



**UNIVERSIDADE FEDERAL DO VALE DO SÃO
FRANCISCO COLEGIADO DE ENGENHARIA
ELETRICA**

**LUCIVANIO DE SOUZA OLIVEIRA JUNIOR
TAYLA DE SOUZA MIRANDA**

Juazeiro-BA

2025

IMPLEMENTAÇÃO DA APROXIMAÇÃO DE DERIVADA DE FUNÇÃO UTILIZANDO DIFERENÇAS FINITAS

Trabalho apresentado no curso
de bacharelado em Engenharia
Elétrica na Universidade
Federal do Vale do São
Francisco.

Professor: Carlos Freitas

Juazeiro-BA

2025

1. O que é a Aproximação de Derivadas usando Diferenças Finitas

As diferenças finitas são técnicas utilizadas para aproximar derivadas de funções com base em valores discretos. São amplamente usadas em métodos numéricos e aplicações computacionais, especialmente na engenharia e física.

Fórmulas Principais:

- Diferença Progressiva: $f'(x) \approx (f(x+h) - f(x)) / h$
- Diferença Regressiva: $f'(x) \approx (f(x) - f(x-h)) / h$
- Diferença Centrada: $f'(x) \approx (f(x+h) - f(x-h)) / (2h)$

Aplicações:

- Aproximação de derivadas em pontos específicos
- Resolução numérica de equações diferenciais
- Simulações computacionais em física e engenharia

Vantagens e Limitações:

- Vantagens: Simplicidade, implementação direta, aplicável a dados experimentais
- Limitações: Erros de truncamento, sensibilidade ao passo h

2. Exemplos utilizados na pratica

CODIGO:

```
1  import math
2
3  def derivada_diferencas_finitas(f, x, h=1e-5, tipo='centrada'):
4      if tipo == 'progressiva':
5          return (f(x + h) - f(x)) / h
6      elif tipo == 'regressiva':
7          return (f(x) - f(x - h)) / h
8      elif tipo == 'centrada':
9          return (f(x + h) - f(x - h)) / (2 * h)
10     else:
11         raise ValueError("Tipo inválido")
12
13     def func(x):
14         return math.sin(x) # exemplo: sin(x)
15
16     x = math.pi / 4
17     h = 0.0001
18
19     resultado = derivada_diferencas_finitas(func, x, h, tipo='centrada')
20
21     print(resultado)
```

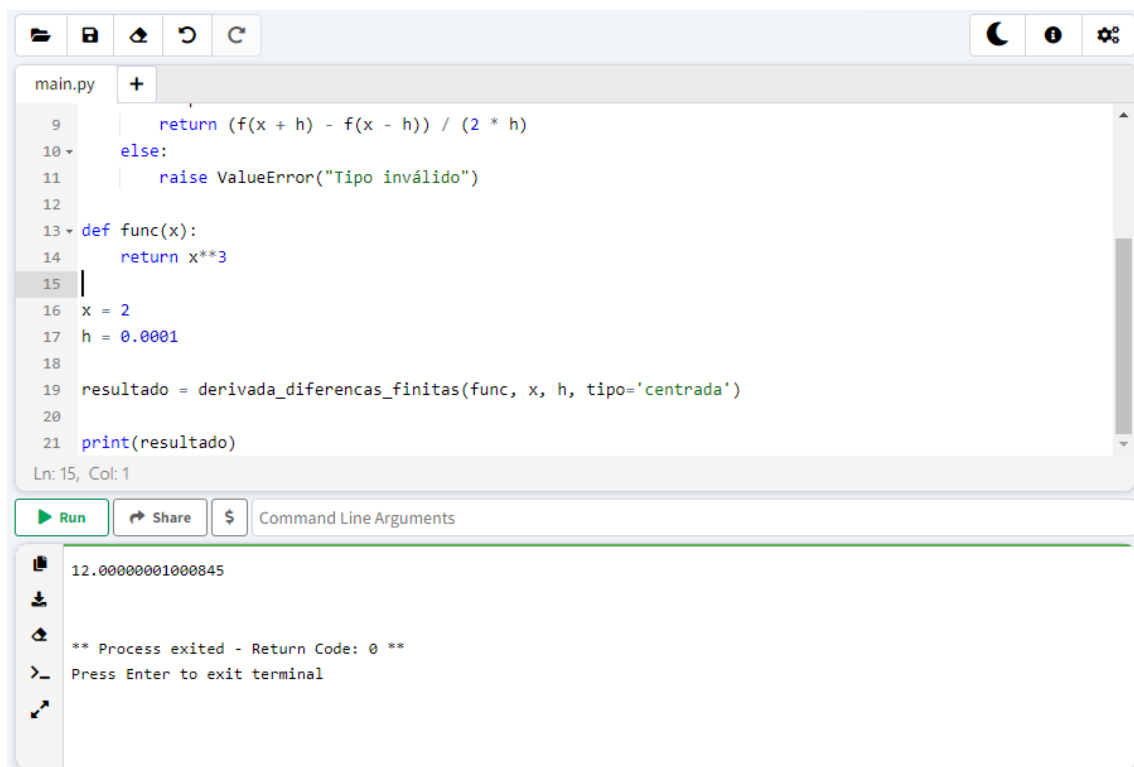
Link Github : <https://github.com/LucivaniaJunior/Codigos-de->

2.2 Testes usados nos Códigos:

2.2.1

$$f(x) = x^3 \quad x = 2$$

Resultado Esperado: 12



The screenshot shows a code editor with a file named 'main.py'. The code defines a function 'derivada_diferencas_finitas' that calculates the derivative of a function 'f' at point 'x' using a central difference method with step size 'h'. It also defines a function 'func(x)' that returns 'x**3'. The script sets 'x = 2' and 'h = 0.0001', then calls the derivative function and prints the result.

```
9         return (f(x + h) - f(x - h)) / (2 * h)
10     else:
11         raise ValueError("Tipo inválido")
12
13 def func(x):
14     return x**3
15
16 x = 2
17 h = 0.0001
18
19 resultado = derivada_diferencas_finitas(func, x, h, tipo='centrada')
20
21 print(resultado)
```

Ln: 15, Col: 1

Run Share Command Line Arguments

12.00000001000845

** Process exited - Return Code: 0 **

Press Enter to exit terminal

Alterações Usadas:

Pontos:

x: 2

h: 0,0001

Função Matemática:

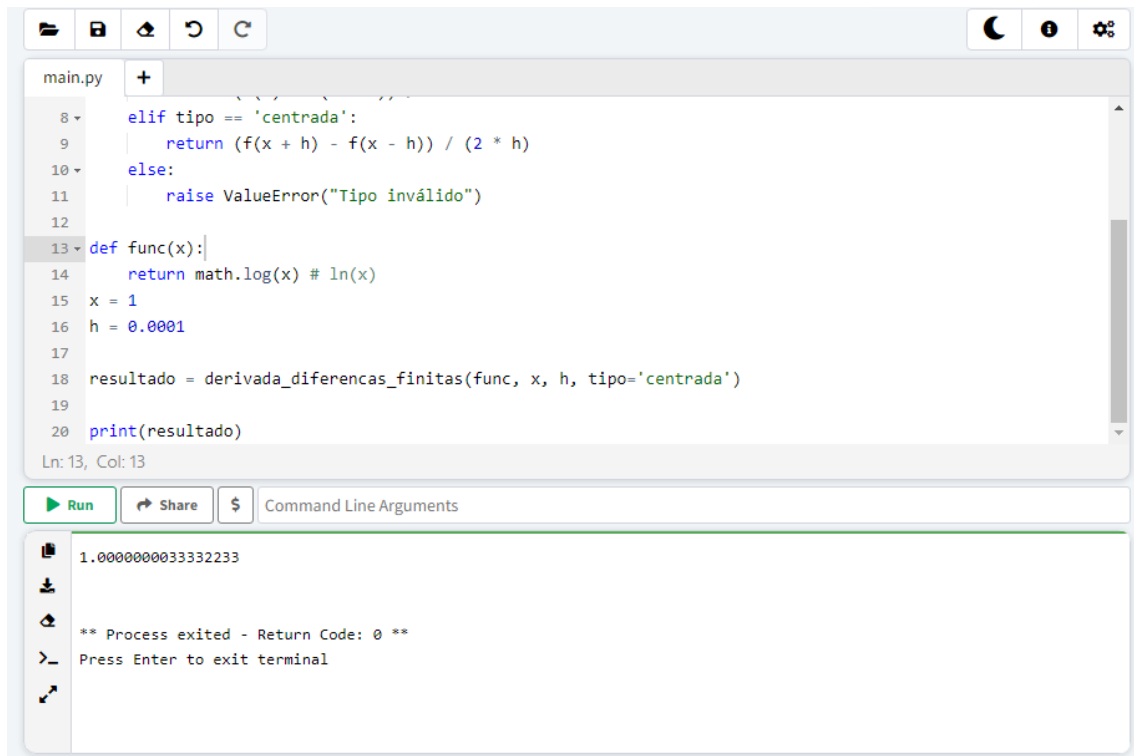
x^{**3}

Resultado Obtido : 12.00000001000845

2.2.2

$f(x) = \ln(x)$ $x = 1$

Resultado Esperado: 1



```
main.py +
8 elif tipo == 'centrada':
9     return (f(x + h) - f(x - h)) / (2 * h)
10 else:
11     raise ValueError("Tipo inválido")
12
13 def func(x):
14     return math.log(x) # ln(x)
15 x = 1
16 h = 0.0001
17
18 resultado = derivada_diferencas_finitas(func, x, h, tipo='centrada')
19
20 print(resultado)
Ln: 13, Col: 13

Run Share $ Command Line Arguments

1.0000000033332233

** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Alterações Usadas:

Pontos:

x: 1

h: 0,0001

Função Matemática:

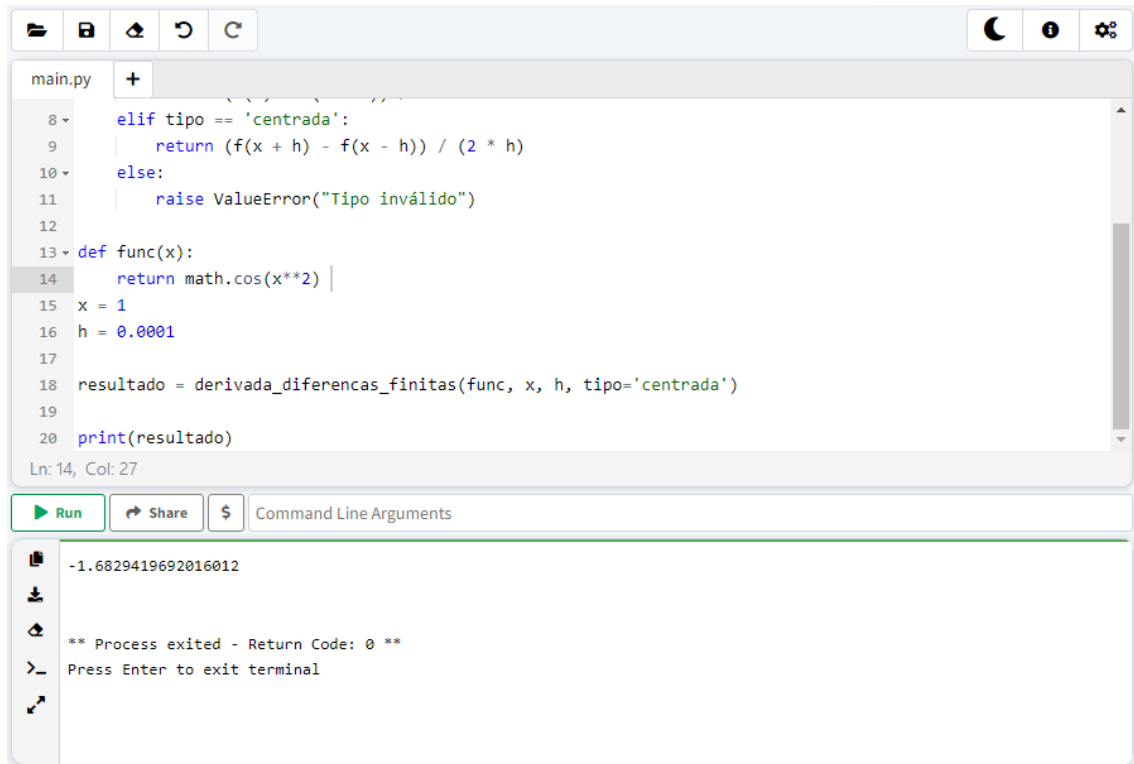
`math.log(x)`

Resultado Obtido: 1.0000000033332233

2.2.3

$f(x) = \cos(x^2)$ $x = 1$

Resultado Esperado: -1,683



```
main.py +
8 elif tipo == 'centrada':
9     return (f(x + h) - f(x - h)) / (2 * h)
10 else:
11     raise ValueError("Tipo inválido")
12
13 def func(x):
14     return math.cos(x**2)
15 x = 1
16 h = 0.0001
17
18 resultado = derivada_diferencas_finitas(func, x, h, tipo='centrada')
19
20 print(resultado)
Ln: 14, Col: 27

Run Share Command Line Arguments

-1.6829419692016012
** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Alterações Usadas:

Pontos:

x: 1

h: 0,0001

Função Matemática:

x^{**2}

Resultado Obtido: -1.6829419692016012

IMPLEMENTAÇÃO DA SOMA DE RIEMMAN PARA CÁLCULO DE INTEGRAL

Trabalho apresentado no curso de bacharelado em Engenharia Elétrica na Universidade Federal do Vale do São Francisco.

Professor: Carlos Freitas

Juazeiro-BA

2025

3. Soma de RIEMMAN e sua utilização no calculo de derivadas

A soma de Riemann é um método numérico usado para aproximar o valor de uma integral definida. Ela consiste em dividir a área sob uma curva em retângulos e somar suas áreas.

Tipos de Soma de Riemann:

- Pela esquerda: Usa o valor da função no início de cada subintervalo
- Pela direita: Usa o valor da função no final de cada subintervalo
- Pelo ponto médio: Usa o valor da função no ponto médio de cada subintervalo

Aplicações:

- Cálculo de áreas sob curvas
- Aproximação de integrais definidas
- Cálculos em física, economia e estatística

Ligação com o Conceito de Integral:

À medida que o número de subintervalos aumenta (e sua largura tende a zero), a soma de Riemann se aproxima do valor exato da integral definida.

4. Exemplos utilizados na pratica

```
Code Blame 26 lines (22 loc) · 583 Bytes
1 def soma_riemann(f, a, b, n, metodo='meio'):
2     largura = (b - a) / n
3     soma = 0.0
4     for i in range(n):
5         if metodo == 'esquerda':
6             x = a + i * largura
7         elif metodo == 'direita':
8             x = a + (i + 1) * largura
9         elif metodo == 'meio':
10            x = a + (i + 0.5) * largura
11        else:
12            raise ValueError("Método inválido")
13        soma += f(x)
14    return soma * largura
15
16 import math
17
18 def funcao(x):
19     return math.sin(x)
20
21 a = 0
22 b = math.pi
23 n = 1000
24
25 resultado = soma_riemann(funcao, a, b, n, metodo='meio')
26 print(resultado)
```

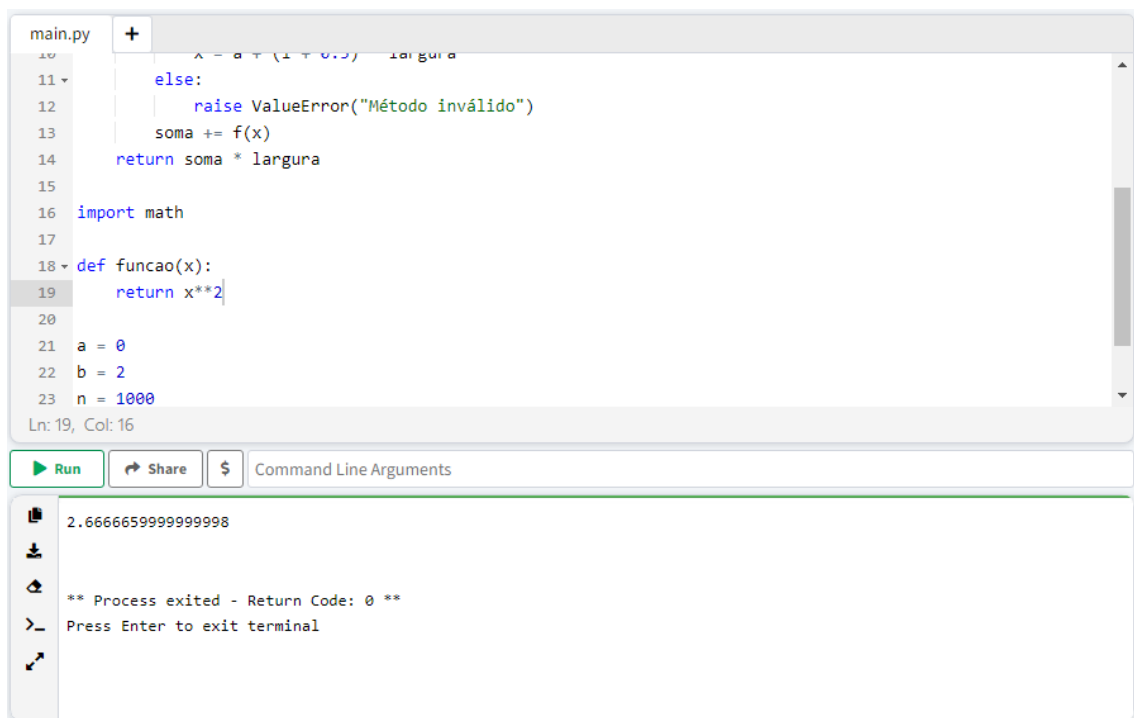
Link do Github: <https://github.com/LucivaniaJunior/Codigos-de-Calculo1/blob/main/Soma%20de%20RIEMANN>

4.2 Testes usados nos Códigos

4.2.1

$f(x) = x^2$ de 0 a 2

Resultado Esperado: 2,6667



```
main.py +
10 x = a + (1 + 0.5) * largura
11 else:
12     raise ValueError("Método inválido")
13 soma += f(x)
14 return soma * largura
15
16 import math
17
18 def funcao(x):
19     return x**2
20
21 a = 0
22 b = 2
23 n = 1000
Ln: 19, Col: 16

Run Share Command Line Arguments

2.6666659999999998

** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Alterações Usadas

Pontos:

a: 0

b: 2

n: 1000

Função Matemática:

x^2 (X^2)

Resultado Obtido: 2.6666659999999998

4.2.1

$f(x) = \sin(x)$ de 0 a π

Resultado Esperado: 2

```
main.py +
10 x = a + (1 + 0.5) * largura
11 else:
12     raise ValueError("Método inválido")
13 soma += f(x)
14 return soma * largura
15
16 import math
17
18 def funcao(x):
19     return math.sin(x)
20
21 a = 0
22 b = math.pi
23 n = 1000
Ln: 22, Col: 12
```

Run Share Command Line Arguments

2.0000008224672676

** Process exited - Return Code: 0 **

Press Enter to exit terminal

Alterações Usadas

Pontos:

a: 0

b: math.pi (valor de pi)

n: 1000

Função Matemática:

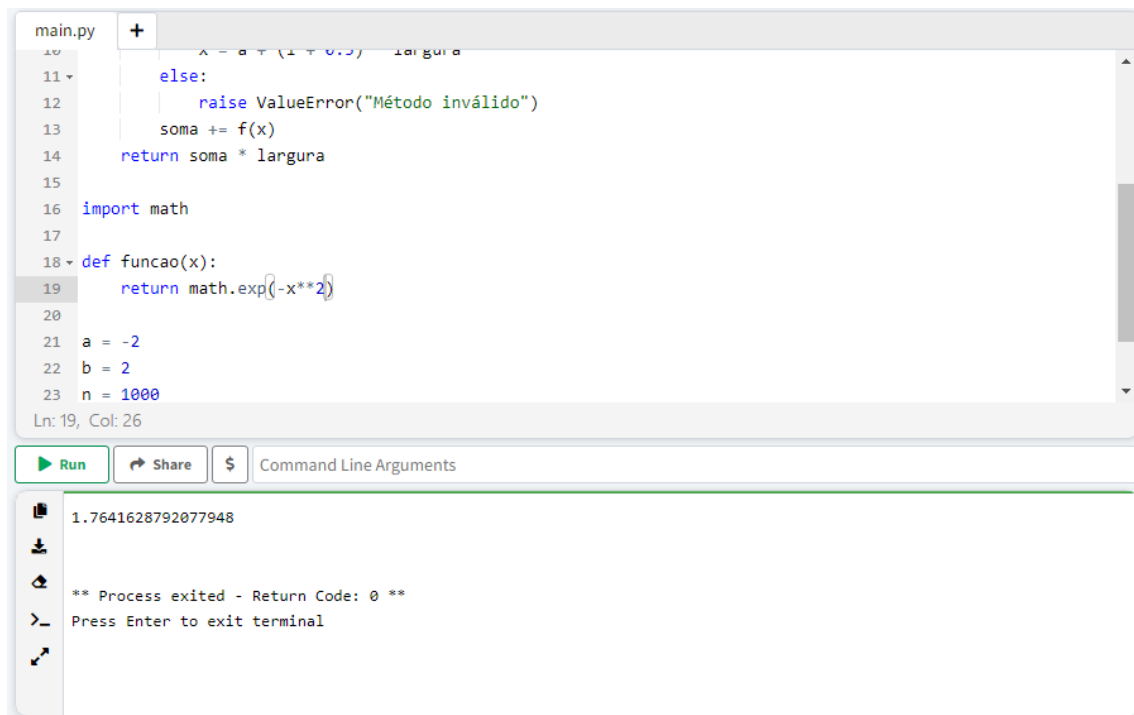
math.sin(x)

Resultado Obtido: 2.0000008224672676

4.2.3

$$f(x) = e^{-x^2} \quad \text{de } -2 \text{ a } 2$$

Resultado Esperado: 1,7640



```
main.py +
10     lambda = a + (1 + 0.5) * largura
11     else:
12         raise ValueError("Método inválido")
13     soma += f(x)
14     return soma * largura
15
16 import math
17
18 def funcao(x):
19     return math.exp(-x**2)
20
21 a = -2
22 b = 2
23 n = 1000
Ln: 19, Col: 26

Run Share $ Command Line Arguments

1.7641628792077948

** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Alterações Usadas:

Pontos:

a: -2

b: 2

n: 1000

Função Matemática:

`math.exp(-x**2)`

Resultado Obtido : 1.7641628792077948

5. CONCLUSÃO

Diferenças Finitas: Usada para aproximar derivadas.

Soma de Riemann : Usada para aproximar integrais (Áreas sob curvas).