Университет ИТМО

Факультет ФПИ и КТ

### Отчёт

### по лабораторной работе 1

### «Решение системы линейных алгебраических уравнений СЛАУ»

Вариант 11

Метод Гаусса с выбором главного элемента по столбцам

Студен:

Ляо Ихун

Гр.P3211

Преподаватель:  
Малышева Татьяна Алексеевна

### Программа:

<https://github.com/LiaoYihong-1/CM_lab1>

1.Основные операции метода Гаусса

**/\*\***

**\* Basic method Gaussian**

**\* @param triangle**

**\* @return**

**\*/**

**protected double[] solver(Matrix triangle) {**

**int j = 0;**

**for (int i = triangle.getRows() - 1; i >= 0; i--) {**

**j++;**

**for (int k = 0; k < j; k++) {**

**if (j == 1) {**

**int targetColumn = triangle.getColumns() - j - 1;**

**triangle.setElement(i, targetColumn, triangle.getElement(i, triangle.getColumns() - 1) / triangle.getElement(i, targetColumn));**

**} else {**

**double result = 0;**

**double b = triangle.getElement(i,triangle.getColumns()-1);**

**for(int z = 0 ; z < j; z++){**

**if(j-z == 1){**

**double coefficient = triangle.getElement(i,triangle.getColumns()-2-z);**

**result = b/coefficient;**

**triangle.setElement(triangle.getRows()-1, triangle.getColumns()-1-j, result);**

**}else {**

**double coefficient = triangle.getElement(i, triangle.getColumns() - 2 - z);**

**double number = triangle.getElement(triangle.getRows() - 1, triangle.getColumns() - 2 - z);**

**b = b - coefficient \* number;**

**}**

**}**

**}**

**}**

**}**

**return triangle.getLine(triangle.getRows() - 1);**

**}**

2.Выбор главного элемента и получить треугольник

/\*\*

\* One of the most important function. He performs the process of choosing main elements and getting triangle

\* @return Matrix

\*/

public Matrix preparedMainMatrixs(){

for(int i = 0; i<matrixProduct.getColumns()-1; i++){

//choose main elements

for(int j=i+1;j< matrixProduct.getRows();j++){

if(matrixProduct.getElement(i,i)<matrixProduct.getElement(j,i)){

matrixProduct.exchangeLine(i,j);

k++;

}

}

//get triangle

double[] origin = matrixProduct.getLine(i);

for(int j=i+1;j< matrixProduct.getRows();j++){

if(matrixProduct.getElement(i,i)!=0.0) {

matrixProduct.mulLine(-matrixProduct.getElement(j, i) / matrixProduct.getElement(i, i), i);

matrixProduct.addLine(i, j);

matrixProduct.setLine(origin, i);

}

}

}

triangle.init(matrixProduct);

return this.matrixProduct;

}

### Цель:

При помощи метода Гаусса с выбором главного элемента, пишим решаюшую СЛАУ программу. Понимать что хотя при прямых методов погрешность не избежна.

### Описание программы

1. В программе три команды:
   1. File – которая позволяет ввод матрицу через файл
   2. Input – которая позволяет пользователи сам вводить матрицу через клавитуру
   3. Exit – уход

Все команды можно описать в любом виде(например file, File, FILE, fiLE)

1. Ключевые классы:
   1. Matrix:
      1. Основые операции матрицы
   2. MatrixBuilder:
      1. Помагает получить матрицу в правильном виде за обработку метода Гаусса
   3. Model:
      1. Ввод и вывод матрицы
      2. Решает уравлении при методе Гаусса
   4. Caculator:
      1. Получение
2. Ключевые методы:
   1. solver() в классе model. Основный процесс метод Гаусса
   2. preparedMainMatrixs() в классе MatrixBuilder. Логика для того чтобы получить матрица в виде треугольника после выбора главного элементов.

### Примеры

文本

描述已自动生成文本

描述已自动生成

图形用户界面, 文本, 应用程序, 聊天或短信

描述已自动生成文本

描述已自动生成

### Вывод

При примеров видим что хотя при помощи прямых методов, наша функция иногда не может получать точные результаты(но погрешность очень маленькая). Это потому что при рачета там всегда есть окружение, переполнение и так далее(компьтер не может представить число как 1/3). То что мы можем делать, это улучшить алготрим чтобы снижать погрешность.

Ещё там когда сохраняю результат, проблема пространства возникает. Мы не знаем ранг матрицы будет сколько. Конечно мы можем создать 20 переменных в начале, но когда n большое это расход пространства.Поэтому я сохраняю все результаты в последней строке, где больше не важно после первого шага метода Гаусса, чтобы экономить память