UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI - UFSJ
DEPARTAMENTO DE ENGENHARIA ELÉTRICA - DEPEL
COORDENAÇÃO DE ENGENHARIA ELÉTRICA - COELE

GABRIEL AUGUSTO SILVA BATISTA

TRABALHO 6

São João del-Rei – MG
Setembro de 2023

Definição das funções: As funções f1 e f2 são definidas usando expressões lambda. A função f1 é um polinômio de grau 5 e a função f2 é a raiz quadrada de x.

```
3   #Funções
4   f1 = lambda x: 3 + 4*x - 3*x**2 + 5*x**3 - x**4 + 4*x**5
5   f2 = lambda x: np.sqrt(x)
```

Limites de integração: Os limites de integração para as duas funções são definidos. Para f1, a integral é calculada de 0 a 2. Para f2, a integral é calculada de 0 a 20.

```
7   #Limites de integração
8   a1, b1 = 0, 2
9   a2, b2 = 0, 20
```

Integrais analíticas: As integrais analíticas das duas funções são calculadas. Estes são os valores “verdadeiros” das integrais, que serão usados para calcular o erro absoluto.

```
11  # Integrais analíticas
12  I1_analitico = 2**6 - 2**5/2 + 5*2**4/4 - 2**5/5 + 4*2**6/6 - 3
13  I2_analitico = 2/3 * 20**(3/2)
```

Números de pontos: Listas de números de pontos são definidas para cada método de integração e para cada função.

```
15  # Pontos
16  pontostrape1 = [2, 5, 13]
17  pontosSimp1 = [3, 5, 13]
18  pontosGauss1 = [1, 2, 3]
19  pontostrape2 = [2, 5, 99]
20  pontosSimp2 = [3, 5, 99]
```

Implementação dos métodos de integração: As funções trapezio, simpson e quadratura são definidas para implementar a regra do trapézio, a regra de Simpson e a quadratura gaussiana, respectivamente.

```
22  #Fórmulas
23  def trapezio(f, a, b, n):
24      h = (b - a) / n
25      x = np.linspace(a, b, n+1)
26      y = f(x)
27      return (h / 2) * (y[0] + 2 * np.sum(y[1:-1]) + y[-1])
28
29  def simpson(f, a, b, n):
30      h = (b - a) / n
31      x = np.linspace(a, b, n+1)
32      y = f(x)
33      return (h / 3) * (y[0] + 4 * np.sum(y[1:-1:2]) + 2 * np.sum(y[2::2]) + y[-1])
34
35  def quadratura(f, a, b, n):
36      x, w = np.polynomial.legendre.leggauss(n)
37      t = 0.5 * (x + 1) * (b - a) + a
38      return np.dot(w, f(t)) * 0.5 * (b - a)
```

Cálculo das integrais e dos erros absolutos: Para cada função e para cada método de integração, a integral é calculada para diferentes números de pontos. O erro absoluto é então calculado como a diferença absoluta entre o resultado numérico e o resultado analítico.

```
# Calculando as integrais usando os métodos implementados
print(f'Função 1 integral (Trapézio) para os pontos {pontostrape1}')
for n in pontostrape1:
    I1_trapz = trapezio(f1, a1, b1, n)
    E1_trapz = abs(I1_analitico - I1_trapz)
    print(f'Resultado: {I1_trapz}')
    print(f'Erro absoluto {E1_trapz}\n')

print(f'Função 1 integral (Simpson) para os pontos {pontosSimp1}')
for n in pontosSimp1:
    I1_simpson = simpson(f1, a1, b1, n)
    E1_simpson = abs(I1_analitico - I1_simpson)
    print(f'Resultado: {I1_simpson}')
    print(f'Erro absoluto {E1_simpson}\n')

print(f'Função 1 integral (Quadratura) para os pontos {pontosGauss1}')
for n in pontosGauss1:
    I1_gauss = quadratura(f1, a1, b1, n)
    E1_gauss = abs(I1_analitico - I1_gauss)
    print(f'Resultado: {I1_gauss}')
    print(f'Erro absoluto {E1_gauss}\n')

print(f'Função 2 integral (Trapézio) para os pontos {pontostrape2}')
for n in pontostrape2:
    I2_trapz = trapezio(f2, a2, b2, n)
    E2_trapz = abs(I2_analitico - I2_trapz)
    print(f'Resultado: {I2_trapz}')
    print(f'Erro absoluto {E2_trapz}\n')

print(f'Função 2 integral (Simpson) para os pontos {pontosSimp2}')
for n in pontosSimp2:
    I2_simpson = simpson(f2, a2, b2, n)
    E2_simpson = abs(I2_analitico - I2_simpson)
    print(f'Resultado: {I2_simpson}')
```

Quadratura Gaussiana com subdivisão do intervalo: Para a função f2, a integral também é calculada usando a quadratura gaussiana com subdivisão do intervalo. O intervalo de integração é dividido em 4 subintervalos e a integral é calculada com 5 pontos em cada subintervalo.

```
#Gaussiana com subdivisão do intervalo
def gaussian_quadrature_subdiv(f, a, b, n, m):
    subintervalos = np.linspace(a, b, m+1)
    result = 0
    for i in range(m):
        result += quadratura(f, subintervalos[i], subintervalos[i+1], n)
    return result

I2_gauss_subdiv = gaussian_quadrature_subdiv(f2, a2, b2, 5, 4)
E2_gauss_subdiv = abs(I2_analitico - I2_gauss_subdiv)

print(f'Função 2 integral (Quadratura com subdivisão) para 5 pontos em cada subintervalo:')
print(f'Resultado: {I2_gauss_subdiv}')
print(f'Erro absoluto {E2_gauss_subdiv}\n')
```

Impressão dos resultados: Os resultados das integrais e dos erros absolutos são impressos para cada função, para cada método de integração e para cada número de pontos.

```
Função 1 integral (Trapézio) para os pontos [2, 5, 13]
Resultado: 89.0
Erro absoluto 12.266666666666666

Resultado: 66.71424
Erro absoluto 34.55242666666666

Resultado: 62.92867896782327
Erro absoluto 38.3379876988434

Função 1 integral (Simpson) para os pontos [3, 5, 13]
Resultado: 52.37128486511202
Erro absoluto 48.89538180155465

Resultado: 50.99750400000001
Erro absoluto 50.26916266666665

Resultado: 55.85981766974138
Erro absoluto 45.406848996925284

Função 1 integral (Quadratura) para os pontos [1, 2, 3]
Resultado: 24.0
Erro absoluto 77.26666666666667

Resultado: 58.88888888888888
Erro absoluto 42.37777777777779

Resultado: 62.26666666666668
Erro absoluto 38.999999999999986

Função 2 integral (Trapézio) para os pontos [2, 5, 99]
Resultado: 53.9834563766817
Erro absoluto 5.645023023312689

Resultado: 58.11438686953494
Erro absoluto 1.5140925304594504

Resultado: 59.609983299019966
```

```
Resultado: 59.609983299019966
Erro absoluto 0.018496100974424223

Função 2 integral (Simpson) para os pontos [3, 5, 99]
Resultado: 49.11790884064053
Erro absoluto 10.510570559353859

Resultado: 53.3138622200573
Erro absoluto 6.314617179937088

Resultado: 59.32071427905753
Erro absoluto 0.3077651209368568

Função 2 integral (Quadratura com subdivisão) para 5 pontos em cada subintervalo:
Resultado: 59.635524393644964
Erro absoluto 0.007044993650573872
```