Nama : Bravely Dirgayuska
NIM : 21120122140142
Kelas : Metode Numerik C

“Tugas 3 - Implementasi Integrasi Numerik untuk Menghitung Estimasi nilai Pi”

Soal

Diinginkan implementasi penghitungan nilai integral fungsi tersebut secara numerik dengan metode:

1. integrasi Reimann (Metode 1)
2. Integrasi trapezoid (Metode 2)
3. Integrasi Simpson 1/3 (Metode 3)

Tugas mahasiswa:
Mahasiswa membuat kode sumber dengan bahasa pemrograman yang dikuasai untuk mengimplementasikan solusi di atas, dengan ketentuan:
Dua digit NIM terakhir % 3 = 0 mengerjakan dengan Metode 1
Dua digit NIM terakhir % 3 = 1 mengerjakan dengan Metode 2
Dua digit NIM terakhir % 3 = 2 mengerjakan dengan Metode 3

1. Sertakan kode testing untuk menguji kode sumber tersebut untuk menyelesaikan problem dengan ketentuan sebagai berikut:
    - Menggunakan variasi nilai N = 10, 100, 1000, 10000
2. Hitung galat RMS dan ukur waktu eksekusi dari tiap variasi N. Nilai referensi pi yang digunakan adalah 3.14159265358979323846
3. Mengunggah kode sumber tersebut ke Github dan setel sebagai publik. Berikan deskripsi yang memadai dari project tersebut. Masukkan juga dataset dan data hasil di repositori tersebut.
4. Buat dokumen docx dan pdf yang menjelaskan alur kode dari (1), analisis hasil, dan penjabarannya. Sistematika dokumen: Ringkasan, Konsep, Implementasi Kode, Hasil

Pengujian, dan Analisis Hasil. Analisis hasil harus mengaitkan antara hasil, galat, dan waktu eksekusi terhadap besar nilai N.

Jawab :

Konsep

Integrasi Riemann

Integrasi Riemann merupakan metode dasar untuk menghitung luas di bawah kurva pada interval tertentu dengan cara membagi interval tersebut menjadi sejumlah sub-interval kecil yang sama panjang. Setiap sub-interval ini kemudian diwakili oleh persegi panjang, di mana tinggi persegi panjang tersebut ditentukan oleh nilai fungsi pada titik tertentu dalam sub-interval tersebut (misalnya, titik kiri, kanan, atau titik tengah). Luas total dihitung dengan menjumlahkan luas semua persegi panjang tersebut. Akurasi metode ini bergantung pada jumlah sub-interval yang digunakan; semakin banyak sub-interval, semakin mendekati hasil integrasi sebenarnya.

Integrasi Trapezoid

Metode Trapezoid adalah teknik numerik untuk menghitung integral suatu fungsi dengan membagi interval integrasi menjadi sejumlah sub-interval dan mengganti area di bawah kurva dengan sejumlah trapezoid. Setiap trapezoid dibentuk dengan menghubungkan titik-titik pada kurva dengan garis lurus. Luas tiap trapezoid dihitung dengan rumus dasar trapezoid, dan total integral diperoleh dengan menjumlahkan luas semua trapezoid. Metode ini lebih akurat dibandingkan metode Riemann sederhana karena memperhitungkan perubahan fungsi di antara titik-titik sub-interval.

Integrasi Simpson 1/3

Metode Simpson 1/3 adalah teknik integrasi numerik yang lebih canggih dibandingkan metode Riemann dan Trapezoid, menggunakan polinomial kuadrat untuk mendekati fungsi yang akan diintegralkan. Interval integrasi dibagi menjadi sejumlah sub-interval yang genap, kemudian setiap dua sub-interval berurutan digunakan untuk membentuk polinomial kuadrat. Integral dihitung dengan menjumlahkan kontribusi masing-masing polinomial kuadrat, yang diperoleh dengan rumus khusus Simpson 1/3. Metode ini memberikan akurasi yang lebih tinggi,

terutama untuk fungsi yang halus, karena menggunakan pendekatan polinomial yang lebih baik dalam mengaproksimasi fungsi.

1.

Implementasi Kode :

```
import time
import numpy as np

def reimann_integration(N):
    dx = 1.0 / N
    total_area = 0.0
    for i in range(N):
        x = (i + 0.5) * dx
        total_area += 4.0 / (1.0 + x * x)
    pi_approx = total_area * dx
    return pi_approx

def calculate_rms_error(approx_pi, reference_pi):
    return np.sqrt((approx_pi - reference_pi) ** 2)

def main():
    reference_pi = 3.14159265358979323846
    N_values = [10, 100, 1000, 10000]

    for N in N_values:
        start_time = time.time()
        approx_pi = reimann_integration(N)
        execution_time = time.time() - start_time
        rms_error = calculate_rms_error(approx_pi, reference_pi)

        print(f"N = {N}")
        print(f"Approximated Pi: {approx_pi}")
        print(f"RMS Error: {rms_error}")
        print(f"Execution Time: {execution_time} seconds\n")

if __name__ == "__main__":
    main()
```

Penjelasan Codingan :

Kode di atas mengimplementasikan solusi integrasi Riemann untuk menghitung nilai π (pi) menggunakan berbagai nilai N yang berbeda. Fungsi `reimann_integration(N)` bertugas menghitung integral dengan membagi interval [0, 1] menjadi N subinterval, menghitung lebar tiap subinterval (dx), dan mengakumulasi total area di bawah kurva fungsi f(x) $\frac{4}{1+x^2}$ untuk tiap titik tengah subinterval. Fungsi `calculate_rms_error(approx_pi, reference_pi)` digunakan untuk menghitung galat RMS (Root Mean Square Error) antara nilai π yang diaproksimasi dan nilai referensi π yang diberikan (3.14159265358979323846). Fungsi utama `main()` mendefinisikan nilai referensi π dan daftar variasi nilai N yang akan diuji. Untuk setiap nilai N, fungsi ini mengukur waktu eksekusi, menghitung nilai π yang diaproksimasi, menghitung galat RMS, dan mencetak hasilnya termasuk nilai π yang diaproksimasi, galat RMS, serta waktu eksekusi. Hasil ini memberikan gambaran tentang akurasi dan efisiensi metode integrasi Riemann dengan berbagai ukuran subinterval.

Hasil Pengujian:

```
N = 10
Approximated Pi: 3.1424259850010987
RMS Error: 0.0008333314113055934
Execution Time: 0.0 seconds

N = 100
Approximated Pi: 3.1416009869231254
RMS Error: 8.333333332277704e-06
Execution Time: 0.0 seconds

N = 1000
Approximated Pi: 3.1415927369231227
RMS Error: 8.333332957022321e-08
Execution Time: 0.0 seconds

N = 10000
Approximated Pi: 3.141592654423134
RMS Error: 8.3334095180021e-10
Execution Time: 0.001994609832763672 seconds
```

Analisis Hasil :

Hasil output dari kode tersebut memberikan analisis tentang seberapa baik metode integrasi Riemann mengaproksimasi nilai π (pi) dengan berbagai nilai N, yang merupakan jumlah subinterval yang digunakan dalam perhitungan. Dengan meningkatkan nilai N dari 10 hingga 10,000, kita dapat mengamati bahwa nilai π yang diaproksimasi mendekati nilai referensi π secara bertahap. Untuk N = 10, aproksimasi mungkin memiliki galat RMS yang relatif besar, menunjukkan bahwa jumlah subinterval yang kecil memberikan hasil yang kurang akurat. Namun, saat N ditingkatkan menjadi 100, 1,000, dan akhirnya 10,000, galat RMS berkurang secara signifikan, mencerminkan peningkatan akurasi dalam aproksimasi nilai π. Di sisi lain, waktu eksekusi meningkat dengan bertambahnya N, menunjukkan trade-off antara akurasi dan efisiensi waktu. Dengan N yang lebih besar, meskipun aproksimasi menjadi lebih akurat, komputasi membutuhkan waktu yang lebih lama. Analisis ini menyoroti bahwa meskipun metode integrasi Riemann efektif untuk menghitung nilai π, ada batasan praktis terkait dengan waktu komputasi, terutama untuk nilai N yang sangat besar.

3. Link Github : https://github.com/Brvlyd/Tugas-Implementasi-Integrasi-Numerik-untuk-Menghitung-Estimasi-nilai-Pi_Metode-Numerik_Bravely