Nama : Annisa Charisma Wijayanti

NIM : 211220122140086

Mata Kuliah : Metode Numerik – Kelas D

Jurusan : Teknik Komputer

Link Github : <https://github.com/annisacharisma/Implementasi-Interpolarisasi_Annisa-Charisma-Wijayanti_Metode-Numerik>

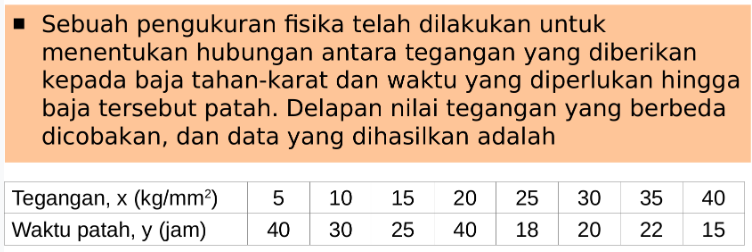
**IMPLEMENTASI INTERPOLASI**

Diinginkan aplikasi untuk mencari solusi dari problem pengujian yang memperoleh data terbatas (data terlampir) dengan interpolasi masing-masing menggunakan metode:

1. polinom Langrange
2. polinom Newton

Tugas mahasiswa:

1. Mahasiswa membuat kode sumber dengan bahasa pemrograman yang dikuasai untuk mengimplementasikan solusi di atas
2. Sertakan kode testing untuk menguji kode sumber tersebut untuk menyelesaikan problem dalam gambar. Plot grafik hasil interpolasi dengan 5 <= x <= 40
3. Mengunggah kode sumber tersebut ke Github dan setel sebagai publik. Berikan deskripsi yang memadai dari project tersebut
4. Buat dokumen docx dan pdf yang menjelaskan alur kode dari (1), analisis hasil, dan penjabarannya.



1. **Polinom Langrange**

Metode Lagrange disukai karena konsepnya sederhana. Namun, secara komputasi metode ini kurang efisien dibanding metode Newton. Secara umum derajat ≤ n untuk (n+1) titik berbeda, persamaan polinom Lagrange dirumuskan:

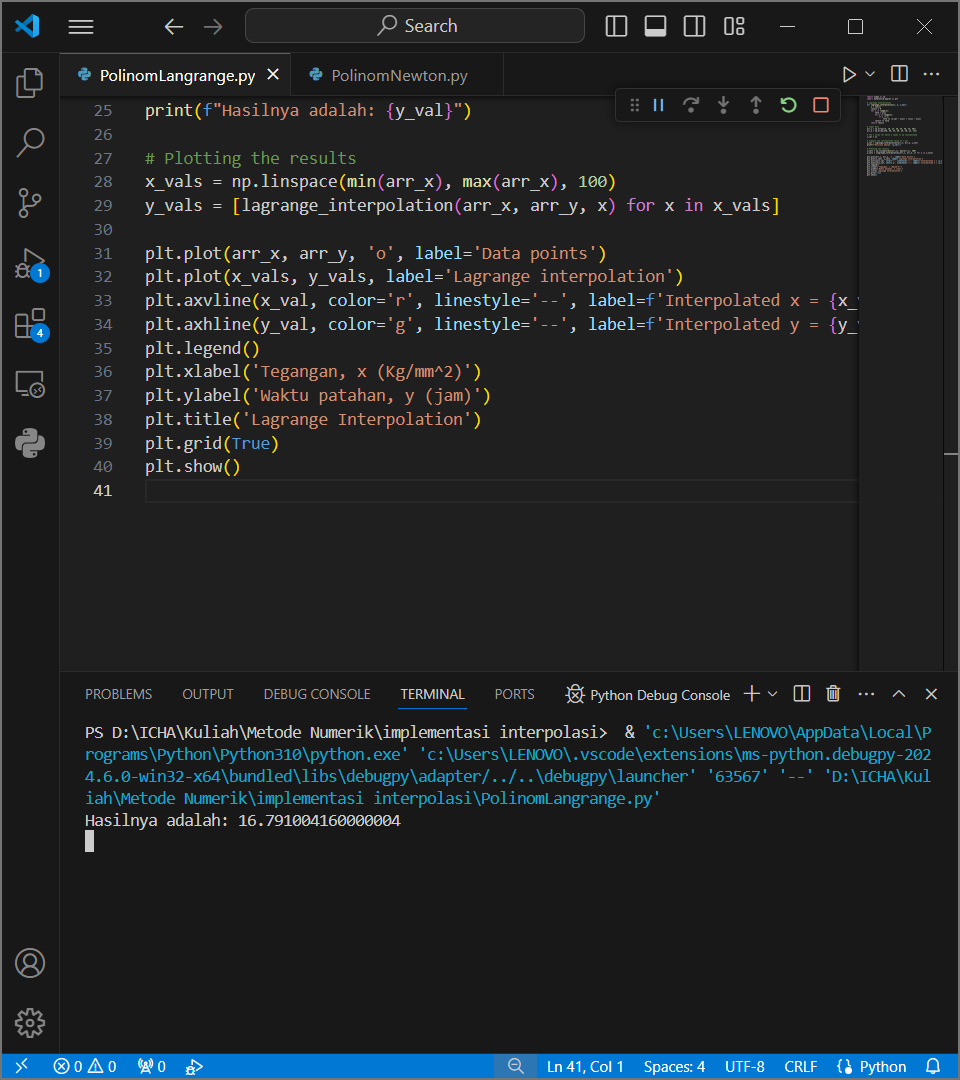
Souce Code:

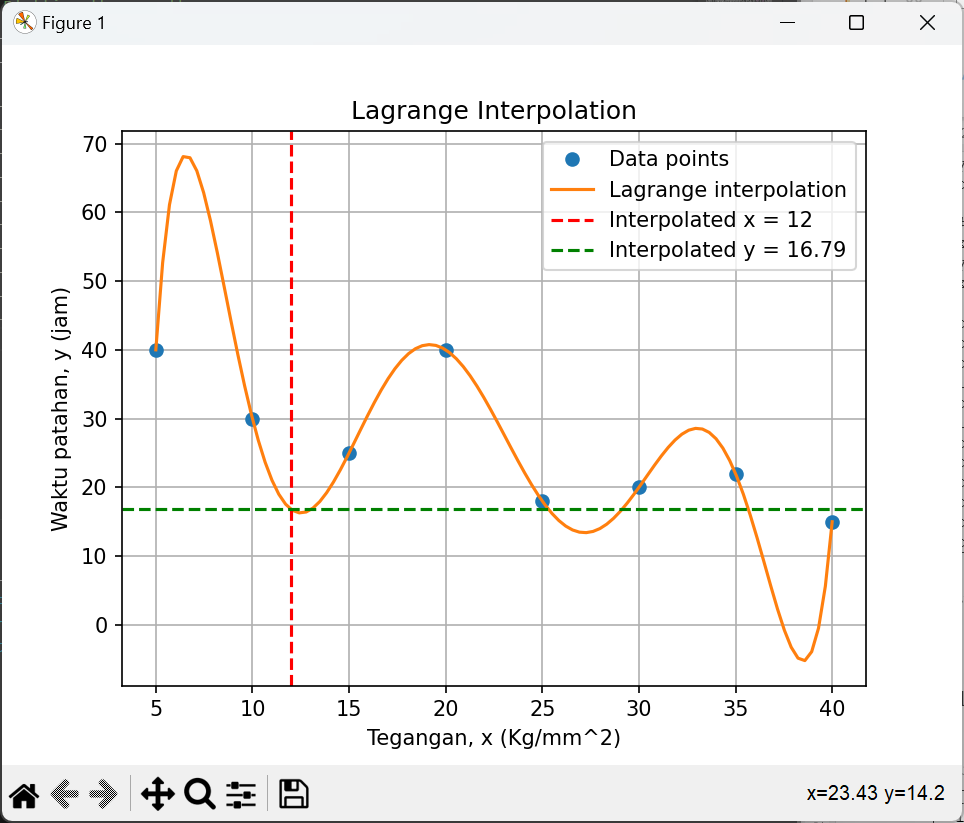
|  |
| --- |
| import numpy as np  import matplotlib.pyplot as plt  # Lagrange Interpolation  def lagrange\_interpolation(x, y, x\_val):  n = len(x)  result = 0  for i in range(n):  term = y[i]  for j in range(n):  if j != i:  term \*= (x\_val - x[j]) / (x[i] - x[j])  result += term  return result  # input data  arr\_x = np.array([5, 10, 15, 20, 25, 30, 35, 40])  arr\_y = np.array([40, 30, 25, 40, 18, 20, 22, 15])  # The x value for which y needs to be interpolated  x\_val = 12  # Compute the interpolated value at x\_val  y\_val = lagrange\_interpolation(arr\_x, arr\_y, x\_val)  print(f"Hasilnya adalah: {y\_val}")  # Plotting the results  x\_vals = np.linspace(min(arr\_x), max(arr\_x), 100)  y\_vals = [lagrange\_interpolation(arr\_x, arr\_y, x) for x in x\_vals]  plt.plot(arr\_x, arr\_y, 'o', label='Data points')  plt.plot(x\_vals, y\_vals, label='Lagrange interpolation')  plt.axvline(x\_val, color='r', linestyle='--', label=f'Interpolated x = {x\_val}')  plt.axhline(y\_val, color='g', linestyle='--', label=f'Interpolated y = {y\_val:.2f}')  plt.legend()  plt.xlabel('Tegangan, x (Kg/mm^2)')  plt.ylabel('Waktu patahan, y (jam)')  plt.title('Lagrange Interpolation')  plt.grid(True)  plt.show() |

Penjelasan Source Code:

* Fungsi lagrange\_interpolation menerima tiga parameter, yaitu x yang berisi nilai-nilai x dari data titik, y array dari nilai-nilai y yang sesuai, dan x\_val adalah nilai x yang ingin dicari nilai interpolasinya. Fungsi ini menghitung nilai interpolasi dengan membentuk polinom Lagrange. Setiap iterasi dalam fungsi menghitung sebuah polinom basis Lagrange untuk setiap titik data, yang kemudian digunakan untuk membentuk polinom interpolasi akhir.
* Selanjutnya mendefiniskan dua array arr\_x dan arr\_y yang berisi data nilai dari input yang diberikan.
* Menentukan nilai x pada titik yang ingin dihitung interpolasi nilai y nya menggunakan fungsi x\_val
* Selanjutnya memanggil fungsi lagrange\_interpolation dengan data titik dan x\_val untuk menghitung nilai y yang diinterpolasi pada x\_val. Hasil ini disimpan dalam y\_val.
* Untuk visualisasi, nilai-nilai x dalam rentang minimum dan maksimum arr\_x dibuat dalam array x\_vals menggunakan np.linspace.
* Plot grafik dibuat menggunakan matplotlib, yang menampilkan titik-titik data asli, kurva interpolasi Lagrange, serta garis-garis vertikal dan horizontal untuk menunjukkan titik interpolasi x\_val dan nilai y yang diinterpolasi.

Hasil:





1. **Polinom Newton**

Persamaan polinomial pada Newton memudahkan komputasi karena fungsinya bisa disederhanakan dengan cara faktorisasi. Metode ini menggunakan konsep perbedaan terbagi (divided differences) untuk membangun polinom interpolasi secara iteratif. Interpolasi polinom Newton dapat didefinisikan sebagai berikut:

Koefisien adalah perbedaan terbagi yang dihitung dengan:

Perbedaan terbagi dihitung sebagai:

Source Code:

|  |
| --- |
| import numpy as np  import matplotlib.pyplot as plt  # Newton Interpolation  def newton\_interpolation(x, y, x\_val):  n = len(y)  # Create a divided difference table  coef = np.zeros([n, n])  coef[:, 0] = y  for j in range(1, n):  for i in range(n - j):  coef[i][j] = (coef[i + 1][j - 1] - coef[i][j - 1]) / (x[i + j] - x[i])    # Evaluate the interpolation polynomial at x\_val  result = coef[0, 0]  product\_term = 1  for i in range(1, n):  product\_term \*= (x\_val - x[i - 1])  result += coef[0, i] \* product\_term  return result  # input data  arr\_x = np.array([5, 10, 15, 20, 25, 30, 35, 40])  arr\_y = np.array([40, 30, 25, 40, 18, 20, 22, 15])  # The x value for which y needs to be interpolated  x\_val = 12 s  # Compute the interpolated value at x\_val  y\_val = newton\_interpolation(arr\_x, arr\_y, x\_val)  print(f"Hasilnya adalah: {y\_val}")  # Plotting the results  x\_vals = np.linspace(min(arr\_x), max(arr\_x), 100)  y\_vals = [newton\_interpolation(arr\_x, arr\_y, x) for x in x\_vals]  plt.plot(arr\_x, arr\_y, 'o', label='Data points')  plt.plot(x\_vals, y\_vals, label='Newton interpolation')  plt.axvline(x\_val, color='r', linestyle='--', label=f'Interpolated x = {x\_val}')  plt.axhline(y\_val, color='g', linestyle='--', label=f'Interpolated y = {y\_val:.2f}')  plt.legend()  plt.xlabel('Tegangan, x (Kg/mm^2)')  plt.ylabel('Waktu patahan, y (jam)')  plt.title('Newton Interpolation')  plt.grid(True)  plt.show() |

Penjelasan Source Code:

* Fungsi newton\_interpolation digunakan untuk menghitung nilai interpolasi menggunakan metode Newton. Fungsi tersebut menerima tiga parameter, yaitu x : array yang berisi titik-titik x (independen), y array yang berisi titik-titik y (dependen), dan x\_val adalah nilai x yang ingin diinterpolasi.
* Inisialisasi ‘n’ adalah panjang array y, dan coef adalah tabel perbedaan terbagi (divided difference table) yang diinisialisasi sebagai matriks nol dengan ukuran n x n.
* Selanjutnya, tabel perbedaan terbagi dihitung melalui loop ganda yang mengisi tabel coef dengan perbedaan terbagi.
* Setelah itu, nilai interpolasi di x\_val dihitung dengan cara menginisialisasi variabel result dengan nilai coef[0, 0] dan kemudian mengalikan produk dari (x\_val - x[i - 1]) serta menambahkan kontribusi dari setiap koefisien ke dalam result. Fungsi ini kemudian mengembalikan nilai interpolasi di x\_val.
* Lalu mendefiniskan dua array arr\_x dan arr\_y yang berisi data nilai dari input yang diberikan
* Menentukan nilai x pada titik yang ingin dihitung interpolasi nilai y nya menggunakan fungsi x\_val
* Menggunakan fungsi newton\_interpolation untuk menghitung nilai interpolasi di x\_val dan mencetak hasilnya.
* Untuk memvisualisasikan hasil, rentang nilai x dibuat dari nilai minimum hingga maksimum arr\_x menggunakan np.linspace. Nilai y untuk rentang x tersebut dihitung dengan menerapkan fungsi newton\_interpolation pada setiap nilai dalam x\_vals. Plot titik-titik data asli dan hasil interpolasi dibuat menggunakan plt.plot. Garis vertikal pada x\_val dan garis horizontal pada y\_val ditambahkan untuk menandai nilai interpolasi.

Hasil:

