МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА № 2

по дисциплине

“Вычислительная математика”

Вариант “Прямой метод Гаусса”

Выполнила:

Студентка группы P3232

Копалина Майя
Алексеевна

Преподаватель:

Перл Ольга

Вячеславовна

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2024

Задание:

Придумать алгоритм и написать код, решающий системы линейных уравнений с помощью прямого метода Гаусса.

$$\begin{cases} a_{11}\cdot x_1 + a_{12}\cdot x_2 + .. + a_{1n}\cdot x_n = b_1 \\ a_{21}\cdot x_1 + a_{22}\cdot x_2 + .. + a_{2n}\cdot x_n = b_2 \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ a_{n1}\cdot x_1 + a_{n2}\cdot x_2 + .. + a_{nn}\cdot x_n = b_n \end{cases}$$

Описание метода:

Метод Гаусса – это метод решение квадратных систем линейных алгебраических уравнений (СЛАУ), суть которого заключается в последовательном исключение неизвестных переменных с помощью элементарных преобразований строк.

Прямой ход метода Гаусса – это поочерёдное преобразования уравнений системы для последующего избавления от переменных неизвестных.

Обратный ход метода Гаусса – это вычисление переменных неизвестных от последнего уравнения к первому.

Элементарные преобразования нужны, чтобы привести данную матрицу к ступенчатому (или треугольному) виду. В таком виде под главной диагональю матрицы должны быть одни нули.

- Можно переставлять строки матрицы местами;
- Если в матрице есть одинаковые (или пропорциональные) строки, можно удалить их все, кроме одной;
- Можно умножать или делить строку на любое число (кроме нуля);
- Нулевые строки удаляются;
- Можно прибавлять к строке строку, умноженную на число, отличное от нуля.

Словесное описание алгоритма:

- Начинаем с исходной системы линейных уравнений.
- Нормализуем систему, деля каждую строку на ее ведущий элемент. Используем метод исключения, чтобы последовательно обнулить элементы под диагональю матрицы.
- Получаем треугольную систему, которую легко решаем методом подстановки.
- Находим значения неизвестных и следим за точностью вычислений.

Алгоритм решения системы линейных уравнений методом Гаусса:

Входные данные:

- n: количество уравнений (размерность матрицы А)
- matrix: расширенная матрица А, где столбец n содержит правые части уравнений

Выходные данные:

- result: список решений системы (x_i)
- residuals: список невязок (разница между левой и правой частями уравнений после подстановки решений)
- пустой список ([]): в случае, если система не имеет решений

Прямой ход:

1.1. Поиск строки с максимальным элементом по столбцу (i):

a. Для каждой строки i (от 0 до n-1):
b. Инициализация max_row = i (строка с максимальным элементом по умолчанию)
c. Для каждой строки j (от i+1 до n-1):
d. Сравнение модулей элементов a_ji и a_max_row,i:
e. Если abs(a_ji) > abs(a_max_row,i):
f. Обновить max_row = j (запомнить строку с бОльшим модулем элемента)

1.2. Проверка на нулевой ведущий элемент:

Проверка abs(a_i,i) < 1e-10:

a. Если да:
   Установить isSolutionExists = False** (система не имеет решений или бесконечное множество решений)
   Вернуть пустой список []** (выход из функции)

1.3. Для каждой строки j (от i+1 до n-1):

- Вычислить коэффициент m_ji = a_ji / a_ii
- Для каждого элемента k (от i до n и правой части) выполнить элементарное преобразование строки j: a_jk -= m_ji * a_ik (вычитание для исключения x_i из уравнения j)

Решение треугольной системы:

Инициализировать список решений result нулями (длиной n).

2.1. Подстановка назад:

- Для каждой строке i (от n-1 до 0 с шагом -1) вычислить решение x_i = a_in / a_ii (деление правой части уравнения i на диагональный элемент)
- Для каждой строки j (от i-1 до 0 с шагом -1): вычислить поправку к правой части уравнения j: a_jn -= a_ji * x_i (исключение x_i из уравнения j)

Вычисление невязок (опционально):
Инициализировать список residuals нулями (длиной n).
Для каждой строки i (от 0 до n-1):

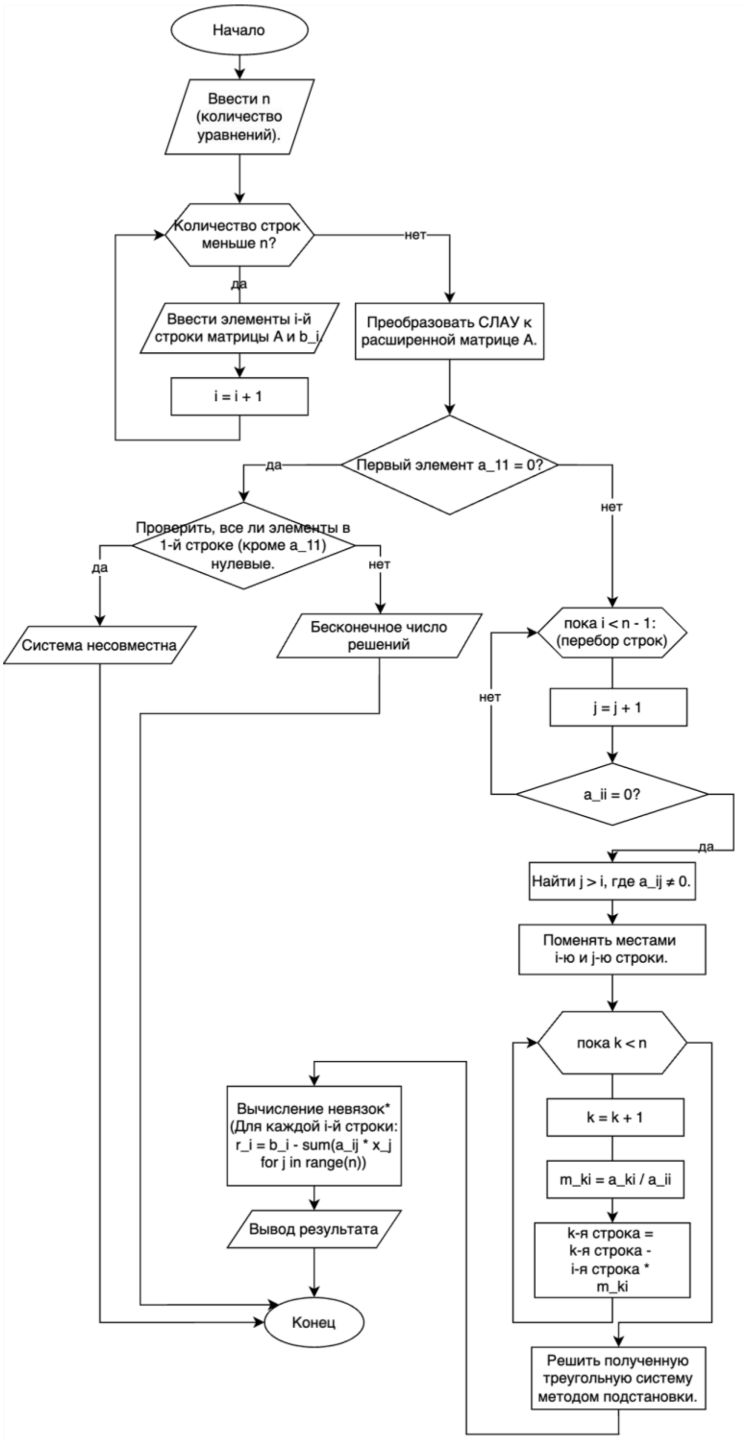
- Вычислить невязку r_i = b_i - sum(a_ij * x_j for j in range(n)) (разница между правой частью и суммой произведения элементов строки i на решения)
- Добавить невязку r_i в список residuals.

Вывод:

- Если isSolutionExists = True (система имеет единственное решение):
- Вернуть result + residuals (список решений и невязок)
- Иначе (система не имеет решений или бесконечное множество решений):
- Вернуть пустой список [] (выход из функции)

Примечания:
Алгоритм использует частичную перестановку строк для улучшения стабильности численного метода.
Проверка на нулевой ведущий элемент позволяет определить несовместность системы или бесконечное множество решений.
Вычисление невязок является опциональным шагом и служит для оценки точности полученного решения.

Блок-схема:



Примечание:
Подробнее про опциональное вычисление невязок можно прочитать выше.

Программа, написанная на python:

```
class Solution:
    isSolutionExists = True
    errorMessage = "The system has no roots of equations or has an infinite set of them."

    @staticmethod
    def solveByGauss(n, matrix):
        for i in range(n):
            max_row = i
            for j in range(i + 1, n):
                if abs(matrix[j][i]) > abs(matrix[max_row][i]):
                    max_row = j
            matrix[i], matrix[max_row] = matrix[max_row], matrix[i]

            if abs(matrix[i][i]) < 1e-10:
                Solution.isSolutionExists = False
                return []

            for j in range(i + 1, n):
                ratio = matrix[j][i] / matrix[i][i]
                for k in range(n + 1):
                    matrix[j][k] -= ratio * matrix[i][k]

        if Solution.isSolutionExists:
            result = [0] * n
            for i in range(n - 1, -1, -1):
                result[i] = matrix[i][n] / matrix[i][i]
                for j in range(i - 1, -1, -1):
                    matrix[j][n] -= matrix[j][i] * result[i]
            residuals = []
            for i in range(n):
                residual = 0
                for j in range(n):
                    residual += matrix[i][j] * result[j]
                residuals.append(residual - matrix[i][n])
            return result + residuals
        else:
            return []
```

Примеры работы программы:

Для того, чтобы продемонстрировать работу программы на различных данных, включая граничные случаи и исключительные ситуации, предлагаю следующие примеры:

Пример 1:
Ввод:
3
2 -1 1 5
1 2 -1 5
1 1 1 6
Вывод:
2.0
3.0
1.0
0.0
0.0
Результат:
Решение системы уравнений с корнями x = 2, y = 3, z = 1, и остатками равными 0.

Пример 2:
Ввод:
3
1 1 1 6
2 -1 1 5
1 2 -1 5
Вывод:
2.0
3.0
1.0
0.0
0.0
Результат:
Решение системы уравнений с корнями x = 2, y = 3, z = 1, и остатками равными 0.

Пример 3:
Ввод:
2
2 -1 5
4 -2 10
Вывод:
The system has no roots of equations or has an infinite set of them.
Результат:

Система уравнений либо не имеет корней, либо имеет бесконечное множество решений.

Пример 4:
Ввод:
3
0 0 0 5
0 0 0 10
0 0 0 15
Вывод:
The system has no roots of equations or has an infinite set of them.
Результат:
Система уравнений либо не имеет корней, либо имеет бесконечное множество решений.

Пример 5:
Ввод:
4
2 -1 1 -1 -3
3 -2 -3 -2 -9
1 -3 -2 -4 -10
-1 -4 -3 -5 -13
Вывод:
-2.0
-3.0
-4.0
-1.0
Результат:
Решение системы уравнений с корнями w = -2, x = -3, y = -4, z = -1, и остатками равными 0.

Вывод:

В результате выполнения лабораторной работы я разобралась с принципом прямого метода решения СЛАУ - метода Гаусса. Асимптотическая сложность данного метода равна O(n^3), где n - размерность матрицы системы.

Метод Гаусса применяется для решения систем линейных уравнений любого размера, но в отличие от итерационных методов, он обеспечивает точное решение. Однако, при работе с большими матрицами метод Гаусса может быть менее эффективным из-за высокой асимптотической сложности.

При проверке корректности работы метода Гаусса убедилась, что полученные результаты совпадают с ожидаемыми значениями, что говорит о правильности реализации алгоритма.

Результат выполнения программы на платформе.

Тесты пройдены.

Assignment results

Score	Plagiarism
73.875 / 100	35.353535%

Number	Status	Score
1	Success	12 / 12
2	Success	5.7749996 / 11
3	Success	5.7749996 / 11
4	Success	5.7749996 / 11
5	Success	11 / 11
6	Success	11 / 11
7	Success	5.7749996 / 11
8	Success	5.7749996 / 11
9	Success	11 / 11