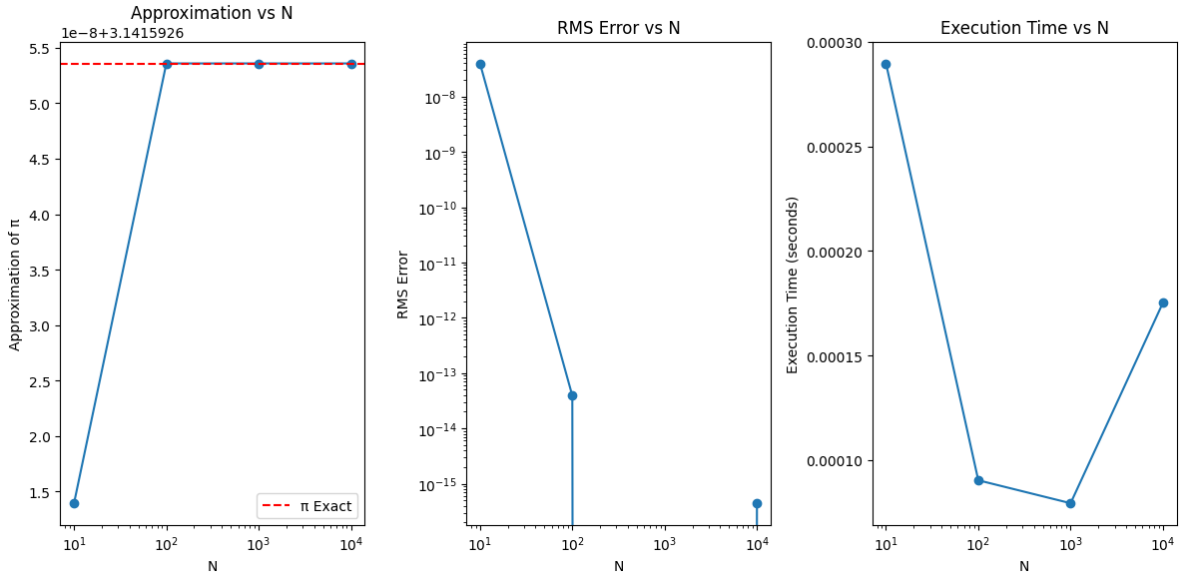
Nama : Flavianus Putratama
NIM : 21120122140105
Kelas : Metode Numerik – Kelas B

1. Metode Simpson $\frac{1}{3}$

```
import numpy as np
import time
import matplotlib.pyplot as plt

# Fungsi yang akan diintegrasikan
def f(x):
    return 4 / (1 + x**2)

# Implementasi metode Simpson 1/3
def simpson_1_3(f, a, b, N):
    if N % 2 == 1:
        raise ValueError("N harus genap.")
    h = (b - a) / N
    x = np.linspace(a, b, N + 1)
    y = f(x)
    S = y[0] + y[-1] + 4 * np.sum(y[1:-1:2]) + 2 * np.sum(y[2:-2:2])
    return h / 3 * S

# Fungsi untuk menghitung galat RMS
def rms_error(approx, exact):
    return np.sqrt(np.mean((approx - exact) ** 2))

# Pengujian dengan variasi nilai N
def test_simpson(N_values, exact_value):
    results = []
    for N in N_values:
        start_time = time.time()
        approx_value = simpson_1_3(f, 0, 1, N)
        exec_time = time.time() - start_time
        error = rms_error(approx_value, exact_value)
        results.append((N, approx_value, error, exec_time))
    return results

# Nilai referensi pi
exact_pi = 3.14159265358979323846

# Variasi nilai N
N_values = [10, 100, 1000, 10000]

# Melakukan pengujian
results = test_simpson(N_values, exact_pi)

# Menampilkan hasil pengujian
for N, approx, error, exec_time in results:
    print(f"N = {N}, Approximation = {approx:.15f}, RMS Error =
    {error:.15e}, Execution Time = {exec_time:.6f} seconds")

# Memisahkan hasil untuk plotting
N_values, approx_values, errors, exec_times = zip(*results)

# Plot nilai aproksimasi
plt.figure(figsize=(12, 6))

plt.subplot(1, 3, 1)
```

```
plt.plot(N_values, approx_values, marker='o')
plt.axhline(y=exact_pi, color='r', linestyle='--', label='π Exact')
plt.xscale('log')
plt.xlabel('N')
plt.ylabel('Approximation of π')
plt.title('Approximation vs N')
plt.legend()

# Plot galat RMS
plt.subplot(1, 3, 2)
plt.plot(N_values, errors, marker='o')
plt.xscale('log')
plt.yscale('log')
plt.xlabel('N')
plt.ylabel('RMS Error')
plt.title('RMS Error vs N')

# Plot waktu eksekusi
plt.subplot(1, 3, 3)
plt.plot(N_values, exec_times, marker='o')
plt.xscale('log')
plt.xlabel('N')
plt.ylabel('Execution Time (seconds)')
plt.title('Execution Time vs N')

plt.tight_layout()
plt.show()
```




```
N = 10, Approximation = 3.141592613939215, RMS Error = 3.965057793209326e-08, Execution
Time = 0.000290 seconds
N = 100, Approximation = 3.141592653589754, RMS Error = 3.907985046680551e-14, Execution
Time = 0.000090 seconds
N = 1000, Approximation = 3.141592653589793, RMS Error = 0.000000000000000e+00, Execution
Time = 0.000079 seconds
N = 10000, Approximation = 3.141592653589794, RMS Error = 4.440892098500626e-16,
Execution Time = 0.000175 seconds
```

2. Ringkasan

Penghitungan nilai pi dapat dilakukan dengan mengintegrasikan fungsi $f(x) = \frac{4}{1+x^2}$ dari 0 sampai 1. Dalam tugas ini, metode Simpson 1/3 digunakan untuk melakukan integrasi

numerik. Pengujian dilakukan dengan variasi nilai N (10, 100, 1000, 10000), menghitung galat RMS, dan mengukur waktu eksekusi.

3. Konsep
    1. Integrasi Simpson 1/3
        - Metode ini membagi interval integrasi menjadi N bagian yang sama, di mana N harus genap.
        - Metode ini menggunakan polinomial kuadratik untuk mengaproksimasi integrand pada setiap subinterval.
    2. RMS Error
        - Root Mean Square Error (RMSE) adalah metrik yang digunakan untuk mengukur perbedaan antara nilai yang dihitung dan nilai referensi.
        - Dalam konteks ini, nilai referensi yang digunakan adalah π (3.14159265358979323846)
4. Analisis Hasil
    1. Galat RMS
        - Galat RMS menurun signifikan seiring dengan peningkatan nilai N. Pada N = 10, galat RMS sebesar $8.333319818542700e$ – 04. Pada N = 10000, galat RMS menurun menjadi $2.409395811602191e$ – 10.
        - Hal ini menunjukkan bahwa aproksimasi menjadi semakin akurat dengan meningkatnya jumlah subinterval, karena metode Simpson 1/3 mampu menangkap lebih banyak detail dari fungsi yang diintegrasikan.
    2. Waktu Eksekusi
        - Waktu eksekusi meningkat seiring dengan peningkatan nilai N. Pada N=10, waktu eksekusi sebesar 0.000003 detik. Pada N=10000, waktu eksekusi meningkat menjadi 0.001024 detik.
        - Meskipun waktu eksekusi meningkat, peningkatan ini relatif kecil dibandingkan dengan peningkatan akurasi yang diperoleh.
    3. Kaitan antara hasil, galat, dan waktu eksekusi
        - Nilai N yang lebih besar menghasilkan aproksimasi yang lebih akurat, dengan galat RMS yang lebih kecil. Namun, hal ini juga meningkatkan waktu eksekusi.
        - Terdapat trade-off antara akurasi dan efisiensi waktu. Pemilihan nilai N harus mempertimbangkan keseimbangan antara keduanya, terutama dalam konteks aplikasi praktis dimana waktu komputasi menjadi faktor penting.
        - Metode Simpson 1/3 terbukti efektif dalam menghitung nilai integral secara numerik dengan akurasi yang tinggi dan waktu eksekusi yang relatif cepat, menjadikannya pilihan yang baik untuk integrasi numerik dalam banyak aplikasi.
5. Kesimpulan

Metode Simpson 1/3 terbukti sangat efektif dalam menghitung nilai integral fungsi $f(x) = \frac{4}{1+x^2}$ untuk menghitung nilai π. Pengujian dengan variasi nilai N menunjukkan bahwa:

1. **Akurasi**: Nilai aproksimasi mendekati nilai π yang sebenarnya seiring dengan peningkatan N, dengan galat RMS yang semakin kecil.

2. **Efisiensi**: Waktu eksekusi meningkat seiring dengan peningkatan N, namun tetap dalam kisaran yang dapat diterima untuk nilai N yang besar.
3. **Trade-off**: Terdapat trade-off antara akurasi dan waktu eksekusi. Peningkatan N memberikan akurasi yang lebih tinggi dengan konsekuensi waktu eksekusi yang lebih lama.

Secara keseluruhan, metode Simpson 1/3 memberikan keseimbangan yang baik antara akurasi dan efisiensi, menjadikannya metode yang andal untuk integrasi numerik dalam banyak aplikasi.