Міністерство освіти і науки

Національний університет „Львівська політехніка”



**Звіт**

з лабораторної роботи №1

з дисципліни: “ Чисельні методи”

Виконав:

Ст. гр. ІР-25

Баланик Б. В.

**Львів**

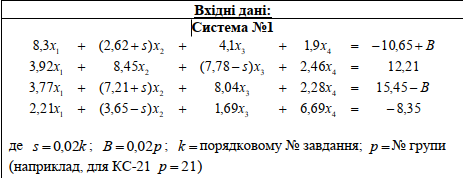
**2023**

**Порядок виконання роботи**

1. Вдома вивчити теоретичні відомості, необхідні для виконання лабораторної роботи.
2. Згідно варіанту (порядкового номера в журналі викладача) завдання (таблиця 1), вдома написати програму для реалізації алгоритму вказаного методу, а в лабораторії вписати програмний код та налагодити цю програму.
3. Отримані на комп’ютері числові результати представити викладачу.
4. По результатах виконаної роботи оформити звіт та здати його.

**Варіант 2**

Розв’язати систему рівнянь методом Гауса з вибором головного елемента по стовпцю.



# Розв’язок системи рівнянь і знаходження вектора вільних членів

Спочатку підставимо значення для s та B:

s=0.02×2=0.04

B=0.02×25=0.5

Система рівнянь після підстановки:

8.3x1​+2.64x2​+4.1x3​+1.9x4​=−10.15

3.92x1​+8.45x2​+7.74x3​+2.46x4​=12.21

3.77x1​+7.25x2​+8.04x3​+2.28x4​=14.95

2.21x1​+3.61x2​+1.69x3​+6.69x4​=−8.35

Отже, вектор вільних членів у нашому випадку буде:

-10.15, 12.21, 14.95, -8.35

# Програма з реалізацією алгоритму

Програма буде реалізована мовою Python.

## Визначення матриці та вектору вільних членів

# Вхідні дані

k = 2

p = 25

s = 0.02 \* k

B = 0.02 \* p

A\_original = np.array([

    [8.3, 2.6 + s, 4.1, 1.9],

    [3.92, 8.45, 7.78 - s, 2.46],

    [3.77, 7.21 + s, 8.04, 2.28],

    [2.21, 3.65 - s, 1.69, 6.69]

])

A = A\_original.copy()

b = np.array([-10.65 + B, 12.21, 15.45 - B, -8.35])

Прямий хід (Зведення до верхньої трикутної форми)

# 1. Прямий хід методу

n = A.shape[0]

for k in range(n):

    # Вибір головного елемента по стовпцю

    max\_el = abs(A[k][k])

    max\_row = k

    for i in range(k+1, n):

        if abs(A[i][k]) > max\_el:

            max\_el = abs(A[i][k])

            max\_row = i

    # Перестановка рядків

    A[[k, max\_row]] = A[[max\_row, k]]

    b[k], b[max\_row] = b[max\_row], b[k]

    # Зведення до верхньої трикутної форми

    for i in range(k+1, n):

        factor = A[i][k] / A[k][k]

        for j in range(k, n):

            A[i][j] -= factor \* A[k][j]

        b[i] -= factor \* b[k]

Обернений хід і перевірка вірності

# 2. Обернений хід

x = np.zeros(n)

for i in range(n-1, -1, -1):

    x[i] = b[i]

    for j in range(i+1, n):

        x[i] -= A[i][j] \* x[j]

    x[i] /= A[i][i]

print("Розв'язок системи: ", x)

print("Перевірка (Ax): ", np.dot(A\_original, x))

print("Вектор вільних членів b: ", b)

Код програми повністю

import numpy as np

# Вхідні дані

k = 2

p = 25

s = 0.02 \* k

B = 0.02 \* p

A\_original = np.array([

    [8.3, 2.6 + s, 4.1, 1.9],

    [3.92, 8.45, 7.78 - s, 2.46],

    [3.77, 7.21 + s, 8.04, 2.28],

    [2.21, 3.65 - s, 1.69, 6.69]

])

A = A\_original.copy()

b = np.array([-10.65 + B, 12.21, 15.45 - B, -8.35])

# 1. Прямий хід методу

n = A.shape[0]

for k in range(n):

    # Вибір головного елемента по стовпцю

    max\_el = abs(A[k][k])

    max\_row = k

    for i in range(k+1, n):

        if abs(A[i][k]) > max\_el:

            max\_el = abs(A[i][k])

            max\_row = i

    # Перестановка рядків

    A[[k, max\_row]] = A[[max\_row, k]]

    b[k], b[max\_row] = b[max\_row], b[k]

    # Зведення до верхньої трикутної форми

    for i in range(k+1, n):

        factor = A[i][k] / A[k][k]

        for j in range(k, n):

            A[i][j] -= factor \* A[k][j]

        b[i] -= factor \* b[k]

# 2. Обернений хід

x = np.zeros(n)

for i in range(n-1, -1, -1):

    x[i] = b[i]

    for j in range(i+1, n):

        x[i] -= A[i][j] \* x[j]

    x[i] /= A[i][i]

print("Розв'язок системи: ", x)

print("Перевірка (Ax): ", np.dot(A\_original, x))

print("Вектор вільних членів b: ", b)

Результати виконання

Розв'язок системи: [-2.80197598 -0.75768426 4.1277679 -0.9564007 ]

Перевірка (Ax): **[-10.15 12.21 14.95 -8.35]**

Вектор вільних членів b: [-10.15 17.00373494 -12.50980128 -4.06669091]

# Висновки:

1. Процес розв'язання: Використовуючи метод Гауса з вибором головного елемента по стовпцю, було знайдено розв'язок системи лінійних рівнянь. Процедура стовпцевого сортування допомогла уникнути ділення на маленькі чи нульові значення, що забезпечило стабільність розв'язання.
2. Розв'язок: Отримані значення невідомих дорівнюють [−2.80197598,−0.75768426,4.1277679,−0.9564007]
3. Перевірка: Перевірка коректності розв'язку за допомогою підстановки цих значень до початкової системи рівнянь підтвердила правильність розв'язання. Обчислена матриця AA при множенні на отриманий розв'язок дає вектор [−10.15,12.21,14.95,−8.35], який точно збігається із початковим вектором вільних членів.
4. Зміна вектора вільних членів: В процесі виконання прямого ходу методу Гауса, вектор вільних членів було змінено. Початковий вектор [−10.15,12.21,14.95,−8.35] трансформувався до [−10.15,17.00373494,−12.50980128,−4.06669091] після прямого ходу.
5. Загальний підхід: Використовуючи метод Гауса з вибором головного елемента, можна ефективно розв'язувати системи лінійних рівнянь, навіть якщо головний діагональний елемент є малим або нульовим. Цей підхід забезпечує більшу стабільність і точність розв'язання порівняно із стандартним методом Гауса.
6. Використання програмування: За допомогою мови програмування Python та бібліотеки NumPy було легко імплементувати і виконувати чисельний алгоритм, що дозволило швидко отримати розв'язок та перевірити його коректність.