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 E.A.



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 𝑎𝑖𝑖 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):

""" Красивый вывод в консоль таблицы коэффициентов """

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 or n > 20:

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 or n > 20:

n = int(input("Порядок матрицы должен быть > 0 и <= 20. 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 or n > 20:

n = int(input("Порядок матрицы должен быть > 0 и <= 20. 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):

""" Решение системы, используя метод Гаусса с выбором главного элемента по столбцам """

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():

""" Начальная функция для определения типа ввода данных от пользователя """

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**

**Изображение выглядит как текст, диаграмма, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.**

**Примеры работы программы**

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

**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. Результаты, полученные моей программой, почти не отличаются от результатов, полученных при вычислении с помощью библиотеки numpy.  
В сравнении с итерационными методами, в данном методе при увеличении порядка системы количество действий стремительно растёт, однако не накапливается погрешность.