**ФПИиКТ**

**Рабочий протокол и отчет по**

**домашней работе №1**

**Вариант №5**

**(Прямой метод Гаусса)**

Ибадуллаев Алибаба Эльбрус оглы

Группа: P3215

Преподаватель: Татьяна Алексеевна

Санкт-Петербург

2022г.

## Цель работы

Поработать с решением СЛАУ при помощи различных методов (прямых и итерационных)

## Задание лабораторной работы

Реализовать программу нахождения решения СЛАУ при помощи прямого метода Гаусса

* Вычисление определителя
* Вывод треугольной матрицы (включая преобразованный столбец В)
* Вывод вектора неизвестных: x1, x2, …, xn
* Вывод вектора невязок: r1, r, …, rn

## Описание метода, расчетные формулы

Основан на приведении матрицы системы к треугольному виду так, чтобы ниже

ее главной диагонали находились только нулевые элементы.

Прямой ход метода Гаусса состоит в последовательном исключении неизвестных

из уравнений системы. Сначала с помощью первого уравнения исключается x1 из

всех последующих уравнений системы. Затем с помощью второго уравнения

исключается x2 из третьего и всех последующих уравнений и т.д.

Этот процесс продолжается до тех пор, пока в левой части последнего (n-го)

уравнения не останется лишь один член с неизвестным xn , т. е. матрица системы

будет приведена к треугольному виду.

Обратный ход метода Гаусса состоит в последовательном вычислении искомых

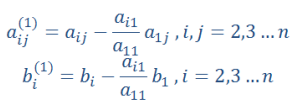
неизвестных: решая последнее уравнение, находим единственное в этом

уравнении неизвестное xn . Далее, используя это значение, из предыдущего

уравнения вычисляем xn−1 и т. д. Последним найдем x1 из первого уравнения.

Метод имеет много различных вычислительных схем, но в каждой из них

основным требованием является det A ≠ 0 .



## Листинг программы (по крайне мере, где реализован сам метод)

// straight stroke (process of exception an elements of matrix to get triangle-matrix)  
for(int i = 0; i < n - 1 ; i++){  
 while (arr[i][i] == 0 ){  
 if (stop == n - i) {  
 System.*err*.println("The error in swapping lines!");  
 System.*exit*(0);  
 }  
 *swapLines*(arr, b, i);  
 stop++;  
 swaps++;  
 }  
  
 for (int k = i+1; k < n; k++){  
 c = arr[k][i] / arr[i][i];  
 arr[k][i] = 0;  
 for (int j = i+1; j < n; j++){  
 arr[k][j] = arr[k][j] - c \* arr[i][j];  
 }  
 b[k] = b[k] - c \* b[i];  
 }  
 stop = 0;  
 System.*out*.println();  
 System.*out*.println("\t<<Iteration №" + (i+1)+">>");  
 io.showExtendedMatrix(arr, b);  
}  
  
  
// reverse stroke (process of getting an unknown variables)  
for (int i = n - 1 ; i >= 0 ; i--){  
 s = 0;  
 for (int j = i+1; j<n; j++){  
 if (Float.*isNaN*(x[j])) s=s+0;  
 else s = s + arr[i][j]\*x[j];  
 }  
 if (Float.*isInfinite*(s)) {  
 System.*err*.println("This system has no solution!\nBecause: "+"0!="+b[i+1]);  
 System.*exit*(-1);  
 }  
 x[i] = (b[i] - s)/arr[i][i] == -0 ? 0:(b[i] - s)/arr[i][i] ;  
}

## Примеры и результаты работы программы

<<INPUT MATRIX>>

A = [1.0, 2.0, 3.0]

[4.0, 5.0, 6.0]

[7.0, 8.0, 10.0]

B = [1.0, 4.0, 4.0]

<<RESULTS>>

<<TRIANGLE MATRIX>>

A = [1.0, 2.0, 3.0]

[0.0, -3.0, -6.0]

[0.0, 0.0, 1.0]

B = [1.0, 0.0, -3.0]

Det A = -3.0

X = [-2.0, 6.0, -3.0]

r = [0.0, 0.0, 0.0]

## Выводы

Я поработать с решением СЛАУ при помощи прямого метода Гаусса