**Факультет ПИиКТ**

**Дисциплина: вычислительная математика**

Лабораторная работа №1

*“Решение системы линейных алгебраических уравнений СЛАУ”*

Выполнил: Иванов Илья

Преподаватель: Перл О.В.

Группа: Р3212

Вариант: метод Гаусса

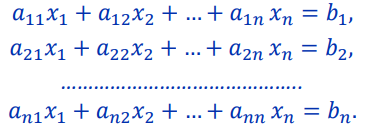
Санкт-Петербург, 2020 г.

**Описание метода**

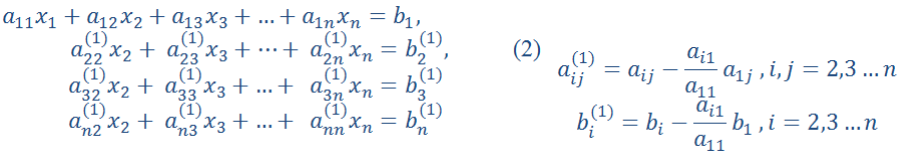
Метод Гаусса – точный метод решения СЛАУ, алгоритм которого реализует последовательное исключение неизвестных из уравнений системы. Данный метод разделяет решение СЛАУ на два этапа (прямой и обратный ход).

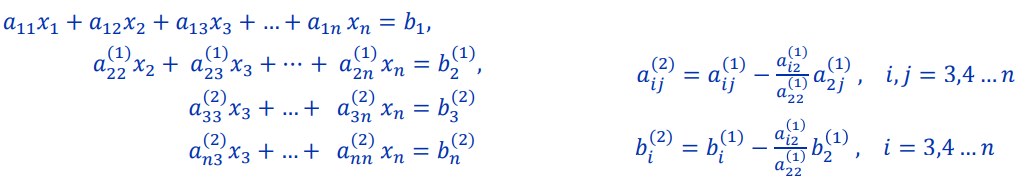
Первый этап состоит из последовательного исключения неизвестных из каждого уравнения. Коэффициенты рассматриваемого уравнения делятся на *ведущий элемент*, то есть элемент, строка и столбец которого совпадают (). Пользуясь полученным уравнением, можно исключить из системы соответствующую неизвестную. Сперва с помощью первого уравнения вычитается первая неизвестная, затем с помощью второй вторая и тд. В результате каждого исключения система уравнений становится меньше, а матрица системы постепенно приведется к треугольному виду. В конце этапа останется лишь один неизвестный член

Рассмотрим общий пример:

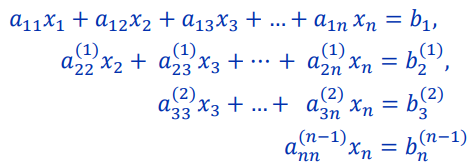


Сперва исключается , затем и тд. до . Элементы, стоящие под ведущим элементом, станут равны нулю.





.........................................................................



Второй этап состоит из нахождения неизвестных начиная с последнего . Далее пользуясь значением найденных на предыдущих этапах переменных, находим текущий. Так продолжается до нахождения

**Блок-схема численного метода**

Изображение выглядит как текст, карта

Автоматически созданное описание

**Листинг численного метода**

**GaussMethod.java**

**public class** GaussMethod {  
 **private double**[][] **matrixA**;  
 **private double**[] **matrixB**;  
 **private int size**;  
 **private** SolvingMessageService **ms**;  
 **private double detA** = 1.0;  
 **private double**[] **result**;  
 **private double**[] **residual**;  
  
 **public** GaussMethod(**double**[][] matrixA, **double**[] matrixB, **int** size){  
 **this**.**matrixA** = matrixA;  
 **this**.**matrixB** = matrixB;  
 **this**.**size** = size;  
 **result** = **new double**[size];  
 **residual** = **new double**[size];  
 **ms** = **new** SolvingMessageService();  
 }  
  
 **public void** solve(){  
 **try** {  
 *//строки которые вычитаем* **for** (**int** i = 0; i < **size** - 1; i++)  
 {  
 checkLeader(i);  
  
 *//строки из которох вычитаем* **for** (**int** j = i + 1; j < **size**; j++){  
 **double** c = **matrixA**[j][i] / **matrixA**[i][i];  
 **matrixA**[j][i] = 0;  
  
 **for** (**int** k = i + 1; k < **size**; k++){  
 **matrixA**[j][k] -= c \* **matrixA**[i][k];  
 }  
  
 **matrixB**[j] -= c \* **matrixB**[i];  
 }}  
  
 **if**(**matrixA**[**size** - 1][**size** - 1] == 0.0 && **matrixB**[**size** - 1] != 0.0) **throw new** CannotSolveException();  
 checkLeader(**size** - 1);  
  
 **for**(**int** i = **size** - 1; i >= 0; i--){  
 **double** s = 0.0;  
  
 **for**(**int** j = i + 1; j < **size**; j++) s += **matrixA**[i][j] \* **result**[j];  
  
 **result**[i] = (**matrixB**[i] - s) / **matrixA**[i][i];  
 }  
  
 calculateDeterminant();  
 calculateResidual();  
  
 **ms**.showMatrixMessage(**matrixA**, **matrixB**, **true**);  
 **ms**.showDeterminant(**detA**);  
 **ms**.showUnknowns(**result**);  
 **ms**.showResidual(**residual**);  
  
 } **catch** (CannotSolveException e){  
 **ms**.noSolutionMessage();  
 } **catch** (EndlessSolutionException e) {  
 **ms**.infinitySolutionMessage();  
 }  
 }  
  
 **public void** checkLeader(**int** it) **throws** EndlessSolutionException {  
 **boolean** status = **false**;  
  
 **if** (**matrixA**[it][it] == 0){  
 **for**(**int** i = it + 1; i < **size**; i++){  
 **if**(**matrixA**[i][it] != 0){  
 swapLine(i, it);  
 status = **true**;  
 **detA** \*= -1;  
 **break**;  
 }  
 }  
  
 **if**(!status) **throw new** EndlessSolutionException();  
 }  
 }  
  
 **public void** swapLine(**int** line1, **int** line2){  
 **double** temp;  
 **for**(**int** j = 0; j < **size**; j++){  
 temp = **matrixA**[line1][j];  
 **matrixA**[line1][j] = **matrixA**[line2][j];  
 **matrixA**[line2][j] = temp;  
 }  
  
 temp = **matrixB**[line1];  
 **matrixB**[line1] = **matrixB**[line2];  
 **matrixB**[line2] = temp;  
 }  
  
 **public void** calculateDeterminant(){  
 **for**(**int** i = 0; i < **size**; i++)  
 **detA** \*= **matrixA**[i][i];  
 }  
  
 **public void** calculateResidual(){  
 **for**(**int** i = 0; i < **size**; i++){  
 **for**(**int** j = 0; j < **size**; j++){  
 **residual**[i] += **matrixA**[i][j] \* **result**[j];  
 }  
  
 **residual**[i] = **matrixB**[i] - **residual**[i];  
 }  
 }  
}

**Пример работы программы и ее результаты**

**Пример №1**

Добро пожаловать!

Данная программа решает СЛАУ методом Гаусса.

Выберите способ ввода данных (консоль {к}| файл {ф}| случайная генерация чисел {г})

г

Ввод данных осуществляется через: генерация случайных чисел

Введите размерность матрицы(количество уравнений {n<=20, n - целое число}).

6

..............................................

Размерность: 6

..............................................

Расширенная матрица элементов будет выглядеть следующим образом:

| 87,361 99,048 51,042 46,848 21,482 5,182 | 42,708|

| 31,230 51,686 48,238 5,817 95,203 13,542 | 80,391|

| 52,387 98,684 13,413 50,877 55,871 11,845 | 6,527|

| 34,726 36,920 39,984 92,678 25,381 2,217 | 88,510|

| 29,757 84,895 58,142 47,742 40,459 80,691 | 72,685|

| 51,526 40,701 76,190 38,797 91,114 49,626 | 84,180|

..............................................

Расширенная матрица элементов после преобразования:

| 87,361 99,048 51,042 46,848 21,482 5,182 | 42,708|

| 0,000 16,278 29,991 -10,930 87,524 11,690 | 65,124|

| 0,000 0,000 -89,582 49,165 -168,257 -19,476 | -176,265|

| 0,000 0,000 0,000 85,698 -15,452 -3,346 | 33,702|

| 0,000 0,000 0,000 0,000 -134,805 55,256 | -55,723|

| 0,000 0,000 0,000 0,000 0,000 57,565 | -55,587|

..............................................

Определитель матрицы: 84717735863,689

..............................................

Таблица неизвестных:

x1 = -1,6156

x2 = 0,5266

x3 = 2,3415

x4 = 0,3587

x5 = 0,0175

x6 = -0,9656

..............................................

Таблица невязок:

|2.8421709430404007E-14|

|1.4210854715202004E-14|

|0.0|

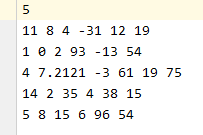
|0.0|

|0.0|

|0.0|

**Пример №2**

test.txt



Добро пожаловать!

Данная программа решает СЛАУ методом Гаусса.

Выберите способ ввода данных (консоль {к}| файл {ф}| случайная генерация чисел {г})

ф

Ввод данных осуществляется через: файл

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Данные файла должны соответствовать следующим требованиям:

\*Первая строка должна содержать количество уравнений {n<=20}.

#n - целое число.

#Строка не должна содержать букв. Только цифры от 0 до 9.

\*Далее следует n строк, содержащие элементы расширенной матрицы. (Каждая строка содержит n + 1 элементов)

#Каждый элемент разделен пробелом от другого пробелом.

#Запись не должна содержать букв.

#Целая часть должна отделяться от дробной точкой {.}!

Пример:

4

1 2 3 4 10

5 6 7 8 11

9 8 7 6 13

5 4 3 2 14

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Введите абсолютный путь до файла. Если файл находится в директории с проектом, то можно ввести название файла.

test.txt

Проверьте название файла и выполнение всех условий. Хотите продолжить? ({да} | {нет})

да

..............................................

Размерность: 5

..............................................

Расширенная матрица элементов будет выглядеть следующим образом:

| 11,000 8,000 4,000 -31,000 12,000 | 19,000|

| 1,000 0,000 2,000 93,000 -13,000 | 54,000|

| 4,000 7,212 -3,000 61,000 19,000 | 75,000|

| 14,000 2,000 35,000 4,000 38,000 | 15,000|

| 5,000 8,000 15,000 6,000 96,000 | 54,000|

..............................................

Расширенная матрица элементов после преобразования:

| 11,000 8,000 4,000 -31,000 12,000 | 19,000|

| 0,000 -0,727 1,636 95,818 -14,091 | 52,273|

| 0,000 0,000 5,227 639,194 -68,734 | 377,370|

| 0,000 0,000 0,000 -2440,740 332,467 | -1427,471|

| 0,000 0,000 0,000 0,000 6,379 | -4,545|

..............................................

Определитель матрицы: 651046,720

..............................................

Таблица неизвестных:

x1 = -6,9787

x2 = 13,3419

x3 = 3,1755

x4 = 0,4878

x5 = -0,7125

..............................................

Таблица невязок:

|0.0|

|0.0|

|0.0|

|0.0|

|0.0|

**Вывод:** необходимым и достаточным условием применимости метода является неравенство нулю всех «ведущих элементов». Отсюда следует, что на определенных этапах приходится менять местами уравнения и следить за изменением знака определителя. Одно из преимуществ метода Гаусса является точность решения, при условии, что коэффициенты уравнения даны точные. Метод является простым, по сравнению с выбором главного элемента (перестановка уравнений для нахождения максимального ведущего). Общее количество арифметических действий в методе Гаусса не превосходит . Поэтому он идеально подходит для систем с небольшим числом переменных.