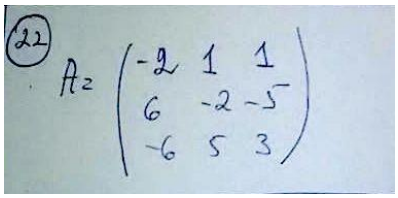
LU-розклад

Підготувала:
Студентка ПМІ-23
Шувар Софія

Мета: Реалізувати алгоритм LU-розкладу. Продемонструвати роботу програми на конкретному прикладі.

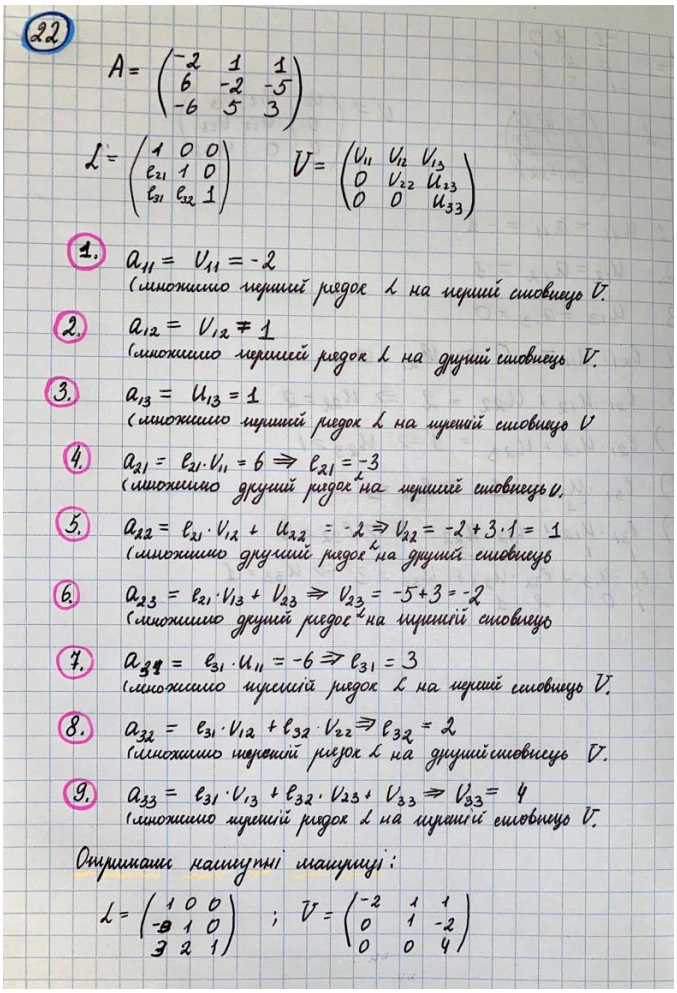
Завдання:



Хід роботи:

1. Розбити Матрицю, подану у завданні на дві матриці L і U, як у запропонованому прикладі.
2. Реалізувати алгоритм LU-розкладу матриці на довільний мові програмування.
3. Продемонструвати результати роботи алгоритму на конкретному прикладі.

Розв'язок системи:



Реалізація алгоритму:

Для реалізації алгоритму я обрала мову програмування Python. Алгоритм написаний з використанням бібліотеки numpy.

Функція user_input(вхідні дані):

Функція наповнює матрицю a введену користувачем з клавіатури.

```
1   import numpy as np
2
3
4   def user_input():
5       n = int(input("Enter matrix size: "))
6       print("Enter matrix:")
7       a = np.array([input().strip().split() for _ in range(n)], int)
8       return a
9
```

Функція lu(реалізація алгоритму):

Аргументи: A – початкова матриця;

Мета алгоритму: представити матрицю A у вигляді добутку нижньої трикутної матриці L та верхньої трикутної матриці U. Позначимо як l_{ij}, u_{ij}, a_{ij} елементи матриць L,U та A відповідно. Тоді маємо:

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

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

Алгоритм складається з наступних частин:

Рядок 12 – отримуємо n – розмір матриці;

Рядок 13 – оголошуємо матрицю U та наповнюємо матрицю її значеннями матриці A;

Рядок 14 – оголошуємо матрицю L розміру n та наповнюємо її нулями.

Рядок 15 – заповнюємо діагональ матриці L одиницями.

Рядок 17 – i = 1, … n розпочинаємо цикл довжини n.

Рядки[17-22] – Цикл від i до j, присвоюємо Lji значення Uji/Uii, а Uj значення (Uj–$ljiUi$) (де Ui,Uj означає i-ті та j-ті рядки матриці)

Рядок 24 – повертаємо отримані трикутні матриці L та U.

```python
def lu(A):
    n = A.shape[0]
    U = A.copy().astype('float64')
    L = np.zeros((n, n), int).astype('float64')
    np.fill_diagonal(L, 1)

    for i in range(n):
        factor = U[i + 1:, i] / U[i, i]
        factor = factor.astype('float64')
        L[i + 1:, i] = factor
        U[i + 1:] -= factor[:, np.newaxis] * U[i]

    return L, U
```

Функція check_result(перевірка результатів):

Аргументи: A – початкова матриця; L – нижня трикутна матриця; U - верхня трикутна матриця.

Функція перевіряє чи добуток матриць L та U є рівним початковій матриці A.

```python
def check_result(A, L, U):
    A_new = L @ U
    return np.array_equal(A, A_new)
```

Функція mat_print(вивід матриці):

Аргументи: A – початкова матриця;

Функція виводить матрицю у гарному вигляді.

```python
def mat_print(A):
    col_maxes = [max([len(("{:g}").format(x)) for x in col]) for col in A.T]
    for x in A:
        for i, y in enumerate(x):
            print(("{:"+str(col_maxes[i])+"g}").format(y), end="  ")
        print("")
```

Функція print_result(вивід):

Основна функція, що викликає усі попередні функції та виводить їх.

```
38
39      def print_result():
40          a = user_input()
41          L, U = lu(a)
42          print("LU decomposition:")
43          print("L: \t")
44          mat_print(L)
45          print("\nU: \t")
46          mat_print(U)
47          print()
48          if check_result(a, L, U):
49              print("Matrix A = Matrix L * Matrix U")
50          else:
51              print("Sth went wrong!")
52
```

main():

```
53
54  if __name__ == "__main__":
55      print_result()
56
```

Результати роботи програми на конкретному прикладі:

```
/Users/sophiyca/venv/bin/python /Users/sophiyca/PycharmProjects/Numerical_analysis/01_Gaussian_elimination/lu_decomposition.py
Enter matrix size: 3
Enter matrix:
-2 1 1
6 -2 -5
-6 5 3
LU decomposition:
L:
 1  0  0
-3  1  0
 3  2  1

U:
-2  1   1
 0  1  -2
 0  0   4

Matrix A = Matrix L * Matrix U

Process finished with exit code 0
```

Висновок

Виконуючи дану практичну роботу, я навчилась розкладати матрицю на верхню та нижні трикутні матриці за допомогою LU-розкладу та реалізувала алгоритм розв`язання на мові Python з використанням можливостей бібліотеки numpy.