Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTAMENTO DE / DEPARTMENT OF
INFORMÁTICA E SISTEMAS

MáquinaCDI

Relatório de Licenciatura em / in Engenharia Informática

Autor / Author

António Domingos Gonçalves Pedroso

António Miguel Grangeiro Rocha

Samuel Frazão Pinto Costa

Supervisor

Arménio António da Silva Correia

Coimbra, Maio 2025

INSTITUTO POLITÉCNICO DE
COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Resumo

Este relatório apresenta os resultados da Atividade 5, focada no desenvolvimento e ampliação de uma Máquina de Cálculo Diferencial e Integral (CDI) implementada em Matlab. Foram implementadas funções para cálculo de derivadas por diferenças finitas (progressivas, regressivas e centradas, incluindo a segunda derivada), integração numérica pelas regras dos Trapézios e de Simpson, e verificação de funções harmônicas de duas variáveis reais. A Máquina CDI foi completada e ampliada com funcionalidades adicionais, incluindo uma aba para conversão entre coordenadas cartesianas, polares e esféricas, e uma aba interativa que utiliza a API do ChatGPT para responder perguntas sobre a aplicação. As implementações foram realizadas em Matlab, com ênfase na precisão numérica e na eficiência computacional. O relatório detalha a fundamentação teórica, os algoritmos implementados, os resultados obtidos e as extensões propostas, contribuindo para a consolidação de competências em análise numérica e modelação matemática.

Abstract

This report presents the outcomes of Activity 5, focused on the development and enhancement of a Differential and Integral Calculus Machine (CDI) implemented in Matlab. Functions were developed for derivative calculations using finite difference methods (forward, backward, and centered, including second derivatives), numerical integration via the Trapezoid and Simpson's rules, and verification of harmonic functions of two real variables. The CDI Machine was completed and extended with additional features, including a tab for conversion between Cartesian, polar, and spherical coordinates, and an interactive tab utilizing the ChatGPT API to answer questions about the application. The implementations were carried out in Matlab, emphasizing numerical accuracy and computational efficiency. The report details the theoretical foundation, implemented algorithms, obtained results, and proposed extensions, contributing to the consolidation of skills in numerical analysis and mathematical modeling.

Epígrafe

Tzzzzz Tzzzzzzz Tzzzzzz.
Arménio, 2025

Agradecimentos

Arménio António da Silva Correia

Rui Manuel Carreira Rodrigues

Índice

Índice de figuras

Lista de Abreviaturas

a	Limite inferior do intervalo de integração ou derivação
b	Limite superior do intervalo de integração ou derivação
CDI	Máquina de Cálculo Diferencial e Integral
$f(x)$	Função real de uma variável usada em derivação ou integração
$f(x, y)$	Função real de duas variáveis usada em verificação de harmonicidade
h	Tamanho do passo (step size) na partição uniforme
I_T	Aproximação da integral pela Regra dos Trapézios
I_S	Aproximação da integral pela Regra de Simpson
ISEC	Instituto Superior de Engenharia de Coimbra
MATLAB	*Matrix Laboratory*
M_2	Máximo absoluto da segunda derivada no intervalo $[a, b]$
M_4	Máximo absoluto da quarta derivada no intervalo $[a, b]$
r	Distância radial da origem em coordenadas esféricas
x	Coordenada Cartesiana ao longo do eixo horizontal
y	Coordenada Cartesiana ao longo do eixo vertical
z	Coordenada Cartesiana ao longo do eixo de altura ou radial em coordenadas cilíndricas/esféricas
x_k	Ponto da partição uniforme, onde $x_k = a + kh$
ρ	Distância radial no plano xy em coordenadas polares ou cilíndricas
θ	Ângulo azimutal no plano xy (polar/cilíndrico) ou ângulo de elevação em coordenadas esféricas (em radianos)
ϕ	Ângulo azimutal no plano xy em coordenadas esféricas (em radianos)

1 Introdução

Os métodos numéricos para cálculo diferencial e integral são ferramentas essenciais na resolução de problemas em diversas áreas da ciência e engenharia, permitindo a aproximação de derivadas e integrais de funções quando soluções analíticas não são viáveis. A Atividade 5 teve como objetivo implementar e ampliar uma Máquina de Cálculo Diferencial e Integral (CDI) em Matlab, abordando técnicas de diferenças finitas, integração numérica e análise de funções harmônicas, com foco em funções reais de uma e duas variáveis.

A atividade foi dividida em quatro partes principais. Na Parte 01, foram implementadas fórmulas de diferenças finitas (progressivas, regressivas e centradas) para cálculo de derivadas primeira e segunda em pontos discretos. Na Parte 02, desenvolveram-se algoritmos para integração numérica usando as regras dos Trapézios e de Simpson, com ênfase na precisão e no controle de erros. A Parte 03 envolveu a verificação de funções harmônicas de duas variáveis reais, uma propriedade fundamental em áreas como física e engenharia. Na Parte 04, a Máquina CDI foi completada e ampliada com funcionalidades adicionais, incluindo uma aba para conversão entre sistemas de coordenadas cartesianas, polares e esféricas, e uma aba interativa que utiliza a API do ChatGPT para responder perguntas sobre a aplicação, agregando interatividade e dinamismo à ferramenta.

As extensões propostas, como a conversão de coordenadas e a integração com a API do ChatGPT, visam aumentar a versatilidade da Máquina CDI, permitindo aplicações em problemas geométricos e facilitando a interação com usuários por meio de respostas automáticas baseadas em inteligência artificial. Todas as implementações foram realizadas em Matlab, aproveitando sua robustez para cálculos numéricos e visualização de resultados.

Este relatório está organizado da seguinte forma: a seção seguinte apresenta a fundamentação teórica dos métodos implementados, seguida por capítulos dedicados às Partes 01 a 04, detalhando os algoritmos, implementações e resultados. Por fim, a conclusão sintetiza as contribuições da atividade, e a bibliografia lista as referências utilizadas.

2 Cálculo Diferencial e Integral

O cálculo diferencial e integral é fundamental para a modelação matemática de fenómenos em ciência e engenharia, permitindo a análise de taxas de variação e acumulação de quantidades. Seja $f(x)$ uma função real de uma variável definida em $[a, b]$, ou $f(x, y)$ uma função de duas variáveis, o objetivo é calcular derivadas (e.g., $f'(x)$, $f''(x)$, ou $\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$) e integrais (e.g., $\int_a^b f(x)\,dx$) para descrever comportamentos dinâmicos ou propriedades geométricas. Estes cálculos são aplicados em áreas como sistemas mecânicos (e.g., mola-massa), física geométrica (e.g., conversão de coordenadas) e análise de campos (e.g., funções harmônicas).

Na prática, soluções analíticas para derivadas e integrais podem ser inviáveis, especialmente quando apenas valores discretos da função são conhecidos ou quando se trata de funções complexas de duas variáveis. Assim, recorre-se a métodos numéricos, como diferenças finitas para aproximação de derivadas, regras de integração numérica (e.g., Trapézios e Simpson) para cálculo de integrais, e técnicas de análise para verificar propriedades como a harmonicidade.

$$\begin{cases} \text{Derivada: } f'(x_k) \approx \frac{f(x_{k+1}) - f(x_k)}{h} \text{ (diferença progressiva)} \\ \text{Integral: } \int_a^b f(x)\,dx \approx \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right] \text{ (regra dos Trapézios)} \\ \text{Harmonicidade: } \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = 0 \end{cases} \tag{2.1}$$

Esta atividade foca na implementação destes métodos em Matlab para desenvolver uma Máquina de Cálculo Diferencial e Integral (CDI). A máquina foi ampliada com funcionalidades adicionais, incluindo uma aba para conversão entre coordenadas cartesianas, polares e esféricas, facilitando aplicações geométricas, e uma aba interativa que utiliza a API do ChatGPT para responder perguntas sobre a aplicação, promovendo maior interatividade e acessibilidade.

3 Diferenças Finitas em 2 Pontos

Fórmulas

As fórmulas de diferenças finitas em 2 pontos são métodos numéricos de primeira ordem para aproximar a derivada primeira de uma função $f(x)$ definida em um intervalo $[a, b]$, conhecida em uma partição uniforme $a = x_0 < x_1 < \cdots < x_n = b$, com passo $h = \frac{b-a}{n}$. Estas fórmulas são essenciais na Máquina de Cálculo Diferencial e Integral (CDI), permitindo estimar taxas de variação em aplicações como física, engenharia e modelação matemática.

As fórmulas de diferenças finitas em 2 pontos são dadas por:

$$\text{Progressiva: } f'(x_k) \approx \frac{f(x_{k+1}) - f(x_k)}{h}, \quad k = 0, 1, \ldots, n-1 \tag{3.1}$$

$$\text{Regressiva: } f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{h}, \quad k = 1, 2, \ldots, n \tag{3.2}$$

onde:

- $f'(x_k)$ → Aproximação da derivada primeira no ponto x_k;
- $f(x_k)$ → Valor da função no ponto x_k;
- h → Tamanho do passo, dado por $h = \frac{b-a}{n}$;
- $x_k = a + kh$ → Pontos da partição uniforme;
- k → Índice do ponto na partição.

A fórmula progressiva usa o ponto seguinte (x_{k+1}) para estimar a derivada, enquanto a regressiva usa o ponto anterior (x_{k-1}). Ambas têm erro de truncatura proporcional a h, sendo simples, mas menos precisas que métodos de ordem superior.

Algoritmo/Função

Algoritmo:

1. Definir a função f, o intervalo $[a, b]$, o tamanho do passo h, e, opcionalmente, o vetor de valores $y = f(x)$;
2. Criar o vetor de pontos $x = [x_0, x_1, \ldots, x_n]$, onde $x_k = a + kh$;
3. Se y não for fornecido, avaliar f nos pontos x_k, armazenando em y;

4. Para a fórmula progressiva:
   - Calcular $f'(x_k) = \frac{y(k+1)-y(k)}{h}$ para $k = 1, \ldots, n-1$;
   - Definir $f'(x_n) = \frac{y(n)-y(n-1)}{h}$ (usando diferença regressiva no último ponto).
5. Para a fórmula regressiva:
   - Calcular $f'(x_1) = \frac{y(2)-y(1)}{h}$;
   - Calcular $f'(x_k) = \frac{y(k)-y(k-1)}{h}$ para $k = 2, \ldots, n$.
6. Retornar os vetores x, y, e as derivadas aproximadas $dydx$.

As implementações em Matlab das fórmulas de diferenças finitas em 2 pontos (progressiva e regressiva) são apresentadas abaixo:

```
function [x, y, dydx] = NDerivacaoDFP(~, f, a, b, h, y)
    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1, n);
    for k = 1:n-1
        dydx(k) = (y(k+1) - y(k)) / h;
    end
    dydx(n) = (y(n) - y(n-1)) / h;
end

function [x, y, dydx] = NDerivacaoDFR(~, f, a, b, h, y)
    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1, n);
    dydx(1) = (y(2) - y(1)) / h;
    for i = 2:n
        dydx(i) = (y(i) - y(i-1)) / h;
    end
end
```

4 Diferenças Finitas em 3 Pontos

Fórmulas

As fórmulas de diferenças finitas em 3 pontos são métodos numéricos para aproximar a derivada primeira e segunda de uma função $f(x)$ definida em um intervalo $[a, b]$, conhecida em uma partição uniforme $a = x_0 < x_1 < \cdots < x_n = b$, com passo $h = \frac{b-a}{n}$. Estas fórmulas oferecem maior precisão que as fórmulas em 2 pontos, sendo amplamente utilizadas na Máquina de Cálculo Diferencial e Integral (CDI) para aplicações em física, engenharia e modelação matemática.

As fórmulas de diferenças finitas em 3 pontos são dadas por:

$$\text{Progressiva: } f'(x_k) \approx \frac{-3f(x_k) + 4f(x_{k+1}) - f(x_{k+2})}{2h}, \quad k = 0, 1, \ldots, n-2 \tag{4.1}$$

$$\text{Regressiva: } f'(x_k) \approx \frac{f(x_{k-2}) - 4f(x_{k-1}) + 3f(x_k)}{2h}, \quad k = 2, 3, \ldots, n \tag{4.2}$$

$$\text{Centrada: } f'(x_k) \approx \frac{f(x_{k+1}) - f(x_{k-1})}{2h}, \quad k = 1, 2, \ldots, n-1 \tag{4.3}$$

$$\text{Segunda Derivada: } f''(x_k) \approx \frac{f(x_{k+1}) - 2f(x_k) + f(x_{k-1})}{h^2}, \quad k = 1, 2, \ldots, n-1 \tag{4.4}$$

onde:

- $f'(x_k)$, $f''(x_k)$ → Aproximações da derivada primeira e segunda no ponto x_k;
- $f(x_k)$ → Valor da função no ponto x_k;
- h → Tamanho do passo, dado por $h = \frac{b-a}{n}$;
- $x_k = a + kh$ → Pontos da partição uniforme;
- k → Índice do ponto na partição.

As fórmulas progressiva e regressiva em 3 pontos são de segunda ordem, com erro de truncatura proporcional a h^2, enquanto a fórmula centrada também é de segunda ordem, sendo mais precisa para pontos internos. A fórmula da segunda derivada é usada para estimar curvaturas, com erro proporcional a h^2.

Algoritmo/Função

Algoritmo:

1. Definir a função f, o intervalo $[a, b]$, o tamanho do passo h, e, opcionalmente, o vetor de valores $y = f(x)$;
2. Criar o vetor de pontos $x = [x_0, x_1, \ldots, x_n]$, onde $x_k = a + kh$;
3. Se y não for fornecido, avaliar f nos pontos x_k, armazenando em y;
4. Para a fórmula progressiva:
   - Calcular $f'(x_k) = \frac{-3y(k)+4y(k+1)-y(k+2)}{2h}$ para $k = 1, \ldots, n-2$;
   - Nos pontos finais ($k = n-1, n$), usar a fórmula regressiva em 3 pontos.
5. Para a fórmula regressiva:
   - Calcular $f'(x_k) = \frac{y(k-2)-4y(k-1)+3y(k)}{2h}$ para $k = 3, \ldots, n$;
   - Nos pontos iniciais ($k = 1, 2$), usar a fórmula progressiva em 3 pontos.
6. Para a fórmula centrada:
   - Calcular $f'(x_k) = \frac{y(k+1)-y(k-1)}{2h}$ para $k = 2, \ldots, n-1$;
   - Nos pontos extremos ($k = 1, n$), usar fórmulas progressiva ou regressiva.
7. Para a segunda derivada:
   - Calcular $f''(x_k) = \frac{y(k+1)-2y(k)+y(k-1)}{h^2}$ para $k = 2, \ldots, n-1$;
   - Nos pontos extremos ($k = 1, n$), estimar usando diferenças de primeiras derivadas.
8. Retornar os vetores x, y, e as derivadas aproximadas $dydx$.

As implementações em MATLAB das fórmulas de diferenças finitas em 3 pontos e da segunda derivada são apresentadas abaixo:

```
function [x, y, dydx] = NDerivacaoDFP3(~, f, a, b, h, y)
    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1, n);
    for i = 1:n-2
        dydx(i) = (-3*y(i) + 4*y(i+1) - y(i+2))/(2*h);
    end
    dydx(n-1) = (y(n-3) - 4*y(n-2) + 3*y(n-1))/(2*h);
    dydx(n) = (y(n-2) - 4*y(n-1) + 3*y(n))/(2*h);
```

```
end

function [x, y, dydx] = NDerivacaoDFR3(~, f, a, b, h, y)
    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1, n);
    dydx(1) = (-3*y(1) + 4*y(2) - y(3))/(2*h);
    dydx(2) = (-3*y(2) + 4*y(3) - y(4))/(2*h);
    for i = 3:n
        dydx(i) = (y(i-2) - 4*y(i-1) + 3*y(i))/(2*h);
    end
end

function [x, y, dydx] = NDerivacaoDFC(~, f, a, b, h, y)
    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1, n);
    dydx(1) = (-3*y(1) + 4*y(2) - y(3))/(2*h);
    for i = 2:n-1
        dydx(i) = (y(i+1) - y(i-1))/(2*h);
    end
    dydx(n) = (y(n-2) - 4*y(n-1) + 3*y(n))/(2*h);
end

function [x, y, dydx] = NDerivacaoD2(~, f, a, b, h, y)
    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1, n);
    temp1 = (-3*y(1) + 4*y(2) - y(3))/(2*h);
    temp2 = (-3*y(2) + 4*y(3) - y(4))/(2*h);
    temp3 = (-3*y(3) + 4*y(4) - y(5))/(2*h);
    dydx(1) = (-3*temp1 + 4*temp2 - temp3)/(2*h);
```

```
    for i = 2:n-1
        dydx(i) = (y(i+1) - 2*y(i) + y(i-1))/(h*h);
    end
    tempn2 = (y(n-4) - 4*y(n-3) + 3*y(n-2))/(2*h);
    tempn1 = (y(n-3) - 4*y(n-2) + 3*y(n-1))/(2*h);
    tempn = (y(n-2) - 4*y(n-1) + 3*y(n))/(2*h);
    dydx(n) = (tempn2 - 4*tempn1 + 3*tempn)/(2*h);
end
```

5 Integração Numérica

Fórmulas

A integração numérica é um método essencial para aproximar o valor de integrais definidas, $\int_a^b f(x)\,dx$, quando a integração analítica não é viável. As Regras dos Trapézios e de Simpson são técnicas amplamente utilizadas na Máquina de Cálculo Diferencial e Integral (CDI), oferecendo aproximações precisas para funções $f(x)$ em uma partição uniforme do intervalo $[a, b]$, com $a = x_0 < x_1 < \cdots < x_n = b$ e passo $h = \frac{b-a}{n}$. Estas fórmulas são aplicadas em áreas como física, engenharia e análise de dados.

As fórmulas de integração numérica são dadas por:

$$\text{Regra dos Trapézios: } I_T(f) = \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right] \tag{5.1}$$

$$|E_T| \leq \frac{b-a}{12}h^2 M_2, \quad M_2 = \max_{x\in[a,b]} |f''(x)| \tag{5.2}$$

$$\text{Regra de Simpson: } I_S(f) = \frac{h}{3}\left[f(x_0) + 4\sum_{\substack{i=1\\ i \text{ ímpar}}}^{n-1} f(x_i) + 2\sum_{\substack{i=2\\ i \text{ par}}}^{n-2} f(x_i) + f(x_n)\right] \tag{5.3}$$

$$|E_S| \leq \frac{b-a}{180}h^4 M_4, \quad M_4 = \max_{x\in[a,b]} |f^{(4)}(x)| \tag{5.4}$$

onde:

- $I_T(f)$, $I_S(f)$ → Aproximações da integral $\int_a^b f(x)\,dx$;
- $f(x_i)$ → Valor da função no ponto $x_i = a + ih$;
- h → Tamanho do passo, dado por $h = \frac{b-a}{n}$;
- n → Número de subintervalos (para Simpson, n deve ser par);
- $|E_T|$, $|E_S|$ → Erros de truncatura, dependentes das derivadas $f''(x)$ e $f^{(4)}(x)$;
- M_2, M_4 → Máximos absolutos da segunda e quarta derivadas no intervalo $[a, b]$.

A Regra dos Trapézios tem erro proporcional a h^2, enquanto a Regra de Simpson, mais precisa, tem erro proporcional a h^4, exigindo que n seja par para alternar os coeficientes

4 e 2.

Algoritmo/Função

Algoritmo (Regra dos Trapézios):

1. Definir a função f, os limites de integração $[a, b]$, e o número de subintervalos n;
2. Calcular o tamanho do passo $h = \frac{b-a}{n}$;
3. Avaliar $f(x_i)$ nos pontos $x_i = a + ih$, para $i = 1, \ldots, n-1$;
4. Somar $s = \sum_{i=1}^{n-1} f(x_i)$;
5. Calcular a aproximação: $T = \frac{h}{2}[f(a) + 2s + f(b)]$;
6. Calcular o valor exato (simbólico) usando $\int_a^b f(x)\,dx$;
7. Calcular o erro absoluto: $|T - \text{valor exato}|$;
8. Retornar T, o valor exato, e o erro.

Algoritmo (Regra de Simpson):

1. Definir a função f, os limites de integração $[a, b]$, e o número de subintervalos n (par);
2. Calcular o tamanho do passo $h = \frac{b-a}{n}$;
3. Avaliar $f(x_i)$ nos pontos $x_i = a + ih$, para $i = 1, \ldots, n-1$;
4. Calcular as somas: $s_{\text{ímpar}} = \sum_{\substack{i=1 \\ i \text{ ímpar}}}^{n-1} f(x_i)$, $s_{\text{par}} = \sum_{\substack{i=2 \\ i \text{ par}}}^{n-2} f(x_i)$;
5. Calcular a aproximação: $S = \frac{h}{3}[f(a) + 4s_{\text{ímpar}} + 2s_{\text{par}} + f(b)]$;
6. Calcular o valor exato (simbólico) usando $\int_a^b f(x)\,dx$;
7. Calcular o erro absoluto: $|S - \text{valor exato}|$;
8. Retornar S, o valor exato, e o erro.

As implementações em Matlab das Regras dos Trapézios e de Simpson, incluindo cálculo simbólico do valor exato e do erro, são apresentadas abaixo:

```
function [T, exactValue, errorValue] = Fregra_trapezios(~, f, a, b, n)
    h = (b - a)/n;
    s = sum( f(a + (1:n-1)*h) );
    T = (h/2)*( f(a) + 2*s + f(b) );

    syms x;
    funcaoTransformada = symfun( f(x), x );
    expressao = int(funcaoTransformada, x, a, b);
    exactValue = double(expressao);
```

```
    errorValue = abs(exactValue - T);
end

function [S, exactValue, errorValue] = Fregra_simpson(~, f, a, b, n)
    h = (b - a) / n;
    sumOdd  = 0;
    sumEven = 0;
    for i = 1:(n-1)
        xi = a + i*h;
        if mod(i,2) == 0
            sumEven = sumEven + f(xi);
        else
            sumOdd  = sumOdd  + f(xi);
        end
    end
    S = (h/3) * ( f(a) + 4*sumOdd + 2*sumEven + f(b) );

    syms x
    funcaoTransformada = symfun( f(x), x );
    expressao  = int(funcaoTransformada, x, a, b);
    exactValue = double(expressao);

    errorValue = abs(exactValue - S);
end
```

6 Conversão de Coordenadas

Fórmulas

A conversão entre sistemas de coordenadas é uma ferramenta essencial na Máquina de Cálculo Diferencial e Integral (CDI), permitindo representar pontos em sistemas Polar, Cilíndrico e Esférico a partir de coordenadas Cartesianas, e vice-versa, para aplicações em física, engenharia e geometria. Estas conversões utilizam relações matemáticas que preservam a posição dos pontos no espaço, ajustando-se às características de cada sistema.

As fórmulas de conversão são dadas por:

- **Cartesiana para Polar** (2D, ignora z):

$$\rho = \sqrt{x^2 + y^2}, \quad \theta = \arctan\left(\frac{y}{x}\right) \tag{6.1}$$

- **Polar para Cartesiana** (2D):

$$x = \rho\cos(\theta), \quad y = \rho\sin(\theta) \tag{6.2}$$

- **Cartesiana para Cilíndrico** (3D):

$$\rho = \sqrt{x^2 + y^2}, \quad \theta = \arctan\left(\frac{y}{x}\right), \quad z = z \tag{6.3}$$

- **Cilíndrico para Cartesiana** (3D):

$$x = \rho\cos(\theta), \quad y = \rho\sin(\theta), \quad z = z \tag{6.4}$$

- **Cartesiana para Esférico** (3D):

$$r = \sqrt{x^2 + y^2 + z^2}, \quad \phi = \arctan\left(\frac{y}{x}\right), \quad \theta = \arccos\left(\frac{z}{r}\right) \tag{6.5}$$

- **Esférico para Cartesiana** (3D):

$$x = r\sin(\theta)\cos(\phi), \quad y = r\sin(\theta)\sin(\phi), \quad z = r\cos(\theta) \tag{6.6}$$

onde:

- $\rho \rightarrow$ Distância radial no plano xy (Polar e Cilíndrico);
- $\theta \rightarrow$ Ângulo azimutal no plano xy (Polar e Cilíndrico) ou ângulo de elevação (Esférico, em radianos);
- $\phi \rightarrow$ Ângulo azimutal no plano xy (Esférico);
- $z \rightarrow$ Coordenada de altura (Cilíndrico) ou raio total (Esférico);
- $r \rightarrow$ Distância radial da origem (Esférico);
- $x, y, z \rightarrow$ Coordenadas Cartesianas.

A função $\arctan$ ajusta o quadrante com base nos sinais de x e y. Os ângulos são fornecidos em graus e convertidos para radianos nos cálculos. As conversões são implementadas no MATLAB com funções nativas, garantindo precisão numérica.

Algoritmo/Função

Algoritmo:

1. Obter valores de entrada (e.g., x, y, z para Cartesianas; θ, ρ, z para Polar/Cilíndrico; ou ϕ, θ, r para Esférico) como texto e convertê-los para números;
2. Validar as entradas, assegurando que são numéricos, escalares, não vazios e não `NaN`, exibindo mensagens de erro caso contrário (e.g., "Um ou mais valores não são expressões matemáticas válidas.");
3. Identificar o tipo de conversão selecionado ("Cartesiana para Polar/Cilíndrico/Esférico" ou "Polar/Cilíndrico/Esférico para Cartesiana");
4. Executar a conversão usando funções MATLAB:
   - **Cartesiana para Polar**: `cart2pol(x, y)` retorna θ, ρ;
   - **Polar para Cartesiana**: `pol2cart(theta, rho)` retorna x, y;
   - **Cartesiana para Cilíndrico**: `cart2pol(x, y, z)` retorna θ, ρ, z;
   - **Cilíndrico para Cartesiana**: `pol2cart(theta, rho, z)` retorna x, y, z;
   - **Cartesiana para Esférico**: `cart2sph(x, y, z)` retorna ϕ, θ, r;
   - **Esférico para Cartesiana**: `sph2cart(phi, theta, r)` retorna x, y, z;
5. Formatar os resultados com duas casas decimais e exibi-los em etiquetas na interface.

Um exemplo em MATLAB para converter coordenadas esféricas ($\phi = 45°, \theta = 60°, r = 2$) para Cartesianas é apresentado abaixo:

```
phi = deg2rad(45);  % Azimute em radianos
theta = deg2rad(60);  % Eleva  o em radianos
```

```
r = 2;  % Raio
[x, y, z] = sph2cart(phi, theta, r);
fprintf('X: %.2f, Y: %.2f, Z: %.2f\n', x, y, z);
```

7 Imagens UI

z = f(x,y)

y = f(x)

IntegNum

Coordenadas

AI Helper

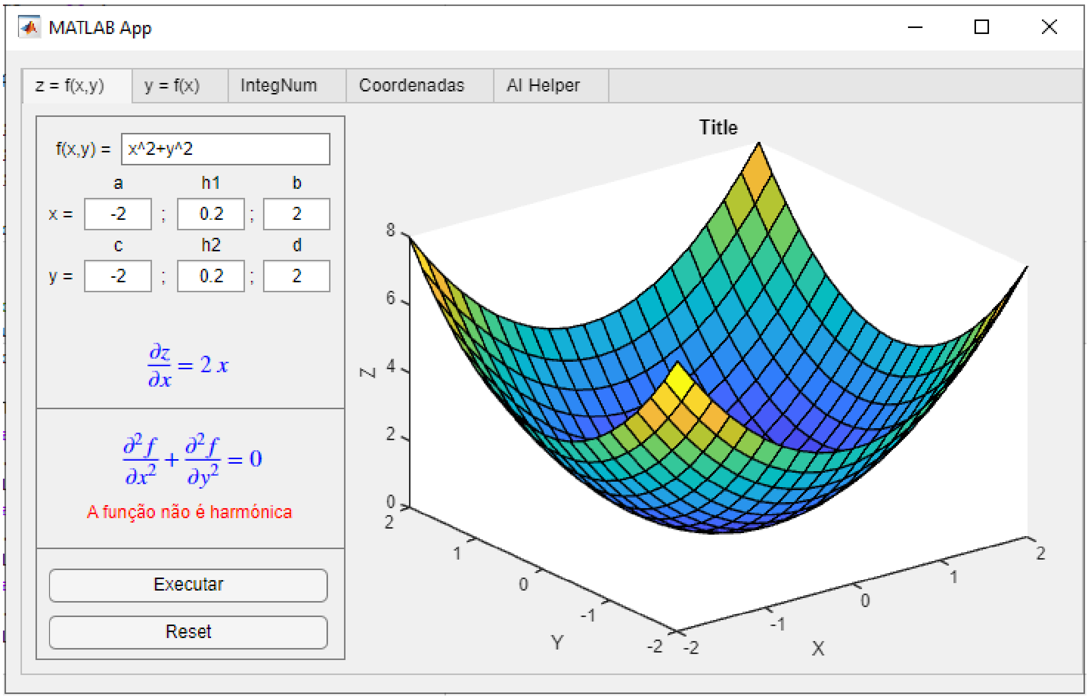

Figura 7.1: z = f(x,y)

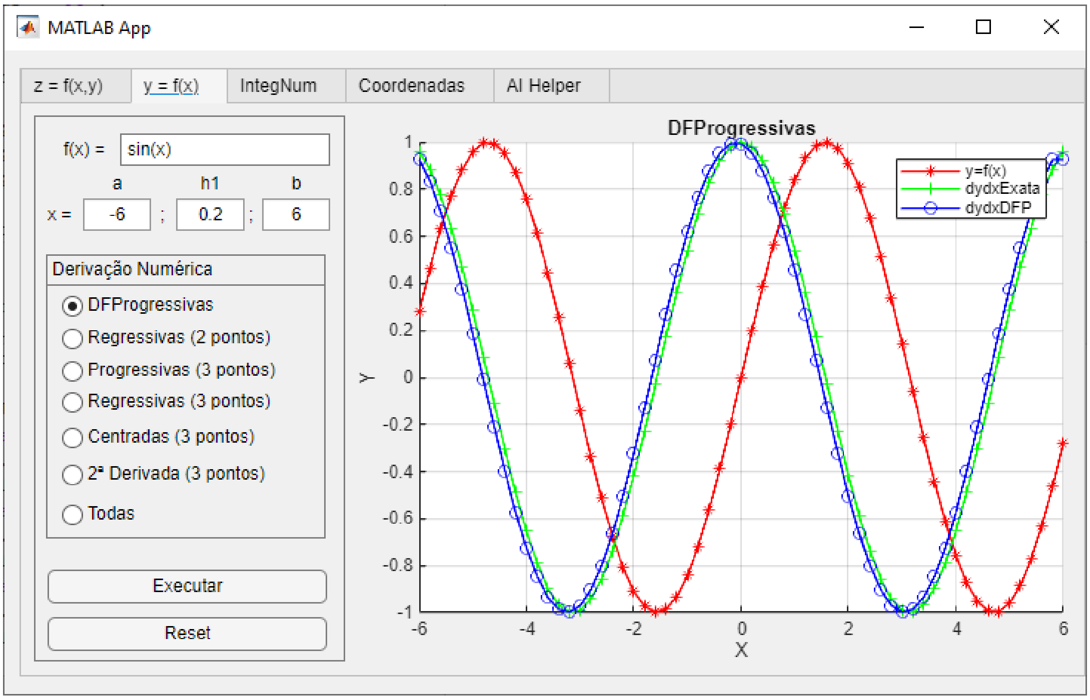

Figura 7.2: y = f(x)

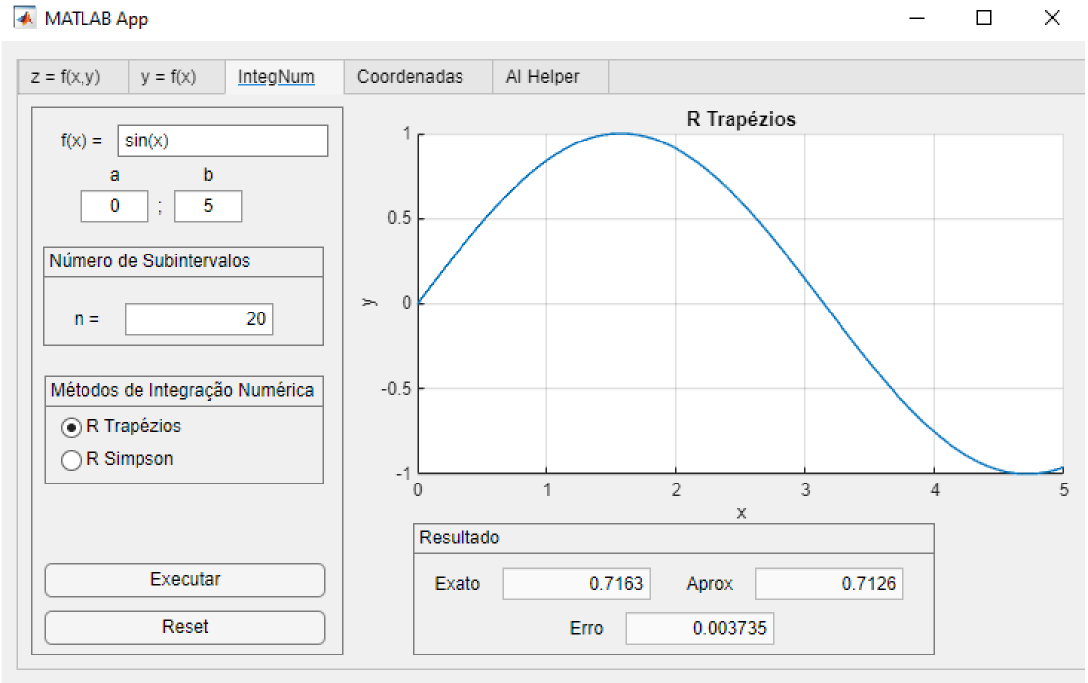

Figura 7.3: IntegNum

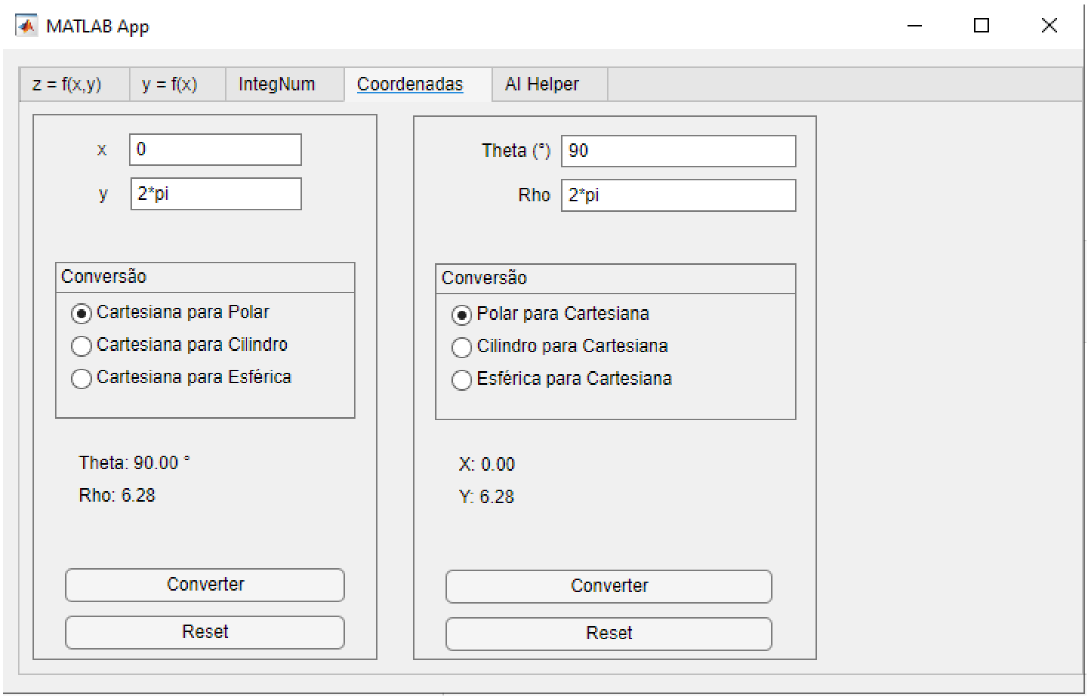

Figura 7.4: Coordenadas

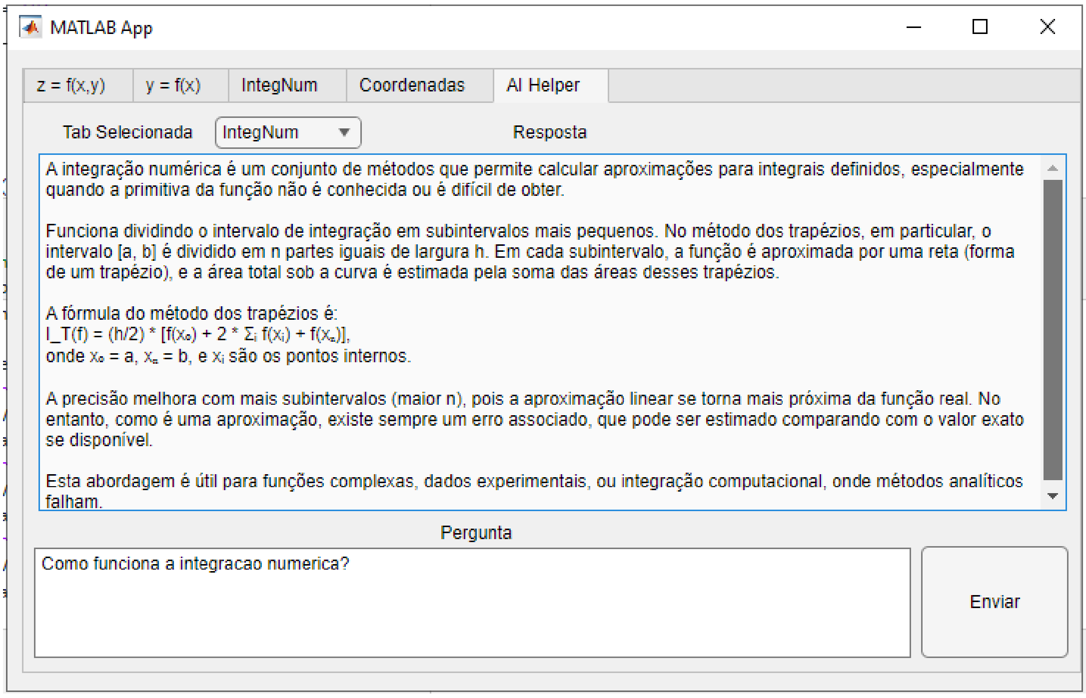

Figura 7.5: AI Helper

8 Conclusão

A realização da Atividade 5 proporcionou um aprofundamento significativo no estudo de métodos numéricos para cálculo diferencial e integral, culminando no desenvolvimento e ampliação da Máquina de Cálculo Diferencial e Integral (CDI) em Matlab. Foram implementadas funções para aproximação de derivadas por diferenças finitas em 2 pontos (progressivas e regressivas) e 3 pontos (progressivas, regressivas, centradas, e segunda derivada), integração numérica pelas Regras dos Trapézios e de Simpson, e verificação de funções harmônicas de duas variáveis reais. Estas implementações, realizadas com as funções `NDerivacaoDFP`, `NDerivacaoDFR`, `NDerivacaoDFP3`, `NDerivacaoDFR3`, `NDerivacaoDFC`, `NDerivacaoD2`, `Fregra_trapezios`, `Fregra_simpson` e `isHarmonica`, demonstraram precisão e eficiência na resolução de problemas práticos.

A Máquina CDI foi completada e enriquecida com funcionalidades inovadoras, incluindo uma aba para conversão entre coordenadas cartesianas, polares e esféricas, que facilita aplicações geométricas, e uma aba interativa que utiliza a API do ChatGPT para responder perguntas sobre a aplicação, promovendo maior acessibilidade e interatividade. A integração numérica destacou-se pela inclusão de cálculos simbólicos para valores exatos e erros absolutos, enquanto as fórmulas de diferenças finitas ofereceram soluções robustas para derivação numérica, com especial relevância em pontos extremos do intervalo.

A comparação entre os métodos revelou a importância de balancear precisão e custo computacional. As fórmulas de diferenças finitas em 3 pontos e a Regra de Simpson mostraram maior precisão devido a erros de truncatura de ordem superior, enquanto as fórmulas em 2 pontos e a Regra dos Trapézios foram mais simples e adequadas para casos menos exigentes. A verificação de funções harmônicas reforçou a aplicabilidade dos métodos em física e engenharia, como na análise de campos.

Em suma, esta atividade consolidou competências em análise numérica, programação em Matlab e desenvolvimento de ferramentas computacionais, contribuindo para a compreensão de técnicas fundamentais em cálculo diferencial e integral. A ampliação da Máquina CDI com funcionalidades criativas demonstrou a versatilidade e o potencial das abordagens numéricas na resolução de problemas complexos.

Referências bibliográficas

[1] Wikipédia, a enciclopédia livre. (2025) Diferenciação numérica. Acedido em 21 de maio de 2025. [Online]. Available: https://pt.wikipedia.org/wiki/Diferencia%C3%A7%C3%A3o_num%C3%A9rica

[2] ——. (2025) Integração numérica. Acedido em 21 de maio de 2025. [Online]. Available: https://pt.wikipedia.org/wiki/Integra%C3%A7%C3%A3o_num%C3%A9rica

[3] ——. (2025) Trapezoidal rule. Acedido em 21 de maio de 2025. [Online]. Available: https://en.wikipedia.org/wiki/Trapezoidal_rule

[4] ——. (2025) Simpson's rule. Acedido em 21 de maio de 2025. [Online]. Available: https://en.wikipedia.org/wiki/Simpson%27s_rule

[5] MathWorks. (2025) Symbolic math toolbox – integração e derivação numérica. Acedido em 21 de maio de 2025. [Online]. Available: https://www.mathworks.com/help/symbolic/numerical-integration.html

[6] Universidade Federal do Rio Grande do Sul. (2025) Diferenças finitas. Acedido em 21 de maio de 2025. [Online]. Available: https://www.ufrgs.br/reamat/CalculoNumerico/livro-sci/dn-diferencas_finitas.html

[7] Universidade Federal de São João del-Rei. (2025) Derivação numérica. Acedido em 21 de maio de 2025. [Online]. Available: https://www.ufsj.edu.br/portal2-repositorio/File/nepomuceno/mn/09MN_Derivacao.pdf

[8] OpenAI. (2025) Openai api documentation. Acedido em 21 de maio de 2025. [Online]. Available: https://platform.openai.com/docs/api-reference

[9] M. Ferreira and P. Costa, "Métodos numéricos para integração: Uma análise comparativa entre trapézios e simpson," *Revista de Matemática Computacional*, vol. 15, no. 2, pp. 33–50, 2025, acedido em 21 de maio de 2025. [Online]. Available: https://www.revmatcomp.pt/artigos/2025/ferreira_integracao

[10] L. Santos, *Cálculo Numérico: Derivação, Integração e Aplicações*. Porto: Editora Universitária do Porto, 2025, acedido em 21 de maio de 2025.

[11] F. Coutinho, P. Silva, and R. Rodrigues, "Como escrever um bom relatório em latex," ISEC/IPC, Coimbra, Tech. Rep., Nov. 2022.

Instituto Superior
de Engenharia
Politécnico de Coimbra