

Atividade 03

Métodos Numéricos para Derivação e Integração

**Licenciatura em Engenharia Informática
Análise Matemática II**

Alunos:

Francisco Ruivo – 2021142022

Daniel Rodrigues - 2021142013

2021 / 2022

Índice

Conteúdo

1. Introdução	3
2. Métodos Numéricos para Derivação	4
2.1. Fórmulas de Diferenças finitas em 2 pontos	4
2.1.1. Progressivas	5
2.1.2. Regressivas	6
2.2. Fórmulas de Diferenças finitas em 3 pontos	7
2.2.1. Progressivas	7
2.2.2. Regressivas	8
2.2.3. Centradas	9
2.3. 2ª Derivada	10
3. Métodos Numéricos para Integração	12
3.1. Regra dos Trapézios.....	13
3.2 Regra de Simpson	15
4.Exemplos de aplicação	17
4.1 Integração	17
4.2 Derivação	18
5. Conclusão	20
6. Autoavaliação e heteroavaliação	21
7. Bibliografia	22

1. Introdução

1.1. Enunciado da atividade proposta e interpretação do mesmo

Parte01: Implemente em Matlab funções para as fórmulas de diferenças finitas seguintes

Frequentemente somos confrontados com a necessidade de determinar valores da derivada de uma função num conjunto de pontos, conhecendo apenas o valor da função nesses pontos.

Questão: Como aproximar o valor da derivada num ponto?

Resposta: Aplicar uma das fórmulas de diferenças finitas: progressivas, regressivas ou centradas.

Seja f uma função definida em $[a, b]$ e suficientemente regular, conhecida num conjunto de pontos da partição uniforme $a = x_0 < x_1 < \dots < x_n = b$

Parte02: Implemente as regras dos trapézios e Simpson seguintes

Parte03:

Avaliar se uma função real de duas variáveis reais é harmónica.
Outras operações e avaliações...

Parte04: Máquina Cálculo Diferencial e Integral

Com base na máquina representada na figura seguinte, que foi objeto de trabalho nas aulas práticas, complete a máquina e amplie as suas potencialidades para o cálculo diferencial e integral de funções reais de uma variável real e de duas variáveis reais.

Este trabalho surge do âmbito da unidade curricular de Análise Matemática 2 do curso de Engenharia Informática do Instituto Superior de Engenharia de Coimbra e consiste na implementação em MATLAB de métodos de Derivação e Integração Numérica.

O principal objetivo é a implementação de funções, através do desenvolvimento de uma GUI em linguagem de programação MATLAB.

Para calcular uma derivada numericamente, utilizámos 6 fórmulas de diferenças divididas finitas: Progressivas (2 pontos), Regressivas (2 pontos), Progressivas (3 pontos), Regressivas (3 pontos), Centradas (3 pontos) e 2ª Derivada (3 pontos).

No caso do cálculo integral, calculado de forma numérica, foram implementadas 2 regras: a Regra dos Trapézios e a Regra de Simpson.

2. Métodos Numéricos para Derivação

2.1. Fórmulas de Diferenças finitas em 2 pontos

O método mais simples para obter um valor aproximado da derivada é usar o método das diferenças finitas. Uma diferença finita é uma expressão do tipo $f(x + b) - f(x + a)$, que ao ser dividida por $(b - a)$, passa a ser designado de quociente das diferenças. A técnica das diferenças finitas tem como finalidade obter uma aproximação da derivada de uma função via fórmulas discretas que apenas requerem um conjunto finito de pares ordenados, designado da seguinte forma $y_k = f(x_k)$.

Nota: $dydx$ representa a derivada no matlab.

2.1.1. Progressivas

Fórmula:

Fórmulas de diferenças finitas em 2 pontos:

$$\text{Progressivas} \gg f'(x_k) := \frac{f(x_{k+1}) - f(x_k)}{h}$$

- $f'(x_k)$ → Aproximação do valor da derivada em x_k ;
- $f(x_{k+1})$ → Valor da função na próxima abscissa;
- $f(x_k)$ → Valor da função na abscissa atual;
- h → Cálculo do passo.

Algoritmo:

1. Alocação de memória de x ;
2. Definir o número de pontos a n ;
3. Se forem inseridos 4 elementos, y toma o valor de $f(x)$;
4. Alocação de memória da derivada;
5. Para i de 1 a $n-1$, calcular o valor aproximado da derivada de f , no ponto atual;
6. Cálculo do valor aproximado da derivada de f , no ponto atual em n .

Função em MatLab:

```
function [x,y,dydx] = DIF2PONTOS_PROG(f,a,b,h,y)

x = a:h:b;                                %alocação de memoria

n=length(x);                              % Número de pontos (tamanho do vetor de abcissas)

if nargin==4
    y=f(x);                               % y é a função f(x), se forem inseridos 4 elementos, y recebe o valor de f(x)
end

dydx=zeros(1,n);                          % Alocação de memória

for i=1:n-1
    dydx(i)=(y(i+1)-y(i))/h;               % Derivada (aproximada) de f no ponto atual
end

dydx(n)=(y(n) - y(n-1))/(h);               % Derivada (aproximada) de f no ponto atual, em n
end
```

2.1.2. Regressivas

Fórmula:

$$\text{Regressivas} \gg f'(x_k) := \frac{f(x_k) - f(x_{k-1})}{h}$$

- $f'(x_k) \rightarrow$ Aproximação do valor da derivada em x_k ;
- $f(x_k) \rightarrow$ Valor da função na abscissa atual;
- $f(x_{k-1}) \rightarrow$ Valor da função na abscissa anterior;
- $h \rightarrow$ Cálculo do passo.

Algoritmo:

1. Alocação de memória de x ;
2. Definir o número de pontos a n ;
3. Se forem inseridos 4 elementos, y toma o valor de $f(x)$;
4. Alocação de memória da derivada;
5. Valor aproximado da derivada de f no ponto atual, em 1;
6. Para i de 2 a n , calcular o valor aproximado da derivada de f , no ponto atual;

Função em MatLab:

```
function [x,y,dydx] = DIF2PONTOS_REG(f,a,b,h,y)

x = a:h:b;                % Alocação de memória
n = length(x);            % Número de pontos (tamanho do vetor de abcissas)
if nargin == 4             % y é a função f(x), se forem inseridos 4 elementos, y recebe o valor de f(x)
    y = f(x);
end
dydx = zeros(1,n);        % Alocação de memória
dydx(1) = (y(2)-y(1))/h;  % derivada (aproximada) de f no ponto atual, em 1
for i = 2:n
    dydx(i) = (y(i)-y(i-1))/h; % derivada (aproximada) de f no ponto atual, para a iésima iteração
end
end
```

2.2. Fórmulas de Diferenças finitas em 3 pontos

2.2.1. Progressivas

Fórmula:

Fórmulas de diferenças finitas em 3 pontos:

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

- $f'(x_k)$ → Aproximação do valor da derivada em x_k ;
- $f(x_k)$ → Valor da função na abscissa atual;
- $f(x_{k+1})$ → Valor da função na abscissa seguinte;
- $f(x_{k+2})$ → Valor da função em 2 abscissas a seguir;
- h → Cálculo do passo.

Algoritmo:

1. Alocação de memória de x;
2. Definir o número de pontos a n;
3. Se forem inseridos 4 elementos, y toma o valor de $f(x)$;
4. Alocação de memória da derivada;
5. Para i de 1 a n - 2, calcular o valor aproximado da derivada de f , no ponto atual;
6. Cálculo do valor aproximado da derivada de f , no ponto atual em n-1;
7. Cálculo do valor aproximado da derivada de f , no ponto atual em n.

Função em MatLab:

```
function [x,y,dydx] = DIF3PONTOS_PROG(f,a,b,h,y)
x = a:h:b;                                     %alocação de memoria de x
n=length(x);                                   % Número de pontos (tamanho do vetor de abscissas)

if nargin==4
    y=f(x);                                     % y é a função f(x), se forem inseridos 4 elementos, y recebe o valor de f(x)
end

dydx=zeros(1,n);                               % Alocação de memória da derivada

for i=1:n-2
    dydx(i)=(-3*y(i) + 4*y(i+1) - y(i+2))/(2*h); % Derivada (aproximada) de f no ponto atual
end

dydx(n-1)=(y(n-3) - 4*y(n-2) + 3*y(n-1))/(2*h); % Derivada (aproximada) de f no ponto atual em n-1
dydx(n)=(y(n-2) - 4*y(n-1) + 3*y(n))/(2*h);    % Derivada (aproximada) de f no ponto atual em n
end
```

2.2.2. Regressivas

Fórmula:

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

- $f'(x_k)$ → Aproximação do valor da derivada em x_k ;
- $f(x_{k-2})$ → Valor da função em 2 abscissas anteriores;
- $f(x_{k-1})$ → Valor da função na abscissa anterior;
- $f(x_k)$ → Valor da função na abscissa atual;
- h → Cálculo do passo.

Algoritmo:

1. Alocação de memória de x;
2. Definir o número de pontos a n;
3. Se forem inseridos 4 elementos, y toma o valor de $f(x)$;
4. Alocação de memória da derivada;
5. Valor aproximado da derivada de f no ponto atual, em 1;
6. Valor aproximado da derivada de f no ponto atual, em 2;
7. Para i de 3 a n, calcular o valor aproximado da derivada de f , no ponto atual.

Função em MatLab:

```
function [x,y,dydx] = DIF3PONTOS_REG(f,a,b,h,y)
x = a:h:b;                                % Alocação de memória para x
n = length(x);                             % Número de pontos (tamanho do vetor de abscissas)
if nargin == 4                             % y é a função f(x), se forem inseridos 4 elementos, y recebe o valor de f(x)
    y = f(x);
end
dydx = zeros(1,n);                         % Alocação de memória da derivada
dydx(1) = (-3*y(1) + 4*y(2) - y(3))/(2*h); % Derivada (aproximada) de f no ponto atual, em 1
dydx(2) = (-3*y(2) + 4*y(3) - y(4))/(2*h); % Derivada (aproximada) de f no ponto atual, em 2
for i = 3:n
    dydx(i) = (y(i-2) - 4*y(i-1) + 3*y(i))/(2*h); % Para i de 3 a n, calcular a derivada (aproximada) de f no ponto atual
end
end
```


2.2.3. Centradas

Fórmula:

$$\text{Centradas} \gg f'(x_k) := \frac{f(x_{k+1}) - f(x_{k-1}))}{2h}$$

- $f'(x_k)$ → Aproximação do valor da derivada em x_k ;
- $f(x_{k+1})$ → Valor da função na abscissa seguinte;
- $f(x_{k-1})$ → Valor da função na abscissa anterior;
- h → Cálculo do passo.

Algoritmo:

1. Alocação de memória de x;
2. Definir o número de pontos a n;
3. Se forem inseridos 4 elementos, y toma o valor de $f(x)$;
4. Alocação de memória da derivada;
5. Valor aproximado da derivada de f no ponto atual, em 1;
6. Para i de 2 a n - 1, calcular o valor aproximado da derivada de f , no ponto atual;
7. Valor aproximado da derivada de f no ponto atual, em n.

Função em MatLab:

```
function [x,y,dydx] = DIF3PONTOS_CENT(f,a,b,h,y)

x = a:h:b;                                % Alocação de memória para x
n = length(x);                             % Número de pontos (tamanho do vetor de abscissas)
if nargin == 4                             % y é a função f(x), se forem inseridos 4 elementos, y recebe o valor de f(x)
    y = f(x);
end
dydx = zeros(1,n);                         % Alocação de memória para a derivada
dydx(1) = (-3*y(1) + 4*y(2) - y(3))/(2*h); % Derivada (aproximada) de f no ponto atual, em 1.
for i = 2:n-1
    dydx(i) = (y(i+1)-y(i-1))/(2*h);        % Para i de 2 a n-1, calcular a derivada (aproximada) de f no ponto atual
end
dydx(n) = (y(n-2) - 4*y(n-1) + 3*y(n))/(2*h); % Derivada (aproximada) de f no ponto atual
end
```

2.3. 2ª Derivada

Fórmula:

$$\text{2ª derivada} \gg f''(x_k) := \frac{f(x_{k+1}) - 2f(x_k) + f(x_{k-1}))}{h^2}$$

- $f''(x_k)$ → Aproximação do valor da 2ª derivada em x_k ;
- $f(x_{k+1})$ → Valor da função na abscissa seguinte;
- $f(x_k)$ → Valor da função na abscissa atual;
- $f(x_{k-1})$ → Valor da função na abscissa anterior;
- h → Cálculo do passo.

Algoritmo:

1. Alocação de memória de x ;
2. Definir o número de pontos a n ;
3. Se forem inseridos 4 elementos, y toma o valor de $f(x)$;
4. Alocação de memória da derivada;
5. Cálculo da primeira derivada nos pontos: $x=a$, $x=a+h$, $x=a+2*h$;
6. Cálculo da 2ª derivada no ponto $x=a$;
7. Para i de 2 a $n - 1$, calcular o valor aproximado da derivada de f , no ponto atual;
8. Cálculo da primeira derivada nos pontos: $x=b-2*h$, $x=b-h$, $x=b$;
9. Cálculo da 2ª derivada no ponto $x=b$.

Função em MatLab:

```
function [x,y,dydx] = 2DERIVADA(f,a,b,h,y)

x=a:h:b;                                     % Alocação de memória
n=length(x);                                % Número de pontos (tamanho do vetor de abcissas)

if nargin==4
    y=f(x);                                  % y é a função f(x), se forem inseridos 4 elementos, y recebe o valor de f(x)
end

dydx=zeros(1,n);                             % Alocação de memória

temp1=(-3*y(1) + 4*y(2) - y(3))/(2*h);        % Primeira derivada no ponto x = a
temp2=(-3*y(2) + 4*y(3) - y(4))/(2*h);        % Primeira derivada no ponto x = a + h
temp3=(-3*y(3) + 4*y(4) - y(5))/(2*h);        % Primeira derivada no ponto x = a + 2*h

dydx(1)=(-3*temp1 + 4*temp2 - temp3)/(2*h);    % Segunda derivada no ponto x = a

for i=2:n-1
    dydx(i)=(y(i+1) - 2*y(i) + y(i-1))/(h*h); % Derivada (aproximada) de f no ponto atual
end

tempn2=(y(n-4) - 4*y(n-3) + 3*y(n-2))/(2*h); % Primeira derivada no ponto x = b - 2*h
tempn1=(y(n-3) - 4*y(n-2) + 3*y(n-1))/(2*h); % Primeira derivada no ponto x = b - h
tempn=(y(n-2) - 4*y(n-1) + 3*y(n))/(2*h);    % Primeira derivada no ponto x = b

dydx(n)=(tempn2 - 4*tempn1 + 3*tempn)/(2*h);   % Segunda derivada no ponto x = b
end
```

3. Métodos Numéricos para Integração

Na Matemática existe uma grande variedade de algoritmos cujo principal objetivo é aproximar o valor de uma dada integral definida de uma função sem o uso de uma expressão analítica para a sua primitiva.

Normalmente, estes métodos são constituídos pelas seguintes fases:

- Decomposição do domínio em subintervalos (um intervalo contido de subintervalos);
- Integração aproximada da função de cada subintervalo;
- Soma dos resultados numéricos obtidos.
- Razões da necessidade de usar a integração numérica:
- Algumas funções não admitem uma primitiva de forma explícita;
- A primitiva da função é muito complicada para ser avaliada;
- Quando não se dispõe de uma expressão analítica para o integrando, mas conhece-se os seus valores em um conjunto de pontos do domínio.

A Integração Numérica de uma função $f(x)$ num intervalo $[a, b]$ consiste no cálculo da área delimitada por essa função, recorrendo à interpolação polinomial, como forma de obtenção de um polinómio $p_n(x)$.

3.1. Regra dos Trapézios

A regra dos trapézios é um método de integração numérica usado para aproximar o integral definido e pode também ser visto como o resultado da média da parte esquerda e direita da soma de Riemann, e por vezes pode mesmo ser definido desta forma. Este método está dividido em 3 passos:

- Preencher a parte inferior da função (em cada subintervalo) com trapézios;
- Calcular as suas áreas;
- Somar os resultados do cálculo das várias áreas.

Quanto mais subintervalos existirem, mais precisa(certeira) é a sua aproximação.

Fórmula:

Regra dos Trapézios

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

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

- $I_T(f) \rightarrow$;
- $f(x_0) \rightarrow$ Valor da função na abscissa x_0 ;
- $f(x_1) \rightarrow$ Valor da função na abscissa x_1 ;
- $f(x_{n-1}) \rightarrow$ Valor da função na abscissa anterior;
- $f(x_n) \rightarrow$ Valor da função na abscissa atual;
- $h \rightarrow$ Cálculo do passo.

Algoritmo:

1. Cálculo do passo;
2. x toma o valor de a (primeira abscissa);
3. s inicializa a 0;
4. Para i de 1 a $n - 1$, somar o h com x , e somar s com $f(x)$;
5. Cálculo da Regra dos Trapézios.

Função em Matlab:

```
% RTrapezios Integracao Numerica - Formula da Regra dos Trapezios
%
% INPUT:  f - funcao integranda
%         [a, b] - intervalo de integracao
%         n - numero de iterações
%
% OUTPUT: area - Valor da area calculada pela Regra dos Trapezios

function area = RTRAPEZIOS(f,a,b,n)

h=(b-a)/n;           % h é o valor dos subintrevalos
x=a;                 % 'x' recebe o valor de 'a' (primeira abcissa)
s=0;                 % s inicializa a 0
for i=1:n-1
    x=x+h;           % Soma de h e x
    s=s+f(x);        % Soma de f(x) e s
end
area = (h/2)*(f(a)+2*s+f(b)); % Calculo da regra dos trapezios
end
```

3.2 Regra de Simpson

A Regra de Simpson é um método de Integração Numérica utilizado para a aproximação de integrais definidos e baseia-se em aproximar o integral definido pela área sob arcos de parábola que interpolam a função.

Para conseguir uma melhor aproximação deve-se dividir o intervalo de integração em intervalos mais pequenos, aplicar a fórmula de Simpson para cada um deles e somar os resultados.

Fórmula:

$$I_S(f) = \frac{h}{3} [f(x_0) + 4f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)]$$
$$|E_S| \leq \frac{b-a}{180} h^4 M_4, \quad M_4 = \max_{x \in [a,b]} |f^{(4)}(x)|$$

- $I_S(f)$ → Cálculo da regra de simpson;
- $f(x_0)$ → Valor da função na abscissa x_0 ;
- $f(x_1)$ → Valor da função na abscissa x_1 ;
- $f(x_{n-1})$ → Valor da função na abscissa anterior;
- $f(x_n)$ → Valor da função na abscissa atual;
- h → Cálculo do passo;
- n → Número de subintervalos.

Algoritmo:

1. Cálculo do passo;
2. x toma o valor de a (primeira abscissa);
3. s inicializa a 0;
4. Para i de 1 a $n - 1$, somar o h com x , se i for par soma-se 2 vezes o valor de $f(x)$ com s , e se for ímpar, soma-se 4 vezes o valor de $f(x)$ com s ;
5. Cálculo da Regra de Simpson.

Função em Matlab:

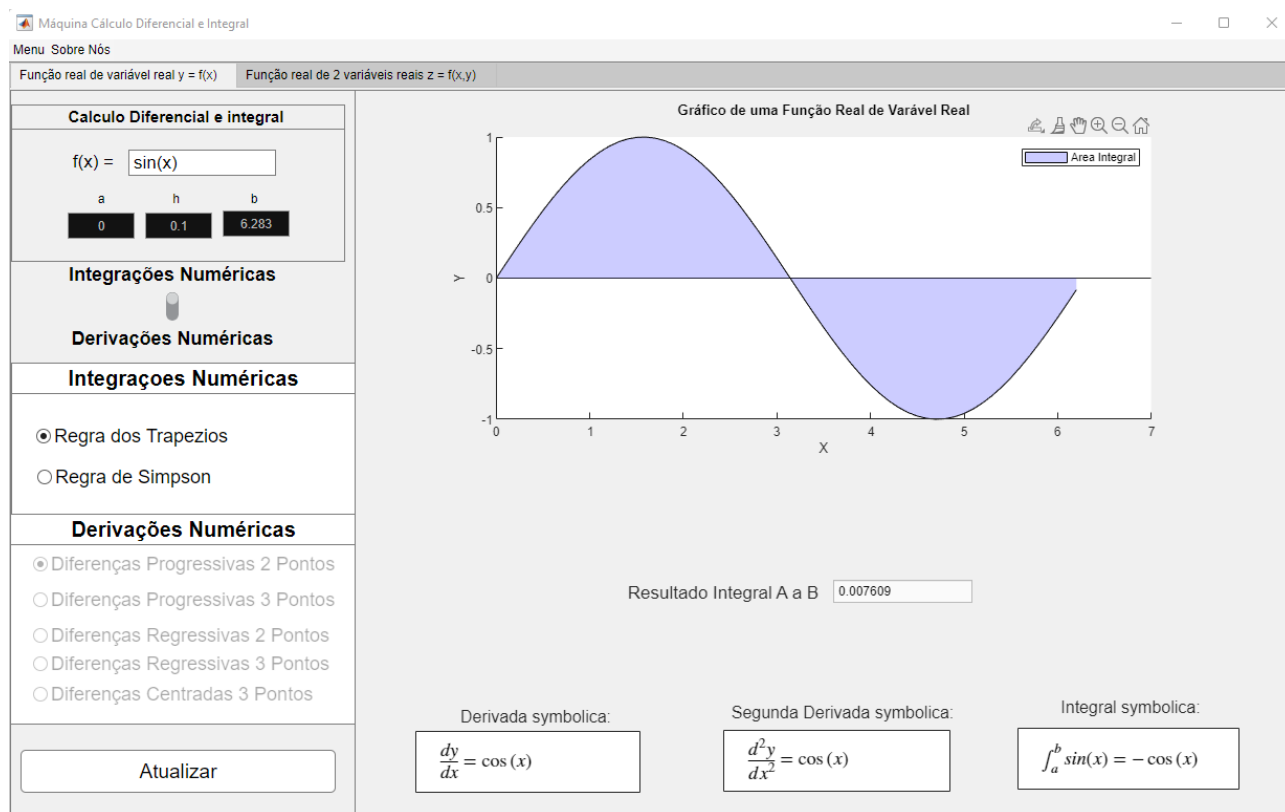
```
function area = RSIMPSON(f,a,b,n)

h=(b-a)/n;           % Valor de cada subintervalo (passo)
x=a;                 % 'x' recebe o valor de 'a' (primeira abcissa)
s=0;                 % Inicializacao da variavel 's' a 0

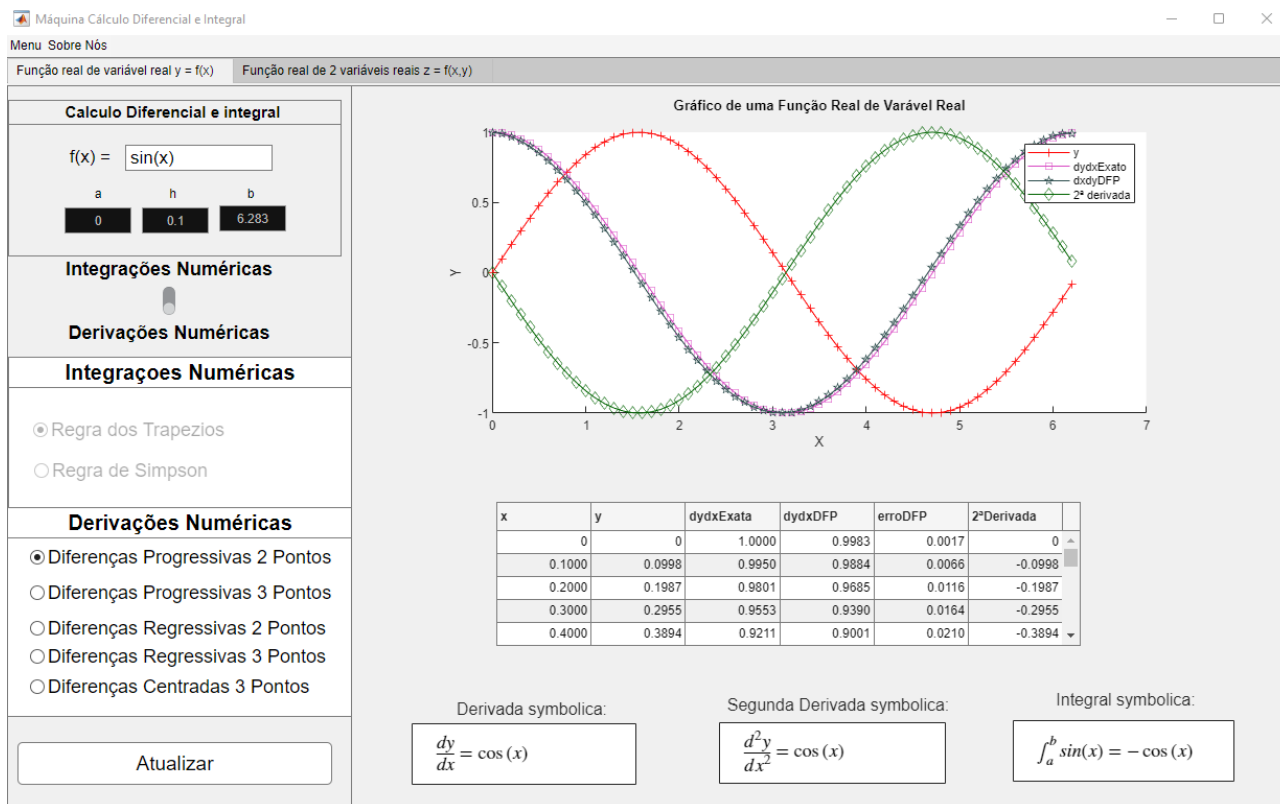
for i=1:n-1
    x=x+h;           % x toma o valor da soma de x com h
    if mod(i,2) == 0 % Se i for par
        s=s+2*f(x); % Soma de s com 2 vezes f(x)
    else
        s=s+4*f(x); % Soma de s com 4 vezes f(x)
    end
end
area = (h/3)*(f(a)+s+f(b));
end
```


4.Exemplos de aplicação

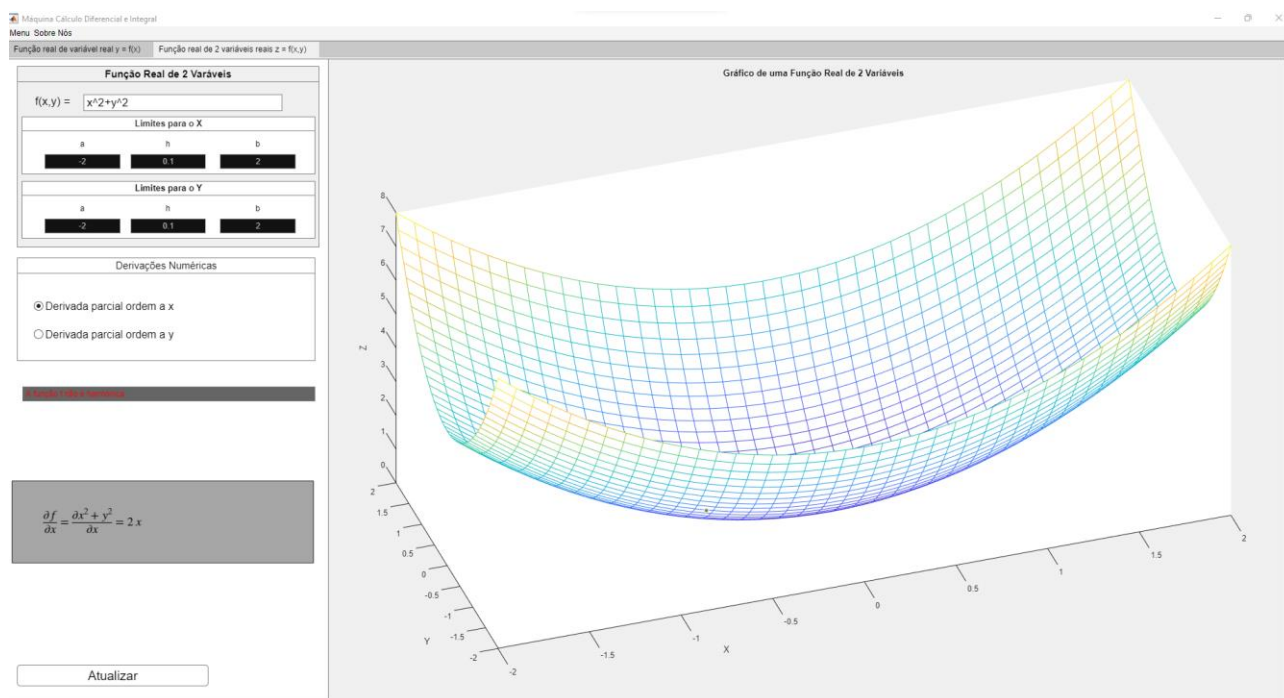
4.1 Integração



4.2 Derivação



4.3 Função real de 2 Variáveis Reais




5. Conclusão

Com a elaboração deste trabalho concluímos que foram desenvolvidos bastantes conhecimentos sobre métodos numéricos e a aplicação dos mesmos. Para além da componente matemática que este trabalho possui, também se desenvolveu bastante conhecimento em programação com MatLab. O facto de lidar com os pequenos desafios e limitações que esta linguagem de programação nos proporciona desenvolveu também o nosso trabalho de investigação para resolução dos mesmos.

Ao contrário dos outros trabalhos, em que haviam métodos que se destacavam mais que outros, neste trabalho todos os métodos se portaram muito bem. Todas as fórmulas de diferenças finitas conseguiram aproximar-se bastante bem das soluções exatas calculadas com o MATLAB. A mesma coisa se aplica às regras implementadas, como a do Trapézio e a de Simpson. Ambas produziram resultados bastante interessantes.

6. Autoavaliação e heteroavaliação

		100																													
2021/2022 » Licenciaturas em Eng. Informática » 1A / 2ªSem		Análise Matemática II																													
NºAluno		Nome Completo do Aluno		LEI(Norma, Política, ECI)	Trabalho de Estudo			Aulas Frequentadas			Gabinete	Fóruns/Facebook	AEB - Aulas/Glossário/Fóruns	25			20		20		5		10		5		15		Trab&At		
					TP(9)			TP(9)			P(9)	R	G	F		Matlab			A00Trabalho - Group			A05Trab_TP		Mín-Testes		AtividadesParticipação		Trabalhos&At [0..5]		Trabalhos&At [0..20]	
202114204		Francisco Carreira Ruivo		x								0	0	0	4,5	4,5		4		5								3,08		12,30	
2021142013		Daniel Ferreira Rodrigues		x								0	0	0	4,5	3,5		4		5								2,88		11,50	
																												0,00		0,00	
#					0			0			0	0	0	0	0	2			2		2		2				0		2		
observações:																															
													</																		

7. Bibliografia

<https://www.wolframalpha.com>

https://www.math.tecnico.ulisboa.pt/~abravo/CI/notas_cdi_i-full.pdf

https://pt.wikipedia.org/wiki/An%C3%A1lise_num%C3%A9rica

<https://www.math.tecnico.ulisboa.pt/~calves/cursos/RTrap.HTM>

<https://www.math.tecnico.ulisboa.pt/~calves/courses/integra/capiiii33.html>

<https://www.mathworks.com/help/matlab/ref/diff.html>

<https://www.mathworks.com/help/symbolic/sym.int.html>

<https://www.mathworks.com/help/matlab/ref/area.html>

https://pt.wikipedia.org/wiki/M%C3%A9todo_das_diferen%C3%A7as_finitas

https://pt.wikipedia.org/wiki/Integração_numérica