Mini relatório de MetComp: Tarefa erro Integral

Aluno: Marcelo Camaran Lucas

Foi pedido para confeccionar um programa que tivesse as caracteristicas de retornar os valores da integral numérica e erros correspndente ao cálculo. Também criar um gráfico para demosntrar a progressão do erro conforme a diminuição do intervalo de dx. Para realizar este trabalho, foi usado dois métodos, o de trapézio e o Simpson.

Os parâmetros utilizados foram os seguintes:

$$f\left(x\right) = \frac{1}{\left(x^2+1\right)}$$

e

$$x \, \in \, \left(-3, \, 3\right)$$

E para o número de fatias N foi criado um aranjo que varia conforme

$$N \, = \, \left(2^2, \, \ldots \, , \, 2^{20}\right)$$

O método do trapézio é feito separando a área abaixo da curva em N trapézios, assim a precisão é maior que usando retângulos. A largura de cada fatias no intervalo de (a,b) pode ser representada da seguinte maneira:

$$\Delta x \, = \, \frac{(b-a)}{N}$$

Assim cada fatia pela altura média dos dois lados da fatia vai resultar na área do trapézio. A Integral pode ser escrita da seguinte forma:

$$\int_a^b f\left(x\right) dx \, \cong \, \left[\frac{f\left(x_0\right) + f\left(x_N\right)}{2} + \sum_{i=1}^{N-1} f\left(x_i\right)\right] \, \Delta x$$

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

#Metodo do Trapézio

def f(x):
    return 1/(math.pow(x, 2) + 1)

n = []
for i in np.arange(2, 22, 2):
    n.append(int(math.pow(2, i)))
#print (n)

a = -3    #valor inicial
b = 3     #valor final

def integral_trapezio(f, a, b, n):
```

```
    dx = (b-a)/n
    soma_1 = (f(a) + f(b))/2
    soma_2 = 0
    for i in range (n):
        x = a + i*dx
        soma_2 += f(x)
    return (soma_1 + soma_2)*dx

print(integral_trapezio(f, a, b, 10))  #o valor de N=10 é apenas para teste
```

```
2.5546898806889313
```

Podemos calcular o erro com a seguinte equação:

$$\varepsilon = |I_{exata} - I(N)|$$

Clique duas vezes (ou pressione "Enter") para editar

```
def f_analitica(a, b):
    return math.atan(b) - math.atan(a)

print(f_analitica(a, b))

erro_trapezio = []
for i in range(10):
    erro_trapezio.append(math.fabs(f_analitica(a, b) - integral_trapezio(f, a, b, n[i])
print(erro_trapezio)
```

```
2.498091544796509
[0.22498537828041432, 0.036097225861289495, 0.009287121729578907, 0.0023382568842156
```

```
x_grafico = []
for i in n:
    x_grafico.append((b-a)/i)
#print(y)

plt.figure()
plt.grid()
plt.plot(x_grafico, erro_trapezio, 'r')
plt.xscale('log')
plt.yscale('log')
plt.xlabel('log(Erro)')
plt.ylabel('log(Δx)')
plt.show()

def inclinacao(erro, a, b, n):
    dx1 = (b - a)/n[0]
    dx2 = (b - a)/n[9]
    return (math.log(dx2) - math.log(dx1))/(math.log(erro[9]) - math.log(erro[0]))

print(f'Inclinação da reta do trapézio: {inclinacao(erro_trapezio, a, b, n)}')
```

Na segunda parte da tarefa, foi utilizado o método simpson, este método utiliza de aproximar a função por uma constante para cada fatia.

Para cada par de fatias, é criado uma coleção de parábolas, elas dão os 3 pontos que precisamos para criar uma parábola. A integral pode ser aproximada como:

$$\int_a^b f(x)\,dx \cong \frac{h}{3}\left[f(a) + f(b) + 4\sum_{i=1}^{\frac{N}{2}} f(a + (2i-1)h) + 2\sum_{i=1}^{\frac{N}{2}-1} f(a + 2ih)\right]$$

h é a largura das fatias

```
#Método de Simpson

def integral_simpson(f, a, b, n):
    soma_par = 0
    soma_impar = soma_par
    dx = (b - a)/n
    for i in range(1, n, 2):
        soma_impar += f(a + i*dx)
    for i in range(2, n, 2):
        soma_par += f(a + i*dx)
    return (dx/3)*(f(a) + f(b) + 4*soma_impar + 2*soma_par)

print(integral_simpson(f, a, b, 10))    #o valor de N=10 é apenas para teste

erro_simpson = []
for i in range(10):
    erro_simpson.append(math.fabs(f_analitica(a, b) - integral_simpson(f, a, b, n[i])))
print(erro_simpson)

plt.figure()
plt.grid()
plt.plot(x_grafico, erro_simpson, 'r')
plt.xscale('log')
plt.yscale('log')
plt.xlabel('log(Erro)')
plt.ylabel('log(Δx)')
plt.show()

print(f'Inclinação da reta de Simpson: {inclinacao(erro_simpson, a, b, n)}')
```

Para concluir, com base na análise dos gráficos e a inclinação, podemos ver que o método Simpson é o que mais tem precisão, julgando a base de erro dos dois métodos para a mesma função.

Podemos notar também que, o valor da inclinação corresponde a taxa de decéscimo do erro em razão da largura das fatias, portanto, novamente mostrando que o método simpson é mais efetivo.

✓ 1s conclusão: 19:14