NAME:- KETAN DILIP ATTARDE

REG NO:- 24-27-06

MTECH DATA SCIENCE

# LU Decomposition and System Solving Algorithm

## LU Decomposition

### Input

- Square matrix $A$ of size $n \times n$

### Output

- Lower triangular matrix $L$ with $L_{ii} = 1$
- Upper triangular matrix $U$

### Steps

1. **Initialization:**
    - Initialize $L$ as the identity matrix of size $n \times n$.
    - Initialize $U$ as a zero matrix of the same size as $A$.
2. **Decomposition:**
    - For each column $k$ from 0 to $n - 1$:
        - Update the $k$-th row of $U$ to eliminate entries below the diagonal.
        - Update the $k$-th column of $L$ to eliminate entries above the diagonal.

- Ensure $L_{ii} = 1$ by dividing the entries in the column by the pivot element $U[k, k]$.

3. **Output:**
    - Return the computed matrices $L$ and $U$.

# Solving System of Equations

## Input

- Coefficient matrix $A$ of size $n \times n$
- Constant vector $b$ of size $n$

## Output

- Solution vector $x$

## Steps

1. **LU Decomposition:**
    - Use the LU decomposition algorithm to compute matrices $L$ and $U$ from $A$.
2. **Forward Substitution:**
    - Solve the system $Ly = b$ for $y$ using forward substitution.
3. **Backward Substitution:**
    - Solve the system $Ux = y$ for $x$ using backward substitution.
4. **Output:**
    - Return the computed solution vector $x$.

```
import numpy as np
```

```
def lu_decomposition(A):
    n = len(A)
    L = np.eye(n)
    U = np.zeros_like(A, dtype=float)

    for k in range(n):
        U[k, k:] = A[k, k:] - L[k, :k] @ U[:k, k:]
        L[k+1:, k] = (A[k+1:, k] - L[k+1:, :k] @ U[:k, k]) / U[k, k]

    return L, U


def solve_system_with_lu_decomposition(A, b):
    L, U = lu_decomposition(A)

    # Forward Substitution (Ly = b)
    y = np.linalg.solve(L, b)

    # Backward Substitution (Ux = y)
    x = np.linalg.solve(U, y)

    return x


# Example Usage
A = np.array([[2, 1, -1],
              [-3, -1, 2],
              [-2, 1, 2]]).astype(np.float64)

b = np.array([8, -11, -3]).astype(np.float64)

solution = solve_system_with_lu_decomposition(A, b)
print("Solution:", solution)
```

```
Solution: [ 2.  3. -1.]
```