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

Факультет программной инженерии и компьютерных технологий

Вычислительная математика

Лабораторная работа 1

**«Решение системы линейных алгебраических уравнений СЛАУ»**

Вариант 13

Выполнила:

Павличенко Софья Алексеевна, Р3215

Проверила:   
Малышева Татьяна Алексеевна

Санкт-Петербург 2025г.

Оглавление

[Цель 3](#_Toc191403469)

[Описание метода 3](#_Toc191403470)

[Этапы метода: 3](#_Toc191403471)

[Расчётные формулы 3](#_Toc191403472)

[Листинг программы 4](#_Toc191403473)

[Результат работы программы 6](#_Toc191403474)

[Выводы 7](#_Toc191403475)

# Цель

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

# Описание метода

**Метод Гаусса с выбором главного элемента по столбцу** — это модификация классического метода Гаусса, где на каждом шаге для повышения точности и стабильности выбирается **наибольший по модулю элемент в текущем столбце** (ведущий элемент), и соответствующая строка меняется местами с текущей.

### **Этапы метода:**

1. **Выбор главного элемента:**
   * В текущем столбце (начиная с главной диагонали) находим элемент с **наибольшим модулем**.
   * Меняем строки местами, чтобы этот элемент стал ведущим.
2. **Прямой ход (приведение к верхнетреугольному виду):**
   * Для всех строк ниже ведущей зануляем элементы под диагональю, используя элементарные преобразования.
3. **Обратный ход (нахождение решения):**
   * Подставляем найденные значения переменных, двигаясь **снизу вверх**, решая треугольную систему.

# Расчётные формулы

– определитель после приведения матрицы к треугольному виду, где *‒* число перестановок строк (или столбцов) матрицы при ее приведении к треугольному виду.

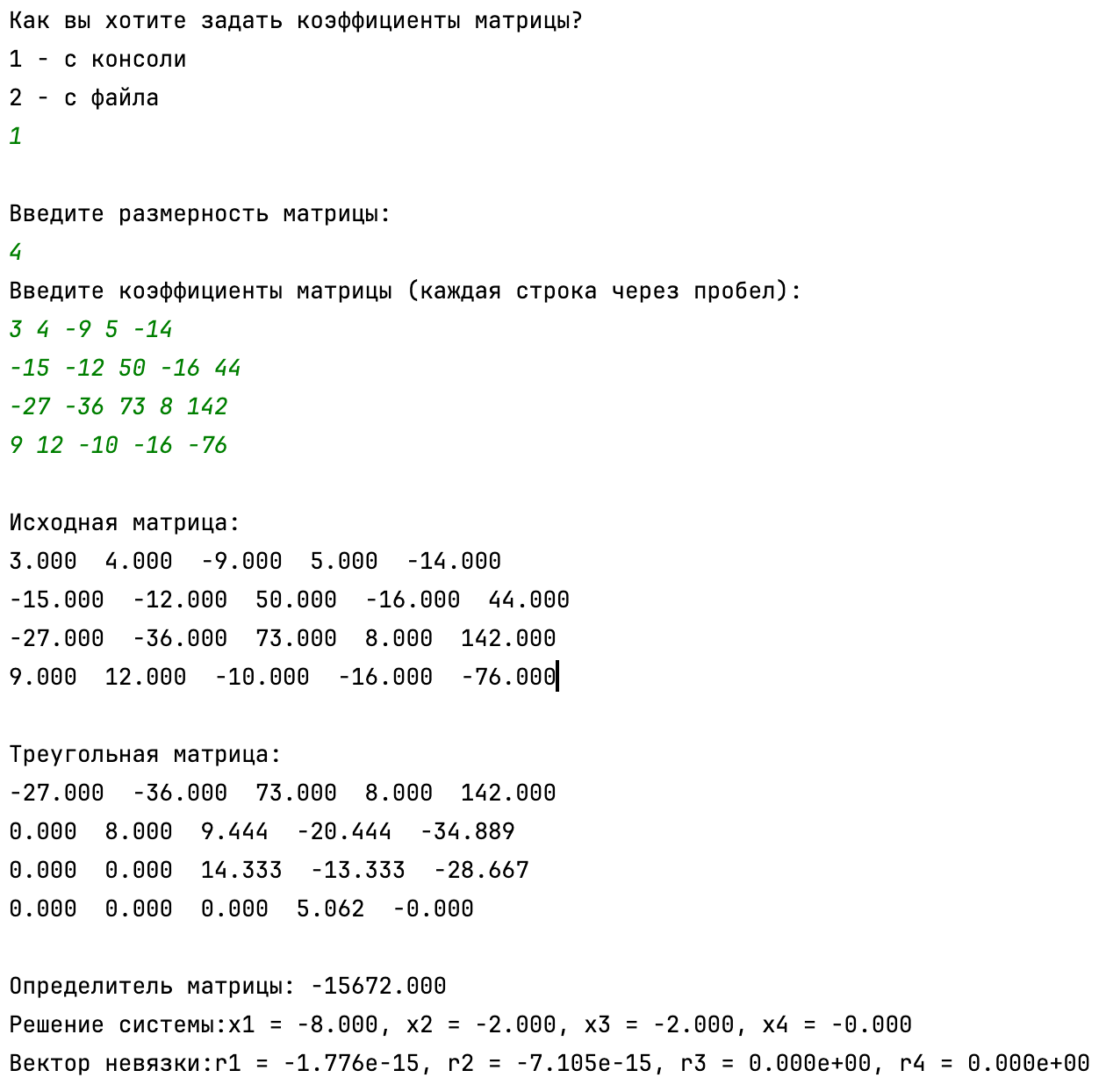
– вектор невязки.

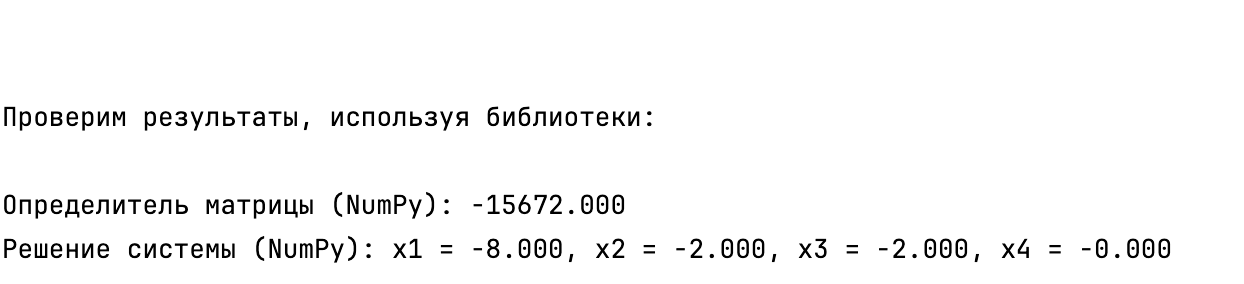
# Листинг программы

import copy  
import numpy as np  
  
  
def to\_upper\_triangular(matrix, n):  
 *"""Прямой ход метода Гаусса — приведение к верхнетреугольному виду"""* triangular\_matrix = copy.deepcopy(matrix) *# Создаём копию, чтобы не изменять исходную матрицу* det = 1 *# Определитель* swap\_count = 0 *# Количество перестановок строк* for i in range(n):  
 *# Выбор главного элемента по столбцу (поиск максимума по модулю)* pivot\_row = i  
 for row in range(i + 1, n):  
 if abs(triangular\_matrix[row][i]) > abs(triangular\_matrix[pivot\_row][i]):  
 pivot\_row = row  
  
 *# Меняем строки, если найден другой главный элемент* if pivot\_row != i:  
 triangular\_matrix[i], triangular\_matrix[pivot\_row] = triangular\_matrix[pivot\_row], triangular\_matrix[i]  
 swap\_count += 1  
  
 *# Если найден нулевой элемент на диагонали — система вырожденная* if triangular\_matrix[i][i] == 0:  
 return triangular\_matrix, 0, swap\_count  
  
 det \*= triangular\_matrix[i][i]  
  
 *# Зануление элементов под главной диагональю* for elimination\_row in range(i + 1, n):  
 k = triangular\_matrix[elimination\_row][i] / triangular\_matrix[i][i]  
 for col in range(i, n + 1):  
 triangular\_matrix[elimination\_row][col] -= k \* triangular\_matrix[i][col]  
  
 *# Корректируем знак определителя в зависимости от количества перестановок* return triangular\_matrix, det \* (-1) \*\* swap\_count, swap\_count  
  
  
def solve\_slae(triangular\_matrix, n):  
 *"""Обратный ход метода Гаусса - находим переменные"""* solution = [0] \* n  
 for i in range(n - 1, -1, -1):  
 *# Находим значение x\_i* solution[i] = triangular\_matrix[i][n] / triangular\_matrix[i][i]  
  
 *# Вычитаем найденное значение из предыдущих уравнений* for j in range(i - 1, -1, -1):  
 triangular\_matrix[j][n] -= triangular\_matrix[j][i] \* solution[i]  
  
 return solution  
  
  
def compute\_residual(matrix, n, solution):  
 *"""Вычисляет вектор невязки: разницу между Ax и b."""* residual = [0] \* n  
 for i in range(n):  
 Ax\_i = sum(matrix[i][j] \* solution[j] for j in range(n))  
 residual[i] = Ax\_i - matrix[i][n]  
  
 return residual  
  
  
*# --- Ввод данных ---*. . .

*# --- Решение СЛАУ ---  
  
# print\_matrix(matrix, "Исходная матрица:")*triangular\_matrix, determinant, swap\_count = to\_upper\_triangular(matrix, n)  
print\_matrix(triangular\_matrix, "Треугольная матрица:")  
  
print(f"Количество перестановок: {swap\_count}")  
print(f"Определитель матрицы: {determinant:.3f}")  
  
if determinant == 0:  
 print("Система несовместна или имеет бесконечно много решений.")  
else:  
 solution = solve\_slae(triangular\_matrix, n)  
 print("Решение системы:" + ", ".join(f"x{i+1} = {x:.3f}" for i, x in enumerate(solution)))  
  
 residual = compute\_residual(matrix, n, solution)  
 print("Вектор невязки:" + ", ".join(f"r{i+1} = {r:.30e}" for i, r in enumerate(residual)))  
  
print("\n\nПроверим результаты, используя библиотеки:\n")  
  
A = np.array([row[:-1] for row in matrix], dtype=float)  
b = np.array([row[-1] for row in matrix], dtype=float)  
  
print(f"Определитель матрицы (NumPy): {np.linalg.det(A):.3f}")  
  
*# Проверка рангов*rank\_A = np.linalg.matrix\_rank(A) *# Ранг коэффициентной матрицы*rank\_Ab = np.linalg.matrix\_rank(np.column\_stack((A, b))) *# Ранг расширенной матрицы*if rank\_A == rank\_Ab and rank\_A == n:  
 print("Решение системы (NumPy):", ", ".join(f"x{i+1} = {xi:.3f}" for i, xi in enumerate(np.linalg.solve(A, b))))  
else:  
 print("Система несовместна или имеет бесконечно много решений.")

# Результат работы программы





# Выводы

В результате выполнения данной лабораторной работой я познакомилась с численными методами решения математических задач на примере систем алгебраических уравнений, реализовав на языке программирования Python метод Гаусса с выбором главного элемента по столбцам.

Результаты, полученные программой, совпали с библиотечными (NumPy) с высокой точностью, что говорит о корректности реализованного алгоритма.