UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO

RELATÓRIO DE IMPLEMENTAÇÃO NUMÉRICA: DIFERENÇAS FINITAS E SOMA DE RIEMANN

Elis Vasconcelos Berti Sanjuan[1]
Carlos Antônio Freitas

1. INTRODUÇÃO

Nesta atividade, desenvolvi dois códigos em Python com o objetivo de aplicar na prática os conteúdos estudados em sala de aula sobre aproximação de derivadas e cálculo de integrais definidas. A primeira parte da atividade envolve o uso de diferenças finitas para estimar derivadas e aplicar essa técnica ao método de Newton-Raphson para encontrar raízes de funções. A segunda parte consiste na implementação da Soma de Riemann para obter aproximações numéricas de integrais.

Antes de apresentar os códigos e suas análises, farei uma exposição teórica sobre os fundamentos matemáticos de cada tema, abordando definições, fórmulas, propriedades e resultados relevantes relacionados às diferenças finitas e às somas de Riemann.

2. DIFERENÇAS FINITAS

A diferença finita é uma técnica utilizada para aproximar a derivada de uma função $f(x)$ em um ponto, usando apenas valores da função próximos a esse ponto. É muito útil quando não se conhece a expressão analítica da derivada ou quando se deseja resolver numericamente um problema.

2.1 Definições

- **Diferença progressiva**:

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

- **Diferença regressiva**:

[1] Graduanda em Engenharia Eletrica na Universidade Federal do Vale do São Franscisco. E-mail: elis.sanjuan@discente.univasf.edu.br

$$f'(x) \approx \frac{f(x) - f(x-h)}{h}$$

- **Diferença centrada** (mais precisa, usada no seu código):

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

Esse método tem erro da ordem h^2, o que significa que quanto menor for o valor de h , maior será a precisão do resultado.

2.2 Aplicação no Método de Newton-Raphson

O método de Newton-Raphson é um processo iterativo para encontrar raízes de funções. A fórmula é:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Quando a derivada exata $f'(x)$ não está disponível, pode-se usar uma derivada aproximada por diferenças finitas. Isso torna o método aplicável mesmo em contextos numéricos.

2.3 Exemplo: DIferenças finitas com Newton-Raphson

Vamos aplicar o método de Newton-raphson para encontrar uma raiz da função

$$f(x) = x^2 - 2$$

Sabemos que a raiz exata é $\sqrt{2} = 1{,}4142$. usando derivada numérica por diferença centrada com $h = 10^{-5}$e chute inicial de $x_0 = 1.0$.

Passo a passo:
Definimos a função:

$$f(x)\, x^2 - 2$$

Escolhemos o valor inicial:

$$x_0 = 1$$

Calculamos $f(x_0) = 1$:

$$f(1.0) = 1.0^2 - 2 = -1.0$$

Aproximamos a derivada com a diferença centrada:

$$f(1.0) = \frac{\big(f(1.0 + h) - f(1.0 - h)\big)}{2h}$$

Com $h = 10^{-5}$:

$$f(1.0) = \frac{\big(f(1.00001) - f(0.99999)\big)}{2 \cdot 10^{-5}}$$

Atualizamos o valor de x usando Newton-Raphson:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 1.0 - \frac{(-1.0)}{2.0} = 1.5$$

Repetimos os passos com $x_1 = 1.5$ até que $|f(x_n)| < \epsilon \, com \, \epsilon \, = \, 10^{-6}$

Após algumas iterações, o código retorna:

- Raiz aproximada: 1.414214
- Número de iterações: 5

3. SOMA DE RIEMANN

A Soma de Riemann é uma técnica para aproximar integrais definidas. Se uma função $f(x)$ é contínua no intervalo $[a, \, b]$, a integral definida pode ser interpretada como a área sob a curva entre esses dois pontos. A soma de Riemann substitui essa área por retângulos e soma suas áreas.

3.1 Definição

Dado um número n

- Dividimos o intervalo $[a,\ b]$ em n partes de comprimento $\Delta x = \frac{b-a}{n}$
- Para cada subintervalo, escolhemos um ponto x_i^* e calculamos:

$$Sn = \sum_{i=0}^{n-1} f(x_i^*) \cdot \Delta x$$

Os principais tipos são:

- **Soma à esquerda**: $x_i^* = a + i \cdot \Delta x$
- **Soma à direita**:$x_i^* = a + (i+1) \cdot \Delta x$
- **Soma no ponto médio**: $x_i^* = a + (i + 0.5) \cdot \Delta x$ ← mais precisa, usada no seu código

3.2 Justificativa teórica

Conforme o número de subintervalos n Teorema Fundamental do Cálculo garante que:

$$\int_a^b f(x)dx = \lim_{n\to\infty} \quad Sn$$

3.3 Exemplo Soma de Riemann com Ponto Médio

Vamos aproximar a integral:

$$\int_0^1 x^2\, dx$$

Usando a soma de Riemann com ponto médio e $n = 10$ subintervalos. Sabemos que o valor exato é:

$$\int_0^1 x^2\, dx = \frac{1}{3} \approx 0{,}3333$$

Passo a passo:

Função:

$$f(x) \ = \ x^2$$

Intervalo de integração:

$$[a, b] = [0,1],\ n = 10$$

Calculamos o tamanho de cada subintervalo:

$$\Delta x = \frac{(b - a)}{n} = \frac{(1 - 0)}{10} = 0.1$$

Calculamos os pontos médios e a área de cada retângulo:

Para $i = 0\ a\ 9$:

- $x_i = a + (i + 0.5)\Delta x$
- $f(x_i) = (x_i)^2$
- Área do retângulo:$f(xi) \cdot \Delta x$

Exemplos de alguns cálculos:

- $x_0 = 0.05 \rightarrow f(0.05) = 0.0025 \rightarrow A_0 = 0.0025 \cdot 0.1 = 0.00025$
- $x_1 = 0.15 \rightarrow f(0.15) = 0.0225 \rightarrow A_1 = 0.00225$
- ...
- $x_9 = 0.95 \rightarrow f(0.95) = 0.9025 \rightarrow A_9 = 0.09025$

Somando todas as áreas:

$$S_{10} = \sum_{i=0}^{9} f(x_i) \cdot \Delta x \approx 0.332500$$

O valor obtido foi:

- Aproximação: 0,332500

- Erro em relação ao valor exato (~0.3333) muito pequeno, evidenciando a precisão da soma de Riemann com ponto médio mesmo com poucos subintervalos.

4. DERIVADAS COM DIFERENÇAS FINITAS E MÉTODO DE NEWTON

A primeira parte do trabalho consistiu em criar duas funções que trabalham juntas: uma que calcula a derivada de forma aproximada utilizando diferenças centrais, e outra que aplica o método de Newton-Raphson, substituindo a derivada analítica pela aproximada.

A primeira função foi:

```
def diferenca_central(f, x, h):
   return (f(x + h) - f(x - h)) / (2 * h)
```

Essa fórmula é uma das mais utilizadas para estimar derivadas numericamente e apresenta boa precisão, principalmente com h pequeno.

A segunda função foi:

```
def newton_iterativo(f, x0, epsilon=1e-6, max_iter=100)
```

Ela implementa o método de Newton-Raphson, que busca encontrar raízes de funções pela fórmula:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \text{ ou } x_{novo} = x - \frac{f(x\ \ )}{f'(x\ \ )}$$

Como eu não utilizo a derivada exata da função, substituo $f'(x)$ pela derivada numérica gerada pela " ***diferenca_central*** ".
O código imprime os valores atualizados de x a cada iteração, e termina quando $|f(x)| < \epsilon$ ou ao alcançar o número máximo de iterações.

4.1 Testes e Resultados: Diferenças Finitas com Newton

4.1.1 Teste 1: Função $f(x) = x^2 - 2$, chute inicial: 1.0

```
f = lambda x: x**2 - 2
```

A raiz esperada é $\sqrt{2} \approx 1.414214$. O código retornou:

```
Raiz aproximada: 1.414214
```

O valor foi atingido rapidamente e com alta precisão, confirmando a estabilidade do método mesmo com a derivada aproximada.

4.1.2 Teste 2: Função $f(x) = \cos(x) - x$, chute inicial: 0.5

```
from math import cos
f = lambda x: cos(x) - x
```

Essa função possui uma raiz conhecida próxima de 0.739085. O resultado foi:

```
Raiz aproximada: 0.739085
```

O método também funcionou muito bem com funções transcendentais. A convergência foi rápida e o valor obtido foi praticamente exato

4.1.3 Teste 3: Função $f(x) = x^3 - 3x + 2$, chute inicial: -2.0

```
f = lambda x: x**3 – 3*x + 2
```

Sabemos que uma das raízes reais dessa função é $x = -2$. O código retornou:

```
Raiz aproximada: -2.000000
```

Mais uma vez, o método foi eficaz e convergiu para a raiz correta com rapidez, mesmo em uma função polinomial de grau 3.

5. CÁLCULO DE INTEGRAL PELA SOMA DE RIEMANN

A primeira função que implementei tem como finalidade aproximar numericamente o valor de uma integral definida, utilizando a técnica da Soma de Riemann com ponto médio. O princípio dessa abordagem é dividir o intervalo de integração $[a,\ b]$ em n subintervalos de mesma largura, e somar as áreas dos retângulos formados pela altura da função avaliada no ponto médio de cada subintervalo.

A estrutura da função que criei é a seguinte:

```
def soma_riemann(f, a, b, n=100):
```

Ela recebe como parâmetros:

- f : a função a ser integrada (passada como lambda);
- a e b : os extremos inferior e superior do intervalo de integração;
- n : o número de subdivisões do intervalo (quanto maior, maior a precisão).

A lógica é simples: primeiro, cálculo a largura dos subintervalos, depois percorro cada um deles utilizando um laço for . Em cada iteração, determino o ponto médio x_i, avalio $f(x_i)$, multiplico pela largura Δx e somo todas essas áreas parciais.

5.1 Testes e Resultados: Soma de Riemann

5.1.1 Teste 1: Função $f(x) = x^2$ no intervalo $[0,\ 1]$

f = lambda x: x**2

a = 0

b = 1

n = 10

Neste caso, a integral exata é $\frac{1}{3} \approx 0{,}333$. Ao rodar o código, obtive:

Valor aproximado da integral: 0.332500

Mesmo com apenas 10 subintervalos, o valor obtido foi muito próximo do exato. Isso mostra que o ponto médio oferece boa aproximação com poucos cálculos.

5.1.2 Teste 2: Função$f(x) = \sin(x)$, no intervalo $[0, \pi]$

```
from math import sin, pi
f = lambda x: sin(x)
a = 0
b = pi
n = 100
```

A integral de sin(x) de 0 a π é 2. O resultado foi:

Valor aproximado da integral: 1.999832

Aqui, o número de subintervalos maior $(n = 100)$ tornou a aproximação praticamente perfeita. Isso mostra que o método é eficiente mesmo para funções trigonométricas.

5.1.3 Teste 3: Função $f(x) = e^x$, no intervalo $[0, 1]$

```
from math import exp
f = lambda x: exp(x)
a = 0
b = 1
n = 50
```

Sabemos que $\int_0^1 e^x \, dx = e - 1 \approx 1.718281$. O código retornou:

Valor aproximado da integral: 1.718281

Neste exemplo, o valor foi praticamente idêntico ao valor real, evidenciando a precisão do algoritmo implementado para funções exponenciais.

6. CONSIDERAÇÕES FINAIS

Com esta atividade, consegui aplicar de forma prática os conceitos de derivada e integral definidos numericamente, utilizando as técnicas de diferenças finitas e Soma de Riemann. Os testes realizados comprovaram que as implementações estão corretas e funcionam bem para diversos tipos de funções, com excelente precisão mesmo em casos mais complexos.

Essas ferramentas, mesmo simples, são poderosas e muito úteis para resolver problemas reais em que a solução analítica é difícil ou impossível de obter. Além disso, reforçam minha compreensão sobre o funcionamento prático dos conceitos estudados em Cálculo I, conectando teoria e programação.