double, nalezy uwazac na rozmiar tablicy i macierzy |

**Zwraca**

*this

#### 4.1.3.12 operator[]()

```
double * Matrix::operator[] (
            int a ) const
```

Operator pozwalajacy na dostep do konkretnych komorek macierzy. Jesli macierz ma jedna wiersz lub kolumne to nalezy odwolac sie do niej - np A[0][n]

**Parametry**

| | | |
|---|---|---|
| in | *a* | indekswiersza do ktorego chcemy sie odwolac |

#### 4.1.3.13 print() [1/3]

```
const void Matrix::print ( )
```

Wyswietla macierz

#### 4.1.3.14 print() [2/3]

```
const void Matrix::print (
            int precision )
```

Wyswietla macierz

**Parametry**

| | | |
|---|---|---|
| in | *precision* | ustawia precyzje wyswietlania liczb w macierzy |

#### 4.1.3.15 print() [3/3]

```
const void Matrix::print (
            int precision,
            int width )
```

Wyswietla macierz

**Parametry**

| | | |
|---|---|---|
| in | *precision* | ustawia precyzje wyswietlania liczb w macierzy |
| in | *width* | ustawia szerokosc wyswietlenia pojedyj komorki macierzy |

#### 4.1.3.16 set_d()

```
void Matrix::set_d (
            int y,
            int x )
```

Ustawia rozmiar macierzy na y na x

**Parametry**

| | | |
|---|---|---|
| in | *y* | Liczba wierszy ktore chcemy posiadac po operacji |
| in | *x* | Liczba kolumn ktore chcemy posiadac po operacji |

#### 4.1.3.17 size_x()

```
const int Matrix::size_x ( ) const
```

Zwraca wartosc x rozmiaru macierzy

**Zwraca**

rozmiar x macierzy

#### 4.1.3.18 size_y()

```
const int Matrix::size_y ( ) const
```

Zwraca wartosc y rozmiaru macierzy

**Zwraca**

rozmiar y macierzy

#### 4.1.3.19 swap_x()

```
void Matrix::swap_x (
            int x1,
            int x2 )
```

Zamienia ze soba kolumny macierzy

**Parametry**

| | | |
|---|---|---|
| in | *x1* | indeks kolumny ktora ma byc zamieniona |
| in | *x2* | indeks kolumny ktora ma byc zamieniona |

#### 4.1.3.20 swap_y()

```
void Matrix::swap_y (
            int y1,
            int y2 )
```

Zamienia ze soba kolumny macierzy

**Parametry**

| | | |
|---|---|---|
| in | *y1* | indeks wiersza ktory ma byc zamieniony |
| in | *y2* | indeks wiersza ktory ma byc zamieniony |

### 4.1.4 Dokumentacja przyjaciół i funkcji związanych

#### 4.1.4.1 operator*

```
Matrix operator* (
            double a,
            Matrix A )  [friend]
```

Funkcja zaprzyjazniona - operator mnozenia macierzy przez skalar

**Parametry**

| | | |
|---|---|---|
| in | *a* | skalar przez ktory ma byc pomnozona macierz |

**Zwraca**

macierz iloczynu

#### 4.1.4.2 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            Matrix & A )  [friend]
```

Zaprzyjazniona funkcja operator strumieniowy

**Parametry**

| | | |
|---|---|---|
| in | *os* | strumietorego chcemy przeslac macierz |
| in | *A* | Macierz ktora przesylamy do strumienia |

Dokumentacja dla tej klasy została wygenerowana z plików:

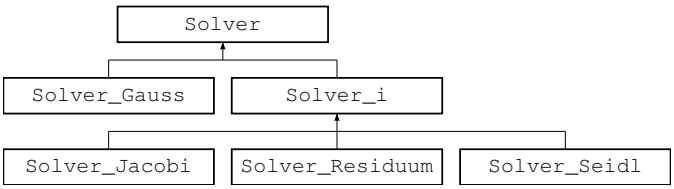
- C:/Users/Piotrek/source/repos/Projekt/Matrix.h
- C:/Users/Piotrek/source/repos/Projekt/Matrix.cpp

## 4.2 Dokumentacja klasy Solver

< Abstrakcyjna klasa do rozwiazywania ukladow rowna

```
#include <Solver.h>
```

Diagram dziedziczenia dla Solver




### Metody publiczne

- virtual bool solv (Matrix A)=0
- Matrix **results** ()

### Atrybuty chronione

- Matrix wyniki

  *Macierz wynikow.*

### 4.2.1 Opis szczegółowy

< Abstrakcyjna klasa do rozwiazywania ukladow rowna

#### 4.2.1.1 solv()

```
virtual bool Solver::solv (
            Matrix A )  [pure virtual]
```

wirtualna metoda rozwiazywania rowna1]Parametry `in` *A* macierz dolaczona Nx(N+1) gdzie lewa czesc NxN jest macierza wspolczynnikow, a ostatni wiersz jest wektorem wynikow rownania w postaci macierzowej

---

Implementowany w Solver_i, Solver_Residuum, Solver_Seidl, Solver_Gauss i Solver_Jacobi.

Dokumentacja dla tej klasy została wygenerowana z plików:

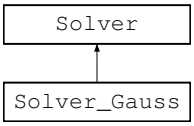
- C:/Users/Piotrek/source/repos/Projekt/Solver.h
- C:/Users/Piotrek/source/repos/Projekt/Solver.cpp

## 4.3 Dokumentacja klasy Solver_Gauss

<Odpowiada za rozwiazywanie ukladow rownaoda Gaussa

`#include <Solver_Gauss.h>`

Diagram dziedziczenia dla Solver_Gauss



### Metody publiczne

- bool solv (Matrix A)

### Dodatkowe Dziedziczone Składowe

### 4.3.1 Opis szczegółowy

<Odpowiada za rozwiazywanie ukladow rownaoda Gaussa

### 4.3.2 Dokumentacja funkcji składowych

#### 4.3.2.1 solv()

```
bool Solver_Gauss::solv (
            Matrix A )  [virtual]
```

Implementacja metody rozwiazywania rownaoda Gaussa

**Parametry**

| | | |
|---|---|---|
| in | *A* | macierz dolaczona Nx(N+1) gdzie lewa czesc NxN jest macierza wspolczynnikow, a ostatni wiersz jest wektorem wynikow rownania w postaci macierzowej |

Implementuje Solver.

Dokumentacja dla tej klasy została wygenerowana z plików:

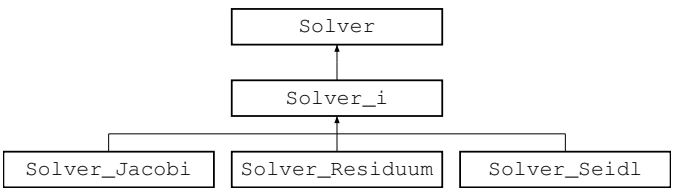
- C:/Users/Piotrek/source/repos/Projekt/Solver_Gauss.h
- C:/Users/Piotrek/source/repos/Projekt/Solver_Gauss.cpp

## 4.4 Dokumentacja klasy Solver_i

< Abstrakcyjna klasa do iteracyjnego rozwiazywania ukladow rowna

`#include <Solver.h>`

Diagram dziedziczenia dla Solver_i



### Metody publiczne

- virtual bool solv (Matrix A)=0
- void set_precision (double precision)

  *Ustawia precyzje generowania rozwiaznia.*

### Metody chronione

- bool norm1 (Matrix &A)
- bool norm2 (Matrix &A)
- bool norm3 (Matrix &A)

### Atrybuty chronione

- double precision = 0.001

  *Precyzja generowania rozwiazania (warunek stopu)*

## 4.4.1 Opis szczegółowy

< Abstrakcyjna klasa do iteracyjnego rozwiazywania ukladow rowna

#### 4.4.1.1 norm1()

```
bool Solver_i::norm1 (
            Matrix & A )  [protected]
```

1. Norma zbieznosci metody iteracyjnej

#### 4.4.1.2 norm2()

```
bool Solver_i::norm2 (
            Matrix & A )  [protected]
```

1. Norma zbieznosci metody iteracyjnej

#### 4.4.1.3 norm3()

```
bool Solver_i::norm3 (
            Matrix & A )  [protected]
```

1. Norma zbieznosci metody iteracyjnej

#### 4.4.1.4 solv()

```
virtual bool Solver_i::solv (
            Matrix A )  [pure virtual]
```

wirtualna metoda rozwiazywania rowna1]Parametry `in` *A* macierz dolaczona Nx(N+1) gdzie lewa czesc NxN jest macierza wspolczynnikow, a ostatni wiersz jest wektorem wynikow rownania w postaci macierzowej

---

Implementuje Solver.

Implementowany w Solver_Residuum, Solver_Seidl i Solver_Jacobi.

Dokumentacja dla tej klasy została wygenerowana z plików:

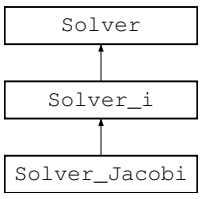
- C:/Users/Piotrek/source/repos/Projekt/Solver.h
- C:/Users/Piotrek/source/repos/Projekt/Solver.cpp

## 4.5 Dokumentacja klasy Solver_Jacobi

<Odpowiada za rozwiazywanie ukladow rownaoda Jacobiego

`#include <Solver_Jacobi.h>`

Diagram dziedziczenia dla Solver_Jacobi




### Metody publiczne

- bool solv (Matrix A)

### Dodatkowe Dziedziczone Składowe

### 4.5.1 Opis szczegółowy

<Odpowiada za rozwiazywanie ukladow rownaoda Jacobiego

### 4.5.2 Dokumentacja funkcji składowych

#### 4.5.2.1 solv()

```
bool Solver_Jacobi::solv (
            Matrix A )  [virtual]
```

Implementacja metody rozwiazywania rownaoda Jacobiego

**Parametry**

| | | |
|---|---|---|
| in | *A* | macierz dolaczona Nx(N+1) gdzie lewa czesc NxN jest macierza wspolczynnikow, a ostatni wiersz jest wektorem wynikow rownania w postaci macierzowej |

Implementuje Solver_i.

Dokumentacja dla tej klasy została wygenerowana z plików:

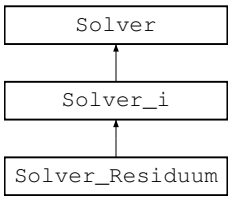
- C:/Users/Piotrek/source/repos/Projekt/Solver_Jacobi.h
- C:/Users/Piotrek/source/repos/Projekt/Solver_Jacobi.cpp

## 4.6 Dokumentacja klasy Solver_Residuum

<Odpowiada za rozwiazywanie ukladow rownaoda minimalizacji residuow

`#include <Solver_Residuum.h>`

Diagram dziedziczenia dla Solver_Residuum




### Metody publiczne

- bool solv (Matrix A)

### Dodatkowe Dziedziczone Składowe

### 4.6.1 Opis szczegółowy

<Odpowiada za rozwiazywanie ukladow rownaoda minimalizacji residuow

### 4.6.2 Dokumentacja funkcji składowych

#### 4.6.2.1 solv()

```
bool Solver_Residuum::solv (
            Matrix A )  [virtual]
```

Implementacja metody rozwiazywania rownaoda residuow

**Parametry**

| | | |
|---|---|---|
| in | *A* | macierz dolaczona Nx(N+1) gdzie lewa czesc NxN jest macierza wspolczynnikow, a ostatni wiersz jest wektorem wynikow rownania w postaci macierzowej |

Implementuje Solver_i.

Dokumentacja dla tej klasy została wygenerowana z plików:

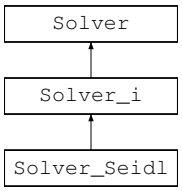
- C:/Users/Piotrek/source/repos/Projekt/Solver_Residuum.h
- C:/Users/Piotrek/source/repos/Projekt/Solver_Residuum.cpp

## 4.7 Dokumentacja klasy Solver_Seidl

<Odpowiada za rozwiazywanie ukladow rownaoda Gaussa-Seidla

`#include <Solver_Seidl.h>`

Diagram dziedziczenia dla Solver_Seidl




### Metody publiczne

- bool solv (Matrix A)

### Dodatkowe Dziedziczone Składowe

### 4.7.1 Opis szczegółowy

<Odpowiada za rozwiazywanie ukladow rownaoda Gaussa-Seidla

### 4.7.2 Dokumentacja funkcji składowych

#### 4.7.2.1 solv()

```
bool Solver_Seidl::solv (
            Matrix A )  [virtual]
```

Implementacja metody rozwiazywania rownaoda Gaussa-Seidla

**Parametry**

| | | |
|---|---|---|
| in | *A* | macierz dolaczona Nx(N+1) gdzie lewa czesc NxN jest macierza wspolczynnikow, a ostatni wiersz jest wektorem wynikow rownania w postaci macierzowej |

Implementuje Solver_i.

Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Piotrek/source/repos/Projekt/Solver_Seidl.h
- C:/Users/Piotrek/source/repos/Projekt/Solver_Seidl.cpp

# Rozdział 5

# Dokumentacja plików

## 5.1 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Matrix.cpp

```
#include "Matrix.h"
#include <iostream>
#include <algorithm>
#include <iomanip>
```

### Funkcje

- Matrix operator∗ (double a, Matrix A)
- std::ostream & operator<< (std::ostream &os, Matrix &A)

### 5.1.1 Dokumentacja funkcji

#### 5.1.1.1 operator∗()

```
Matrix operator* (
            double a,
            Matrix A )
```

Funkcja zaprzyjazniona - operator mnozenia macierzy przez skalar

**Parametry**

| | | |
|---|---|---|
| in | *a* | skalar przez ktory ma byc pomnozona macierz |

**Zwraca**

macierz iloczynu

#### 5.1.1.2 operator<<()

```
std::ostream& operator<< (
            std::ostream & os,
            Matrix & A )
```

Zaprzyjazniona funkcja operator strumieniowy

**Parametry**

| | | |
|---|---|---|
| in | *os* | strumietorego chcemy przeslac macierz |
| in | *A* | Macierz ktora przesylamy do strumienia |

## 5.2 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Matrix.h

```
#include <iostream>
```

### Komponenty

- class Matrix

  < *Dynamic allocated Matrix*

## 5.3 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Projekt.cpp

```
#include <iostream>
#include <iomanip>
#include "SLE.h"
```

### Funkcje

- int **main** ()

## 5.4 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/SLE.h

```
#include "Matrix.h"
#include "Solver.h"
#include "Solver_Gauss.h"
#include "Solver_Jacobi.h"
#include "Solver_Residuum.h"
#include "Solver_Seidl.h"
```

## 5.5 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver.cpp

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

## 5.6 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver.h

```
#include "Matrix.h"
```

### Komponenty

- class Solver

  < *Abstrakcyjna klasa do rozwiazywania ukladow rownaclass Solver_i*
  < Abstrakcyjna klasa do iteracyjnego rozwiazywania ukladow rownaDoxyCompactItemize

## 5.7 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Gauss.cpp

```
#include "Solver_Gauss.h"
```

## 5.8 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Gauss.h

```
#include "Solver.h"
#include "Matrix.h"
```

### Komponenty

- class Solver_Gauss
  *<Odpowiada za rozwiazywanie ukladow rownaoda Gaussa*

## 5.9 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Jacobi.cpp

```
#include "Solver_Jacobi.h"
#include <algorithm>
#include <iostream>
```

## 5.10 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Jacobi.h

```
#include "Solver.h"
#include "Matrix.h"
```

### Komponenty

- class Solver_Jacobi
  *<Odpowiada za rozwiazywanie ukladow rownaoda Jacobiego*

## 5.11 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Residuum.cpp

```
#include "Solver_Residuum.h"
#include <algorithm>
#include <iostream>
```

## 5.12 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Residuum.h

```
#include "Solver.h"
#include "Matrix.h"
```

### Komponenty

- class Solver_Residuum
  *<Odpowiada za rozwiazywanie ukladow rownaoda minimalizacji residuow*

## 5.13 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Seidl.cpp

```
#include "Solver_Seidl.h"
#include <algorithm>
#include <iostream>
```

## 5.14 Dokumentacja pliku C:/Users/Piotrek/source/repos/Projekt/Solver_Seidl.h

```
#include "Solver.h"
#include "Matrix.h"
```

## Komponenty

- class Solver_Seidl
  *<Odpowiada za rozwiazywanie ukladow rownaoda Gaussa-Seidla*

# Indeks