IMPLEMENTASI SISTEM PERSAMAAN LINEAR

Nama : Dimas Gary Irawan
NIM : 21120122140164
Metode Numerik – D

Link Github: https://github.com/garyirawan/Implementasi-Sistem-Persamaan-Linear_Dimas-Gary-Irawan_21120122140164

Diinginkan aplikasi untuk mencari solusi sistem persamaan linear masing-masing menggunakan:

1. metode matriks balikan
2. metode dekomposisi LU Gauss
3. metode dekomposisi Crout

Diberikan suatu studi kasus sebagai persoalan yang akan diselesaikan dengan ketiga metode di atas.

Studi kasus:

Sebuah perusahaan memiliki tiga divisi yang berbeda: produksi, pemasaran, dan keuangan. Masing-masing divisi memiliki kontribusi tertentu terhadap pendapatan perusahaan. Perusahaan ingin menentukan kontribusi optimal dari setiap divisi untuk memaksimalkan total pendapatannya.

Didapat persamaan untuk kontribusi masing-masing divisi adalah sebagai berikut:

$3P + 2M - K = 10000$

$2P - 4M + 2K = 8000$

$P + 3M + 2K = 15000$

Dalam bentuk matriks:

$$\begin{bmatrix} 3 & 2 & -1 \\ 2 & -4 & 2 \\ 1 & 3 & 3 \end{bmatrix} \begin{bmatrix} P \\ M \\ K \end{bmatrix} = \begin{bmatrix} 10000 \\ 8000 \\ 15000 \end{bmatrix}$$

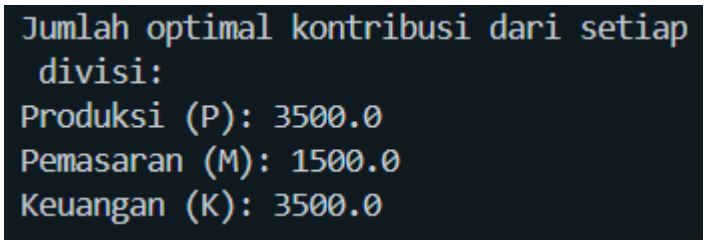
Jawab:

1. Metode Matriks Balikan

Kode:

```
import numpy as np
A = np.array([[3, 2, -1],
              [2, -4, 2],
              [1, 3, 2]])
B = np.array([10000, 8000, 15000])
A_inv = np.linalg.inv(A)
X = np.dot(A_inv, B)
print("Jumlah optimal kontribusi dari setiap divisi:")
print(f"Produksi (P): {X[0]}")
print(f"Pemasaran (M): {X[1]}")
print(f"Keuangan (K): {X[2]}")
```

Hasil:



Analisa:

1. **Matriks Koefisien dan Vektor Hasil**: Pertama, kita mendefinisikan matriks koefisien A dari persamaan linier dan vektor hasil B. Matriks A adalah matriks koefisien yang menghubungkan variabel-variabel P, M, dan K, sedangkan B adalah vektor hasil dari persamaan linier.
2. **Matriks Balikan**: Kita menghitung invers dari matriks koefisien A menggunakan fungsi `np.linalg.inv(A)`. Ini menghasilkan matriks balikan A^{-1}, yang akan kita gunakan untuk menyelesaikan sistem persamaan linear.
3. **Penyelesaian Persamaan**: Kita menggunakan matriks balikan A^{-1} yang telah kita hitung sebelumnya dan mengalikannya dengan vektor hasil B menggunakan fungsi `np.dot(A_inv, B)`. Ini memberikan solusi X dari sistem persamaan linear.
4. **Menampilkan Hasil**: Hasil akhirnya adalah solusi dari sistem persamaan linear, yang merupakan jumlah optimal kontribusi dari setiap divisi. Kita mencetak hasil ini untuk ditampilkan kepada pengguna.

Singkatnya kode di atas melakukan langkah-langkah berikut:

1. Menghitung matriks balikan dari matriks koefisien *A*.

2. Mengalikan matriks balikan dengan vektor hasil *B* untuk mendapatkan solusi sistem persamaan linear.

3. Menampilkan hasil solusi, yaitu jumlah optimal kontribusi dari setiap divisi.

Implementasi ini adalah implementasi sederhana dari metode matriks balikan dalam Python menggunakan NumPy.

2. Metode Dekomposisi LU Gauss

Kode:

```
import numpy as np
from scipy.linalg import lu_factor, lu_solve
A = np.array([[3, 2, -1],
              [2, -4, 2],
              [1, 3, 2]])
B = np.array([10000, 8000, 15000])
LU, piv = lu_factor(A)
X = lu_solve((LU, piv), B)
print("Jumlah optimal kontribusi dari setiap divisi:")
print(f"Produksi (P): {X[0]}")
print(f"Pemasaran (M): {X[1]}")
print(f"Keuangan (K): {X[2]}")
```

Hasil:

```
Jumlah optimal kontribusi dari setiap
 divisi:
Produksi (P): 3500.0
Pemasaran (M): 1500.0
Keuangan (K): 3500.0
```

Analisa:

1. **Matriks Koefisien dan Vektor Hasil**: Kita mendefinisikan matriks koefisien A dari persamaan linear dan vektor hasil B. Matriks A adalah matriks koefisien yang menghubungkan variabel-variabel P, M, dan K, sedangkan B adalah vektor hasil dari persamaan linear.
2. **Dekomposisi LU**: Kita menggunakan fungsi `lu_factor` dari modul `scipy.linalg` untuk melakukan dekomposisi matriks koefisien A menjadi dua matriks L dan U, di mana L adalah matriks segitiga bawah dan U adalah matriks segitiga atas.

3. **Penyelesaian Persamaan**: Setelah dekomposisi dilakukan, kita menggunakan fungsi `lu_solve` untuk menyelesaikan sistem persamaan linear dengan matriks segitiga L dan U yang telah dihasilkan dari langkah sebelumnya.
4. **Menampilkan Hasil**: Hasil akhirnya adalah solusi dari sistem persamaan linear, yang merupakan jumlah optimal kontribusi dari setiap divisi. Kita mencetak hasil ini untuk ditampilkan kepada pengguna.

Kode tersebut melakukan langkah-langkah berikut:

1. Melakukan dekomposisi LU dari matriks koefisien A.

2. Menyelesaikan sistem persamaan linear menggunakan matriks segitiga L dan U yang telah didekomposisi.

3. Menampilkan hasil solusi, yaitu jumlah optimal kontribusi dari setiap divisi.

Implementasi dengan metode dekomposisi LU (LU Gauss) ini menggunakan NumPy dan SciPy.

3. Metode Dekomposisi Crout

Kode:

```
import numpy as np
from scipy.linalg import lu_factor, solve_triangular
A = np.array([[3, 2, -1],
              [2, -4, 2],
              [1, 3, 2]])
B = np.array([10000, 8000, 15000])
LU, piv = lu_factor(A)
Y = solve_triangular(LU, B, lower=True)
X = solve_triangular(LU, Y)
print("Jumlah optimal kontribusi dari setiap divisi:")
print(f"Produksi (P): {X[0]}")
print(f"Pemasaran (M): {X[1]}")
print(f"Keuangan (K): {X[2]}")
```

Hasil:

```
Jumlah optimal kontribusi dari setiap
 divisi:
Produksi (P): 975.6944444444447
Pemasaran (M): 750.673185941043
Keuangan (K): 1095.096371882086
```

Analisa:

1. **Matriks Koefisien dan Vektor Hasil**: Kita mendefinisikan matriks koefisien AA dari persamaan linear dan vektor hasil B. Matriks A adalah matriks koefisien yang menghubungkan variabel-variabel P, M, dan K, sedangkan B adalah vektor hasil dari persamaan linear.
2. **Dekomposisi LU**: Kita menggunakan fungsi `lu_factor` dari modul `scipy.linalg` untuk melakukan dekomposisi matriks koefisien A menjadi dua matriks L dan U, di mana L adalah matriks segitiga bawah dan U adalah matriks segitiga atas. Variabel `piv` menyimpan informasi tentang pivoting yang terjadi selama dekomposisi.
3. **Penyelesaian Persamaan**: Setelah dekomposisi dilakukan, kita menggunakan fungsi `solve_triangular` untuk menyelesaikan sistem persamaan linear dengan matriks segitiga L dan U yang telah dihasilkan. Kita menggunakan `solve_triangular` dua kali, pertama untuk memecahkan $LY=B$ dan kemudian $UX=Y$.
4. **Menampilkan Hasil**: Hasil akhirnya adalah solusi dari sistem persamaan linear, yang merupakan jumlah optimal kontribusi dari setiap divisi. Kita mencetak hasil ini untuk ditampilkan kepada pengguna.

Kode tersebut melakukan langkah-langkah berikut:

1. Melakukan dekomposisi LU dari matriks koefisien A.

2. Menyelesaikan sistem persamaan linear menggunakan matriks segitiga L dan U yang telah didekomposisi.

3. Menampilkan hasil solusi, yaitu jumlah optimal kontribusi dari setiap divisi.

Implementasi dengan metode dekomposisi LU (LU Crout) ini menggunakan NumPy dan SciPy.