МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Лабораторная работа №3

«Метод Якоби решения СЛАУ»

Задание № СЛАУ-04

Выполнил: Снежко Лев Владимирович,

студент 3 курса, 3 группы

Дисциплина: «Численные методы»

Преподаватель: Будник А.М.

Минск, 2024

**Постановка задачи**

Найти решение системы линейных алгебраических уравнений Ax = b с основной матрицей A системы вида:

и столбца свободных членов b вида:

методом Якоби.

Обосновать сходимость метода Якоби, выбрать начальное приближение:

x0 = b

и вычислить количество итераций, необходимых для достижения точности

ε = 1e-5

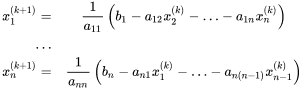
Использовать следующий критерий останова:

**Алгоритм решения**

При решении задачи использовалось значение параметра ε=1e-5

Перед применением алгоритма необходимо проверить условия сходимости метода Якоби. В случае настоящих входных данных (см. раздел «Постановка задачи») выполняется достаточное условие сходимости (подробнее см. раздел «Результаты и их анализ»)

Будем строить последовательность приближений x(k). Выберем первое приближение x(0). По условию лабораторной работы необходимо взять x(0)= b. Остальные приближения x(k) будем считать по формулам:



Эти формулы можно представить в матричном виде. Пусть А – основная матрица системы – представима в виде А = L + D + U, где

L – нижняя треугольная матрица с нулями на главной диагонали

D – диагональная матрица

U – верхняя треугольная матрица с нулями на главной диагонали

Тогда



**Листинг**

import numpy as np

def Jacobi(

        A, b, eps: float = 0.001, x\_init = None) -> list[float]:

    def sum(a, x, j: int) -> float:

        S: float = 0

        for i, (m, y) in enumerate(zip(a, x)):

            if i != j:

                S += m\*y

        return S

    def norm(x, y) -> float:

        return max((abs(x0-y0) for x0, y0 in zip(x, y)))

    if x\_init is None:

        y = [a/A[i][i] for i, a in enumerate(b)]

    else:

        y = x\_init.copy()

    x: list[float] = [-(sum(a, y, i) - b[i])/A[i][i]

                      for i, a in enumerate(A)]

    iterations\_count = 0

    while norm(y, x) > eps:

        for i, elem in enumerate(x):

            y[i] = elem

        for i, a in enumerate(A):

            x[i] = -(sum(a, y, i) - b[i])/A[i][i]

        iterations\_count += 1

    return x, iterations\_count

def ldr\_decomposition(A):

    # Инициализация матриц L, D и R

    n = A.shape[0]

    L = np.zeros\_like(A, dtype=np.float64)

    D = np.zeros\_like(A, dtype=np.float64)

    R = np.zeros\_like(A, dtype=np.float64)

    for i in range(n):

        for j in range(n):

            if i == j:  # Если элемент на главной диагонали

                D[i, j] = A[i, j]

                #L[i, j] = 0  # 1 на главной диагонали L

            elif i > j:  # Если элемент ниже главной диагонали

                L[i, j] = A[i, j]

            elif i < j:  # Если элемент выше главной диагонали

                R[i, j] = A[i, j]

    return L, D, R

A = np.loadtxt(fname='A.txt', dtype=np.float64)

B = np.loadtxt('B.txt', dtype=np.float64)

e = np.eye(N=A.shape[0])

L, D, R = ldr\_decomposition(A)

another\_b = -np.linalg.inv(D) @ (L + R)

norm\_b = np.linalg.norm(another\_b, np.inf)

print(norm\_b)

X, iterations\_count = Jacobi(A, B, eps=1e-5)

x\_np = np.array(X)

error = A @ x\_np - B

print(error)

np.savetxt(X=error, fname='error.txt')

np.savetxt(X=x\_np, fname='x\_result.txt')

with open('iterations.txt', 'w') as file:

    file.write(str(iterations\_count))

**Результаты и их анализ**

На описанных данных (см. раздел «Постановка задачи») алгоритм вернул следующее значение столбца x

X =

В результате работы описанного алгоритма был получен вектор невязки:

R1 = A\*x - b,

R1 =

||R1|| = 2.6108e-6

В процессе проверки свойства диагонального преобладания матрицы A получена пара векторов:

D1 = D2=

где D1 – вектор диагональных элементов, D2 – вектор сумм элементов соответствующих строк исключая диагональные элементы.

Нетрудно видеть, что каждый элемент вектора D1 больше соответствующего ему элемента вектора D2, а значит, матрица А обладает свойством диагонального преобладания. Т.е. выполняется достаточное условие сходимости метода Якоби.

Для достижения точности ε = 1e-5 понадобилось провести 11 итераций метода