NAME:- KETAN DILIP ATTARDE

REG NO:- 24-27-06

MTECH DATA SCIENCE

Cholesky Decomposition and System Solving Algorithm

Cholesky Decomposition

Input

- Symmetric positive definite matrix A of size $n \times n$

Output

- Lower triangular matrix L such that $A = LL^T$

Steps

1. **Initialization:**
   - Initialize a zero matrix L of the same size as A.
2. **Decomposition:**
   - For each row i from 0 to $n-1$:
     - For each column j from 0 to i:
       - If $i = j$, compute L_{ii} using $L_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2}$.
       - If $i \neq j$, compute L_{ij} using $L_{ij} = \frac{1}{L_{jj}} \left(A_{ij} - \sum_{k=1}^{j-1} L_{ik} L_{jk} \right)$.

3. **Output:**
    - Return the computed lower triangular matrix L.

Solving System of Equations

Input

- Symmetric positive definite matrix A of size $n \times n$
- Constant vector b of size n

Output

- Solution vector x

Steps

1. **Cholesky Decomposition:**
    - Use the Cholesky decomposition algorithm to compute the lower triangular matrix L from A.
2. **Forward Substitution:**
    - Solve the system $Ly = b$ for y using forward substitution.
3. **Backward Substitution:**
    - Solve the system $L^T x = y$ for x using backward substitution.
4. **Output:**
    - Return the computed solution vector x.

```
import numpy as np
```

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

    for i in range(n):
        for j in range(i+1):
            if i == j:
                L[i, i] = np.sqrt(A[i, i] - np.sum(L[i, :i]**2))
            else:
                L[i, j] = (A[i, j] - np.sum(L[i, :i] * L[j, :i])) / L[j, j]

    return L


def solve_system_with_cholesky_decomposition(A, b):
    L = cholesky_decomposition(A)

    # Solve Ly = b using forward substitution
    y = np.linalg.solve(L, b)

    # Solve L^T x = y using backward substitution
    x = np.linalg.solve(L.T, y)

    return x


# Example Usage
A = np.array([[4, 12, -16],
              [12, 37, -43],
              [-16, -43, 98]]).astype(np.float64)

b = np.array([1, 2, 3]).astype(np.float64)
print("The augmented matrix is:-")
print(np.column_stack((A, b)))
print("-"*50)
solution = solve_system_with_cholesky_decomposition(A, b)
print("The solution is:- ")
```

```
print(solution)
```

```
The augmented matrix is:-
[[  4.  12. -16.   1.]
 [ 12.  37. -43.   2.]
 [-16. -43.  98.   3.]]
--------------------------------------------------
The solution is:-
[28.58333333 -7.66666667  1.33333333]
```