Análise Matemática II – Matemática Computacional | Engenharia Informática

TP 1

Métodos Numéricos para EDO/PVI

Diogo Silva – 2020138438 - LEI
Hugo Ferreira – 2020128305 - LEI
Rúben Mendes – 2020138473 – LEI

Índice

1. Introdução

Este trabalho foi realizado como parte da Unidade de Programa de Análise Matemática II, como uma avaliação do "ensino à distância". Os professores pretendem que os estudantes aprendam a se familiarizar com os métodos numéricos de desenvolvimento de equações diferenciais comuns e problemas de valor inicial, além da programação em Matlab e da linguagem dinâmica, essencial para a aprovação da Unidade de Análise Matemática II do Curso de Engenharia Informática.

Neste relatório, chamaremos os assuntos apresentados no enunciado do trabalho do Professor Armênio Correia, por exemplo, equações diferenciais comuns, problemas iniciais de valor e os diferentes métodos e fórmulas usadas para resolver problemas, o código e as funções são apresentados no Matlab.

Além disso, explicamos o que entendemos dos métodos Ode45 e Ode23 e porque usamos os mesmos. Vamos também fundamentar a escolha do ODE23 como função adicional do nosso trabalho.

1.1 Equação diferencial: definição e propriedades

O que é uma EDO?

Uma Equação Diferencial Ordinária (EDO) é uma equação da forma:

$$F(x, y(x), y'(x), y''(x), \ldots, y^{(n)}(x)) = 0$$

envolvendo uma função incógnita $\boldsymbol{y = y(x)}$ e suas derivadas ou suas diferenciais. $\boldsymbol{x}$ é a variável independente, y é a variável dependente e o símbolo y^k denota a derivada de ordem $\boldsymbol{k}$ da função $y = y(x)$.

Como por exemplo:

$$y'' + 3y' = \sin(x)$$
$$(2y'')^3 + 10y' + 9y = \tan(x)$$

Ordem, tipo e linearidade de uma ED

As equações diferenciais são classificadas quanto ao tipo, ordem e linearidade.

Classificação quanto ao tipo

Quanto ao tipo, as equações diferenciais são classificadas como:

- **Equações diferenciais ordinárias (EDO)** são aquelas que contêm uma ou mais derivadas de variáveis dependentes em relação a uma variável independente.
- **Equações diferenciais parciais (EDP)** são aquelas que envolve as derivadas parciais de uma ou mais variáveis dependentes em relação a uma ou mais variáveis independentes.

Classificação quanto à ordem

Quanto à ordem, uma equação diferencial pode ser de 1ª, 2ª, …, n-ésima ordem, dependendo da derivada de maior ordem presente na equação.

Temos como exemplo das diferentes ordens:

$\operatorname{sen} y\prime + y = 0$ EDO de primeira ordem

$x^2 y\prime\prime + xy\prime + y = 0$ EDO de segunda ordem

$(y\prime\prime\prime)^2 - y\prime\prime + y^2 = 0$ EDO de terceira ordem

Classificação quanto a linearidade

Quanto a linearidade de uma equação diferencial, ela pode ser linear ou não linear. Ela é linear se as incógnitas e suas derivadas aparecem de forma linear. Por exemplo uma equação diferencial ordinária de ordem n é uma equação que pode ser escrita como:

$$a_n(x)\frac{d^n y}{dx^n} + a_{n-1}(x)\frac{d^{n-1} y}{dx^{n-1}} + \ldots + a_1(x)\frac{dy}{dx} + a_0(x)y = g(x)$$

As equações diferenciais ordinárias que não podem ser escritas nessa forma são não lineares.

1.2 Definição de PVI

Uma equação diferencial ao satisfazer algumas condições é denominada de Problema de Valor Inicial (PVI):

$$\begin{cases} \boldsymbol{y' = f(t, y)} & \text{ED} \\ \boldsymbol{t \in [a, b]} & \text{VI} \\ y(a) = y_0 & \text{CI} \end{cases}$$

Seja y uma função de $\boldsymbol{x}$ e $\boldsymbol{n}$ um número inteiro positivo, então numa relação de igualdade que envolva $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{y''}, .., \boldsymbol{y}(\boldsymbol{n})$, é chamada uma **equação diferencial ordinária (EDO).**

Uma função $\boldsymbol{f}$ é a solução de uma equação diferencial se a substituição de $\boldsymbol{y}$ por $\boldsymbol{f}$. resultar em uma identidade para todo o $\boldsymbol{x}$ em algum intervalo.

Quando se impõem condições especiais que determinam um solução particular. Associados a $\boldsymbol{y}(n) = f(x, y', y'', \ldots, y^{(n-1)}$, podem existir condições cujo número coincide com a ordem da equação diferencial ordinária. Se estas condições se referem a um único x, tem-se um Problema de Valor Inicial (PVI).

2. Métodos númericos para resolução de PVI

Para um determinado PVI é possível obter resultados mais ou menos precisos da EDO dada. Dos métodos utilizados temos: **método de Euler**, método cuja aproximação à solução exata é reduzida; o **método Euler Melhorado**, que apresenta soluções parecidas ao Euler tradicional, no entanto, utiliza cálculos diferentes; o **método Runge-Kutta de ordem 2 (RK2)**, cuja aproximação à solução exata é maior que a do método de Euler. Por fim, o **método Runge-Kutta de ordem 4 (RK4)**, o método que tem soluções mais aproximadas à solução original.

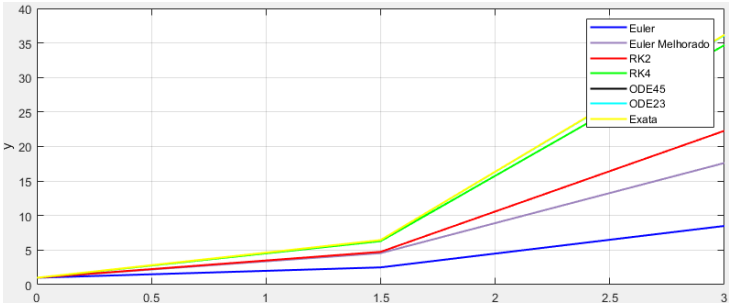
Apresentamos um problema de valor inicial:

$$\begin{cases} y'(t) = f(t, y(t)) \\ t \in [a, b] \\ y(t_0) = y_0 \end{cases}$$

Fazemos a partição regular do intervalo e obtemos:

$$h = \frac{(b-a)}{n}$$

$$t_0 = a, t_1 = t_0 + h, t_2 = t_1 + h, \dots, t_n = t_{n-1} + h = b$$



2.1. Método de Euler

Método de Euler tem o seu nome relacionado com Leonhard Euler um matemático e físico oriundo da suíça. Este é um método numérico de primeira ordem utilizado para resolver equações diferenciais ordinárias onde é dado um valor inicial.

O método de Euler é o tipo mais básico utilizado na integração numérica de equações diferenciais ordinárias. Este é bastante utilizado em matemática e ciência computacional.

2.1.1. Fórmulas necessárias para o método de Euler

$$y(t_i + 1) \approx ?$$

$$y_{i+1} = y_i + h \times f(t_i, y_i), i = 0,1,2,\dots$$

1ª Iteração

$$i = 0$$

$$y_1 = y_0 + h_f\ (t_0, y_0)$$

2ª Iteração

$i = 1$

$y_2 = y_1 + h_f\ (t_1\ , y_1)$

...

2.1.2. Algoritmo

INPUT: F, a, b, n, y_0

OUTPUT: y $\quad\quad$ $\%\ y\ =$

y_0	y_1	...	...	y_n

$h = (b - a)\ /\ n\ ;$

$t(1) = a\ ;$

$y(1) = y_0\ ;$

PARA $i = 1$ até n

$$y_{(i+1)} = y_{(i)} + h \times f\ (t_i\ , y_i)\ ;$$

$$t_{(i+1)} = t_{(i)} + h_{(i)}\ ;$$

FIM PARA

Função em matlab

function $y\ =\ MEuler(f, a, b, n, y0)$

$h = (b - a)/n;$

$t(1)\ =\ a;$

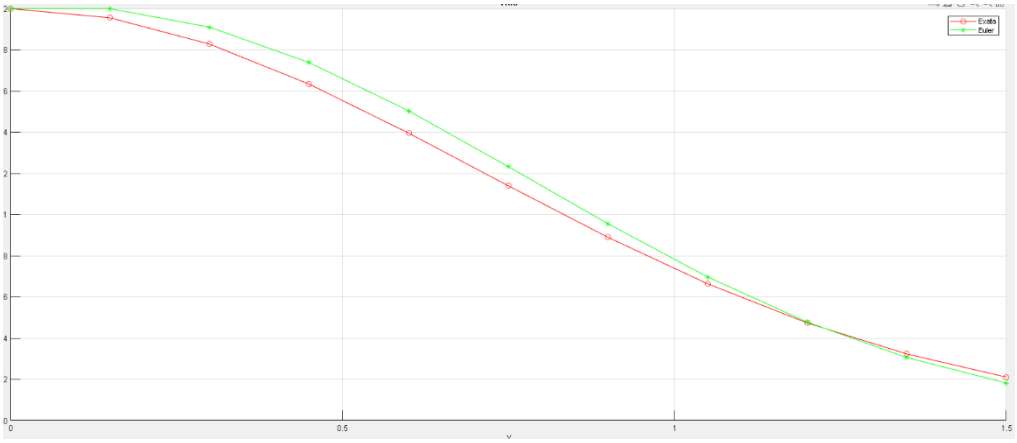
$y(1)\ =\ y0;$

for $i = 1{:}n$

$$y(i+1) = y(i) + h * f(t(i), y(i));$$

$$t(i+1) = t(i) + h;$$

end



2.2.

t	Exata	Euler	erroEuler
0	2.0000	2.0000	0
0.1500	1.9555	2.0000	0.0445
0.3000	1.8279	1.9100	0.0821
0.4500	1.6334	1.7381	0.1047
0.6000	1.3954	1.5035	0.1081
0.7500	1.1396	1.2328	0.0933
0.9000	0.8897	0.9554	0.0657
1.0500	0.6641	0.6975	0.0334
1.2000	0.4739	0.4778	0.0039

Método de Euler Melhorado

O método de Euler melhorado é semelhante em tudo com o método de Euler tradicional. Apenas existe uma diferença entre eles, o método de Euler tradicional utiliza uma média das inclinações em cada ponto para cada iteração, ou seja, tendo um x_0 e um x_1, este método calcula a inclinação em x_0 e a inclinação em x_1 e consegue assim um resultado mais aproximado. Estes são bastante parecidos, obtendo até as mesmas soluções.

2.2.1. Fórmulas de Euler Melhorado

$$y^*_{i+1} = y_i + h \times f(t_i, y_i), i = 0,1,2,\ldots$$

$$y_{i+1} = y_i + h \times \frac{f(t_i, y_i) + f(t_{i+1}, y^*_{i+1})}{2}, i = 0,1,2,\ldots$$

2.2.2. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

$t = a{:}\,h{:}\,b$;

$h = (b-a)/n;$

$y = zeros\ (1, n+1);$

$y(1) = y_0;$

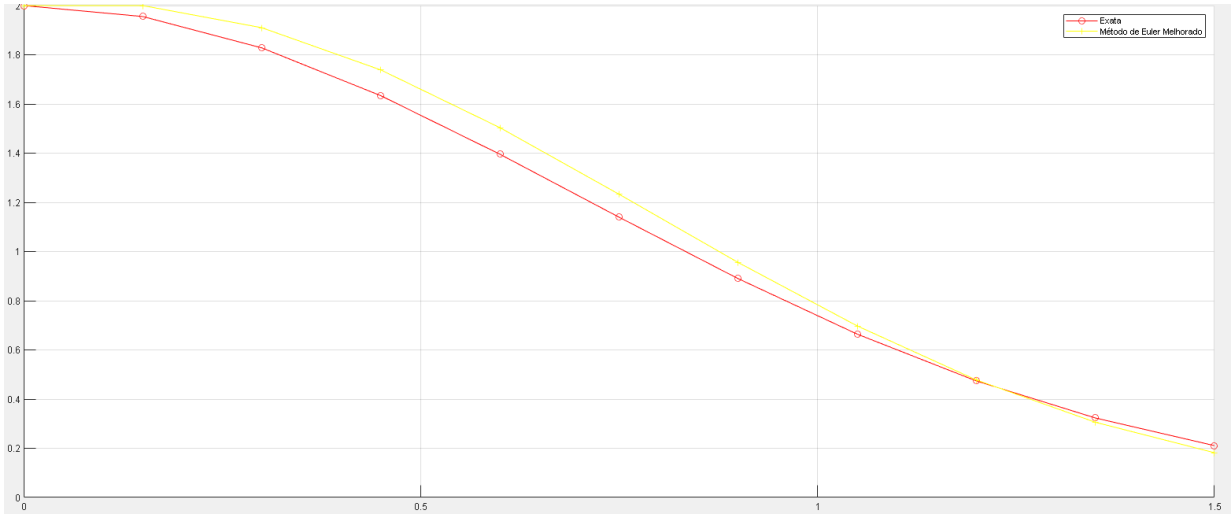
PARA $i = 1$ até n

$y(i+1) = y(i) + h \times f(t(i), y(i));$

FIM PARA

Função em Matlab

```
function y = MeulerMelhorado(f,a,b,n,y0)
        h=(b-a)/n;
        t=a:h:b;
        y=zeros(1,n+1);
        y(1)=y0;
        for i=1:n
            y(i+1)=y(i)+h*f(t(i),y(i)); , y(i);
end
```



t	Exata	Euler Método Melhorado	erroEulerM
0	2.0000	2.0000	0
0.1500	1.9555	2.0000	0.0445
0.3000	1.8279	1.9100	0.0821
0.4500	1.6334	1.7381	0.1047
0.6000	1.3954	1.5035	0.1081
0.7500	1.1396	1.2328	0.0933
0.9000	0.8897	0.9554	0.0657
1.0500	0.6641	0.6975	0.0334
1.2000	0.4739	0.4778	0.0039

2.3. Método RK2

RK2 (Runge-Kutta de ordem 2) é um método que oferece alguma precisão, visto que a sua fórmula utiliza para cada interação dois valores normalmente denominadas de “k”, o primeiro é a inclinação no início do intervalo e o segundo representa a inclinação no final do intervalo, fazendo em seguida a média das inclinações para obter a inclinação para cada interação.

Este método é bastante similar ao método de Euler melhorado visto que até obtêm as mesmas soluções.

2.3.1. Fórmulas do RK2

$$k_1 = h * f(t_i, y_i);\quad k_2 = h * f(t_{i+1}, y_i + k_1);$$

$$y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2), i = 0,1,2,\ldots,n-1$$

2.3.2. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

$h = (b-a)/n;$

$t = a{:}\,h{:}\,b$

$y = zeros(n+1, n);$

$y(1) = y_0;$

PARA $i = 1{:}\,n$

$k_1 = h * f(\,t\,(\,i\,)\,, y\,(\,i\,)\,)\,;$

$k_2 = h * f(\,t\,(i+1)\,, y\,(\,i) + k_1)\,;$

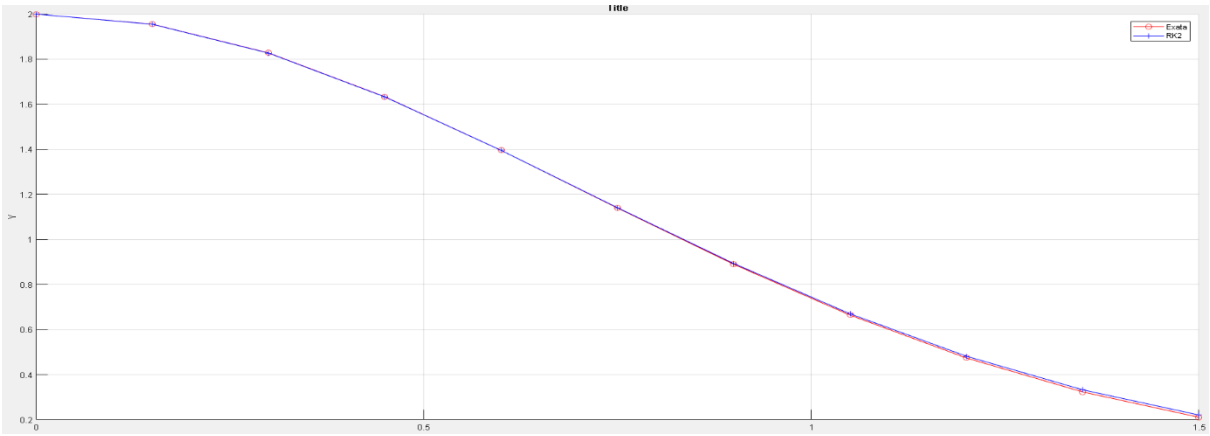
$y\,(i+1) = y\,(\,i\,) + (k_1 + k_2)\backslash 2\,;$

FIM PARA

Função em matlab

```
function y = NRK2 (f,a,b,n,y0)
h=(b-a)\n;
t=a:h:b;
y=zeros (1,n+1);
y (1)=y0;
```

```
for i=1:n
        k1=h*f(t(i),y(i));
        k2=h*f(t(i+1), y(i)+k1);
        y(i+1)=y(i)+(k1+k2)/2;
end
```



t	Exata	RK2	erroRK2
0	2.0000	2.0000	0
0.1500	1.9555	1.9550	0.0005
0.3000	1.8279	1.8270	0.0009
0.4500	1.6334	1.6326	0.0008
0.6000	1.3954	1.3953	0.0001
0.7500	1.1396	1.1410	0.0014
0.9000	0.8897	0.8932	0.0035
1.0500	0.6641	0.6700	0.0059
1.2000	0.4739	0.4818	0.0080

2.4. Método RK4

RK4 (Runge-Kutta de ordem 4) é o método numérico mais preciso de todos os abordados no trabalho prático de Análise Matemática 2. Isto deve-se à sua fórmula, que considera para cada iteração quatro valores, denominados por k, onde o primeiro é a inclinação no início do intervalo, o segundo a inclinação no ponto médio do intervalo a partir da primeira inclinação, o terceiro uma

repetição do segundo valor mas utilizando a inclinação do segundo valor em vez do primeiro e por fim o quarto que é a inclinação no final do intervalo. Assim obtém-se a inclinação para cada iteração, o que o torna tão eficiente.

2.4.1. Fórmulas do RK4

$$k_1 = h * f(t_i, y_i);\quad k_2 = h * f(t_{i+1}, y_i + k_1);$$
$$k_3 = h * f(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_2);\quad k_4 = h * f(t_{i+1}, y_i + k_3);$$
$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), i = 0,1,2,\ldots$$

2.4.2. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

$h = (b - a)/n;$

$t = a{:}\,h{:}\,b\,;$

$y = zeros\,(1, n + 1);$

y(1)=y_0;

PARA $i = 1$ até n

$k_1 = f(\,t(i), y(i)\,);$

$k_2 = f(\,t(i) + (h/2), y(i) + (h * k_1)\,/2\,);$

$k_3 = f(\,t(i) + (h/2), y(i) + h * (k_2/2)\,);$

$k_4 = f(\,t(i) + h\,, y(i) + (h * k_3)\,);$

$y(i + 1) = y(i) + (h/6) * (k_1 + 2 * k_2 + 2 * k_3 + k_4);$

$t(i + 1) = t(i) + h;$

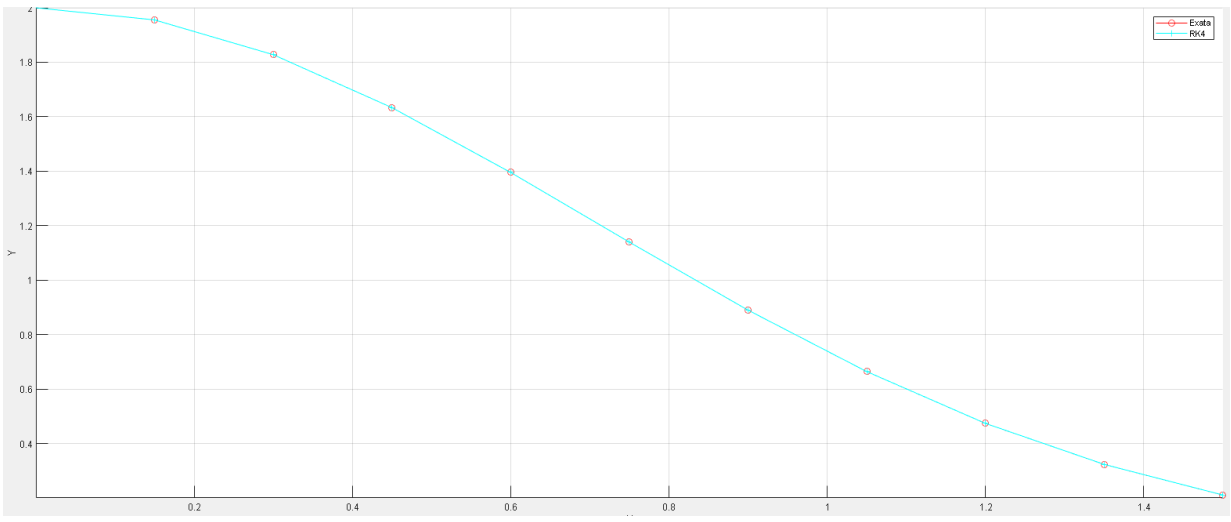
FIM PARA

Função em matlab

```
function y = NRK4(f,a,b,n,y0)
        h = (b-a)/n;
        t=a:h:b;
        y=zeros(1,n+1);
        y(1)=y0;

        for i=1:n
                k1 = f(t(i), y(i));
                k2 = f(t(i)+(h/2), y(i)+(h*k1)/2);
                k3 = f(t(i)+(h/2), y(i)+h*(k2/2));
                k4 = f(t(i)+h, y(i)+(h*k3));

                y(i+1)=y(i)+(h/6)*(k1+2*k2+2*k3+k4);
                t(i+1)=t(i)+h;
        end
```



t	Exata	RK4	erroRK4
0	2.0000	2.0000	0
0.1500	1.9555	1.9555	2.1262e-08
0.3000	1.8279	1.8279	1.8523e-07
0.4500	1.6334	1.6334	3.8408e-07
0.6000	1.3954	1.3954	1.0254e-07
0.7500	1.1396	1.1396	2.7063e-06
0.9000	0.8897	0.8897	9.1085e-06
1.0500	0.6641	0.6641	2.0325e-05
1.2000	0.4739	0.4739	3.5841e-05

2.5. Função ODE45 do Matlab

A função ode45, que é uma implementação do método de Dormand–Prince, é o recomendado para a maior parte dos casos. Esta função utiliza o método de Runge–Kutta de quarta e quinta ordens para simular e estimar o erro de integração.

2.5.1. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

$h = (b - a)/n;$

$tspan = a{:}\, h{:}\, b;$

$h = (b - a)/n;$

$t = a{:}\, h{:}\, b\ ;$

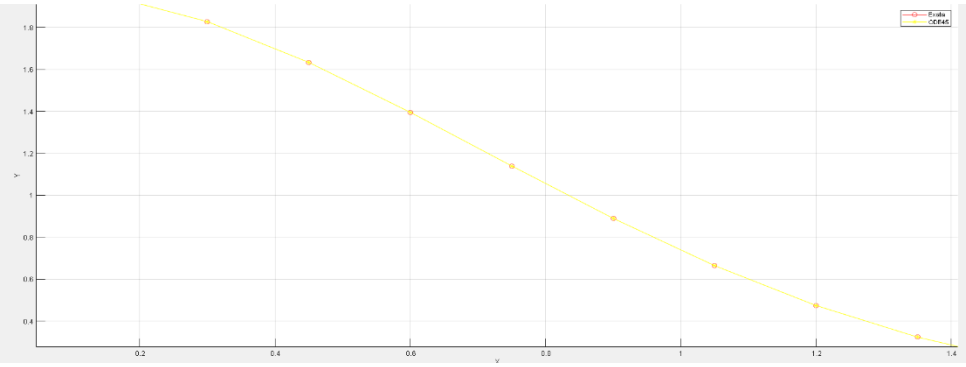
$y = zeros\ (1, n + 1);$

$sol\ =\ ode45(f, t, y);$

$[t, y]\ = deval(sol, t);$

Função em matlab

```
function [t,y]=NODE45(f,a,b,n,y0)
h=(b-a)/n;
t=a:h:b;
y=zeros(1,n+1);
sol=ode45(f,t,y0);
[t,y] = deval(sol,t);
End
```



t	Exata	ODE45	erroODE45
0	2.0000	2.0000	0
0.1500	1.9555	1.9555	1.1454e-09
0.3000	1.8279	1.8279	4.1289e-09
0.4500	1.6334	1.6334	2.1265e-08
0.6000	1.3954	1.3954	4.8469e-08
0.7500	1.1396	1.1396	6.8215e-08
0.9000	0.8897	0.8897	4.4582e-08
1.0500	0.6641	0.6641	6.7253e-08
1.2000	0.4739	0.4739	3.0020e-07

2.6. Função ODE23 do Matlab

A função ode23 é um método de integração para sistemas de equações diferenciais comuns usando fórmulas de Runge-Kutta-Fehlberg de segunda e terceira ordem com tamanho de passo automático.

2.6.1. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

$h = (b - a)/n;$

$tspan = a{:}\,h{:}\,b;$

$h = (b - a)/n;$

$t = a{:}\,h{:}\,b\,;$

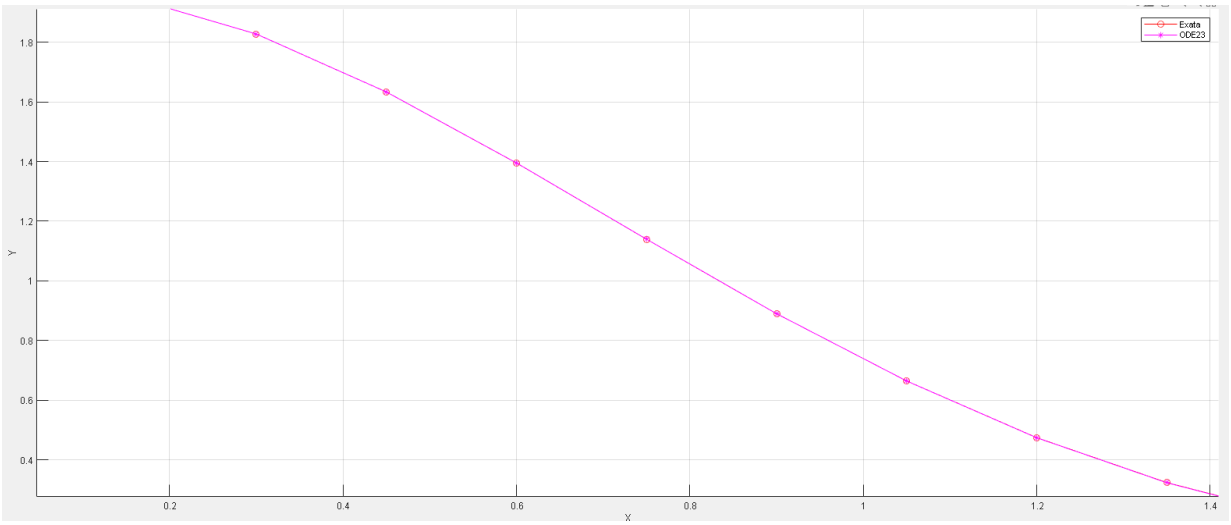
$y = zeros\,(1, n + 1);$

$sol\ = \ ode23(f, t, y);$

$[t, y]\ = deval(sol, t);$

Função em matlab

```
function [t,y]=NODE23(f,a,b,n,y0)
h=(b-a)/n;
t=a:h:b;
y=zeros(1,n+1);
sol=ode23(f,t,y0);
[t,y] = deval(sol,t);
end
```



t	Exata	ODE23	erroODE23
0	2.0000	2.0000	0
0.1500	1.9555	1.9555	3.7756e-06
0.3000	1.8279	1.8279	2.4135e-05
0.4500	1.6334	1.6334	6.1688e-05
0.6000	1.3954	1.3955	9.8931e-05
0.7500	1.1396	1.1397	1.0607e-04
0.9000	0.8897	0.8898	5.5895e-05
1.0500	0.6641	0.6640	6.0808e-05
1.2000	0.4739	0.4736	2.2816e-04

3. Exemplos de aplicação e teste dos métodos

3.1 Exercício 3 do Teste Farol

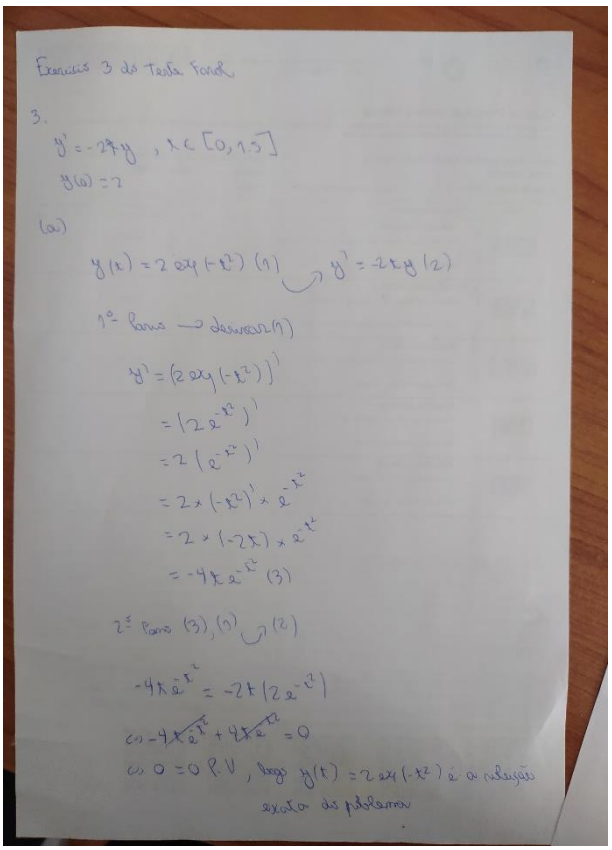
3. Considere o problema de valor inicial $y' = -2ty, \quad y(0) = 2, \quad t \in [0,1.5]$

[2.50] **(a)** Verifique que $y(t) = 2\exp(-t^2)$ é a solução exata do problema.

[5.00] **(b)** Complete a tabela seguinte e interprete os resultados obtidos. Para o preenchimento da coluna das aproximações de Euler, deve apresentar os cálculos das iterações da aplicação da fórmula do método de Euler.

			Aproximações		Erros	
i	t_i	$y(t_i)$ Exata	y_i Euler	y_i RK2	$\lvert y(t_i) - y_i \rvert$ Euler	$\lvert y(t_i) - y_i \rvert$ RK2
0	0	2			0	0
1		1.5576		1.5000		0.0576
2	1					0.0142
3	1.5	0.2108		0.3750		

(a)



b)

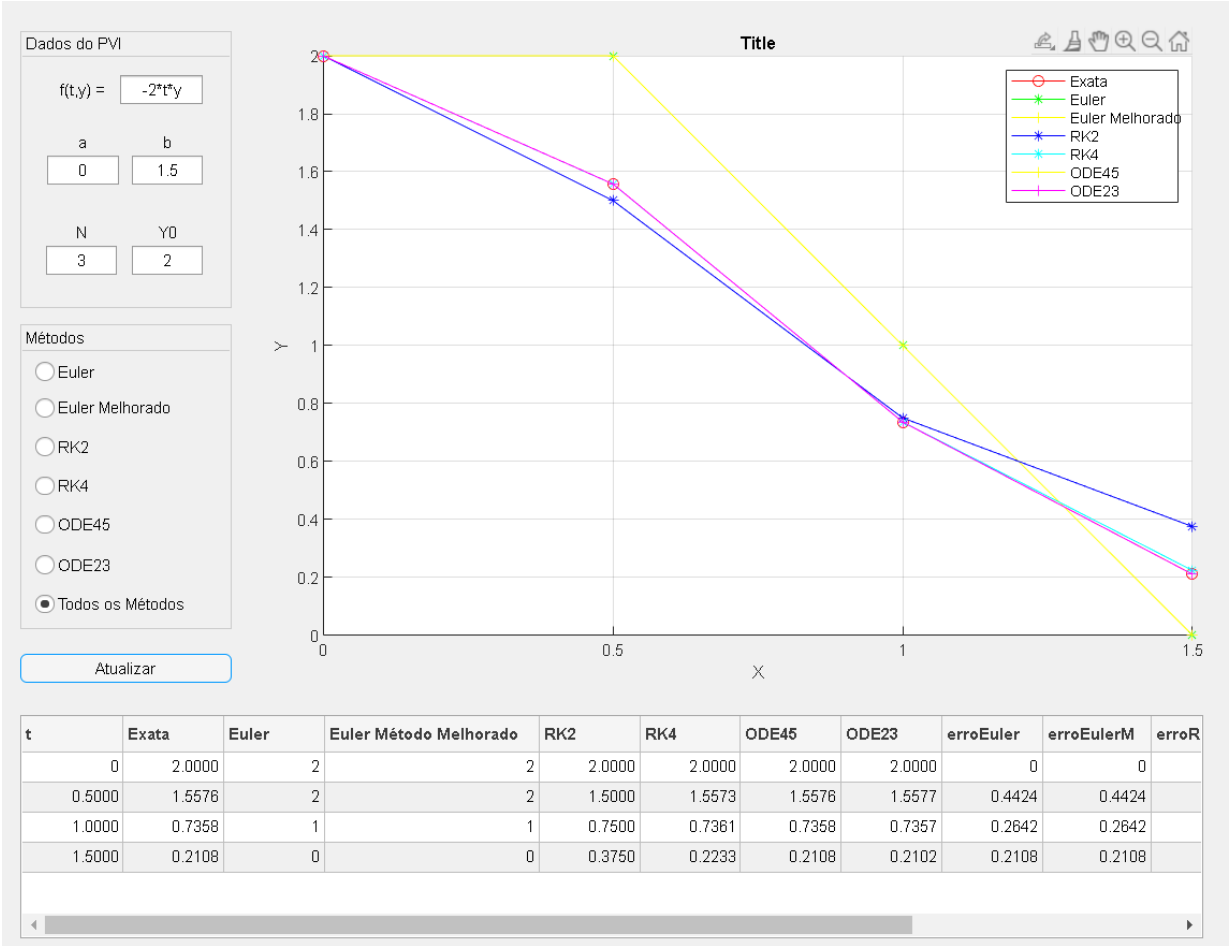
t	Exata	Euler	RK2	erroEuler	erroRK2
0	2.0000	2	2.0000	0	0
0.5000	1.5576	2	1.5000	0.4424	0.0576
1.0000	0.7358	1	0.7500	0.2642	0.0142
1.5000	0.2108	0	0.3750	0.2108	0.1642

3.1.1. PVI - ED de 1ª ordem e Condições Iniciais

$$\begin{cases} y'(t) = f(t, y(t)) \\ t \in [a, b] \\ y(t_0) = y_0 \end{cases} \Leftrightarrow \begin{cases} y'(t, y) = -2ty \\ t \in [0,1.5] \\ y(0) = 2 \end{cases}$$

$$h = \frac{(b-a)}{n} = \frac{(1.5-0)}{3} = 0.5$$

3.1.2. Exemplos de output – App com gráfico e tabela



t	Exata	Euler	Euler Método Melhorado	RK2	RK4	ODE45	ODE23	erroEuler	erroEulerM	erroR
0	2.0000	2	2	2.0000	2.0000	2.0000	2.0000	0	0	
0.5000	1.5576	2	2	1.5000	1.5573	1.5576	1.5577	0.4424	0.4424	
1.0000	0.7358	1	1	0.7500	0.7361	0.7358	0.7357	0.2642	0.2642	
1.5000	0.2108	0	0	0.3750	0.2233	0.2108	0.2102	0.2108	0.2108	

3.2 Problema de aplicação:

1. If air resistance is proportional to the square of the instantaneous velocity, then the velocity v of a mass m dropped from a height h is determined from

$$m\frac{dv}{dt} = mg - kv^2, \quad k > 0$$

Let $v(0) = 0$, $k = 0.125$, $m = 5$ slugs, and $g = 32\, ft/s^2$.

   (a) Use the Runge-Kutta method with $h = 1$ to find an approximation to the velocity of the falling mass at $t = 5\, s$.

   (b) Use a numerical solver to graph the solution of the initial-value problem.

   (c) Use separation of variables to solve the initial-value problem and find the true value $v(5)$.

2. A mathematical model for the area A (in cm^2) that a colony of bacteria (B. forbiddenkeyworddendroides) occupies is given by

$$\frac{dA}{dt} = A(2.128 - 0.0432A).$$

Suppose that the initial area is $0.24\, cm^2$.

   (a) Use the Runge-Kutta method with $h = 0.5$ to complete the following table.

$t(days)$	1	2	3	4	5
$A(observed)$	2.78	13.53	36.30	47.50	49.40
$A(approximated)$					

   (b) Use a numerical solver to graph the solution of the initial-value problem. Estimate the values $A(1)$, $A(2)$, $A(3)$, $A(4)$, and $A(5)$ from the graph.

   (c) Use separation of variables to solve the initial-value problem and compute the values $A(1)$, $A(2)$, $A(3)$, $A(4)$, and $A(5)$.

3.2.1. Modelação matemática do problema

1.

a)

m dv/dt = mg - kv^2 mv'= mg-kv^2, k>0

Como: k=0.125, m=5 e g=32ft/s^2, temos que: v'=(5*32-0.125*v^2)/5

y'= (5*32-0.125*y^2)/5;

y0 = v(0) = 0;

t E [0,5];

a=0;

b=5;

h=1;

n= ((b-a))/h= ((5-0))/1= 5;

b)

$$\begin{cases} y'(t) = f(t, y(t)) \\ t \in [a, b] \\ y(t_0) = y_0 \end{cases} \Leftrightarrow \begin{cases} y'(t, y) = \frac{5*32-0.125*y^2}{5} \\ t \in [0,5] \\ y(0) = 0 \end{cases}$$

c)

v(5)=35.7213

t	Exata	RK2	erroRK2
0	2.0000	2.0000	0.0000
1	26.4736	19.5849	6.8887
2	34.0331	24.7448	9.2883
3	35.4795	27.6280	7.8515
4	35.7272	29.5376	6.1896
5	35.7687	30.9049	4.8638

2.

a)

$$\frac{dA}{dt} = A(2.218 - 0.0432A)$$

$$\Leftrightarrow A' = A(2.218 - 0.0432A)$$

$$y' = y * (2.218 - 0.0432 * y);$$

$$y_0 = 0.24;$$

$$t \in [1,5];$$

$$a = 1;$$

$$b = 5;$$

$$h = 0{,}5;$$

$$n = \frac{(b-a)}{h} = \frac{(5-1)}{0.5} = 8;$$

T(days)	**1**	**2**	**3**	**4**	**5**
A(Observed)	**2.78**	**13.53**	**36.30**	**47.50**	**49.40**

A(Approximated)	0.2400	1.7401	10.9768	35.4325	47.4155

T(days)	**1**	**2**	**3**	**4**	**5**
A(Observed)	**2.78**	**13.53**	**36.30**	**47.50**	**49.40**
A(Approximated)	0.2400	2.1025	14.3923	40.1006	19.7660

3.2.2. Resolução através da aplicação criada

1.

a)

Método RK2:

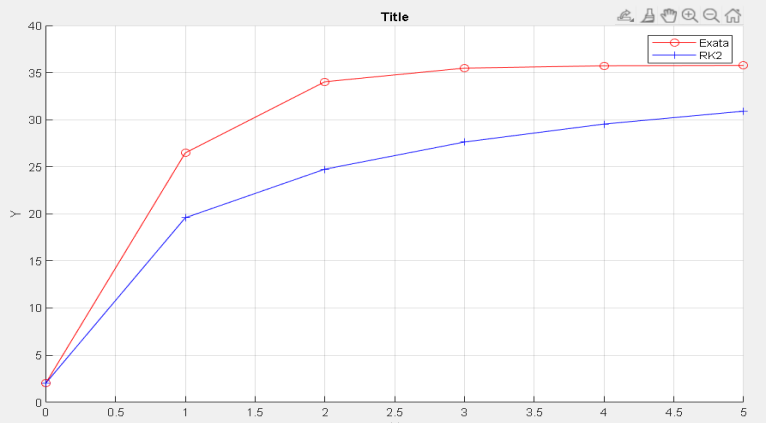
t	Exata	RK2	erroRK2
0	2.0000	2.0000	0.0000
1	26.4736	19.5849	6.8887
2	34.0331	24.7448	9.2883
3	35.4795	27.6280	7.8515
4	35.7272	29.5376	6.1896
5	35.7687	30.9049	4.8638

Método RK4:

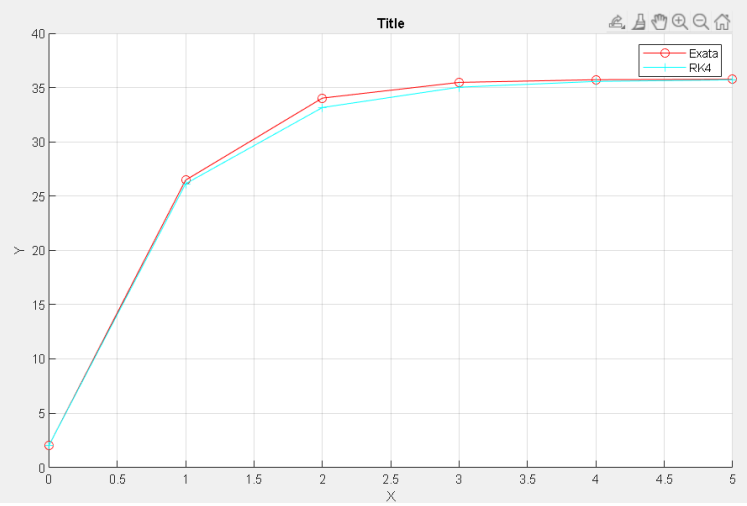
t	Exata	RK4	erroRK4
0	2.0000	2.0000	0.0000
1	26.4736	26.1384	0.3352
2	34.0331	33.1625	0.8707
3	35.4795	35.0396	0.4400
4	35.7272	35.5680	0.1592
5	35.7687	35.7178	0.0510

b)

Método RK2:



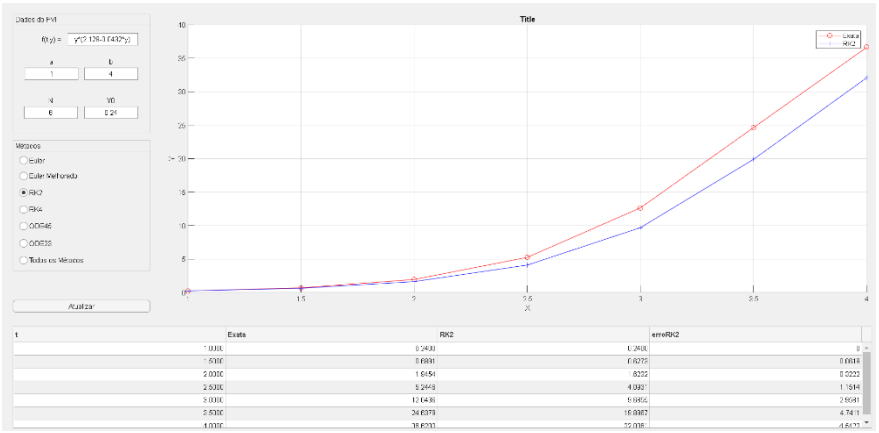
Método RK4:



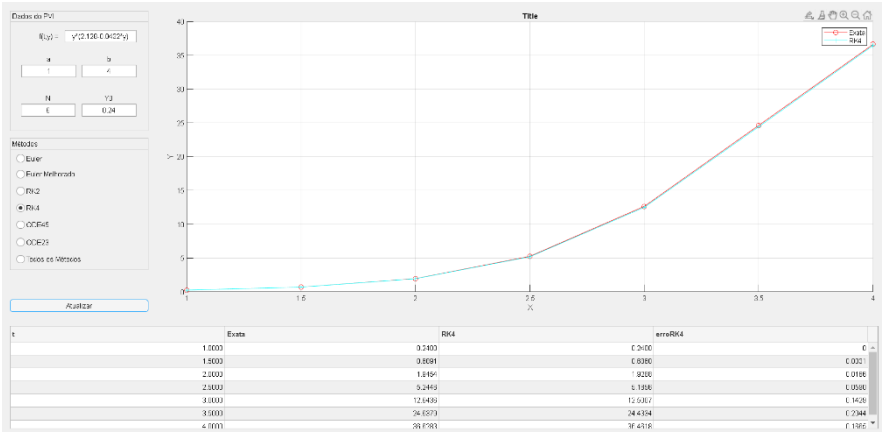
2.

b)

Método RK2:



Método RK4:



4. Conclusão

É bastante evidente a importância das equações diferenciais ordinárias (EDO) na modelagem de problemas em diversas áreas da ciência, bem como o uso de métodos numéricos para resolver tais equações.

Quando estes métodos são utilizados com o auxílio de computadores levam a uma melhor agilidade no estudo das equações diferenciais, pois conseguimos criar tabelas e

gráficos que possibilitam fazer um estudo mais detalhado dos dados. Para resolver a equação diferencial deste trabalho, utilizámos o software MATLAB e realizámos uma comparação entre os métodos numéricos de Euler, Euler Modificado e Runge-Kutta de 2º e 4º ordem.

O método de Runge-Kutta pode ser entendido como um aperfeiçoamento de método de Euler, com uma melhor estimativa da derivada da função. Os métodos de Runge-Kutta de 2ª e de 4ª ordem são métodos desenvolvidos com o objetivo de produzirem resultados com a mesma precisão que os obtidos pelo método de Taylor (método abordado nas aulas), evitando assim o cálculo das derivadas.

O método de Euler é um dos métodos mais antigo conhecido para a resolução de equações diferencias ordinárias. Todavia, os problemas práticos não devem ser resolvidos com este método, uma vez que existem outros métodos que proporcionam resultados com uma melhor precisão relativamente ao método de Euler.

Concluindo, neste trabalho apresentámos métodos de implementação simples e que produzem soluções de bastante efetividade para diversos problemas envolvendo equações diferencias.