МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

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 №8

Выполнил:
Студент группы P3208
Петров В. М.

Преподаватель:
Машина Е.А.

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2025

Цель работы

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

Описание метода

Метод Гаусса с выбором главного элемента по столбцам.

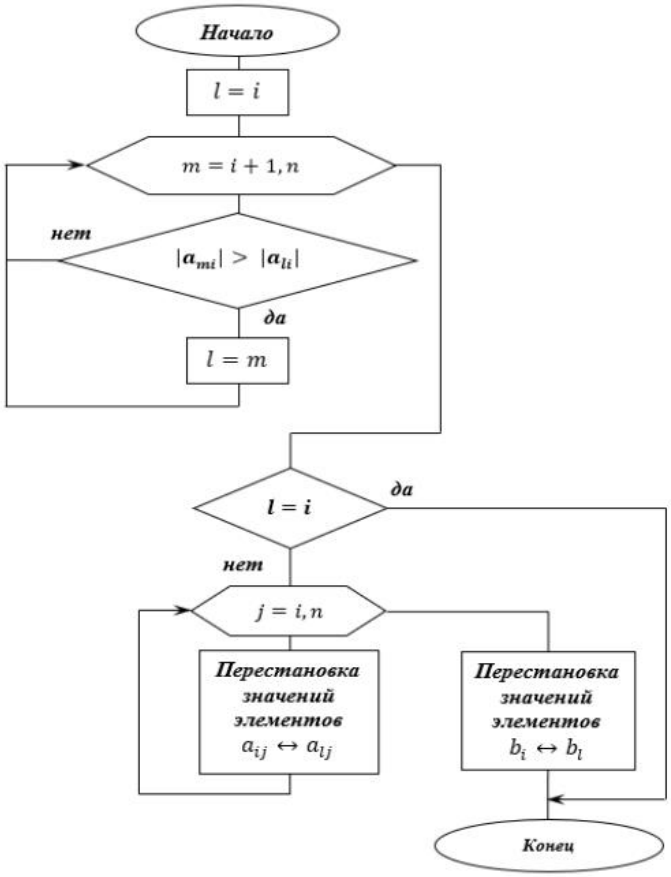
Схема с выбором главного элемента является одной из модификаций метода Гаусса. Идеей является такая перестановка уравнений, чтобы на k-ом шаге исключения ведущим элементом a_{ii} оказывался наибольший по модулю элемент k-го столбца.

Листинг программы

```
import os
import random
from prettytable import PrettyTable
import numpy as np

def print_matrix_in_table(matrix, n):
    table = PrettyTable()
    table.field_names = ["k" + str(i + 1) if i < n else "b" for i in range(n + 1)]
    for row in matrix:
        table.add_row(row)
    print(table)

def get_file():
    """ Получение порядка матрицы и самой матрицы из файла по введённому пути """
    # 1 - ошибка при считывании, 2 - стартовое значение, 0 - успешно
    code_error = 2
    matrix = []
    while code_error != 0:
        if code_error != 2:
            print("Ошибка при чтении данных из файла, в строке должен быть n+1 элемент, а строк всего n (n -
определитель матрицы)")
        file_path = input("Введите путь до файла: ")
        while not os.path.isfile(file_path):
            file_path = input("К сожалению, файла по этому пути не существует, введите путь снова: ")

        code_error = 0
        with open(file_path, 'r') as input_file:
            n = int(input_file.readline())
            if n <= 0:
                code_error = 1
            for string in input_file.readlines():
                line_matrix = list(map(int, string.strip().split()))
                if len(line_matrix) != n + 1:
                    code_error = 1
                    break
                matrix.append(line_matrix)
            if len(matrix) != n:
                code_error = 1

    return matrix

def get_keyboard():
    """ Получение порядка матрицы и самой матрицы с клавиатуры """
    n = int(input("Введите порядок матрицы: n = "))
    while n <= 0:
        n = int(input("Порядок матрицы должен быть > 0. Введите порядок матрицы: n = "))

    matrix = []
    for i in range(n):
        line_matrix = list(map(int, input("коэффициенты уравнения №" + str(i + 1) + ": ").strip().split()))
        while len(line_matrix) != n + 1:
            print("В строке должен быть n+1 элемент, где n - определитель матрицы. Введите строку повторно")
            line_matrix = list(map(int, input("коэффициенты уравнения №" + str(i + 1) + ": ").strip().split()))

        matrix.append(line_matrix)

    return matrix
```

```
def get_random():
    """ Создание матрицы заданной размерности со случайными значениями """
    n = int(input("Введите порядок матрицы: n = "))
    while n <= 0:
        n = int(input("Порядок матрицы должен быть > 0. Введите порядок матрицы: n = "))
    return [[random.randint(-20, 20) for _ in range(n + 1)] for _ in range(n)]

def get_determinant(matrix, n):
    """ Вычисление определителя матрицы """
    determinant = 1
    for i in range(n):
        determinant *= matrix[i][i]
    return determinant

def matrix_transformation(matrix, n):
    """ Преобразование начальной матрицы методом Гаусса с выбором главного элемента по столбцам """
    for col in range(n - 1):
        index_max_num = col
        for row in range(col + 1, n):
            if abs(matrix[row][col]) > abs(matrix[index_max_num][col]):
                index_max_num = row
        matrix[col], matrix[index_max_num] = matrix[index_max_num], matrix[col]

        for row in range(col + 1, n):
            multiplier = matrix[row][col] / matrix[col][col]
            for k in range(n + 1):
                matrix[row][k] -= multiplier * matrix[col][k]

    print("Полученная треугольная матрица:")
    print_matrix_in_table(matrix, n)
    return matrix

def gauss_method(matrix, n):
    new_matrix = matrix_transformation(matrix, n)
    determinant = get_determinant(matrix, n)
    if determinant == 0:
        print("Матрица является несовместной: det = 0")
        return
    else:
        print("Определитель: det = " + str(determinant))

    variables = [0] * n
    residuals = [0] * n
    for i in range(n - 1, -1, -1):
        sum_vars = 0
        for k in range(i + 1, n):
            sum_vars += new_matrix[i][k] * variables[k]
        variables[i] = (new_matrix[i][n] - sum_vars) / new_matrix[i][i]
        residuals[i] = sum_vars + new_matrix[i][i] * variables[i] - new_matrix[i][n]

    print("Полученный вектор неизвестных:")
    print(*variables)
    print("Полученный вектор невязок:")
    print(*residuals)


def start():
    print("Решение системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента по столбцам")
    file_or_keyboard_or_random = ""
    while file_or_keyboard_or_random != "f" and file_or_keyboard_or_random != "k" and file_or_keyboard_or_random !=
"r":
        file_or_keyboard_or_random = input("Как вы хотите ввести матрицу? Введите f, если из файла, k - с клавиатуры,
r - создать рандомную матрицу: ")

    matrix = []
    if file_or_keyboard_or_random == "f":
        matrix = get_file()
        print("Считанная из файла матрица:")
        print_matrix_in_table(matrix, len(matrix))
    elif file_or_keyboard_or_random == "k":
        matrix = get_keyboard()
    elif file_or_keyboard_or_random == "r":
        matrix = get_random()
        print("Созданная случайная матрица:")
        print_matrix_in_table(matrix, len(matrix))
    gauss_method(matrix, len(matrix))

    print()
    matrix_k = [[matrix[row][col] for col in range(len(matrix))] for row in range(len(matrix))]
    coef_b = [matrix[row][-1] for row in range(len(matrix))]
    print("Сравним результаты с вычислениями через библиотеку numpy:")
    try:
        print("Решение системы:", *np.linalg.solve(matrix_k, coef_b))
        print("Определитель матрицы: det = ", np.linalg.det(matrix_k))
    except np.linalg.LinAlgError:
        print("det = 0")

if __name__ == '__main__':
    start()
```

Блок-схема метода



l – номер наибольшего по абсолютной величине элемента матрицы в столбце с номером i (т. е. среди элементов a_{ii}, … , a_{mi}, … , a_{ni});
m – текущий номер элемента, с которым происходит сравнение;
Выбор главного элемента осуществляется по столбцу

Примеры работы программы

```
Решение системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента по столбцам
Как вы хотите ввести матрицу? Введите f, если из файла, k - с клавиатуры, r - создать рандомную матрицу: r
Введите порядок матрицы: n = 6
Созданная случайная матрица:
+-----+-----+-----+-----+-----+-----+-----+
|  k1 |  k2 |  k3 |  k4 |  k5 |  k6 |  b  |
+-----+-----+-----+-----+-----+-----+-----+
|  5  |  13 |  5  |  13 | -17 |  7  | -10 |
|  8  | -20 |  18 |  10 |  5  |  11 |  -6 |
|  12 |  15 | -18 | -15 |  20 | -19 | -12 |
| -17 | -17 |  -7 |  2  | -20 |  16 | -12 |
|  -9 |  -9 | -12 |  3  | -18 |  -8 |  9  |
| -16 |  6  |  17 | -20 |  10 | -11 |  -4 |
+-----+-----+-----+-----+-----+-----+-----+
Полученная треугольная матрица:
+-----+------+-----------------------+--------------------+--------------------+--------------------+---------------------+
|  k1 |   k2 |          k3           |         k4         |         k5         |         k6         |          b          |
+-----+------+-----------------------+--------------------+--------------------+--------------------+---------------------+
| -17 |  -17 |          -7           |         2          |        -20         |         16         |         -12         |
| 0.0 | -28.0 |   14.705882352941178  |  10.941176470588236 | -4.411764705882353 |  18.529411764705884 | -11.647058823529411 |
| 0.0 |  0.0 |   35.14285714285714   | -13.285714285714286 | 25.357142857142858 | -11.499999999999998 | -1.8571428571428577 |
| 0.0 |  0.0 |          0.0          | -20.49318507890961 | 20.825860832137735 |  -12.71215925394548 | -22.847560975609756 |
| 0.0 |  0.0 | 8.881784197001252e-16 |         0.0         | -9.566946705171958 |  7.294276713048044  |  -38.12477465651527 |
| 0.0 |  0.0 | -2.451840894876144e-16 |         0.0         |        0.0         |  -20.4574223734581  |  26.77070750035446  |
+-----+------+-----------------------+--------------------+--------------------+--------------------+---------------------+
Определитель: det = -67093082.99999995
Полученный вектор неизвестных:
-3.7622306013274134 0.6189653708415815 -0.760504134233927 4.9624341454095955 2.9873104355630824 -1.308606089840886
Полученный вектор невязок:
0.0 -1.776356839400250 5e-15 0.0 1.776356839400250 5e-14 0.0 0.0

Сравним результаты с вычислениями через библиотеку numpy:
Решение системы: -3.762230601327413 0.6189653708415813 -0.7605041342339274 4.9624341454095955 2.9873104355630824 -1.308606089840886
Определитель матрицы: det =  -67093082.999999925
```

```
Решение системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента по столбцам
Как вы хотите ввести матрицу? Введите f, если из файла, k - с клавиатуры, r - создать рандомную матрицу: k
Введите порядок матрицы: n = 4
коэффициенты уравнения №1: 1 2 3 4 5
коэффициенты уравнения №2: 1 1 1 1
В строке должен быть n+1 элемент, где n - определитель матрицы. Введите строку повторно
коэффициенты уравнения №2: 1 1 1 1 10
коэффициенты уравнения №3: 3 5 -1 8 0
коэффициенты уравнения №4: 5 1 9 10 4
Полученная треугольная матрица:
+-----+-----+---------------------+---------------------+--------------------+
|  k1 |  k2 |          k3         |          k4         |         b          |
+-----+-----+---------------------+---------------------+--------------------+
|  5  |  1  |          9          |          10         |         4          |
| 0.0 | 4.4 | -6.3999999999999995 |         2.0         |        -2.4        |
| 0.0 | 0.0 |  3.8181818181818175 |  1.1818181818181819 | 5.181818181818182  |
| 0.0 | 0.0 |         0.0         | -1.4761904761904763 | 9.142857142857142  |
+-----+-----+---------------------+---------------------+--------------------+
Определитель: det = -123.99999999999999
Полученный вектор неизвестных:
5.887096774193546 7.032258064516127 3.274193548387097 -6.193548387096773
Полученный вектор невязок:
0.0 1.3322676295501878e-15 0.0 0.0

Сравним результаты с вычислениями через библиотеку numpy:
Решение системы: 5.887096774193546 7.032258064516127 3.274193548387097 -6.193548387096773
Определитель матрицы: det =  -124.00000000000003
```

```
Решение системы линейных алгебраических уравнений методом Гаусса с выбором главного элемента по столбцам
Как вы хотите ввести матрицу? Введите f, если из файла, k - с клавиатуры, r - создать рандомную матрицу: f
Введите путь до файла: try.txt
Считанная из файла матрица:
+----+----+----+
| k1 | k2 | b  |
+----+----+----+
| 5  | 3  | 11 |
| 6  | -4 | -2 |
+----+----+----+
Полученная треугольная матрица:
+-----+-------------------+--------------------+
|  k1 |         k2        |         b          |
+-----+-------------------+--------------------+
|  6  |         -4        |         -2         |
| 0.0 | 6.333333333333334 | 12.666666666666666 |
+-----+-------------------+--------------------+
Определитель: det = 38.0
Полученный вектор неизвестных:
0.9999999999999999 1.9999999999999998
Полученный вектор невязок:
0.0 0.0

Сравним результаты с вычислениями через библиотеку numpy:
Решение системы: 0.9999999999999999 1.9999999999999998
Определитель матрицы: det =  37.99999999999999
```

Вывод

В результате выполнения данной лабораторной работой я познакомился с различными численными методами решения математических задач на примере систем алгебраических уравнений, реализовав на языке программирования Python метод Гаусса с выбором главного элемента по столбцам.
В сравнении с итерационными методами, в данном методе при увеличении порядка системы количество действий стремительно растёт, однако не накапливается погрешность.