**Penjelasan Source Code Metode Dekomposisi Crout Pada Python**

Nama: Muhammad Nio Hastungkoro

NIM: 2112012214015

Matkul: Metode Numerik (A)

Source Code:

|  |
| --- |
| import numpy as np  def LU\_decomposition(A):  n = len(A)  L = np.eye(n)  U = A.astype(float) # Mengonversi tipe data matriks A ke float  for k in range(n-1):  for i in range(k+1, n):  if U[k, k] == 0:  return "Pembagian dengan nol terjadi, tidak dapat dilanjutkan"  factor = U[i, k] / U[k, k]  L[i, k] = factor  U[i, k:] -= factor \* U[k, k:]  return L, U  def solve\_linear\_eq\_LU\_gauss(A, b):  L, U = LU\_decomposition(A)  y = np.linalg.solve(L, b)  x = np.linalg.solve(U, y)  return x  # Testing  A = np.array([[2, 3], [4, 5]], dtype=float) # Mengonversi tipe data matriks A ke float  b = np.array([6, 7], dtype=float) # Mengonversi tipe data vektor b ke float  print("Solusi dengan metode dekomposisi LU Gauss:", solve\_linear\_eq\_LU\_gauss(A, b)) |

Penjelasan:

Mari kita jabarkan alur kode tersebut secara rinci:

* Def crout\_decomposition(A):

Ini adalah deklarasi fungsi crout\_decomposition yang menerima sebuah parameter, yaitu matriks \( A \) (diwakili oleh A). Fungsi ini bertujuan untuk melakukan dekomposisi Crout pada matriks \( A \).

* n = len(A)  
  Baris ini menghitung ukuran matriks \( A \) dan menyimpannya dalam variabel n.
* L = np.zeros((n, n)) dan U = np.zeros((n, n)):

Keduabaris ini membuat matriks nol \( L \) dan \( U \) dengan ukuran \( n \times n \) menggunakan fungsi np.zeros() dari NumPy. Matriks ini akan digunakan untuk menyimpan elemen-elemen matriks \( L \) dan \( U \) dalam dekomposisi Crout.

* for j in range(n):

Ini adalah loop yang berjalan dari 0 hingga \( n-1 \), iterasi ini untuk setiap kolom matriks \( A \).

* U[j, j = 1

Baris ini mengatur elemen diagonal utama matriks \( U \) pada kolom \( j \) menjadi 1, karena setiap elemen diagonal utama matriks \( U \) adalah 1.

* for i in range(j, n):

Ini adalah loop dalam loop luar yang berjalan dari \( j \) hingga \( n-1 \), iterasi ini untuk setiap baris pada atau di bawah diagonal utama matriks \( A \).

* L[i, j] = A[i, j] - np.dot(L[i, :j], U[:j, j])

Baris ini menghitung elemen matriks \( L \) pada baris \( i \) dan kolom \( j \) dengan menggunakan formula dekomposisi Crout.

* 8or i in range(j+1, n):

Ini adalah loop dalam loop luar yang berjalan dari \( j+1 \) hingga \( n-1 \), iterasi ini untuk setiap baris di atas diagonal utama matriks \( A \).

* U[j, i] = (A[j, i] - np.dot(L[j, :j], U[:j, i])) / L[j, j]

Baris ini menghitung elemen matriks \( U \) pada baris \( j \) dan kolom \( i \) dengan menggunakan formula dekomposisi Crout.

* return L, U

Fungsi mengembalikan matriks \( L \) dan matriks \( U \).

* def solve\_linear\_eq\_crout(A, b):

Ini adalah deklarasi fungsi solve\_linear\_eq\_crout yang menerima dua parameter, yaitu matriks \( A \) (diwakili oleh A) dan vektor hasil \( b \) (diwakili oleh b). Fungsi ini bertujuan untuk menyelesaikan sistem persamaan linear \( Ax = b \) dengan menggunakan metode dekomposisi Crout.

* L, U = crout\_decomposition(A)

Baris ini memanggil fungsi crout\_decomposition untuk mendapatkan matriks \( L \) dan matriks \( U \) dari matriks \( A \).

* y = np.linalg.solve(L, b)

Baris ini menggunakan fungsi np.linalg.solve() untuk menyelesaikan sistem \( Ly = b \) dengan matriks \( L \) dan vektor \( b \). Hasilnya disimpan dalam variabel y.

* 1x = np.linalg.solve(U, y)

Baris ini menggunakan fungsi np.linalg.solve() untuk menyelesaikan sistem \( Ux = y \) dengan matriks \( U \) dan vektor \( y \). Hasilnya adalah vektor solusi \( x \), yang disimpan dalam variabel x.

* return x

Fungsi mengembalikan vektor solusi \( x \).

* A = np.array([[1, 2], [3, 4]])

Inisialisasi matriks koefisien \( A \) dengan nilai [[1, 2], [3, 4]].

* b = np.array([5, 6])

Inisialisasi vektor hasil \( b \) dengan nilai [5, 6].

* print("Solusi dengan metode dekomposisi Crout:", solve\_linear\_eq\_crout(A, b))

Baris ini memanggil fungsi solve\_linear\_eq\_crout dengan matriks koefisien \( A \) dan vektor hasil \( b \) sebagai argumen, kemudian mencetak solusi yang diperoleh menggunakan metode dekomposisi Crout.