НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет прикладної математики
Кафедра прикладної математики

Звіт
із лабораторної роботи No1
з кредитного модуля «АЛГОРИТМИ І СИСТЕМИ КОМП'ЮТЕРНОЇ МАТЕМАТИКИ 1. МАТЕМАТИЧНІ АЛГОРИТМИ»
для студентів
спеціальності 113 «Прикладна математика»

Виконала:
Студентка КМ-81
Верзун П. В.

Керівник:
Ст. Викладач
Ліскін В.О.

Київ – 2021

Мета роботи – вивчити правила використання програмних засобів для факторизації матриць і розв’язання систем лінійних алгебраїчних рівнянь, розв’язати задану систему рівнянь із використанням методів:

- Матричний метод (octave)
- LU-факторизації (python)

Кожний метод реалізувати в якому середовищі вказано в дужках вище. Провести порівняльний аналіз вивчених чисельних методів розв’язання СЛАР.

ЗАВДАННЯ

Варіант	Матриця коефіцієнтів системи А				Вектор вільних членів В
2	0,31	0,14	0,30	0,27	1,02
	0,26	0,32	0,18	0,24	1,00
	0,61	0,22	0,20	0,31	1,34
	0,40	0,34	0,36	0,17	1,27

ТЕОРЕТИЧНІ ВІДОМОСТІ

Матричний метод *(метод оберненої матриці)*

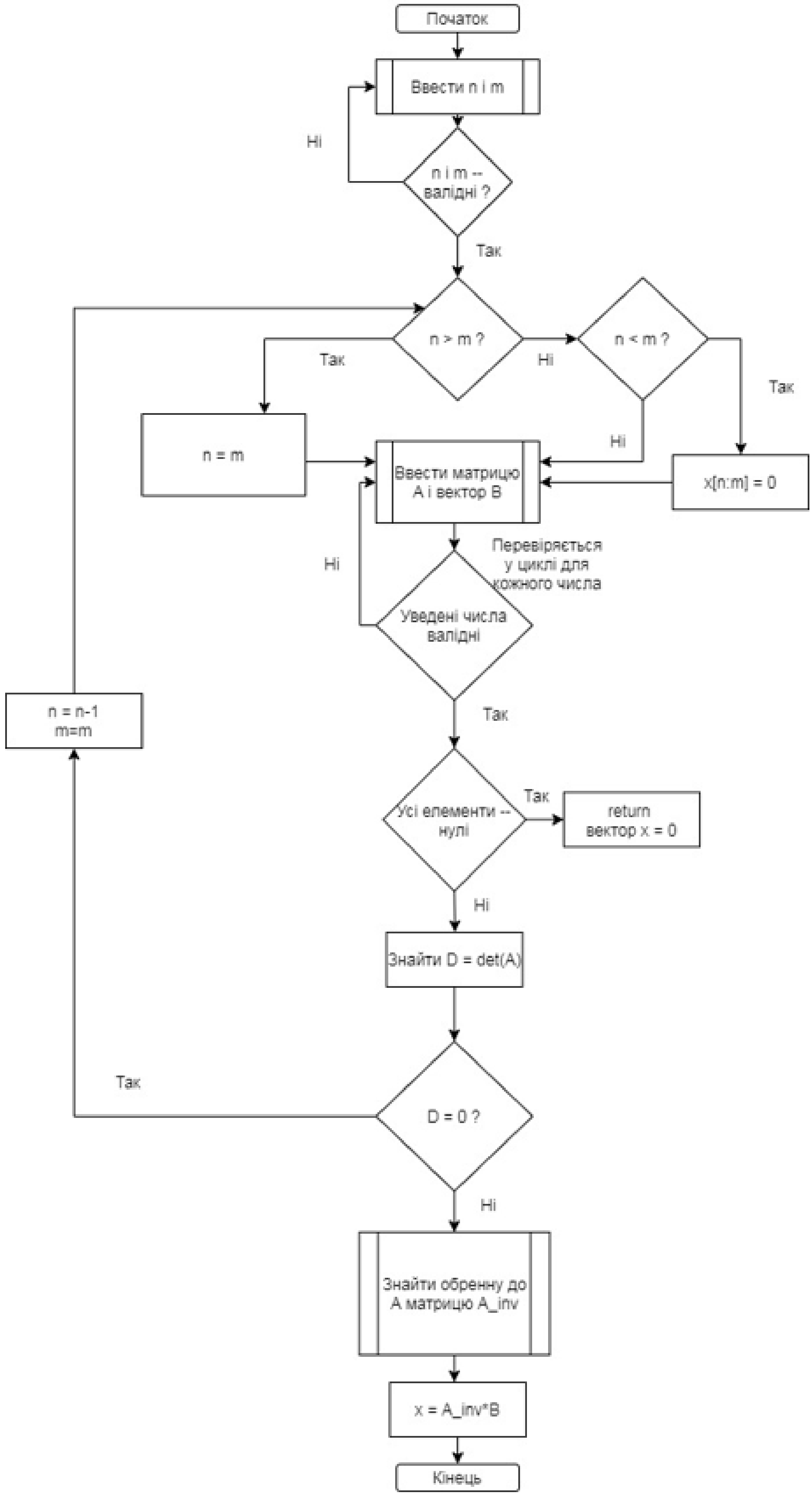
Зобразимо задану систему рівнянь в матричному вигляді:

$$A \cdot \overline{x} = \overline{b}$$

Очевидно, що при множенні зліва обох частин рівності на обернену матрицю *A,* отримуємо відповідь:

$$\overline{x} = A^{-1} \cdot \overline{b}$$

Блок-схема алгоритму:


Початок
Ввести n і m
Ні
n і m -- валідні ?
Так
n > m ?
Так
Ні
n < m ?
Так
n = m
Ні
Ввести матрицю A і вектор B
x[n:m] = 0
Перевіряється у циклі для кожного числа
Ні
Уведені числа валідні
Так
n = n-1
m=m
Усі елементи -- нулі
Так
return
вектор x = 0
Ні
Знайти D = det(A)
Так
D = 0 ?
Ні
Знайти обренну до A матрицю A_inv
x = A_inv*B
Кінець

LU-факторизація

Даний метод є модифікацією методу Гауса. Він потребує $\frac{2n^3}{3}$ арифметичних операцій. Ідея методу полягає в представленні матриці *A* у вигляді добуткі нижньо- та верхньодіагональних матриць L та U з одиницями на головних діагоналях. Елементи даних матриць обчислюються за формулами:

$$l_{ij} = a_{ij} - \sum_{k=0}^{j-1} l_{ik} u_{kj} \ (i \geq \ j)$$

$$u_{ij} = \frac{1}{l_{ii}} \left[a_{ij} - \sum_{k=0}^{i-1} l_{ik} u_{kj} \right] \ (i < j)$$

Отримавши *LU* представлення матриці *A,* легко побачити, що з умови:

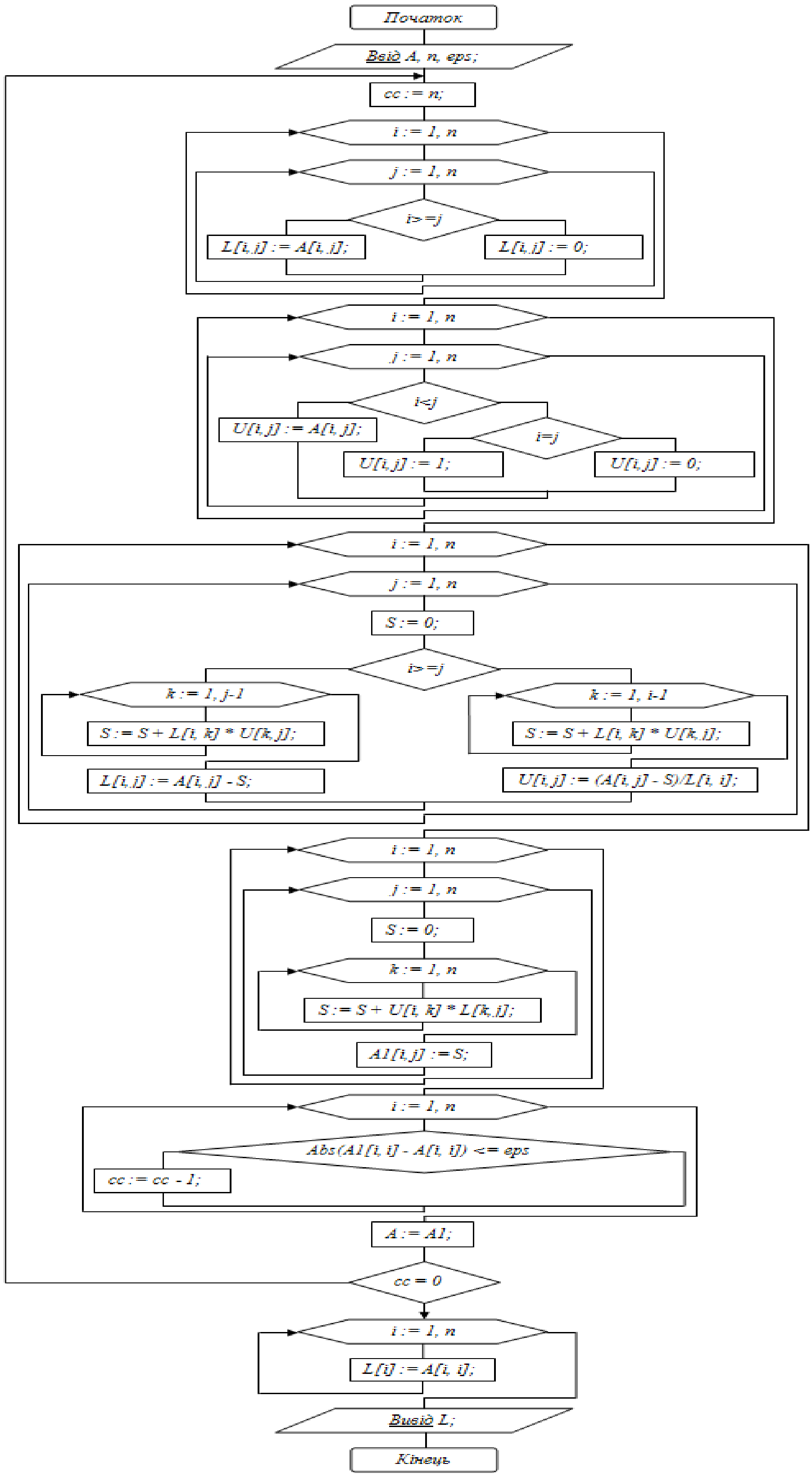
$$A\overline{x} = \overline{b} \Leftrightarrow LU\overline{x} = \overline{b}$$

слідує, що задача розв'язується в два кроки:

1. $L\overline{y} = \overline{b}$
2. $U\overline{x} = \overline{y}$

При цьому, завдяки трикутному вигляду матриць, пошук розв'язку обох рівнянь є тривіальною задачею.

Блок-схема алгоритму:



Початок
Ввід A, n, eps;
cc := n;
i := 1, n
j := 1, n
i>=j
L[i, j] := A[i, j];
L[i, j] := 0;
i := 1, n
j := 1, n
i<j
U[i, j] := A[i, j];
i=j
U[i, j] := 1;
U[i, j] := 0;
i := 1, n
j := 1, n
S := 0;
i>=j
k := 1, j-1
S := S + L[i, k] * U[k, j];
k := 1, i-1
S := S + L[i, k] * U[k, j];
L[i, j] := A[i, j] - S;
U[i, j] := (A[i, j] - S)/L[i, i];
i := 1, n
j := 1, n
S := 0;
k := 1, n
S := S + U[i, k] * L[k, j];
A1[i, j] := S;
i := 1, n
Abs(A1[i, i] - A[i, i]) <= eps
cc := cc - 1;
A := A1;
cc = 0
i := 1, n
L[i] := A[i, i];
Вивід L;
Кінець

ХІД РОБОТИ

В модулі scipy мови програмування python надано функцію linalg.solve() для знаходження розв’язку СЛАР. Дана функція наслідується з бібліотеки LAPACK та використовує метод LU-факторизації, тож скористаємось нею для розв’язання поставленої задачі. Для її використання необхідно гарантувати квадратний вигляд матриці, її невиродженість та повноту рангу.

Матричний метод реалізуємо за допомогою Octave.

1. LU-факторизація на python

```
import numpy as np
from scipy.linalg import solve
from prettytable import PrettyTable

A = np.array([
        [0.31, 0.14, 0.30, 0.27],
        [0.26, 0.32, 0.18, 0.24],
        [0.61, 0.22, 0.20, 0.31],
        [0.40, 0.34, 0.36, 0.17]
        ])

b = np.array([1.02, 1.00, 1.34, 1.27]).reshape((4, 1))



def view_matrix(matrix):
        table = PrettyTable(header=False)
        table.add_rows(matrix)
        print(table)

        return None

def check_rank(matrix):
        rank = np.linalg.matrix_rank(matrix)
        shape = matrix.shape[0]

        if rank == shape:
                print("Ранг матриці є повним та дорівнює {}\n".format(rank))
```

```
            return True

        else:
            print("Ранг матриці є невпоним")
            return False

def get_det(matrix):
        det = np.linalg.det(matrix)

        if det != 0:
            print("Матриця є навиродженою та її визначник дорівнює {}\n".format(det))
            return True

        else:
            print("Матриця є виродженою")
            return False


if __name__ == "__main__":

        print("\nМатриця A:")
        view_matrix(A)
        print("\nВектор вільних членів b:")
        view_matrix(b)

        if check_rank(A) and get_det(A):
            x = solve(A, b)
            print("Розв'язок:")
            table = PrettyTable(header=False)
            table.add_rows([x])
            print(table)
            print("\nA * x = ")
            print(np.dot(A, x).reshape((4,1)))
```

Результат виконання програми:

```
Матриця A:
+------+------+------+------+
| 0.31 | 0.14 | 0.3  | 0.27 |
| 0.26 | 0.32 | 0.18 | 0.24 |
| 0.61 | 0.22 | 0.2  | 0.31 |
| 0.4  | 0.34 | 0.36 | 0.17 |
+------+------+------+------+

Вектор вільних членів b:
+------+
| 1.02 |
| 1.0  |
| 1.34 |
| 1.27 |
+------+
Ранг матриці є повним та дорівнює 4

Матриця є навиродженою та її визначник дорівнює 0.005266240000000001

Розв'язок:
+------+------+------+------+
| [1.] | [1.] | [1.] | [1.] |
+------+------+------+------+

A * x =
[[1.02]
 [1.  ]
 [1.34]
 [1.27]]
```

2. Матричний метод на Octave

```
disp("Матриця A системи рівнянь:")
A = [0.31, 0.14, 0.30, 0.27;
     0.26, 0.32, 0.18, 0.24;
     0.61, 0.22, 0.20, 0.31;
     0.40, 0.34, 0.36, 0.17]
disp("Вектор вільних членів b системи рівнянь:")
b = [1.02; 1.00; 1.34; 1.27]
disp("Розв'язок x системи:")
x = A\b
disp("Перевірка A*x:")
disp(A*x)
```

Результат виконання програми:

```
>> lab1

Матриця A системи рівнянь:
A =

   0.31000   0.14000   0.30000   0.27000
   0.26000   0.32000   0.18000   0.24000
   0.61000   0.22000   0.20000   0.31000
   0.40000   0.34000   0.36000   0.17000

Вектор вільних членів b системи рівнянь:
b =

   1.0200
   1.0000
   1.3400
   1.2700

Розв'язок x системи:
x =

   1.00000
   1.00000
   1.00000
   1.00000

Перевірка A*x:
   1.0200
   1.0000
   1.3400
   1.2700
```

Отже, обома методами було отримано правильні результати з абсолютною точністю.

ВИСНОВКИ

В ході виконання даної лабораторної роботи було засвоєно методи розв'язання СЛАР за допомогою ЕОМ, а саме способи програмної реалізації методу LU-факторизації та матричного методу із використанням python та Octave. Отримані обома методами результати розв'язання поставленої задачі є ідентичними.