Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTAMENTO DE / DEPARTMENT OF
INFORMÁTICA E SISTEMAS

Métodos Numéricos para derivação e integração de funções

Autor / Author

Tiago Fernandes Lopes – a2023144639,

Engenharia Informática

Autor / Author

Daniel Duarte Silva – a2023144551,

Engenharia Informática

Autor / Author

Guilherme Alexandre Neves Martins – a2023144573,

Engenharia Informática

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, maio de 2024

ÍNDICE

ÍNDICE DE FIGURAS

1 INTRODUÇÃO

O cálculo diferencial e integral é uma área central da matemática, essencial para diversas disciplinas como física, engenharia e economia. Divide-se em duas partes principais:

Cálculo Diferencial

O cálculo diferencial estuda as taxas de variação e utiliza derivadas para entender como as quantidades mudam instantaneamente. Isto permite a análise de inclinações de curvas, optimização de funções e modelação de fenómenos dinâmicos.

Cálculo Integral

O cálculo integral foca-se na acumulação de quantidades e no cálculo de áreas sob curvas, utilizando integrais. Isto é útil para determinar áreas, volumes e quantidades acumuladas.

2 MÉTODOS NUMÉRICOS PARA DERIVAÇÃO

A Derivação Numérica, e.g. Fórmula das Diferenças Finitas, é usada para calcular a derivada quando temos apenas um conjunto de pontos da função, ou para funções de derivação complexa ou não deriváveis em todo o seu domínio. À medida que o intervalo h diminui, a derivada numérica se aproxima do valor real, mas sempre haverá um erro de arredondamento. Para reduzir esse erro, usamos múltiplos pontos.

A partir desses pontos dentro de um intervalo $[a, b]$ é possível determinar a função f. Ao saber a função podemos calcular a sua derivada e aplicá-la a qualquer ponto do intervalo $[a, b]$. Utilizam-se 2, 3 ou mais pontos para calcular a derivada através das Fórmulas de Diferenças Finitas.

2.1 Fórmulas de Diferenças Finitas em 2 pontos

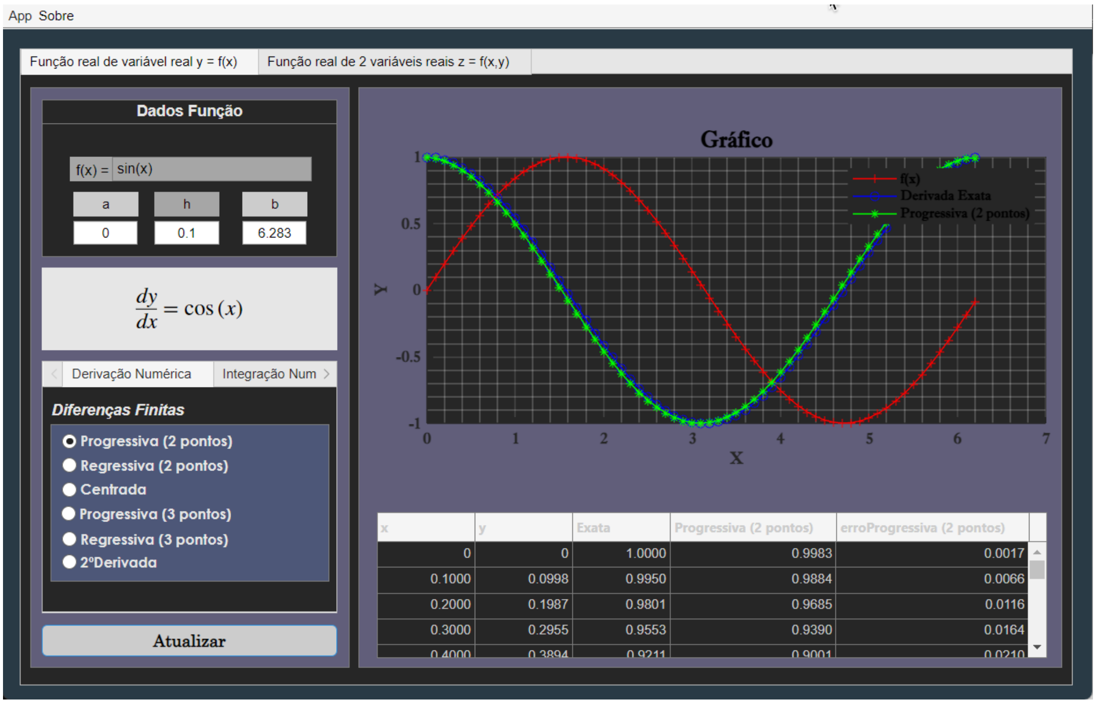
Existem dois métodos para calcular a derivada utilizando apenas dois pontos:

- Progressivas;
- Regressivas.

2.1.1 Progressivas

$$f'(x_k) := \frac{f(x_{k+1}) - f(x_k)}{h} \tag{2.1}$$

- $f(x_k) \rightarrow$ Valor da função no ponto x (ponto que queremos calcular)
- $f(x_{k+1}) \rightarrow$ Valor no próximo ponto
- $h \rightarrow$ Intervalo entre cada ponto



2.1 Progressiva em 2 pontos

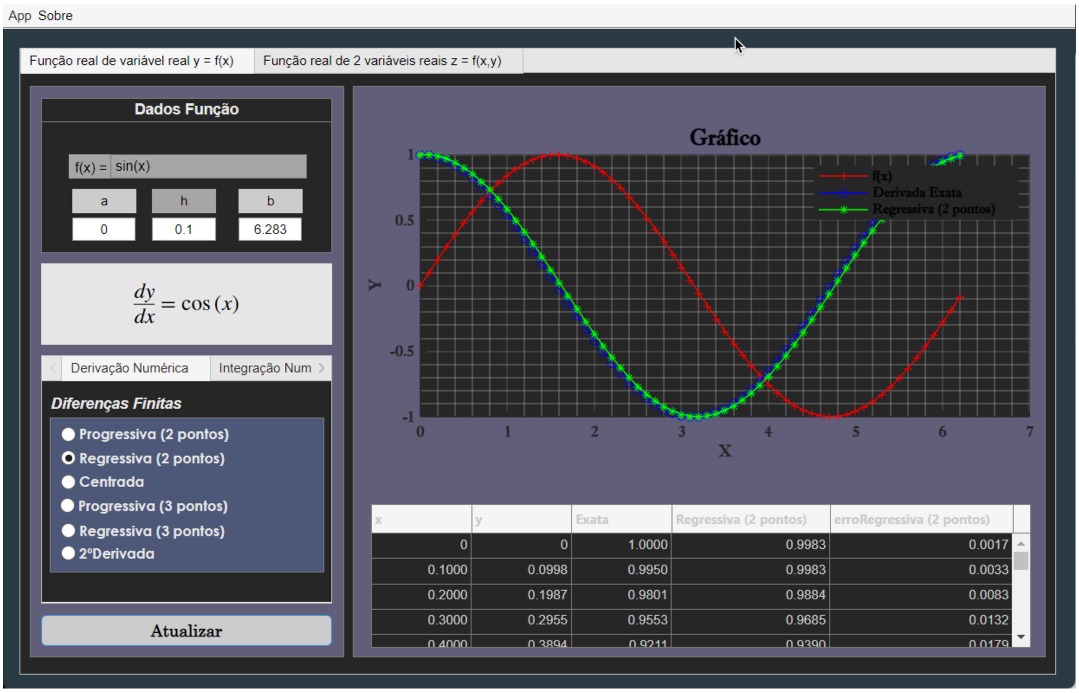
Função em MATLAB

```
function [x,y,dydx] = Progressiva2p(~,f,a,b,h,y)
    % Calcula a derivada numérica de primeira ordem utilizando o método
das diferenças progressivas com 2 pontos.

    % INPUT:
    %   ~ - ignorado, pode ser usado para compatibilidade de chamada
    %   f - função a ser derivada (opcional se y for fornecido)
    %   a - valor inicial do intervalo
    %   b - valor final do intervalo
    %   h - passo de incremento no intervalo
    %   y - valores da função f nos pontos de x (opcional se f for
fornecido)

    % OUTPUT:
    %   x - vetor de pontos no intervalo [a, b] com espaçamento h
    %   y - valores da função f nos pontos de x
    %   dydx - vetor das derivadas de y em cada ponto de x

    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
```

2.1.2 Regressivas

$$f'(x_k) \coloneqq \frac{f(x_k) - f(x_{k-1})}{h} \tag{2.2}$$

- $f(x_k)$ → Valor da função no ponto x (ponto que queremos calcular)
- $f(x_{k-1})$ → Valor no ponto anterior
- h → Intervalo entre cada ponto



x	y	Exata	Regressiva (2 pontos)	erroRegressiva (2 pontos)
0	0	1.0000	0.9983	0.0017
0.1000	0.0998	0.9950	0.9983	0.0033
0.2000	0.1987	0.9801	0.9884	0.0083
0.3000	0.2955	0.9553	0.9685	0.0132
0.4000	0.3894	0.9211	0.9390	0.0179

2.2 Regressiva em 2 pontos

Função em MATLAB

```
function [x,y,dydx] = Regressiva2p(~,f,a,b,h,y)
    % Calcula a derivada numérica de primeira ordem utilizando o método
das diferenças
    % regressivas com 2 pontos.

    % INPUT:
    %   ~ - ignorado, pode ser usado para compatibilidade de chamada
    %   f - função a ser derivada (opcional se y for fornecido)
    %   a - valor inicial do intervalo
    %   b - valor final do intervalo
    %   h - passo de incremento no intervalo
    %   y - valores da função f nos pontos de x (opcional se f for
fornecido)

    % OUTPUT:
    %   x - vetor de pontos no intervalo [a, b] com espaçamento h
    %   y - valores da função f nos pontos de x
    %   dydx - vetor das derivadas de y em cada ponto de x

    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1,n);
    for k = 2:n
        dydx(k) = (y(k) - y(k-1)) / h;
    end
    dydx(1) = (y(2) - y(1)) / h;
end
```

2.2 Fórmulas de Diferenças Finitas em 3 pontos

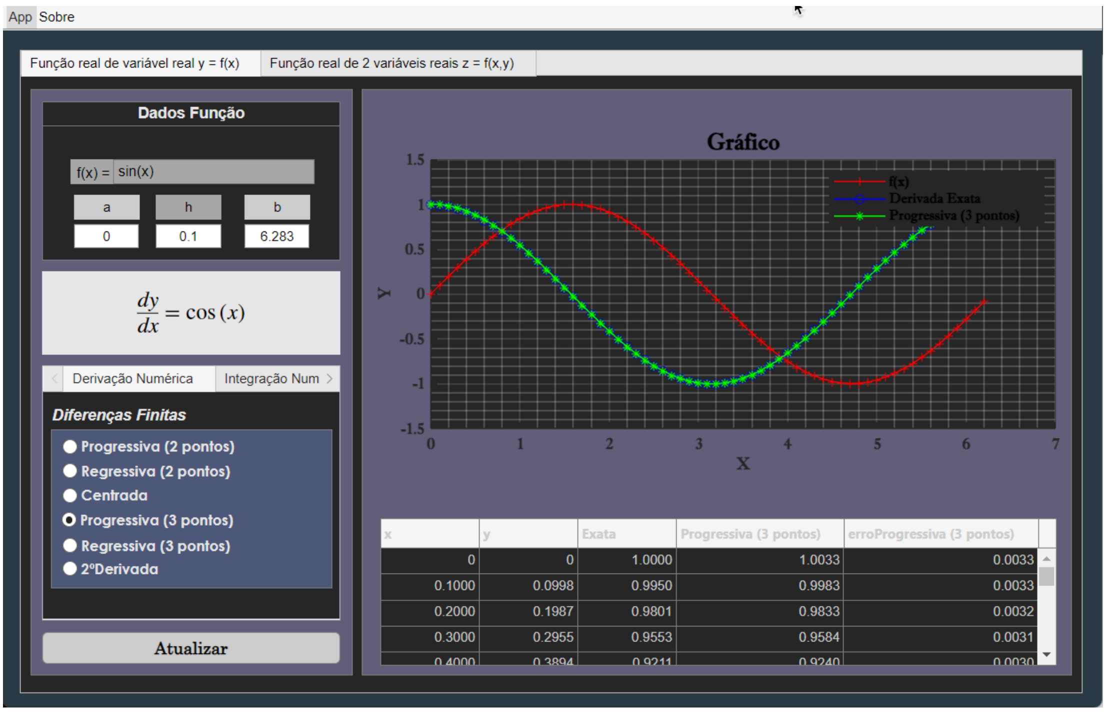
Existem dois métodos para calcular a derivada utilizando apenas três pontos:

- Progressivas;
- Regressivas;
- Centradas.

2.2.1 Progressivas

$$f'(x_k) \coloneqq \frac{-3f(x_k) + 4f(x_{k+1}) - f(x_{k+2})}{2h} \tag{2.3}$$

- $f(x_k) \rightarrow$ Valor da função no ponto x (ponto que queremos calcular)
- $f(x_{k+1}) \rightarrow$ Valor no próximo ponto
- $f(x_{k+2}) \rightarrow$ Valor em 2 pontos a frente
- $h \rightarrow$ Intervalo entre cada ponto



2.3 Progessiva em 3 pontos

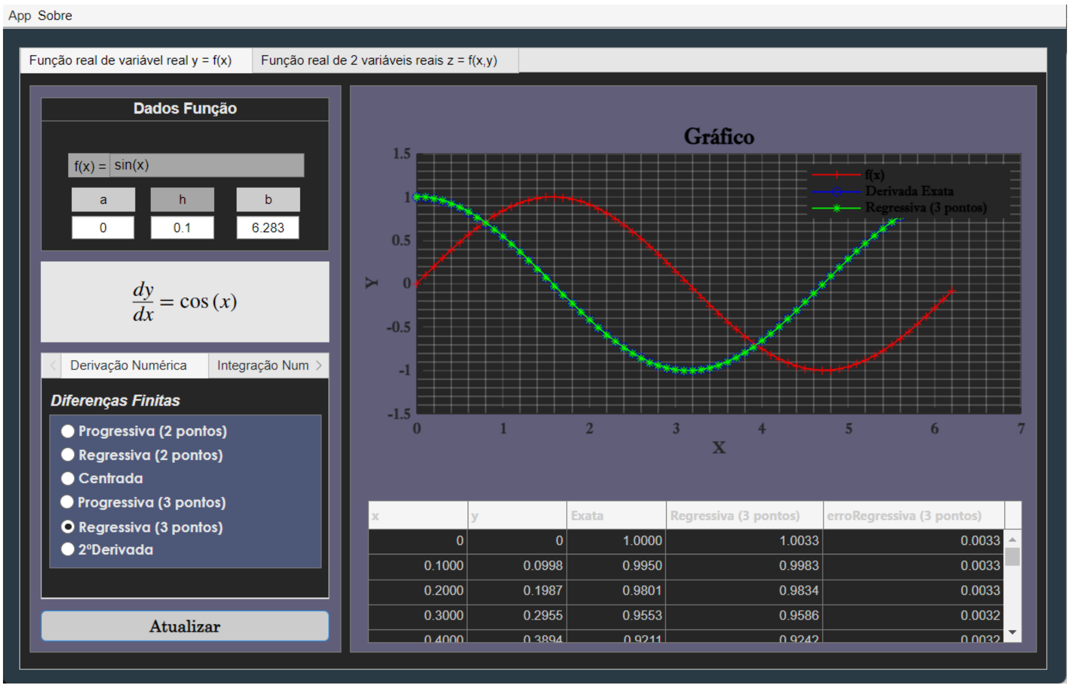
Função em MATLAB

```
function [x,y,dydx] = Progressiva3p(f,a,b,h,y)
    % Calcula a derivada numérica de primeira ordem utilizando o método
das diferenças progressivas com 3 pontos.

    % INPUT:
    %   f - função a ser derivada (opcional se y for fornecido)
    %   a - valor inicial do intervalo
    %   b - valor final do intervalo
    %   h - passo de incremento no intervalo
    %   y - valores da função f nos pontos de x (opcional se f for
fornecido)

    % OUTPUT:
    %   x - vetor de pontos no intervalo [a, b] com espaçamento h
    %   y - valores da função f nos pontos de x
    %   dydx - vetor das derivadas de y em cada ponto de x

    x = a:h:b;
    n = length(x);
    if nargin == 4
        y = f(x);
    end
    dydx = zeros(1,n);
    for k = 1:n-2
        dydx(k) = (-3*y(k) + 4*y(k+1) - y(k+2)) / (2*h);
    end
    dydx(n) = (y(n-2) - 4*y(n-1) + 3*y(n)) / (2*h);
    dydx(n-1) = (y(n-3) - 4*y(n-2) + 3*y(n-1)) / (2*h);
end
```

2.2.2 Regressivas

$$f'(x_k) := \frac{f(x_{k-2}) - 4f(x_{k-1}) + 3f(x_k)}{2h} \tag{2.4}$$

- $f(x_k) \rightarrow$ Valor da função no ponto x (ponto que queremos calcular)
- $f(x_{k-1}) \rightarrow$ Valor no ponto anterior
- $f(x_{k-2}) \rightarrow$ Valor em 2 pontos antes
- $h \rightarrow$ Intervalo entre cada ponto



2.4 Regressiva em 3 pontos

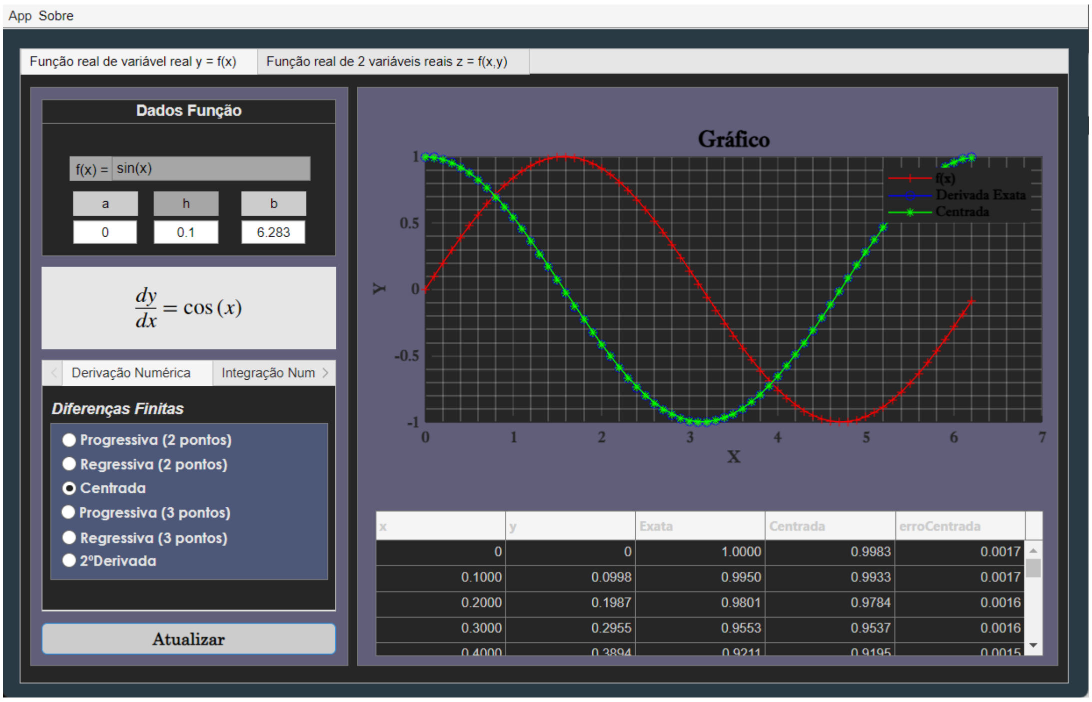
Função em MATLAB

```
function [x,y,dydx] = Regressiva3p(f,a,b,h,y)
    % Calcula a derivada numérica de primeira ordem utilizando o método
das diferenças
    % regressivas com 3 pontos.

    % INPUT:
    %   f - função a ser derivada (opcional se y for fornecido)
    %   a - valor inicial do intervalo
    %   b - valor final do intervalo
    %   h - passo de incremento no intervalo
    %   y - valores da função f nos pontos de x (opcional se f for
fornecido)

    % OUTPUT:
    %   x - vetor de pontos no intervalo [a, b] com espaçamento h
    %   y - valores da função f nos pontos de x
    %   dydx - vetor das derivadas de y em cada ponto de x

    x = a:h:b;
    n = length(x);
    if nargin == 4
        y = f(x);
    end
    dydx = zeros(1,n);
    for k = 3:n
        dydx(k) = (y(k-2) - 4*y(k-1) + 3*y(k)) / (2*h);
    end
    dydx(1) = (-3*y(1) + 4*y(2) - y(3)) / (2*h);
    dydx(2) = (-3*y(2) + 4*y(3) - y(4)) / (2*h);
end
```

2.2.3 Centradas

$$f'(x_k) \coloneqq \frac{f(x_{k+1}) - f(x_{k-1})}{2h} \tag{2.5}$$

- $f(x_{k-1}) \rightarrow$ Valor no ponto anterior
- $f(x_{k+1}) \rightarrow$ Valor no próximo ponto
- $h \rightarrow$ Intervalo entre cada ponto



2.5 Centrada

Função em MATLAB

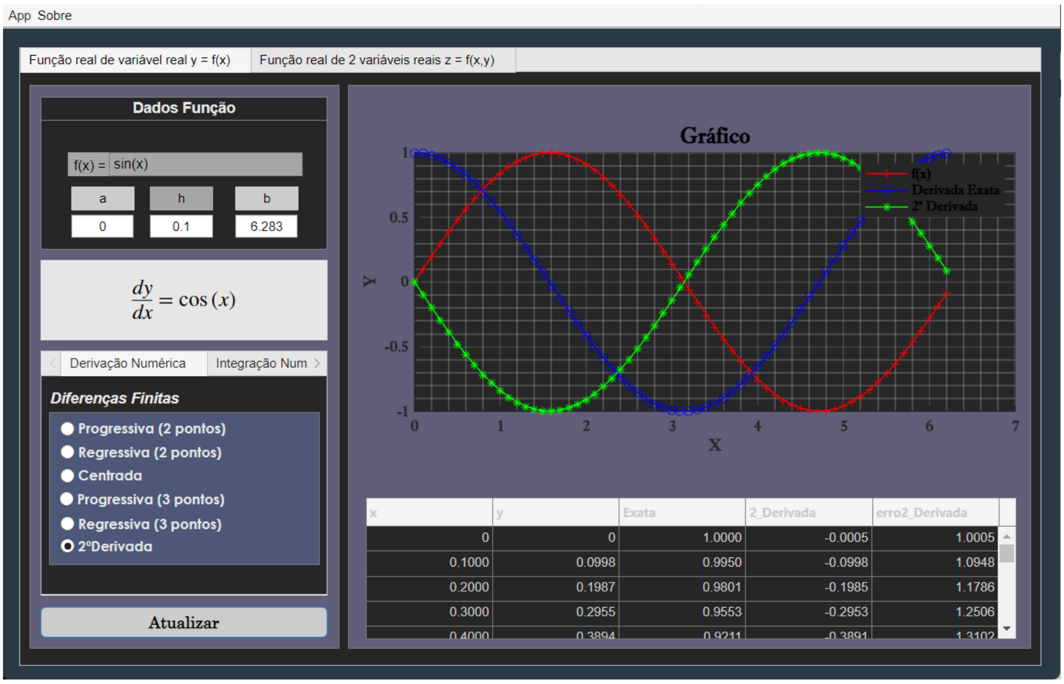
```
function [x,y,dydx] = Centrada(~,f,a,b,h,y)
    % Calcula a derivada numérica de primeira ordem utilizando o método
das diferenças
    % centradas.

    % INPUT:
    %   ~ - ignorado, pode ser usado para compatibilidade de chamada
    %   f - função a ser derivada (opcional se y for fornecido)
    %   a - valor inicial do intervalo
    %   b - valor final do intervalo
    %   h - passo de incremento no intervalo
    %   y - valores da função f nos pontos de x (opcional se f for
fornecido)

    % OUTPUT:
    %   x - vetor de pontos no intervalo [a, b] com espaçamento h
    %   y - valores da função f nos pontos de x
    %   dydx - vetor das derivadas de y em cada ponto de x

    x = a:h:b;
    n = length(x);
    if nargin == 5
        y = f(x);
    end
    dydx = zeros(1,n);
    for k = 2:n-1
        dydx(k) = (y(k+1) - y(k-1)) / (2*h);
    end
    dydx(1) = (y(2) - y(1)) / h;
    dydx(n) = (y(n) - y(n-1)) / h;
end
```

2.3 Segunda derivada

A segunda derivada de uma função pode ser calculada através da seguinte fórmula:

$$f''(x_k) := \frac{f(x_{k+1}) - 2f(x_k) + f(x_{k-1})}{h^2} \tag{2.6}$$

- $f(x_k) \rightarrow$ Valor da função no ponto x (ponto que queremos calcular)
- $f(x_{k-1}) \rightarrow$ Valor no ponto anterior
- $f(x_{k+1}) \rightarrow$ Valor no próximo ponto
- $h \rightarrow$ Intervalo entre cada ponto



2.6 2ª derivada

Função em MATLAB

```
function [x,y,dydx] = SDerivada(f,a,b,h,y)
    function [x,y,dydx] = SDerivada(f,a,b,h,y)
    %SDERIVADA Calcula a segunda derivada numérica de uma função.
    %   Esta função calcula a segunda derivada de uma função utilizando o
método das
    %   diferenças centradas para os pontos internos e diferenças
regressivas ou progressivas
    %   para os pontos das extremidades.

    % INPUT:
    %   f - função a ser derivada (opcional se y for fornecido)
    %   a - valor inicial do intervalo
    %   b - valor final do intervalo
    %   h - passo de incremento no intervalo
    %   y - valores da função f nos pontos de x (opcional se f for
fornecido)

    % OUTPUT:
    %   x - vetor de pontos no intervalo [a, b] com espaçamento h
    %   y - valores da função f nos pontos de x
    %   dydx - vetor das segundas derivadas de y em cada ponto de x

    x = a:h:b;
    n = length(x);
    if nargin == 4
        y = f(x);
    end
    dydx = zeros(1,n);

    for k = 2:n-1
        dydx(k) = (y(k+1) - 2*y(k) + y(k-1)) / (h^2);
    end

    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) / (h*2);

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

3 INTEGRAÇÃO NUMÉRICA

A integração numérica é uma técnica utilizada para calcular o valor aproximado de integrais definidas, especialmente quando a integração analítica é difícil ou impossível de ser realizada. Existem vários métodos de integração numérica, cada um com as suas vantagens e desvantagens, dependendo da função a ser integrada e da precisão requerida. Aqui estão alguns dos métodos mais comuns:

3.1 Regra do Trapézio

A regra do trapézio é um método numérico utilizado para aproximar o valor de integrais definidas. É especialmente útil quando a integral de uma função não pode ser resolvida analiticamente ou quando se trabalha com dados empíricos. Este método envolve a decomposição da área sob a curva numa série de trapézios, cuja área é mais fácil de calcular.

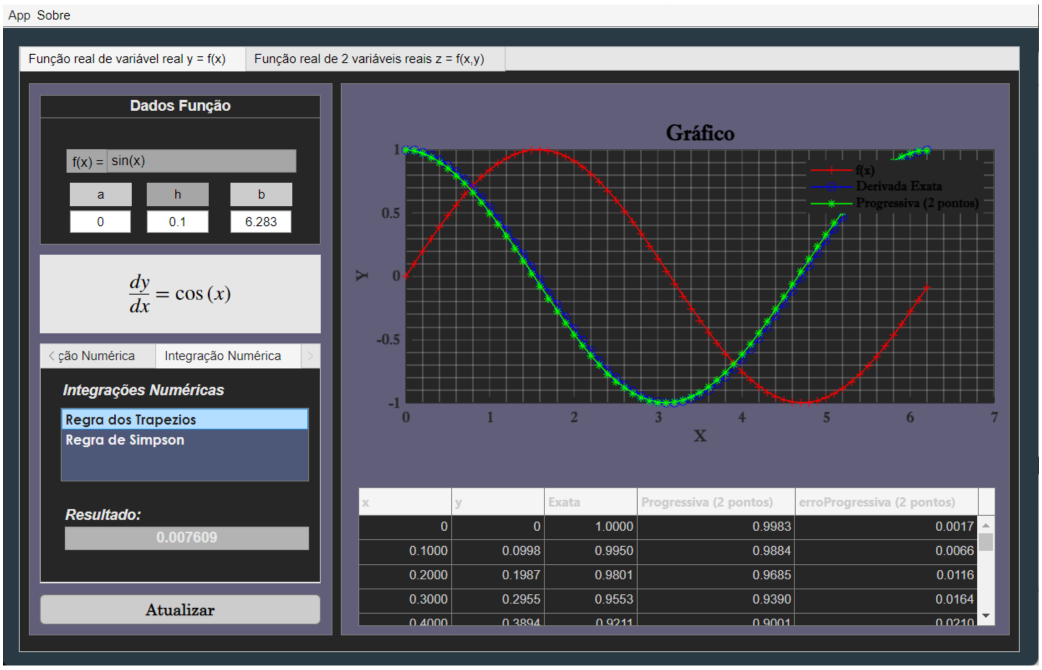
Fórmula:

$$I_T(f) = \frac{h}{2}[f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)] \tag{3.1}$$

$$\text{Erro: } |E_T| \leq \frac{b-a}{12}h^2 M_2, M_2 = max_{x\in[a,b]}|f''(x)| \tag{3.2}$$

Passos de Aplicação:

1. Dividir o intervalo: Dividir o intervalo [a,b] em n subintervalos de igual largura;
2. Calcular os pontos de $\boldsymbol{x_0}, \boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \mathrm{x}_{\boldsymbol{n}}$;
3. Avaliar a Função: Calcular os valores da função f nesses pontos $\boldsymbol{f(x_0)}, \boldsymbol{f(x_1)}, \boldsymbol{f(x_2)}, \ldots, \boldsymbol{f(x_n)}$
4. Aplicar a fórmula: Substituir os valores na fórmula da regra do trapézio para obter a aproximação do integral.



3.1 Regra dos Trapézios

Função em MATLAB

```
function T = RegraTrapezios(f, a, b, n)
    %REGRA_TRAPEZIOS Calcula a integral definida de uma função usando a
regra dos trapézios.
    %   Esta função calcula a integral de uma função f no intervalo [a, b]
    %   utilizando a regra dos trapézios com n subintervalos.

    % INPUT:
    %   f - função a ser integrada
    %   a - valor inicial do intervalo de integração
    %   b - valor final do intervalo de integração
    %   n - número de subintervalos para a regra dos trapézios

    % OUTPUT:
    %   T - valor aproximado da integral de f no intervalo [a, b]

    h = (b-a)/n;
    x = a;
    s = 0;

    for i = 1:n-1
        x = x + h;
        s = s + f(x);
    end

    T = h/2 * (f(a) + 2*s + f(b));
end
```

3.2 Regra de Simpson

A regra de Simpson é um método numérico de integração muito usado para obter aproximações precisas de integrais definidas.

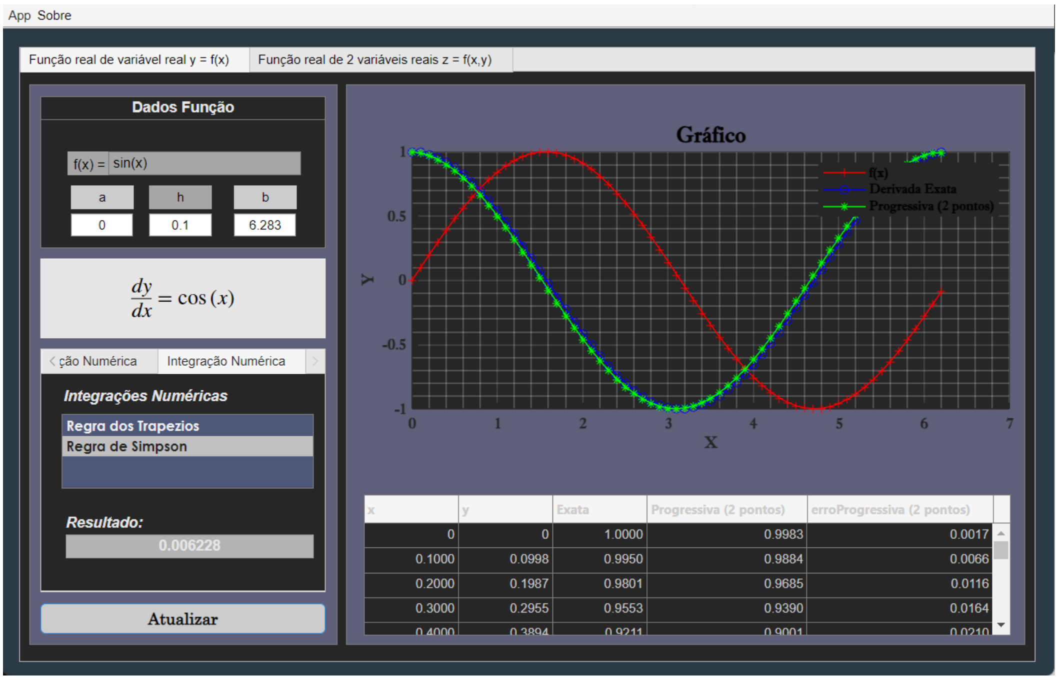
Fórmula:

$$I_S(f) = \frac{h}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)] \tag{3.3}$$

$$\text{Erro: } |E_S| \leq \frac{b-a}{180} h^4 M_4, M_4 = max_{x\in[a,b]}\left|f^{(4)}(x)\right| \tag{3.4}$$

Passos de Aplicação:

1. Definir o Intervalo e o Número de Subintervalos (n deve ser par);
2. Calcular a largura dos Subintervalos;
3. Determinar os Pontos de Amostragem;
4. Calcular os valores em Função dos Pontos;
5. Aplicar a fórmula;



3.2 Regra de Simpson

Função em MATLAB

```
function out_S = RegraSimpson(f, a, b, n)
    %REGRA_SIMPSON Calcula a integral definida de uma função usando a
regra de Simpson.
    %   Esta função calcula a integral de uma função f no intervalo [a, b]
    %   utilizando a regra de Simpson com n subintervalos.

    % INPUT:
    %   f - função a ser integrada
    %   a - valor inicial do intervalo de integração
    %   b - valor final do intervalo de integração
    %   n - número de subintervalos para a regra de Simpson (deve ser par)

    % OUTPUT:
    %   out_S - valor aproximado da integral de f no intervalo [a, b]

    h = (b - a) / n;  % Calcula o tamanho de cada subintervalo
    x = a;            % Inicializa x com o valor inicial do intervalo
    s = 0;            % Inicializa a soma dos termos intermediários

    if mod(n, 2) ~= 0
        error('n deve ser um número par');
    end

    for i = 1:n-1
        x = x + h;
        if mod(i, 2) == 0
            s = s + 2 * f(x);
        else
            s = s + 4 * f(x);
        end
    end

    out_S = h / 3 * (f(a) + s + f(b));
end
```

4 CONCLUSÃO

A integração numérica e os métodos numéricos para derivação permitem-nos resolver problemas que não podem ser resolvidos analiticamente. Métodos de integração como o Regra dos Trapézios e Simpson calculam aproximações para integrais definidas, enquanto métodos de derivação como diferenças finitas e derivadas centradas estimam derivadas de funções discretas ou complexas.

A escolha do método adequado depende de fatores como a precisão necessária, a natureza da função e as restrições computacionais.

5 BIBLIOGRAFIA

Diferenciação numérica. (2023, novembro 7). Wikipédia, a enciclopédia livre. Retrieved 22:21, novembro 7, 2023 from https://pt.wikipedia.org/w/index.php?title=Diferencia%C3%A7%C3%A3o_num%C3%A9rica&oldid=66928287.

Integração numérica. (2022, outubro 23). Wikipédia, a enciclopédia livre. Retrieved 16:10, outubro 23, 2022 from https://pt.wikipedia.org/w/index.php?title=Integra%C3%A7%C3%A3o_num%C3%A9rica&oldid=64614705.

6 AUTOAVALIAÇÃO E HETEROAVALIAÇÃO

Autoavaliação

Tiago Fernandes Lopes: 4/5 valores

Daniel Duarte Silva: 4/5 valores

Guilherme Alexandre Neves Martins: 4/5 valores

Heteroavaliação

Tiago Fernandes Lopes:

- Daniel Duarte Silva: 4/5 valores
- Guilherme Alexandre Neves Martins: 4/5 valores

Daniel Duarte Silva:

- Tiago Fernandes Lopes: 4/5 valores
- Guilherme Alexandre Neves Martins: 4/5 valores

Guilherme Alexandre Neves Martin:

- Daniel Duarte Silva: 5/5 valores
- Tiago Fernandes Lopes: 5/5 valores

Instituto Superior
de Engenharia
Politécnico de Coimbra