Relatório da Tarefa 2 de MAP3121

Gabriel Youssef Campos 10884301 Victor Nascimento Pereira - 10773530

6 de junho de 2022

0 Introdução

Nessa tarefa estudamos o método de Gauss para integração numérica. Tendo em mãos uma fórmula de Gauss para aproximar integrais simples em extremos de integração específicos, nosso objetivo foi adaptar a fórmula de maneira a calcular integrais duplas em extremos de integração quaisquer.

1 Parte teórica

A fórmula de Gauss para aproximar uma função é dada por:

$$\int_a^b f(x)\,dx = \sum_{j=1}^{n} w_j f(x_j) + E_n(f) \tag{1}$$

Os pesos w_j são dados por:

$$w_j = \int_a^b L_j(x)\,dx \tag{2}$$

sendo $L_j(x)$ os polinômios de Lagrange. Pela equação 2 vemos que os pesos w_j não dependem da função $f(x)$; eles são apenas dependentes dos pontos x_j considerados. Daqui em diante vamos chamar de **nós** os pontos pontos x_j; ou seja, os **nós** são os pontos onde devemos calcular os **pesos**.

A partir de fórmulas de Gauss conhecidas para o intervalo $[-1, 1]$, podemos encontrar as fórmulas de Gaus para um intervalo $[a, b]$ qualquer utilizando a **mudança de variável** vista em 3.

$$\int_a^b f(t)\,dt = \int_{-1}^{1} f\left(\left(\frac{b-a}{2}\right)x + \left(\frac{a+b}{2}\right)\right)\cdot\left(\frac{b-a}{2}\right)dx \tag{3}$$

A mudança de variáveis apresentada em 3 precisa ser adaptada de forma que possamos implementar um programa que avalie a integral de $f(x)$ no intervalo $[a, b]$. A adaptação que fizemos pode ser vista em 4.

$$\int_a^b f(t)\,dt = \frac{b-a}{2}\cdot\sum_{i=1}^{n} w_i f\left(\frac{(b-a)x_i + (b+a)}{2}\right) \tag{4}$$

sendo w_1, w_2, ..., w_n, os pesos e x_1, x_2, ..., x_n, as raízes do polinômio de Legendre.

Chegamos a implementar um algoritmo que calculava as raízes do polinômio de Legendre para que pudéssemos usá - las na fórmula de integração de Gauss, porém, com o arquivo *dados.txt* fomos capazes de evitar esses cálculos, já que os pesos e os coeficientes foram fornecidos para diferentes valores de n.

Assim, podemos dizer que o arquivo fornecido simplificou o algoritmo, fazendo com que ele não passasse de uma sequência inteligente de somatórias.

1.1 Exemplo

A fórmula

$$\int_{-1}^{1} f(x)\,dx = \frac{5}{9}f(-\sqrt{0,6}) + \frac{8}{9}f(0) + \frac{5}{9}f(\sqrt{0,6}) \tag{5}$$

é exata para integral de -1 a 1 de polinômios de grau menor ou igual a 5, como verificado para um caso genérico com o código abaixo:

```
#f(x) = 5x^3+3x
import math

def f(x):
    return 5*x**3 + 3*x

result = (5/9)*f(-math.sqrt(0.6))+(8/9)*f(0)+(5/9)*f(math.sqrt(0.6))

print("Integral de -1 a 1 de f(x) = ", result)
```

para nível de comparação é elaborado o segundo código:

```
from scipy.integrate import quad

I = quad(f, -1, 1)

print("Integral de -1 a 1 de f(x) = ", I)
```

que nos leva a comparar o resultado da fórmula dada com um método numérico bem aproximado e apurar a veracidade da afirmação.

2 Parte prática

Aqui iremos apresentar o programa implementado sendo testado para o cálculo de integrais duplas em regiões R do plano.

2.1 Exemplo 1

Vamos calcular o volume do cubo de arestas iguais a 1 e o volume do tetraedro de vértices $(0, 0, 0)$, $(1, 0, 0)$, $(0, 1, 0)$ e $(0, 0, 1)$.

A solução em Python que implementamos para o exemplo 2.1 pode ser vista abaixo:

```
import numpy as np

from le_arquivo import n

# f(x) = 1 cubo
def f1(x):
    return 1

# f(x)= -x+1 tetraedro
def f2(x):
    return -x+1


def integra(x):
    n1=n(x)
    soma1 = 0
    soma2 = 0
    for i in n1:
        soma1 += f1(i[0])*i[1]
        soma2 += f2(i[0])*i[1]
    return soma1, soma2


x = int(input("Digite a precis o de n s (6, 8 ou 10): "))
soma1, soma2 = integra(x)
print("O volume calculado com n", x, " do cubo  : ", soma1,
    "\nO volume calculado com n", x, " do trapesio  : ", soma2)
```

2.2 Exemplo 2

Vamos calcular a área da região do primeiro quadrante dada por:

$$
\begin{aligned}
A &= \int_0^1 \left[\int_0^{1-x^2} dy \right] dx \\
&= \int_0^1 \left[\int_0^{\sqrt{1-y}} dx \right] dy = \frac{2}{3}
\end{aligned}
\tag{6}
$$

A solução em Python que implementamos para o exemplo 2.2 pode ser vista abaixo:

```
import numpy as np
import math

from le_arquivo import n

def f1(x):
    return 1-x**2

def f2(x):
    return math.sqrt(1-x)

def integra(x):
    n1=n(x)
    soma1 = 0
    soma2 = 0
    for i in n1:
        soma1 += f1(i[0])*i[1]
        soma2 += f2(i[0])*i[1]
    return soma1, soma2


x = int(input("Digite a precis o de n s (6, 8 ou 10): "))
soma1, soma2 = integra(x)
print("O volume calculado com n", x, " na ordem dy dx  : ",
    soma1, "\nO volume calculado com n", x, " na ordem dx dy  : ", soma2)
```

2.3 Exemplo 3

Vamos calcular a área e o volume abaixo da região descrita por $z = e^{y/x}$, $0.1 \leq x \leq 0.5$, $x^3 \leq y \leq x^2$. A partir da expressão 7, podemos calcular a área da região abaixo da superfície usando a equação 8.

$$
A = \int\int \sqrt{f_x^2(x,y) + f_y^2(x,y) + 1}\, dy\, dx \tag{7}
$$

$$
\int_{0.1}^{0.5} \int_{x^3}^{x^2} \sqrt{(-ye^{y/x}/(x^2))^2 + (e^{y/x}/x)^2 + 1}\, dy\, dx = 0.105498 \tag{8}
$$

A solução em Python que implementamos para o exemplo 2.3 pode ser vista abaixo:

```
import numpy as np
import math

from le_arquivo import n


def f1(x,y):
    x1 = ((0.2)*x) + 0.3
    y1 = ((x1**2-x1**3)/2)*y + ((x1**2+x1**3)/2)
    return math.sqrt((((-y1*np.exp(y1/x1))/x1**2)**2)+
        ((np.exp(y1/x1)/x1)**2)+1) * (0.2) * ((x**2-x**3)/2)


def f2(x,y):
    x1 = ((0.2)*x) + 0.3
    y1 = ((x1**2-x1**3)/2)*y + ((x1**2+x1**3)/2)
```

```
        return np.exp(y1/x1) * (0.2) * ((x**2-x**3)/2)

    def integraArea(x):
        n1 = n(x)
        F = 0
        soma = 0
        v=0
        for i in n1:
            for j in range(len(n1)):
                v=((i[0]**2-i[0]**3)/2)*n1[j][1] + ((i[0]**2+i[0]**3)/2)
                F += f1(i[0], n1[j][0]) * v
            soma += F*i[1]
        return soma

    def integraVolume(x):
        n1 = n(x)
        F = 0
        soma = 0
        v=0
        for i in n1:
            for j in range(len(n1)):
                v=((i[0]**2-i[0]**3)/2)*n1[j][1] + ((i[0]**2+i[0]**3)/2)
                F += f2(i[0], n1[j][0]) * v
            soma += F*i[1]
        return soma


    x = int(input("Digite a precis o de n s (6, 8 ou 10): "))
    soma1 = integraArea(x)
    soma2 = integraVolume(x)
    print("A area da superf cie descrita", x, " n  : ",
        soma1, "\nO volume abaixo da regiao   ", soma2)
```

2.4 Exemplo 4

Vamos calcular o volume da calota esférica proveniente da revolução de uma curva sobre o eixo y.

$$V = 2\pi \int \int_R d_g(x, y)\, dx\, dy \tag{9}$$

$$R : 0 \leq x \leq e(-y^2), -1 \leq y \leq 1 \tag{10}$$

Foi proposta a seguinte implementação para a resolução do problema:

```
    import numpy as np
    import math

    from le_arquivo import n


    def f1(x, y):
        return np.exp(-y**2)-((x**2+y**2)/4)


    def integra(x):
        n1 = n(x)
        F = 0
        soma = 0
        for i in n1:
            for j in range(len(n1)):
                F += f1(i[0], n1[j][0]) * n1[j][1]
            soma += F*i[1]
        return 2*np.pi*soma


    x = int(input("Digite a precis o de n s (6, 8 ou 10): "))
    soma = integra(x)
    print("O volume calculado com n", x, "   : ", soma)
```