Análise Matemática II

Trabalho Prático Nº1

Métodos Numéricos para EDO/PVI

André Lopes a2019139754
Nuno Honório a2019126457
Rafaela Carvalho a2019127935
Samuel Tavares a2019126468

Índice

1. Introdução

Este trabalho foi realizado no âmbito da Unidade Curricular de Análise Matemática II, como avaliação do “Ensino à Distância”. Pretende-se que os alunos aprendam e consigam familiarizar-se com os métodos numéricos para desenvolvimento de equações diferenciais ordinárias e problemas de valor inicial, além da linguagem de programação e dinâmica do Matlab, ambos essenciais para a aprovação à Unidade Curricular de Análise Matemática II da Licenciatura de Engenharia Informática.

Neste relatório, vamos abordar os temas apresentados no enunciado pelo professor Arménio Correia, como o que são equações diferenciais ordinárias, problemas de valor inicial e os vários métodos e fórmulas usados para resolver os problemas, ainda apresentando o seu código e função feitos no Matlab.

Além disso, vamos explicar o que entendemos dos métodos ODE45 e ODE23 e para que usamos os mesmos. Vamos também fundamentar a escolha do ODE23 como função adicional do nosso trabalho.

Na conclusão, será apresentada a conclusão deste trabalho, qual será a utilizado do mesmo para o futuro e quais foram as dificuldades encontradas.

1. Contexto do trabalho

2.1. O que é uma EDO?

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

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

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:

$$y'' + 3y' + 6y = \sin(x)$$

$$(y'')^3 + 3y' + 6y = \tan(x)$$

2.1.1. Ordem, tipo e linearidade de uma ED

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

2.1.1.1. 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.

2.1.1.2. 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 1ª ordem
$x^2 y\prime\prime + xy\prime + y = 0$	EDO de 2ª ordem
$(y\prime\prime\prime)^2 - y\prime\prime + y^2 = 0$	EDO de 3ª ordem

2.1.1.3. 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.

2.2. Problema de Valor Inicial

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

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

Seja y uma função de $\boldsymbol{x}$ e $\boldsymbol{n}$ um número inteiro positivo, então numa relação de igualdade que envolva $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{y''}, \ldots, \boldsymbol{y}(\boldsymbol{n})$, 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 Numéricos

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.

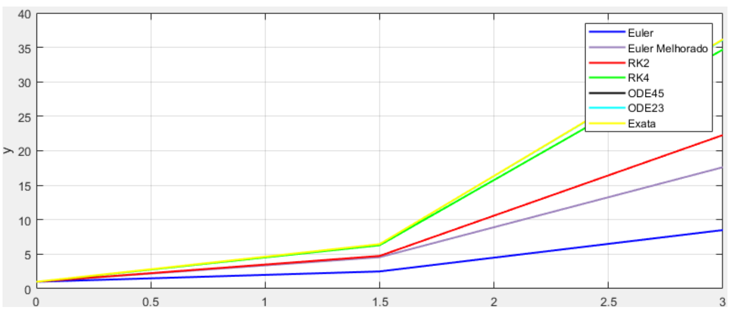
Apresentamos um problema de valor inicial:

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

Fazemos a partição regular do intervalo e obtemos:

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

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



3.1. 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.

3.1.1. Fórmulas de Euler

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

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

1ª Iteração

$$i = 0$$

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

2ª Iteração

$$i = 1$$

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

…

3.1.2. Algoritmo

INPUT: F, a, b, n, y_0

OUTPUT: y $\%\, y =$

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

3.1.3. Função em matlab

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

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

$t(1) = a;$

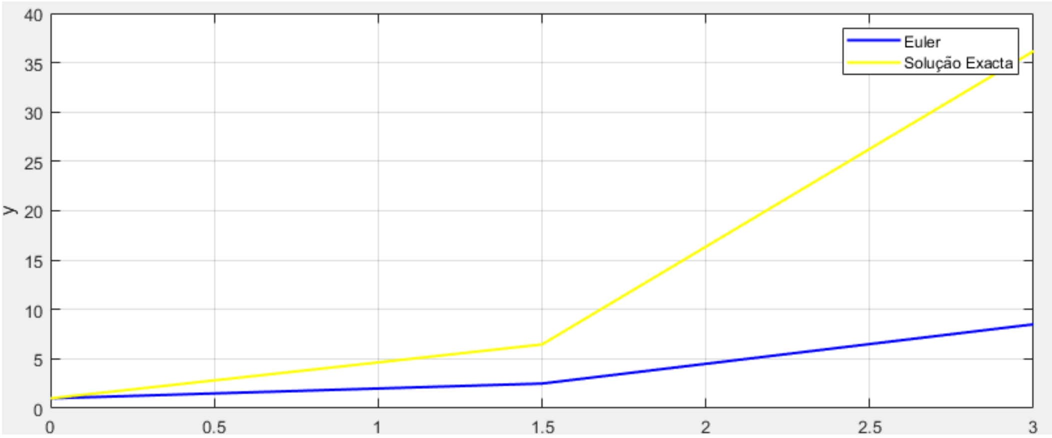
$y(1) = y0;$

for $i = 1{:}n$

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

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

end



	t	Solução Exacta	Euler	Erro Euler
1	0	1	1	0
2	1.5000	6.4634	2.5000	3.9634
3	3	36.1711	8.5000	27.6711

3.2. 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.

3.2.1. Fórmulas de Euler Melhorado

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

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

3.2.2. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

$t = a: h: b$;

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

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

$y(1) = y_0$;

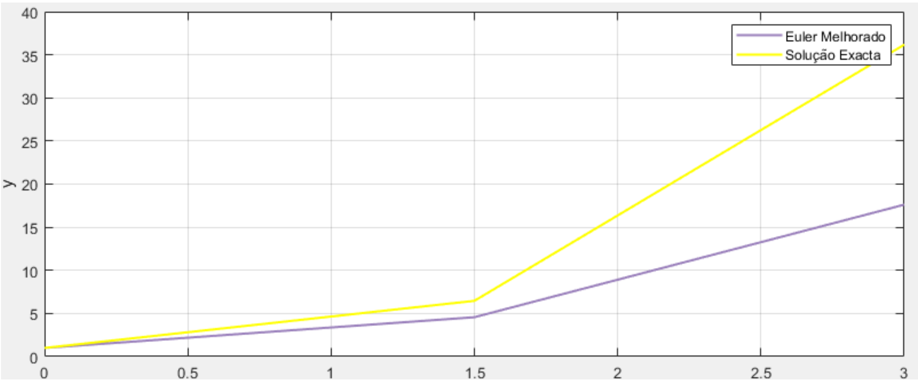
PARA $i = 1$ até n

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

FIM PARA

3.2.3. Função em matlab

```
function y = NEulerMelh(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	Solução Exacta	Euler Melhorado	Erro Euler Melhorado
1	0	1	1	0
2	1.5000	6.4634	4.5625	1.9009
3	3	36.1711	17.6172	18.5539

3.3. 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.

3.3.1. Fórmulas do RK2

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

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

3.3.2. 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 $i = 1{:}n$

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

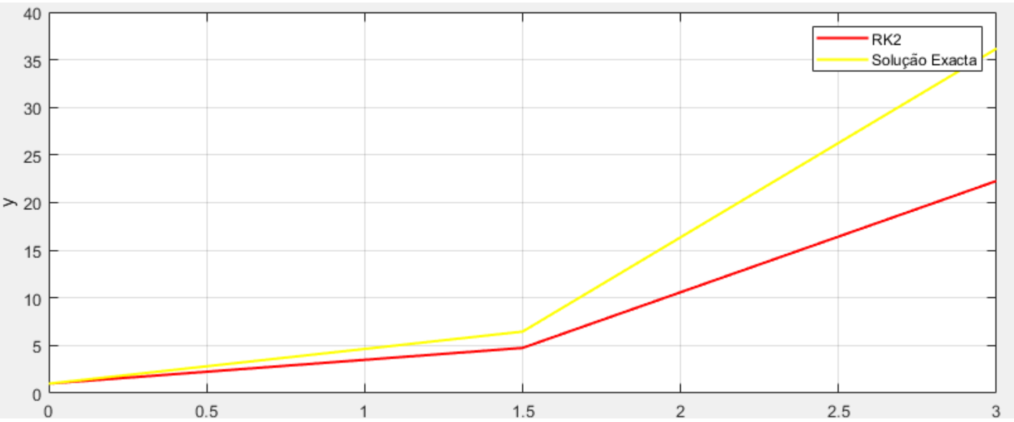
$k_2 = h * f(t(i+1), y(i) + k_1)$;

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

FIM PARA

3.3.3. Função em matlab

```
function y = KRK2 (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	Solução Exacta	RK2	Erro RK2
1	0	1	1	0
2	1.5000	6.4634	4.7500	1.7134
3	3	36.1711	22.2813	13.8898

3.4. 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.

3.4.1. Fórmulas do RK4

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

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

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), i = 0,1,2, \ldots$$

3.4.2. 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

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

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

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

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

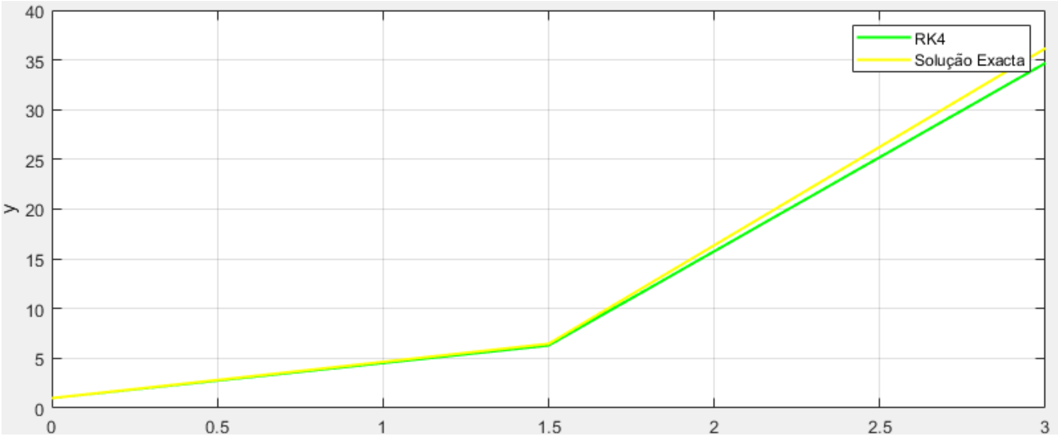
$y(i+1) = y(i) + (h/6) * (k_1 + 2 * k_2 + 2 * k_3 + k_4);$

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

FIM PARA

3.4.3. 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	Solução Exacta	RK4	Erro RK4
1	0	1	1	0
2	1.5000	6.4634	6.2969	0.1665
3	3	36.1711	34.6925	1.4786

3.5. 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.

3.5.1. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

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

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

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

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

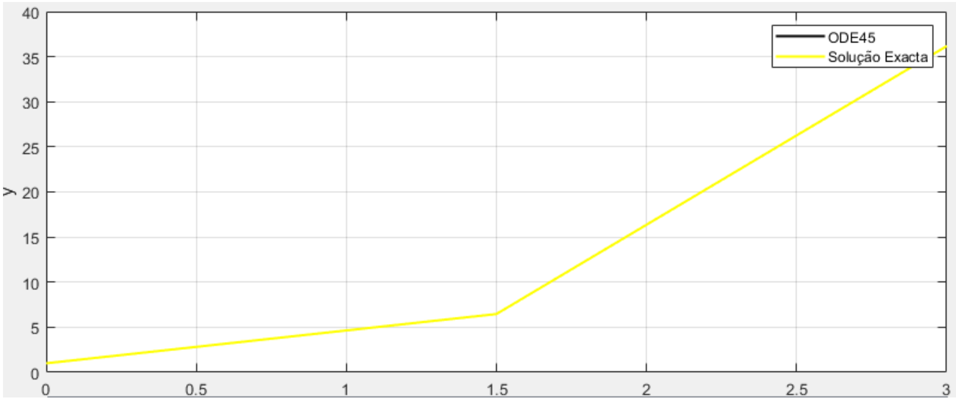
$y = zeros\,(1, n + 1);$

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

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

3.5.2. 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	Solução Exacta	ODE45	Erro ODE45
1	0	1	1	0
2	1.5000	6.4634	6.4634	1.1545e-05
3	3	36.1711	36.1711	4.2685e-05

3.6. 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.

3.6.1. Algoritmo

INPUT: f, a, b, n, y_0

OUTPUT: y

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

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

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

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

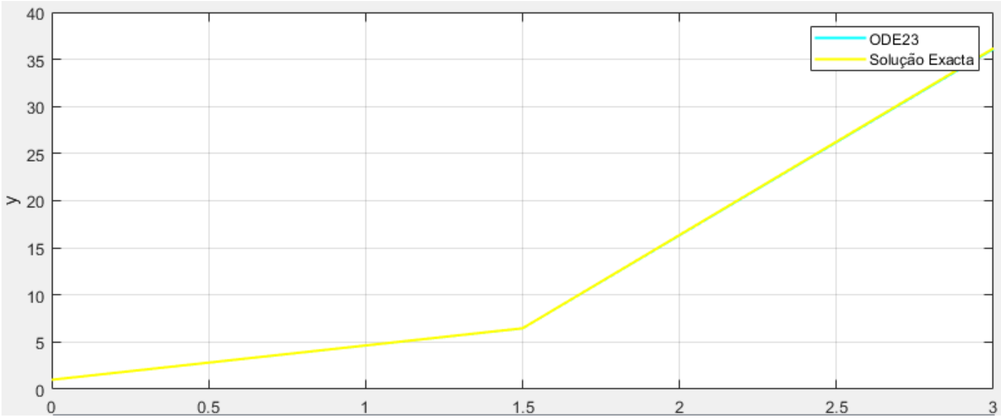
$y = zeros\ (1, n + 1);$

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

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

3.6.2. 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	Solução Exacta	ODE23	Erro ODE23
1	0	1	1	0
2	1.5000	6.4634	6.4568	0.0066
3	3	36.1711	36.1040	0.0671

4. Exemplos de aplicação e teste dos métodos

4.1 Exercício 4 do um teste A de 2015/2016

4.

a)

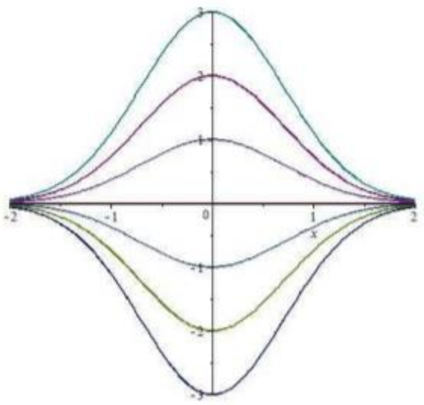


Figura 5

A figura correta é a figura 5 pois uma vez que y=c*exp(-x^2), sabendo que c é uma constante qualquer, ou seja, pode ser negativa ou positiva, podemos ter valores de y negativos ou positivos.

Sabendo que y' nos mostra se a função é crescente ou decrescente num determinado ponto, podemos substituir x por -1, y'=-2*(-1)*y⇔y'=2y.

b)

y' + 2ty = 0 ⇔ y' = -2ty

y' = $(3e^{-t^2})'$ = $3e^{-t^2}$*($-t^2$)' = -2t*3*e^{-t^2} = -6te^{-t^2}

2ty = 2t*3e^{-t^2} = 2t*3e^{-t^2} = 6te^{-t^2}

Logo, e como queremos provar y'=-2ty pois -6te^{-t^2}=-(6te^{-t^2})⇔-6te^{-t^2}=-6te^{-t^2}.

c)

I	Ti	yiExata	yiEuler	YiRK2	YiRK4	ErroEuler	ErroRK 2	ErroRK 4
0	0	3	3	3	3	0	0	0
1				2.2500	2.3359		0.0864	0.0005
2	1	1.1036	3	1.1250	1.1041	1.8964	0.0214	0.0005
3					0.3350		0.2463	0.0188
4	2	0.0549	-3	0.4219	0.0907	3.0549	0.3670	0.0358

d)

A figura correta é a figura 9, tendo em conta os valores do Método RK2, Euler e RK4.

e)

$$\begin{cases} y'(t) = f(t, y(t)) \\ t \in [a, b] \\ y(t_0) = y_0 \end{cases} \Leftrightarrow \begin{cases} y'(t, y) = -2ty \\ t \in [-2,2] \\ y(0) = 3 \end{cases}$$

f)

Euler:

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

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

$t(1) = a;$

$y(1) = y0;$

for $i = 1{:}n$

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

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

end

NRK2:

```
function y = KRK2 (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
```

g)

A script apresentada contém alguns erros, como tal a script corrigida é:

```
clear;

clc;


strF = '-2ty'

f = @(t,y) eval(vectorize(strF));

a = 0;

b = 2;

n = 2;

y0 = 3;


yEuler = NEuler(f,a,b,n,y0);

yRK2 = NRK2(f,a,b,n,y0);

yRK4 = NRK4(f,a,b,n,y0);


t = a:(b-a)/n:b;

sExata = dsolve(['Dy=',strF],['y(',a,')=',num2str(a)]);

yExata = eval(vectorize(char(sExacta)));


plot(t, yExata,'-kd')
```

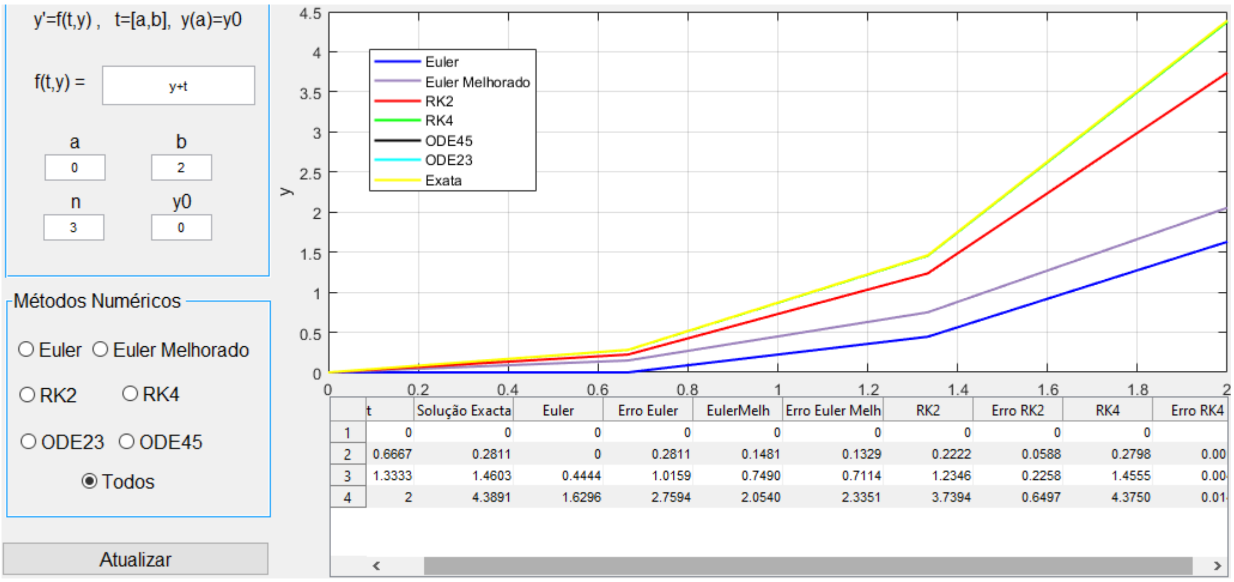
```
hold on

plot(t, yEuler,'-bo')

plot(t, yRK2,'-g*')

plot(t, yRK4,'-r+')

grid on

legend('Exata’,'Euler','RK2','RK4')

hold off


erroEuler = abs(yExata-yEuler);

erroRK2 = abs(yExata-YRK2);

erroRK4 = abs(yExata-YRK4);

tabela=[t.’,yExata.’,yEuler.’,yRK2.’,yRK4.’,…
erroEuler.’,erroRK2.’,erroRk4.’];

disp(tabela);
```

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

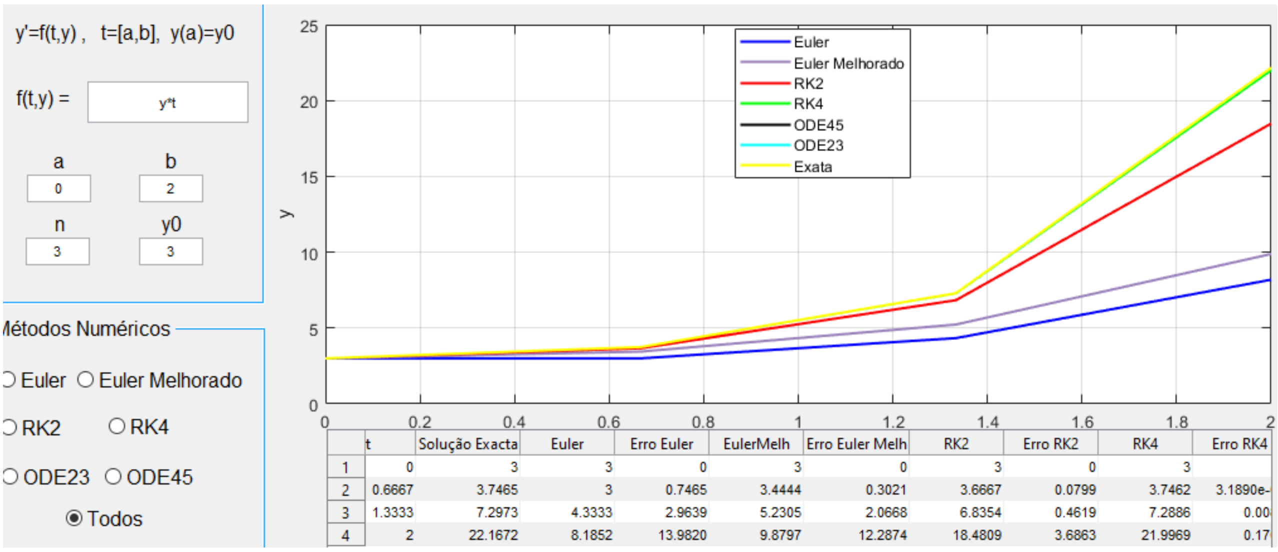
$$\begin{cases} y'(t) = f(t, y(t)) \\ t \in [a, b] \\ y(t_0) = y_0 \end{cases} \Leftrightarrow \begin{cases} y'(t, y) = -2ty \\ t \in [0,2] \\ y(0) = 3 \end{cases}$$

$$h = \frac{(b-a)}{n} = \frac{(2-0)}{2} = 1$$

4.1.2. Exemplos de output - GUI com gráfico e tabela



	t	Solução Exacta	Euler	Erro Euler	EulerMelh	Erro Euler Melh	RK2	Erro RK2	RK4	Erro RK4
1	0	0	0	0	0	0	0	0	0	
2	0.6667	0.2811	0	0.2811	0.1481	0.1329	0.2222	0.0588	0.2798	0.00
3	1.3333	1.4603	0.4444	1.0159	0.7490	0.7114	1.2346	0.2258	1.4555	0.00
4	2	4.3891	1.6296	2.7594	2.0540	2.3351	3.7394	0.6497	4.3750	0.01



	t	Solução Exacta	Euler	Erro Euler	EulerMelh	Erro Euler Melh	RK2	Erro RK2	RK4	Erro RK4
1	0	3	3	0	3	0	3	0	3	
2	0.6667	3.7465	3	0.7465	3.4444	0.3021	3.6667	0.0799	3.7462	3.1890e-
3	1.3333	7.2973	4.3333	2.9639	5.2305	2.0668	6.8354	0.4619	7.2886	0.00
4	2	22.1672	8.1852	13.9820	9.8797	12.2874	18.4809	3.6863	21.9969	0.17

4.2 Problema de aplicação:

Atividade02 .: Problema de Aplicação

Acrescentem ao vosso relatório da atividade02 » Métodos Numéricos para EDO-PVI em MATLAB e GUIs » de AM2 a resolução computacional dos seguintes problemas do livro Differential Equations with Modeling Applications, cujo autor é Dennis G.Zill.

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 $A(5)$.

10 de outubro, Arménio Correia

4.2.1. Modelação matemática do problema

1.

a)

m dv/dt = mg - kv^2 mv’= mg-kv^2

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	Solução Exacta	RK2	Erro RK2
1	0	0	0	0
2	1	25.5296	19.2000	6.3296
3	2	33.8322	24.5588	9.2734
4	3	35.4445	27.5118	7.9327
5	4	35.7213	29.4569	6.2644
6	5	35.7678	30.8457	4.9221

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

4.2.2. Resolução através da aplicação criada

1.

a)

Método RK2:

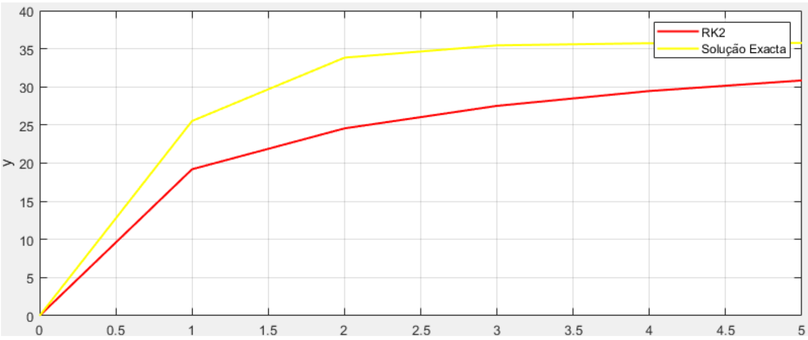
	t	Solução Exacta	RK2	Erro RK2
1	0	0	0	0
2	1	25.5296	19.2000	6.3296
3	2	33.8322	24.5588	9.2734
4	3	35.4445	27.5118	7.9327
5	4	35.7213	29.4569	6.2644
6	5	35.7678	30.8457	4.9221

Método RK4:

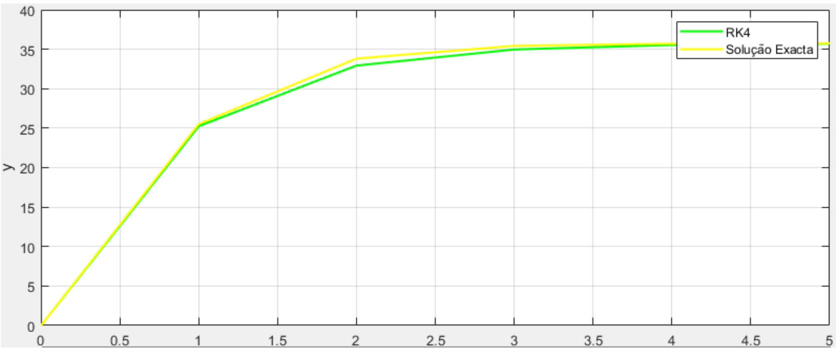
	t	Solução Exacta	RK4	Erro RK4
1	0	0	0	0
2	1	25.5296	25.2570	0.2726
3	2	33.8322	32.9390	0.8932
4	3	35.4445	34.9772	0.4673
5	4	35.7213	35.5503	0.1709
6	5	35.7678	35.7128	0.0550

b)

Método RK2:



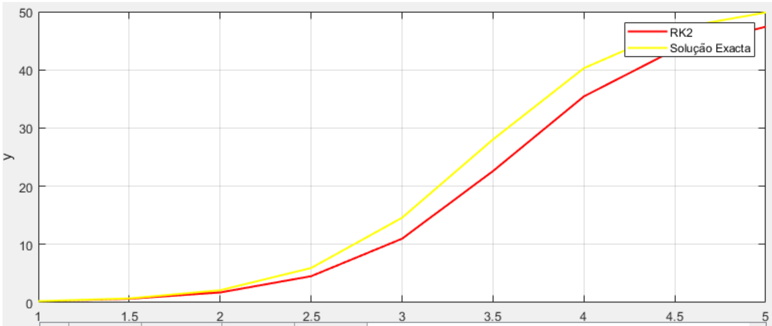
Método RK4:



2.

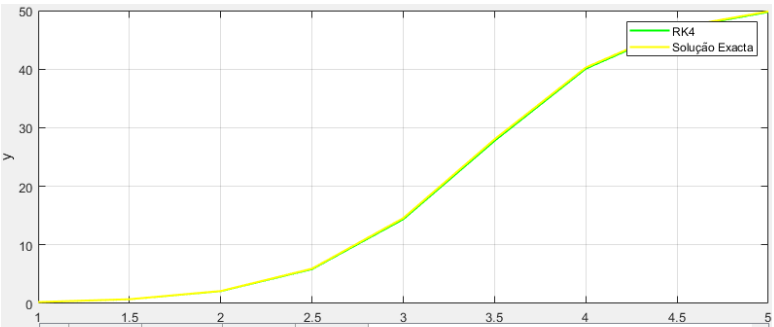
b)

Método RK2:



	t	Solução Exacta	RK2	Erro RK2
1	1	0.2400	0.2400	8.3267e-17
2	1.5000	0.7207	0.6497	0.0710
3	2	2.1240	1.7401	0.3840
4	2.5000	5.9395	4.5306	1.4089
5	3	14.5787	10.9768	3.6019
6	3.5000	28.0270	22.5960	5.4310
7	4	40.2866	35.4325	4.8542

Método RK4:



	t	Solução Exacta	RK4	Erro RK4
1	1	0.2400	0.2400	8.3267e-17
2	1.5000	0.7207	0.7168	0.0039
3	2	2.1240	2.1025	0.0215
4	2.5000	5.9395	5.8614	0.0781
5	3	14.5787	14.3923	0.1864
6	3.5000	28.0270	27.7802	0.2468
7	4	40.2866	40.1006	0.1860

5. 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.