Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №1
по дисциплине «Вычислительная математика»

Вариант: **13 - Метод Гаусса с выбором главного элемента по столбцам**

Преподаватель:
Малышева Татьяна Алексеевна

Выполнил: Телегин Даниил Евгеньевич
Группа: P3211

Санкт-Петербург, 2025 г

Цель работы

Изучить численные методы решения систем линейных алгебраических уравнений и реализовать один из них средствами программирования.

Целью данной лабораторной работы является изучение и реализация метода Гаусса с выбором главного элемента по столбцам для решения системы линейных алгебраических уравнений. Также требуется вычислить определитель матрицы, вывести треугольную форму матрицы, найти вектор неизвестных и вектор невязок, а затем сравнить результат с библиотечными функциями.

Описание метода

Метод Гаусса с выбором главного элемента по столбцам
Метод Гаусса предназначен для решения систем линейных уравнений вида: $Ax = B$
где:

- A — матрица коэффициентов (размерность $n \times n$),
- x — вектор неизвестных (n-мерный),
- B — вектор свободных членов.

Метод состоит из двух этапов:

1. **Прямой ход** (приведение к треугольной форме)
2. **Обратный ход** (нахождение неизвестных методом подстановки)

Выбор главного элемента
На каждом шаге выбирается наибольший по модулю элемент текущего столбца в качестве ведущего. Это уменьшает вычислительные погрешности.
Формула преобразования элементов в прямом ходе:

$$A[j][k] = A[j][k] - \frac{A[j][i]}{A[i][i]} \times A[i][k]$$

$$B[j] = B[j] - \frac{A[j][i]}{A[i][i]} \times B[i]$$

Определитель матрицы
После приведения к треугольному виду определитель вычисляется как произведение диагональных элементов:

$$det(A) = \prod_{i=1}^{n} A[i][i]$$

Вектор невязок
После нахождения решения x, вектор невязок r вычисляется как разница между левыми и правыми частями уравнения: $r = Ax - B$

Листинг программы

https://github.com/quwiier/4_computional_mathematics/tree/master/labs/lab1

main.py

```
import numpy as np

def swap_rows(A, B, row1, row2):
    """Меняет местами две строки в матрице A и векторе B"""
    A[row1], A[row2] = A[row2], A[row1]
    B[row1], B[row2] = B[row2], B[row1]

def gauss_elimination_with_pivoting(A, B):
    """Метод Гаусса с выбором главного элемента по столбцам и подсчётом
перестановок"""
    n = len(A)
    swap_count = 0

    for i in range(n):
        max_row = i
        for k in range(i + 1, n):
            if abs(A[k][i]) > abs(A[max_row][i]):
                max_row = k

        if max_row != i:
            swap_rows(A, B, i, max_row)
            swap_count += 1

        for j in range(i + 1, n):
            factor = A[j][i] / A[i][i]
            for k in range(i, n):
                A[j][k] -= factor * A[i][k]
            B[j] -= factor * B[i]

    return A, B, swap_count

def compute_determinant(U, swap_count):
    """Вычисляет определитель с учетом количества перестановок"""
    det = (-1) ** swap_count
    for i in range(len(U)):
        det *= U[i][i]
    return det

def compute_residual(A, x, B):
```

```
    """Вычисляет вектор невязок"""
    residuals = [sum(A[i][j] * x[j] for j in range(len(A))) - B[i] for i
in range(len(A))]
    return residuals

def back_substitution(A, B):
    """Обратный ход метода Гаусса"""
    n = len(A)
    x = [0] * n
    for i in range(n - 1, -1, -1):
        sum_ax = sum(A[i][j] * x[j] for j in range(i + 1, n))
        x[i] = (B[i] - sum_ax) / A[i][i]
    return x

def main():
    n = int(input("Введите размерность матрицы: "))
    if n > 20:
        print("n > 20, эффективность и точность решения снижена")

    print("Введите коэффициенты матрицы построчно:")
    A = [list(map(float, input().split())) for _ in range(n)]

    print("Введите свободные члены:")
    B = list(map(float, input().split()))

    determinant_numpy = np.linalg.det(np.array(A))
    print(determinant_numpy) # вывод до выполнения алгоритма, чтобы
убедиться, что равен 0, тогда будет ошибка "Система несовместная (0 на
главной диагонали)"
    x_numpy = np.linalg.solve(np.array(A), np.array(B))

    A_copy = [row[:] for row in A]
    B_copy = B[:]

    U, B_transformed, swap_count =
gauss_elimination_with_pivoting(A_copy, B_copy)
    x_gauss = back_substitution(U, B_transformed)
    determinant = compute_determinant(U, swap_count)
    residuals = compute_residual(A, x_gauss, B)


    print("\nТреугольная матрица после преобразований:")
    for i in range(n):
```

```
        print(" ".join(f"{num:.2f}" for num in U[i]) + f" |
{B_transformed[i]:.2f}")

    print("\nОпределитель матрицы:", determinant)
    print("Определитель матрицы (NumPy):", determinant_numpy)
    print(swap_count) # кол-во перестановок для определения знака
детерминанта -1^k, k-колво перестановок

    print("\nРешение системы:", " ".join(f"x{i+1} = {xi:.15f}" for i, xi
in enumerate(x_gauss)))
    print("Решение системы (NumPy):", " ".join(f"x{i+1} = {xi:.15f}" for
i, xi in enumerate(x_numpy)))

    print("\nВектор невязок:", " ".join(f"{r:.30f}" for r in residuals))

if __name__ == "__main__":
    try:
        main()
    except ZeroDivisionError:
        print("Система несовместная (0 на главной диагонали)")
    except Exception as e:
        print(f"Произошла ошибка: {e}")
```

Примеры работы программы

Введите размерность матрицы: 3

Введите коэффициенты матрицы построчно:

2 3 -1

4 -1 2

-3 2 5

Введите свободные члены:

5 3 -2

Треугольная матрица после преобразований:

4.00 -1.00 2.00 | 3.00

0.00 3.50 -2.00 | 3.50

0.00 0.00 7.21 | -1.00

Определитель матрицы: -101.0

Определитель матрицы (NumPy): -101.00000000000003

Решение системы: x1 = 1.049505 x2 = 0.920792 x3 = -0.138614

Решение системы (NumPy): x1 = 1.049505 x2 = 0.920792 x3 = -0.138614

Вектор невязок: 0.000000 0.000000 0.000000

Вывод:

В результате выполнения данной лабораторной работой я познакомился с численными методами решения математических задач на примере систем алгебраических уравнений, реализовав на языке программирования Python метод Гаусса с выбором главного элемента по столбцам, а также сравнив результат с решением через стороннюю библиотеку NumPy.