Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы численного анализа

**Отчёт**

к лабораторной работе

на тему:

Решение систем линейных алгебраических уравнений (СЛАУ)

методом Гаусса и с помощью его модификаций

                               Выполнил студент группы 253505

Снежко Максим Андреевич

Проверил: Анисимов Владимир Яковлевич

Минск 2023

# **ЦЕЛИ**

* Изучить метод Гаусса и его модификации, составить алгоритм метода и программу его реализации, получить численное решение заданной СЛАУ
* Написать алгоритм решения СЛАУ указанными методами, применимый для организации вычислений на ЭВМ
* Составить программу решения СЛАУ по разработанному алгоритму
* Выполнить тестовые примеры и проверить правильность работы программы

**Введение**

Для эффективного решения систем линейных алгебраических уравнений (СЛАУ) в практических задачах часто требуются быстрые методы. Один из классических методов для этой цели - метод Гаусса и его вариации. Основная идея метода Гаусса заключается в поэтапном исключении переменных из системы, чтобы преобразовать ее в эквивалентную систему с верхней треугольной матрицей. Модификации метода Гаусса, включая выбор главного элемента, используются для уменьшения значений коэффициентов матрицы СЛАУ, при этом незначительно уступая в скорости вычислений. Это позволяет снизить погрешности в результате вычислений.

# **КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

Система линейных алгебраических уравнений (СЛАУ) - это набор уравнений, в которых все переменные входят в первой степени и коэффициенты при них являются константами. Общий вид СЛАУ можно записать следующим образом:



Здесь ***А*** и ***b*** заданы, требуется найти ***х***.

Методы решения систем линейных алгебраических уравнений (СЛАУ) можно разделить на две основные категории: прямые (или методы прямого подстановочного типа) и итерационные методы.

1 Прямые методы: Прямые методы применяются для точного решения СЛАУ и обеспечивают точное решение системы. Они основаны на последовательных математических операциях, таких как умножение, вычитание и деление, чтобы найти значения неизвестных.

2 Итерационные методы: Итерационные методы используются для приближенного решения СЛАУ. Они начинают с начального приближения и последовательно уточняют решение на каждой итерации.

Прямые методы обычно требуют более высокой вычислительной мощности и памяти, но они гарантируют точное решение. Наибольшее распространение среди прямых методов получили метод Гаусса и его модификации.

Метод Гаусса обладает рядом преимуществ по сравнению с другими методами:

1 Точное решение: Метод Гаусса гарантирует точное решение СЛАУ при условии, что система уравнений имеет решение и матрица коэффициентов невырожденная.

2 Универсальность: Метод Гаусса применим к разнообразным видам СЛАУ с различными коэффициентами и правыми частями. Он не зависит от специфических характеристик системы и применим к системам с произвольным числом переменных и уравнений.

3 Простота реализации: Алгоритм метода Гаусса относительно прост в понимании и реализации. Он состоит из двух основных этапов: прямого и обратного хода, что делает его доступным для программирования и вычислений вручную.

4 Стабильность: В большинстве случаев метод Гаусса стабилен и надежен. Он не подвержен численным неустойчивостям, которые могут возникнуть при использовании некоторых итерационных методов.

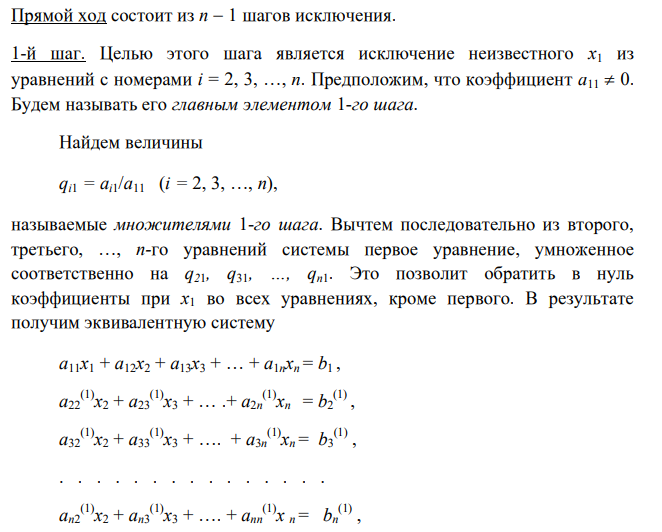
Метод Гаусса, используемый для решения систем линейных алгебраических уравнений (СЛАУ), включает два основных этапа: прямой и обратный ходы.

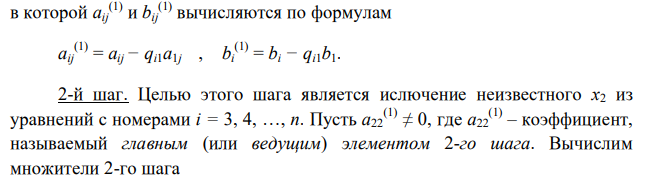
Прямой ход метода Гаусса заключается в преобразовании расширенной матрицы системы так, чтобы получить верхнюю треугольную матрицу с нулями под главной диагональю. Это достигается путем последовательного исключения переменных из уравнений.

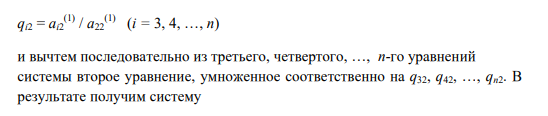
Обратный ход, который иногда называется методом Гаусса-Жордана, представляет собой этап, в котором матрица приводится к диагональному виду, и значения переменных вычисляются от последней к первой. Это также включает в себя последовательное исключение переменных, но на этот раз вверх от главной диагонали.

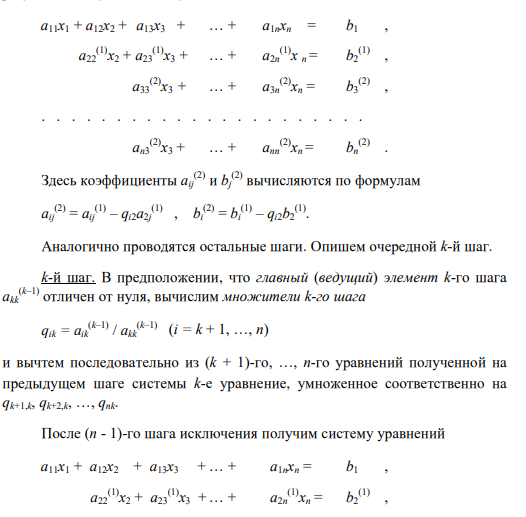
Таким образом, метод Гаусса и метод Гаусса-Жордана отличаются только последовательностью шагов исключения переменных. Оба метода являются прямыми методами решения СЛАУ, и оба обеспечивают точное решение системы, при условии, что система имеет единственное решение и матрица коэффициентов невырожденная.

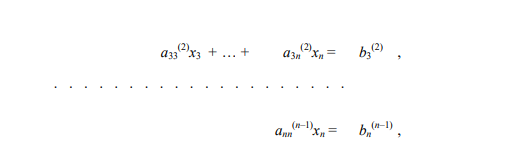
Метод Гаусса идеально подходит для решения систем содержащих больше трех линейных уравнений, для решения систем уравнений, которые не являются квадратными (чего не скажешь про метод Крамера и матричный метод). То есть метод Гаусса - наиболее универсальный метод для нахождения решения любой системы линейных уравнений, он работает в случае, когда система имеет бесконечно много решений или несовместна.



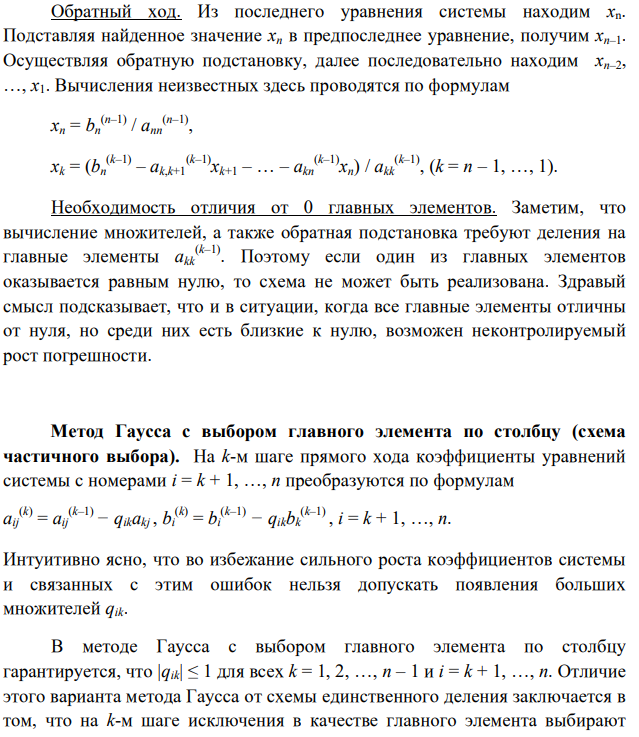


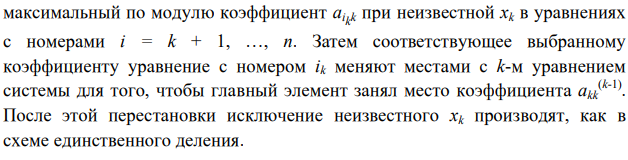


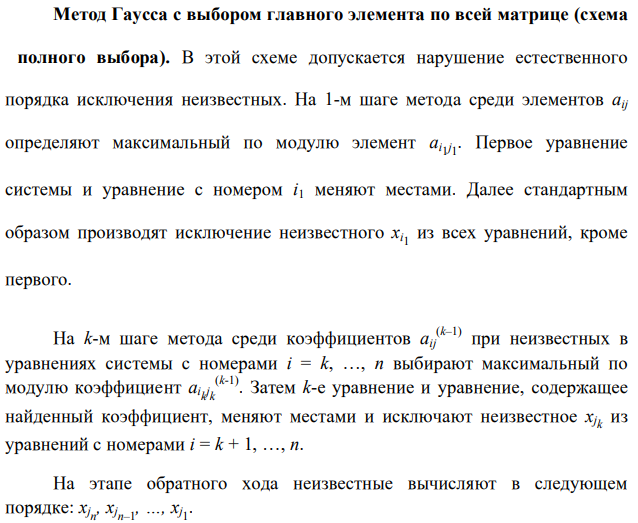




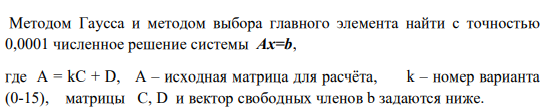




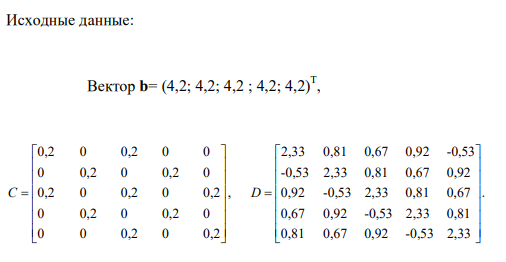




# **ЗАДАНИЕ**



Вариант 27



# **ПРОГРАММНАЯ РЕАЛИЗАЦИЯ**

#include <iostream>

#include <iomanip>

**using** **namespace** std;

**const** **int** n = **5**;

**double**\* **gauss1**(**double** A[n][n], **double**\* b, **int** n) // схема единстенного деления

{

**double**\* x = **new** **double**[n];

**for** (**int** i = **0**; i < n; i++) // прямой ход, приведение к верхетреугольному виду

{

**if** (A[i][i] != **0**)

{

**for** (**int** j = i + **1**; j < n; j++)

{

**double** q = A[j][i] / A[i][i]; // получаем q

**for** (**int** k = i; k < n; k++) // вычитаем i-ую строку умноженную на q из последующих строк

{

**if** (k == i)

{

A[j][k] = **0**; // исключение погрешности для нулевых элементов

}

**else**

{

A[j][k] -= q \* A[i][k];

}

}

b[j] -= q \* b[i];

}

}

**else**

{

cout << "Нельзя решить систему данным методом: ведущий элемент стал равен 0**\n**";

**return** x;

}

}

**for** (**int** i = n - **1**; i >= **0**; i--) // обратный ход, получение решений уравнений

{

**for** (**int** j = n - **1**; j > i; j--)

{

b[i] -= x[j] \* A[i][j];

}

x[i] = b[i] / A[i][i];

}

**return** x;

}

**double**\* **gauss2**(**double** A[n][n], **double**\* b, **int** n)

{

**double**\* x = **new** **double**[n];

**for** (**int** i = **0**; i < n; i++) //прямой ход, приведение к верхетреугольному виду

{

**int** max = abs(A[i][i]);

**int** maxIndex = i;

**for** (**int** j = i + **1**; j < n; j++) //поиск наибольшего главного элемента в столбце

{

**if** (abs(A[j][i]) > max)

{

maxIndex = j;

max = abs(A[j][i]);

}

}

**for** (**int** j = **0**; j < n; j++) //перестановка уравнения с наибольшим главным элементом

{

swap(A[i][j], A[maxIndex][j]);

}

swap(b[i], b[maxIndex]);

**if** (A[i][i] != **0**) // аналогично первой схеме решения

{

**for** (**int** j = i + **1**; j < n; j++)

{

**double** q = A[j][i] / A[i][i];

**for** (**int** k = i; k < n; k++)

{

**if** (k == i)

{

A[j][k] = **0**;

}

**else**

{

A[j][k] -= q \* A[i][k];

}

}

b[j] -= q \* b[i];

}

}

**else**

{

cout << "Нельзя решить систему данным методом: ведущий элемент стал равен 0**\n**";

**return** x;

}

}

**for** (**int** i = n - **1**; i >= **0**; i--)

{

**for** (**int** j = n - **1**; j > i; j--)

{

b[i] -= x[j] \* A[i][j];

}

x[i] = b[i] / A[i][i];

}

**return** x;

}

**double**\* **gauss3**(**double** A[n][n], **double**\* b, **int** n)

{

**double**\* x = **new** **double**[n];

**for** (**int** i = **0**; i < n; i++) //прямой ход

{

**double** max = abs(A[i][**0**]);

**int** maxIndexX = i;

**int** maxIndexY = **0**;

**for** (**int** j = i; j < n; j++) //поиск наибольшего элемента в матрице под текущей строкой

{

**for** (**int** k = **0**; k < n; k++)

{

**if** (abs(A[j][k]) > max)

{

maxIndexX = j;

maxIndexY = k;

max = abs(A[j][k]);

}

}

}

**for** (**int** j = **0**; j < n; j++) //перестановка уравнения с наибольшим главным элементом

{

swap(A[i][j], A[maxIndexX][j]);

}

swap(b[i], b[maxIndexX]);

**if** (A[i][maxIndexY] != **0**)

{

**for** (**int** j = i + **1**; j < n; j++)

{

**double** q = A[j][maxIndexY] / A[i][maxIndexY]; //получаем q

**for** (**int** k = **0**; k < n; k++) //вычитаем i-ую строку умноженную на q из последующих строк

{

**if** (k == maxIndexY)

{

A[j][k] = **0**; //исключение погрешности для нулевых элементов

}

**else**

{

A[j][k] -= q \* A[i][k];

}

}

b[j] -= q \* b[i];

}

}

**else**

{

cout << "Нельзя решить систему данным методом: элементы в оставшихся строках стали нулевыми (система не имеет единственное решение)**\n**";

**return** x;

}

}

**bool**\* xFound = **new** **bool**[n];

**for** (**int** i = **0**; i < n; i++)

{

xFound[i] = **0**;

}

**for** (**int** i = n - **1**; i >= **0**; i--) //обратный ход, получение решений уравнений

{

**int** curX;

**for** (**int** j = **0**; j < n; j++)

{

**if** (A[i][j] != **0** && xFound[j] == **0**)

{

curX = j;

}

**if** (A[i][j] != **0** && xFound[j] == **1**)

{

b[i] -= x[j] \* A[i][j];

}

}

x[curX] = b[i] / A[i][curX];

xFound[curX] = **1**;

}

**return** x;

}

**int** **main**()

{

setlocale(LC\_ALL, "ru");

std::cout.precision(**4**); // Устанавливает точность

std::cout.setf(std::ios::fixed, std::ios::floatfield);

**double** b[n] = { **4.2**, **4.2**, **4.2**, **4.2**, **4.2** };

**double** C[n][n] = { {**0.2**,**0**,**0.2**,**0**,**0**},

{**0**,**0.2**,**0**,**0.2**,**0**},

{**0.2**,**0**,**0.2**,**0**,**0.2**},

{**0**,**0.2**,**0**,**0.2**,**0**},

{**0**,**0**,**0.2**,**0**,**0.2**} };

**double** D[n][n] = { {**2.33**,**0.81**,**0.67**,**0.92**,-**0.53**},

{-**0.53**,**2.33**,**0.81**,**0.67**,**0.92**},

{**0.92**,-**0.53**,**2.33**,**0.81**,**0.67**},

{**0.67**,**0.92**,-**0.53**,**2.33**,**0.81**},

{**0.81**,**0.67**,**0.92**,-**0.53**,**2.33**} };

**double** A[n][n]{};

**for** (**int** i = **0**; i < n; i++)

{

**for** (**int** j = **0**; j < n; j++)

{

A[i][j] = **27** \* C[i][j] + D[i][j];

}

}

**for** (**int** i = **0**; i < n; i++)

{

**for** (**int** j = **0**; j < n; j++)

{

cout << A[i][j] << " ";

}

cout << '\n';

}

cout << "**\n\n**";

**int** choise;

**while** (true)

{

cout << "1. Метод Гаусса (схема единственного деления)**\n**";

cout << "2. Метод Гаусса (схема частичного выбора)**\n**";

cout << "3. Метод Гаусса (схема полного выбора)**\n**";

cout << "4. Выход**\n**";

cin >> choise;

system("cls"); // Очистка экрана консоли

**if** (choise == **1**)

{

**double**\* x = gauss1(A, b, n);

**for** (**int** i = **0**; i < n; i++)

{

cout << x[i] << "**\n**";

}

**return** **0**;

}

**else** **if** (choise == **2**)

{

**double**\* x = gauss2(A, b, n);

**for** (**int** i = **0**; i < n; i++)

{

cout << x[i] << "**\n**";

}

**return** **0**;

}

**else** **if** (choise == **3**)

{

**double**\* x = gauss3(A, b, n);

**for** (**int** i = **0**; i < n; i++)

{

cout << x[i] << "**\n**";

}

**return** **0**;

}

**else** **if** (choise == **4**)

{

**return** **0**;

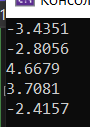
}

}

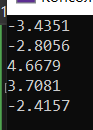
}

# **ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ**

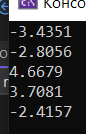
Метод Гаусса (схема единственного деления):



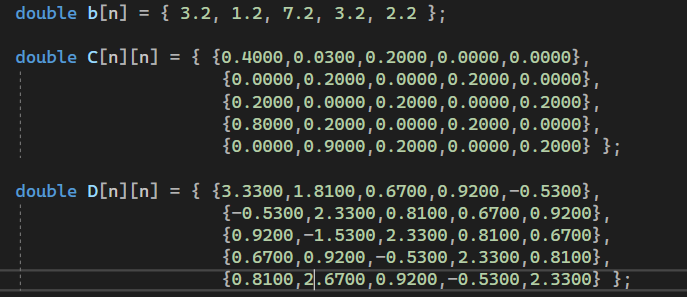
Метод Гаусса (схема частичного выбора):



Метод Гаусса (схема полного выбора):



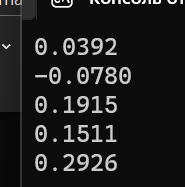
**ТЕСТОВЫЙ ПРИМЕР**

****

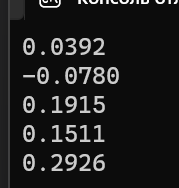
Должны были получится следующие корни: 0.0392, -0.0780, 0.1915, 0.1511, 0.2926

Согласно результатам программы:

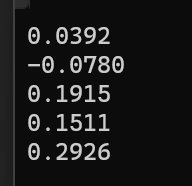
Метод Гаусса (схема единственного деления):



Метод Гаусса (схема частичного выбора):



Метод Гаусса (схема полного выбора):



Что соответствует ожидаемым корням.

double A[n][n] = { {2.1234, 0.91, 0.733, 1.33, 3.76},

{5.546, 1.123, 4.2, 2.28, 1.337},

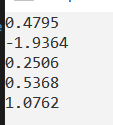
{2.435, 0.123, 0.45, 0.55, 2.66},

{0.12, -0.54, 0.12, -0.542, 3.12},

{-2.12, -1.435, 0.564, 2.554, 0.86,} };

Должны были получится следующие корни: 0.4795, -1.9364, 0.2506, 0.5368, 1.0762

Cогласно результатам программы (метод Гаусса (схема единственного деления), метод Гаусса (схема полного выбора), метод Гаусса (схема полного выбора)):



Что соответствует ожидаемым корням.

double A[n][n] = { {0.1234, 2.91, 3.732, 0.33, -3.76},

{0.546, 1.12, 2.245, 4.28, -1.337},

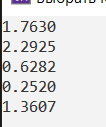
{-2.435, -0.123, 2.45, 3.55, 4.66},

{2.1245, -2.54, 4.12, -2.542, 3.182},

{2.12, -0.435, -0.564, 2.554, 0.86,} };

Должны были получится следующие корни: 1.7630, 2.2925, 0.6282, 0.2520, 1.3607

Cогласно результатам программы (метод Гаусса (схема единственного деления), метод Гаусса (схема полного выбора), метод Гаусса (схема полного выбора)):



Что соответствует ожидаемым корням.

double A[n][n] = { {0, 3.91, 5.732, 0.33, -3.76},

{3.12, 0, -0.1124, 4.28, -1.337},

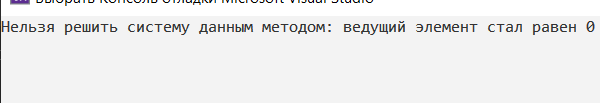
{1.54, -0.123, 0, 3.55, 4.66},

{-0.1245, -2.54, 4.12, 0, 3.182},

{4.12, -0.435, -0.564, 2.554, 0} };

По результату решений не должно быть.

Cогласно результатам программы (метод Гаусса (схема единственного деления), метод Гаусса (схема полного выбора), метод Гаусса (схема полного выбора)):



Что соответствует ожидаемому результату.

double A[n][n] = { {0, 3.91, 5.732, 0.33, -3.76},

{3.12, 0, -0.1124, 4.28, -1.337},

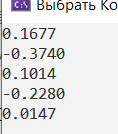
{1.54, -0.123, 0, 3.55, 4.66},

{-0.1245, -2.54, 4.12, 0, 3.182},

{4.12, -0.435, -0.564, 2.554, 0} };

Должны были получится следующие корни: 0.1677, -0.3740, 0.1014, -0.2280, 0.0147.

Cогласно результатам программы (метод Гаусса (схема единственного деления), метод Гаусса (схема полного выбора), метод Гаусса (схема полного выбора)):



Что соответствует ожидаемому результату.

**ОЦЕНКА ПОГРЕШНОСТИ**

Δ(x1) = |x – x1| = |-3.4351 + 3.43512707808| = 0.00002707808

Δ(x2) = |x – x2| = |-2.8056 + 2.80562722508| = 0.00002722508

Δ(x3) = |x – x3| = |4.6679 – 4.66791846713| = 0.00001846713

Δ(x4) = |x – x4| = |3.7081 – 3.70813063443| = 0.00003063443

Δ(x5) = |x – x5| = |-2.4157 + 2.41574544657| = 0.00004544657

# **ВЫВОД**

В ходе выполнения данной лабораторной работы был применён

метод Гаусса по схеме единственного деления, метод Гаусса с выбором

главного элемента по столбцу (схема частичного выбора) и метод Гаусса с

выбором главного элемента по всей матрице (схема полного выбора) для

решения системы линейных уравнений, также были составлены

алгоритмы и созданы реализации соответствующих программ на языке

С++ для решения поставленной задачи. Была произведена оценка

точности результатов.