ATIVIDADE 01 – MÉTODOS NUMÉRICOS PARA EDO/PVI

Coimbra, 30 de abril de 2021

Disciplina: Análise Matemática II
Docente: Arménio Correia
Discentes: João Almeida 2020144466
Nuno Santos 2019110035
Pedro Nogueira 2020136533

Índice

1. Introdução

O presente trabalho foi realizado no âmbito da disciplina de Análise Matemática II, disciplina essa que tem como objetivo perceber a importância da Matemática e o seu papel estruturante enquanto ciência base e, paralelamente, enquanto ferramenta de suporte do raciocínio lógico e estruturado.

Concomitantemente, a realização deste trabalho tem como principal propósito fomentar a investigação, a aprendizagem de competências algorítmicas, o estudo da programação em MATLAB, a investigação da criação de Interface Gráfica do Utilizador (GUI´s). Por outro lado, objetiva-se a confrontação dos resultados alcançados, nos diferentes Métodos Numéricos, mediante a realização de diversos exercícios.

Neste trabalho são apresentados os métodos numéricos de Euler, Euler Melhorado e Runge-Kutta de 2ª ordem e Runge-Kutta de 4ª ordem cujo principal uso é a solução de Equações Diferenciais Ordinárias (EDO,s). Para analise comparativa entre os métodos, foram efetuados cálculos no software MatLab. As conclusões se basearam no resultados obtidos para aos erros considerando os resultados, analítico e aproximados, para uma EDO dada em diversos pontos. concluiu-se que o método Runge-Kutta de 4ª ordem tem erro inferior aos demais sendo neste caso o mais eficiente.

1.1. Equação Diferencial: Definição e Propriedades

Uma equação algébrica é uma equação em que as incógnitas são números, enquanto uma equação diferencial é uma equação em que as incógnitas são funções e a equação envolve derivadas destas funções. Numa equação diferencial em que a incógnita é uma função $y(t)$, t é a variável independente e y é a variável dependente.

1.2. Definição de PVI

Chama-se a **problema de valores iniciais** (ou **problema de Cauchy**) a todo o problema que consiste numa equação diferencial satisfazendo as condições dadas num único ponto do intervalo em que a equação é considerada. Estas condições chamam-se condições iniciais[1].

Um P.V.I pode ser, matematicamente, apresentado da seguinte forma:

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

Estes PVI's podem ser resolvidos de uma forma exata ou aproximada, e o nosso trabalho trata exatamente a segunda forma, através do uso de Métodos Numéricos.

[1] *Esta designação resulta de, inicialmente, estes problemas serem dependentes do tempo, e as condições serem dadas no instante t = 0.*

2. Métodos Numéricos para a Resolução de PVI

Os métodos numéricos são algoritmos aritméticos que apresentam resultados aproximados de EDO´s, desta forma percebe-se que se tem um valor ( ) com sendo o valor aproximado e ( ) o valor exato, sendo os seguintes métodos apresentados e demonstrados. O primeiro método apresentado será o método de Euler que vem ser o método que serve de base para os demais método, após este vira o método de Euler Melhorado que como o nome no diz é um aperfeiçoamento do método de Euler, o método de Runge-kutta de 2º ordem por ultimo teremos o método de Runge-Kutta de 4º ordem.

O problema padrão para o qual desenvolveremos métodos numéricos é da forma $y'(t) = f\left(t_0, y(t)\right)$, ou seja, uma equação diferencial ordinária de primeira ordem, linear ou não, com condição inicial $y(t0) = y0$. Apesar de parecer restritivo tratar problemas deste tipo apenas, veremos que equações ordinárias de ordem superior também serão contempladas, visto que poderão ser convertidas em um problema equivalente, na forma desse problema padrão.

2.1. Cálculo do Passo

O valor do passo, h, será usado por todos os Métodos Numéricos implementados. Assim, a fim de evitar repetição desnecessária, decidimos apresentar aqui a sua definição e fórmula de cálculo. Este valor é o tamanho de cada sub-intervalos no intervalo original [a, b], e pode ser calculado da seguinte forma:

$$\boxed{h = \frac{b-a}{n}}$$

Onde:

- h → Tamanho de cada sub-intervalo (passo);
- a → Limite esquerdo do intervalo;
- b → Limite direito do intervalo;
- n → Número de sub-intervalos.

2.2. Métodos de Euler

Este é o mais simples e mais antigo dos métodos numéricos utilizados na solução particular de equações diferenciais, foi criada no séc. XVIII pelo matemático LEONHARD EULER (1707 – 1783), foi aluno de Johann Bernoulli. Ele seguiu seu passos de Daniel Bernoulli, indo para São Petersburgo em 1727. Durante o resto de sua vida esteve associado à Academia de São Petersburgo (1727-1741 e 1766-1783) e à Academia de Berlim (1741-1766). Euler foi o matemático mais prolífico de todos os tempos; suas obras completas enchem mais de 70volumes. Seus interesses incluíam todas as áreas da matemática e muitos campos de aplicação. O método de Euler foi criado por volta do ano de 1768, este método é conhecido como método da reta secante ou método de Euler. [Boyce, 2006]

O método de Euler é um procedimento numérico de primeira ordem (y') para aproximar a solução da equação diferencial $y' = f(t, y)$ que satisfaz a condição inicial: $y(t0) = y0$.

2.2.1. Fórmulas

$$\boxed{y_{\{i+1\}} = y_i + h \times f(x_i, y_i), \qquad i = 0,1,2, \dots, n-1}$$

Onde:

- $Y_{\{i+1\}}$ → Próximo valor aproximado da solução do problema original (na abcissa $t_{\{i+1\}}$) ;
- Y_i → Valor aproximado da solução do problema original na abcissa atual;
- h → Valor de cada sub-intervalo (passo);
- $\mathrm{f}(\mathrm{t_i}, \mathrm{y_i})$ → Valor da equação em t_i e y_i.

2.2.2. Algoritmo/ Função

Algoritmo:

1. Definir o valor do passo h;
2. Criar um vetor y para guardar a solução e atribuir $y(1) = y_0$;
3. Atribuir o primeiro valor de y;
4. Para $i = 1$ até n, fazemos o cálculo do método de Euler para a i-ésima iteração.

Função MATLAB:

```
function y = MEuler(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));
end
```

2.3. Método de Euler Melhorado ou Modificado

O método de Euler comparado a outros métodos numéricos percebe-se que é uma ferramenta simples, mas que para determinados problemas ele se torna falho, assim na busca por melhorar os erros $e_n = y(x_n) - y_n$ criou-se o **método de Euler melhorado**, que consiste em uma aproximação semelhante ao primeiro método utilizado, onde para fazer a aproximação $y(x_1)$ devemos realizar todos os passos do método de Euler, desta forma observamos que o método de Euler melhorado é uma modificação do método de Euler, onde este método é escrito da seguinte forma:

$$y_{\{n+1\}} = y_n + h\left(\frac{f(x_n, y_n) + f(x_n + h, y_n + h.y_n)}{2}\right)$$

Sendo $n = 0,1,2,3,\ldots$

Este método também se pode referir como Método de Euler Melhorado ou Modificado, ou um método de Runge-Kutta de 2º ordem.

2.3.1. Fórmulas

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

Onde:

- $Y_{\{i+1\}}$ → Próximo valor aproximado da solução do problema original (na abcissa $t_{\{i+1\}}$);
- Y_i → Valor aproximado da solução do problema original na abcissa atual;
- h → Valor de cada sub-intervalo (passo).

Cálculo de K1:

$$\boxed{k_1 = f(t_i, y_i)}$$

Onde:

- $K_1 \rightarrow$ Inclinação no início do intervalo;
- $f(t_i, y_i) \rightarrow$ Valor da equação em t_i e y_i.

Cálculo de K2:

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

Onde:

- $K_2 \rightarrow$ Inclinação do fim do intervalo;
- $t_i \rightarrow$ Valor da abcissa atual;
- $h \rightarrow$ Tamanho de cada sub-intervalo (passo);
- $y_i \rightarrow$ Valor aproximado da solução do problema original na abcissa atual;
- $k_1 \rightarrow$ Inclinação no início do intervalo.

2.3.2. Algoritmo/ Função

Algoritmo:

1. Definir o passo h;
2. Criar um vetor y para guardar a solução;
3. Atribuir o primeiro valor de y (condição inicial) do PVI;
4. Cálculo da inclinação no início do intervalo;
5. Cálculo da inclinação no fim do intervalo;
6. Cálculo da média das inclinações;
7. Cálculo do valor aproximado para a i-ésima iteração

Função MATLAB:

```
function y = MEulerMelh(f,a,b,n,y0)

h = (b-a)/n;
t = a:h:b;
y = zeros(1,n+1);
y(1) = y0;
z(1) = y0;
for i =1:n
    y(i+1) = y(i)+(h/2)*(f(t(i),y(i))+ f(t(i),y(i)));
end
end
```

2.4. Método RK2

Carl David Runge (1856-1927). Matemático e físico alemão. A análise de dados o levou a considerar problemas em computação numérica, e o método de Runge-Kutta tem origem em seu artigo sobre soluções numéricas de equações diferenciais de 1895. O método foi estendido para sistemas de 98 equações em 1901 por M. Wilhelm Kutta (1867-1944). Kutta era um matemático alemão que trabalhava com aerodinâmica e é, também, muito conhecido por suas contribuições importantes à teoria clássica de aerofólio.

Nesta seção trataremos do método mais famoso: o método de Runge-Kutta. Ele nasce do método de Euler, sendo o Runge-Kutta de primeiro grau o próprio método de Euler. O Runge-Kutta de segundo grau é o método de Euler melhorado, como veremos a seguir. Concluiremos a seção com o Runge-Kutta de quarto grau, que é o método mais preciso para a obtenção de soluções aproximadas para um problema de valor inicial. Cada método do Runge-Kutta é uma comparação com um polinômio de Taylor conveniente, daí que surgem os graus em seus nomes. Quando comparado a um polinômio de grau 1, teremos o Runge-Kutta de primeiro grau. Ao fazermos essa comparação, o cálculo da derivada é eliminado, fazendo-se assim avaliações da função f em cada iteração

2.4.1. Fórmulas

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

Onde:

- y_i+1 → Próximo valor aproximado da solução do problema original (na abcissa t_i+1);
- y_i → Valor aproximado da solução do problema <u>original</u> na abcissa atual.

Cálculo de K1:

$$K_1 = h \times f(t_i, y_i)$$

Onde:

- K_1 → Inclinação no início do intervalo;
- h → Valor de cada sub-intervalo;
- $f(t_i,y_i)$ → Valor da equação em x_i e y_i.

Cálculo de K2:

$$K_2 = h \times f(t_{\{i+1\}}, y_i + k_1)$$

Onde:

- K_2 -> Inclinação no fim do intervalo;
- t_i -> Valor da abcissa atual;
- h -> Tamanho de cada sub-intervalo (passo);
- y_i -> Valor aproximado da solução do problema original na abcissa atual;
- K_1 -> Inlinação no início do intervalo

2.4.2. Algoritmo/ Função

Algoritmo:

1. Definir o passo h;
2. Criar um vetor y para guardar a solução;
3. Atribuir o primeiro valor de y (condição inicial) do PVI;
4. Cálculo da inclinação no início do intervalo;
5. Cálculo da inclinação no fim do intervalo;
6. Cálculo da média das inclinações;
7. Cálculo do método de RK2 para a i-ésima iteração.

Função MATLAB:

```
function y = RK2(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
end
```

2.5. Método de RK4

O método de Runge-Kutta de 4ª ordem é o mais usado na solução numérica de problemas com equações diferenciais ordinárias. Este método não necessita do cálculo de qualquer derivada de f, mas depende de outra função que é definida avaliando f em diferentes pontos.

2.5.1. Fórmulas

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

Onde:

- Y_{i+1} → Aproximação pelo método Rk4 de $y(x_{n+1})$;
- Y_i → Valor de y na i-ésima iteração;
- h → Valor de cada sub-intervalo (passo).

E também:

$$k_1 = h * f(t_i, y_i)$$
$$k_2 = h * f(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_1)$$
$$k_3 = h * f(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_2)$$
$$k_4 = h * f(t_{i+1}, y_i + k_3)$$

Onde:

- k_1 → Inclinação no início do intervalo;
- k_2 → Inclinação no ponto médio do intervalo;
- k_3 → Inclinação (novamente) no ponto médio do intervalo;
- k_4 → Inclinação no final do intervalo.

Média ponderada das inclinações:

$$\frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

2.5.2. Algoritmo/ Função

Algoritmo:

1. Definir o passo h;
2. Criar um vetor y para guardar a solução e atribuir $y(t) = y0$;
3. Atribuir o primeiro valor de y;
4. Cálculo de inclinação no início do intervalo;
5. Cálculo da inclinação no ponto médio do intervalo;
6. Cálculo (novamente) da inclinação no ponto médio do intervalo;
7. Cálculo da inclinação no fim do intervalo;
8. Cálculo do método de RK4 para a i-ésima iteração.

Função MATLAB:

```
function y = RK4(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) + h/2, y(i) + 0.5*k1);
    k3 = h*f(t(i) + h/2, y(i) + 0.5*k2);
    k4 = h*f(t(i+1), y(i) + k3);

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

end
```

2.6. Função ODE45 do Matlab

A função ODE45 é uma das função nativas do MATLAB, e é baseada num método de Runge-Kutta.

2.6.1. Fórmulas

$$[t, y] = ode45(f, t, y0)$$

Onde:

- $t \rightarrow$ Vetor das abcissas;
- $f \rightarrow$ Equação diferencial em t e em y;
- $y_0 \rightarrow$ Valor inicial do PVI (condição inicial).

2.6.2. Algoritmo/Função

Algoritmo:

1. Definir o passo h;
2. Aproximação através da função ODE45.

Função MATLAB:

```
function y = ODE45(f,a,b,n,y0)

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

    [~,y] = ode45(f,t,y0);
    y = y.';
end
```

2.7. Método Dormand Prince

Em análise numérica, **Dormand-Prince** é um método para resolução de equações diferenciais ordinárias (EDOs). O método é um membro da família de métodos Runge-Kutta para resolver EDOs. Mais precisamente, ele avalia seis vezes a função para calcular soluções de quarta e quinta ordem. A diferença entre essas soluções é então tomada como o erro da solução (de quarta ordem). Esta estimativa de erro é muito conveniente para algoritmos de integração adaptativos. Outros métodos de integração parecidos são o método de Runge-Kutta-Fehlberg (RKF) e o *método Cash-Karp (RKCK).*

2.7.1. Fórmulas

2.7.2. Algoritmo/Função

Baseámo-nos no referente link Dormand Prince para construção do referente método.

Função MATLAB:

```
function y = DormandPrince(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/5,     y(i) + (h*k1/5));
    k3 = f(t(i) + (3/10)*h, y(i) + (3*h*k1)/40 + (9*h*k2/40));
    k4 = f(t(i) + (4/5)*h,  y(i) + (44*h*k1)/45 - (56*h*k2)/15 + (32*h*k3/9));
    k5 = f(t(i) + (8/9)*h,  y(i) + (19372*h*k1)/6561 - (25360*h*k2)/2187 + (64448*h*k3)/6561 - (212*h*k4)/729);
    k6 = f(t(i) + h,        y(i) + (9017*h*k1)/3168 - (355*h*k2)/33 + (46732*h*k3)/5247 + (49*h*k4)/176 - (5103*h*k5)/18656);
    k7 = f(t(i) + h,        y(i) + (35*h*k1)/384 + (500*h*k3)/1113 + (125*h*k4/192) - (2187*h*k5/6784) + (11*h*k6/84));

    y(i + 1) = y(i) + h*(((35*k1)/384) + ((500*k3)/1113) + (125/192)*k4 - (2187/6784)*k5 + (11/84)*k6);
end
```

3. Exemplos de Aplicação e Teste dos Métodos

3.2. Exercício 3 do Teste Farol

3.2.1. PVI – Equação Diferencial de 1ª Ordem e Condições Iniciais

Resolução do Exercício 3 do Teste "Farol" 2015/2016

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

[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	$\|y(t_i) - y_i\|$ Euler	$\|y(t_i) - y_i\|$ RK2
0	0	2			0	0
1		1.5576		1.5000		0.0576
2	1					0.0142
3	1.5	0.2108		0.3750		

[0.50] **(c)** Qual das figuras seguintes representa graficamente uma solução do PVI dado? Justifique a sua resposta.

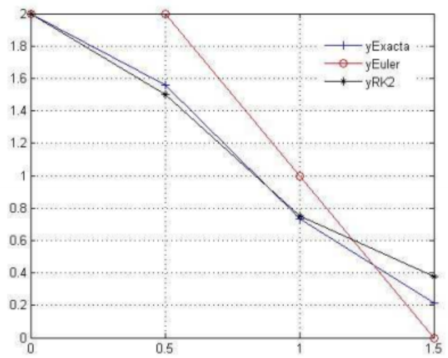


Figura 4

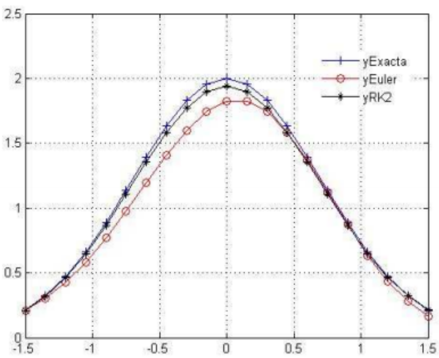


Figura 5

[1.00] **(d)** Estabeleça um PVI cuja solução em modo gráfico coincide com a figura que excluiu na alínea anterior.

[1.00] **(e)** Quais dos comandos seguintes em GeoGebra lhe permitiriam determinar a solução exata do PVI e a solução aproximada do mesmo.

(A)	`SolveODE[-2xy,(0,2)]`	**(B)**	`SolveODE[-2xy,(-1.5,0.2108)]`
(C)	`NSolveODE[{-2xy}, 0, {2}, 1.5]`	**(D)**	`NSolveODE[{-2xy}, -1.5, {0.2108}, 1.5]`

FORMULÁRIO

PVI	**Método de Euler**	**Método de Runge-Kutta (RK2)**
$(P)\begin{cases} y' = f(t,y) \\ t \in [a,b] \\ y(a) = y_0 \end{cases}$	$y_{i+1} = y_i + h \times f(t_i, y_i)$, $i = 0,1,2,\ldots,n-1$	$k1 = h \times f(t_i, y_i)$ $k2 = h \times f(t_{i+1}, y_i + k1)$ $y_{i+1} = y_i + \frac{1}{2}(k1 + k2)$, $i = 0,1,2,\ldots,n-1$

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

```
clc
clear

%help dsolve

syms y(t)

while 1
    %strF = input("f(t,y) = ","s");
    strF = "2*t*y"
    f =@(t,y) eval(vectorize(strF));
    try
        fTeste = f(t,y);
        break
    catch ME
        errordlg("Introduza uma função em t e y","Erro","modal")
    end
end
```

```
strF =
"2*t*y"
```

```
while 1
    a = 1;
    if (isscalar(a) && isreal(a))
        break
    end
end

b = 1.5
```

```
b = 1.5000
```

```
n = 20
```

```
n = 20
```

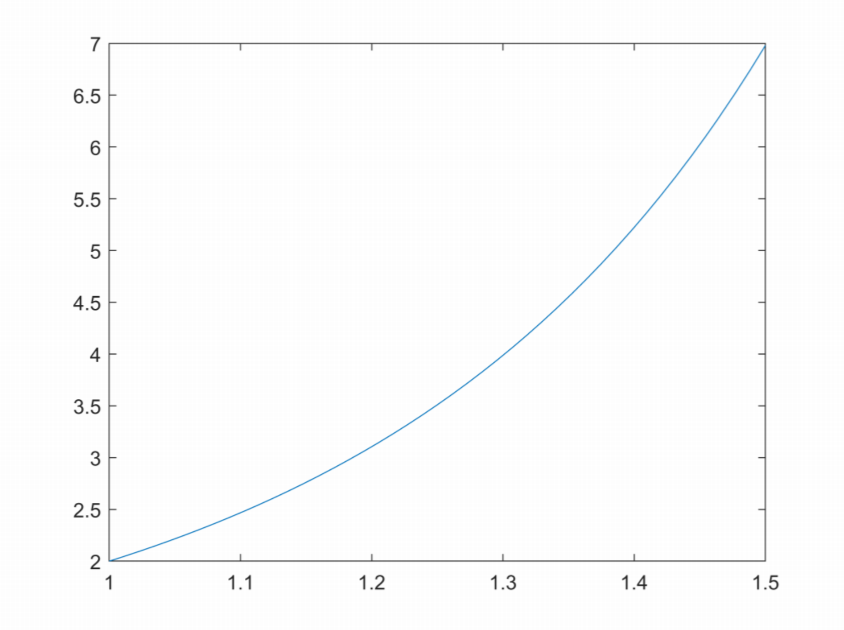
```
y0 = 2
```

```
y0 = 2
```

```
sExata = dsolve(diff(y,t)==f(t,y),y(a)==y0)
```

sExata = $2\,e^{t^2}\,e^{-1}$

```
fplot(sExata,[a,b])
```




```
%help function_handle
g =@(t) eval(vectorize(char(sExata)))
```

```
g = function_handle with value:
    @(t)eval(vectorize(char(sExata)))
```

```
h = (b-a)/n
```

```
h = 0.0250
```

```
t = a:h:b
```

```
t = 1×21
    1.0000    1.0250    1.0500    1.0750    1.1000    1.1250    1.1500    1.1750 ···
```

```
yExata = g(t)
```

```
yExata = 1×21
    2.0000    2.1039    2.2159    2.3368    2.4674    2.6085    2.7611    2.9264 ···
```

```
tabela = [t.',yExata.']
```

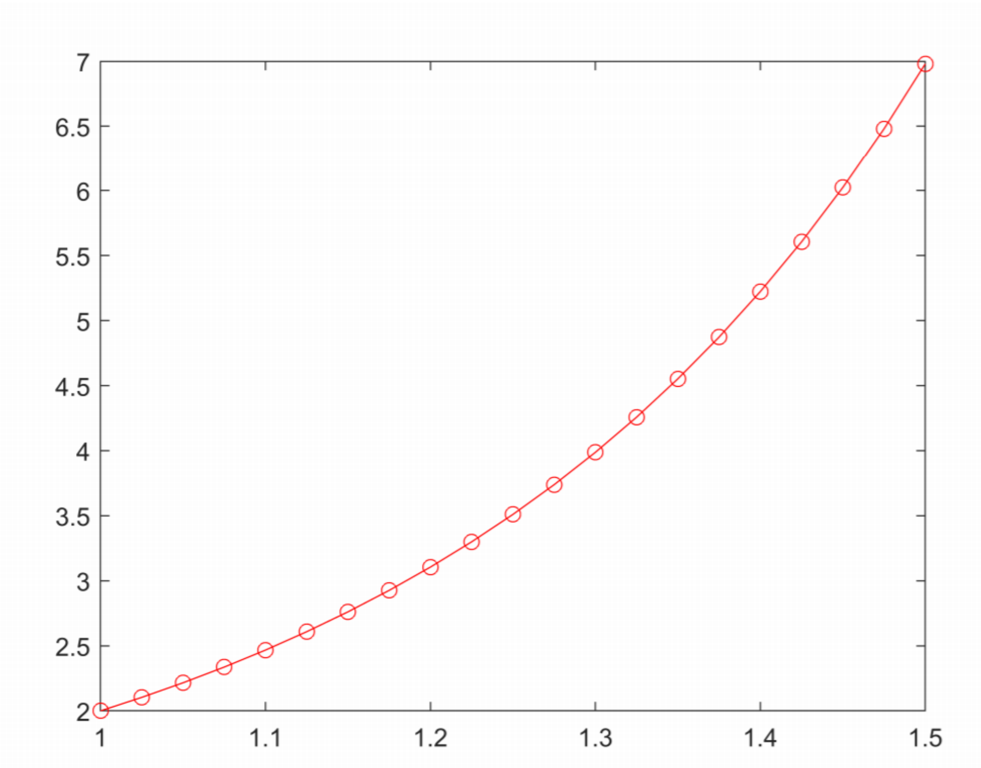
```
tabela = 21×2
    1.0000    2.0000
    1.0250    2.1039
    1.0500    2.2159
    1.0750    2.3368

    1.1000    2.4674
    1.1250    2.6085
    1.1500    2.7611
    1.1750    2.9264
    1.2000    3.1054
    1.2250    3.2995
      ⋮
```

```
array2table(tabela,"VariableNames",{'t','yExata'})
```

ans = 21×2 table

	t	yExata
1	1.0000	2.0000
2	1.0250	2.1039
3	1.0500	2.2159
4	1.0750	2.3368
5	1.1000	2.4674
6	1.1250	2.6085
7	1.1500	2.7611
8	1.1750	2.9264
9	1.2000	3.1054
10	1.2250	3.2995
11	1.2500	3.5101
12	1.2750	3.7388
13	1.3000	3.9874
14	1.3250	4.2579
15	1.3500	4.5524
16	1.3750	4.8733
17	1.4000	5.2234
18	1.4250	5.6056
19	1.4500	6.0234
20	1.4750	6.4803
21	1.5000	6.9807

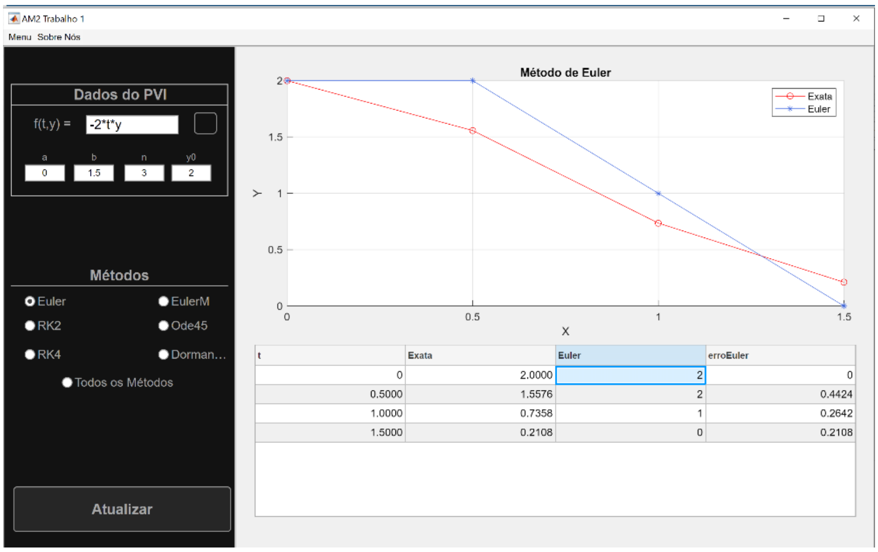
```
plot(t,yExata,"-ro")
```



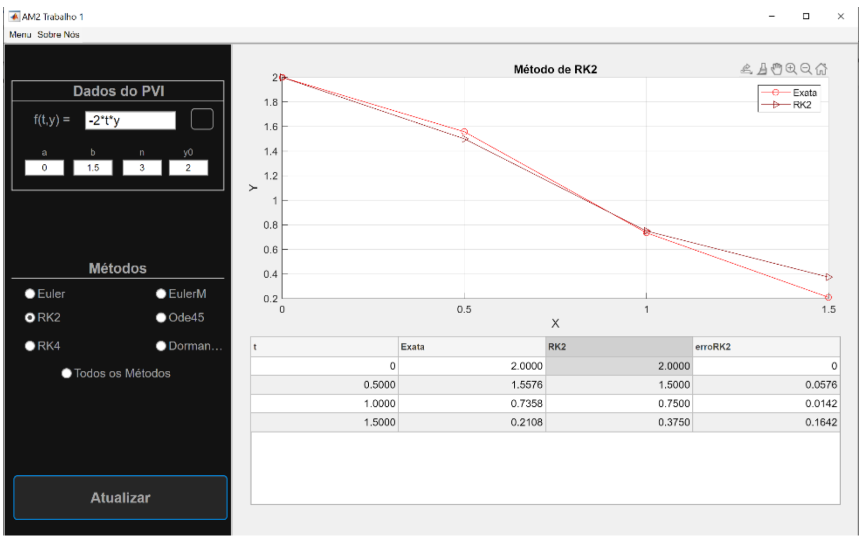
(b) Complete a tabela seguinte e interprete os resultados obtidos. Para o preenchimento da coluna das aproximações de Euler, deve apresentar os cálculos das iterações da aplicação da fórmula do método de Euler.

			Aproximações		Erros	
i	t_i	$y(t_i)$ Exata	y_i Euler	y_i RK2	$\|y(t_i) - y_i\|$ Euler	$\|y(t_i) - y_i\|$ RK2
0	0	2	2	2	0	0
1	0,5	1.5576	2	1.5000	0,4424	0.0576
2	1	0,7358	1	0,7216	0,2642	0.0142
3	1.5	0.2108	0	0.3750	0,2108	0,1642

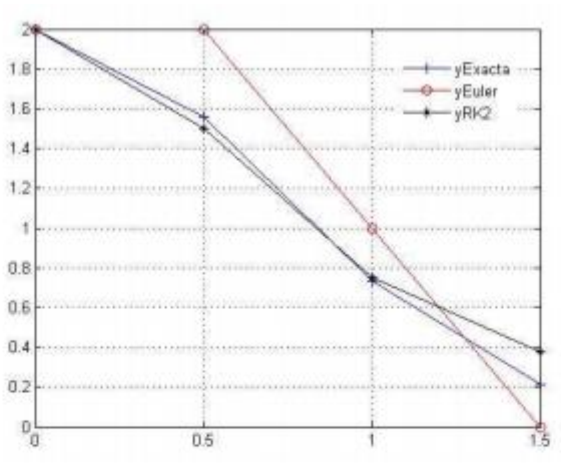
Euler:



RK2:



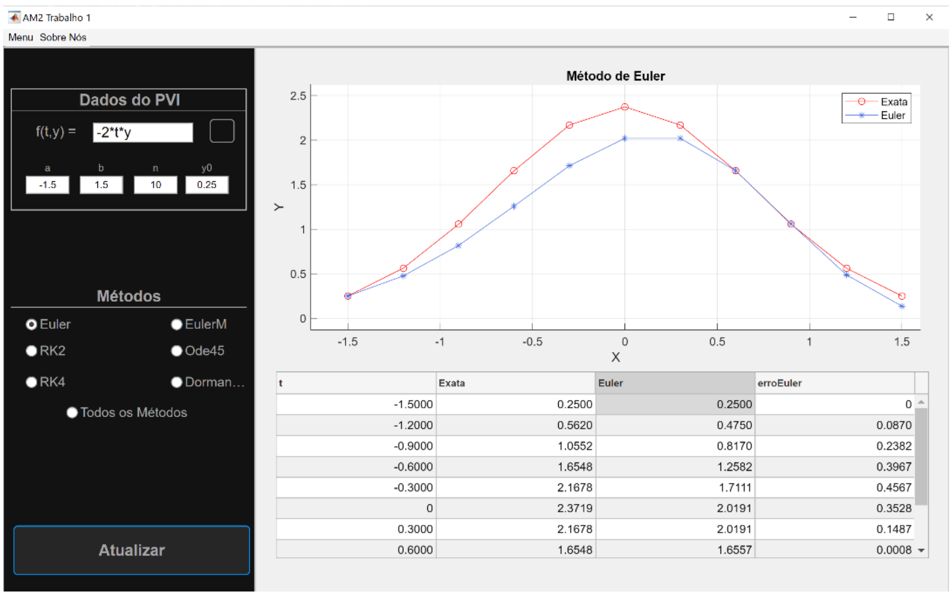
(c) Qual das figuras seguintes representa graficamente uma solução do PVI dado? Justifique a sua resposta.



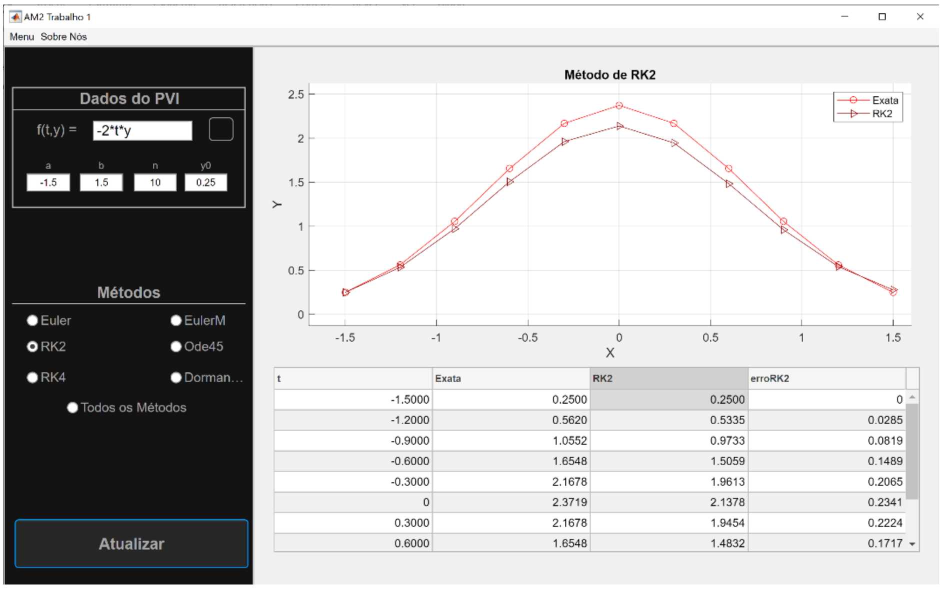
Nos gráficos acima, na nossa aplicação, é possível identificar que este gráfico corresponde ao nosso PVI.

(d) Estabeleça um PVI cuja solução em modo gráfico coincide com a figura que excluiu na alínea anterior.

Euler:

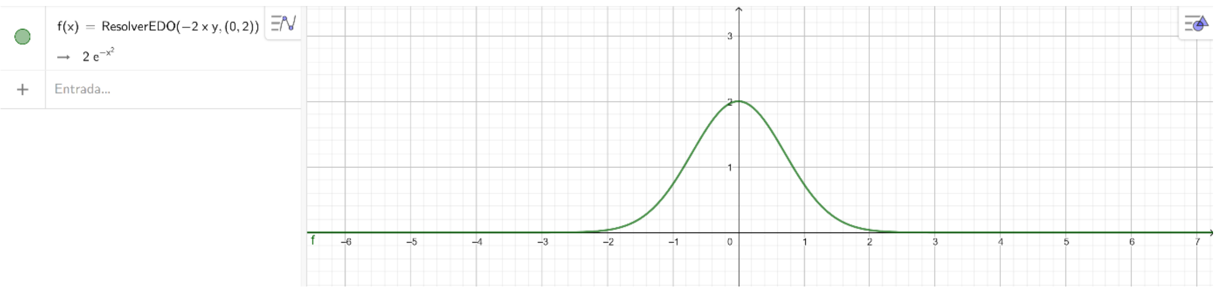


RK2:



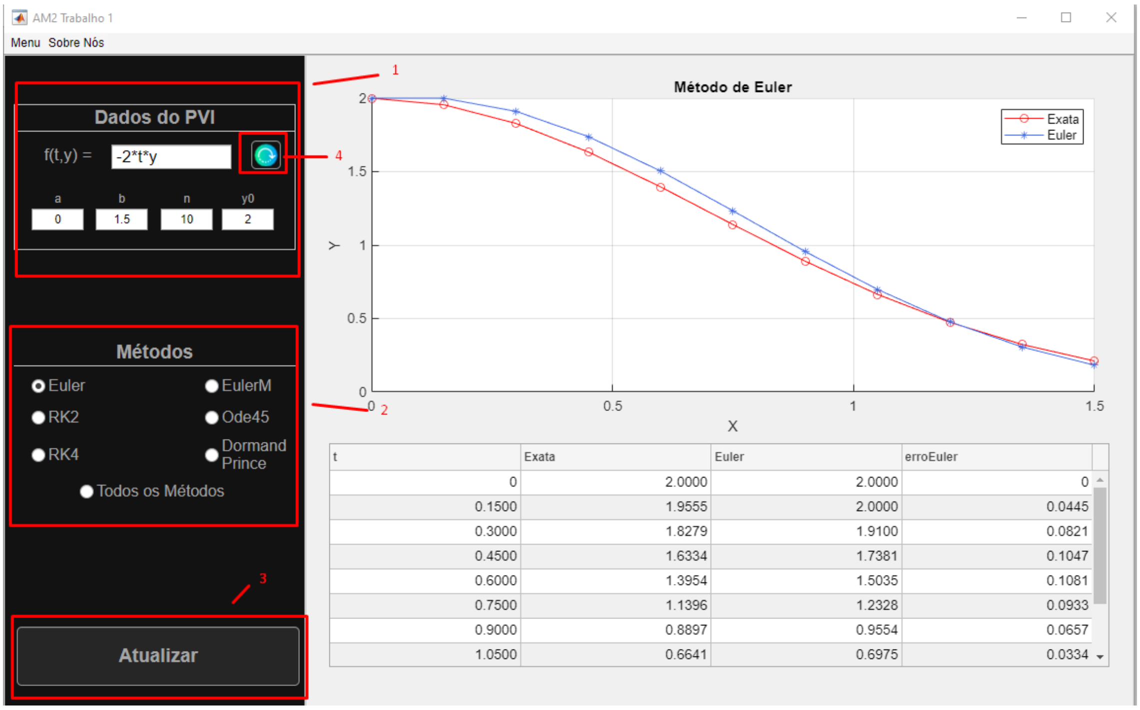
(e) Quais dos comandos seguintes em GeoGebra lhe permitiriam determinar a solução exata do PVI e a solução aproximada do mesmo.

(A)	`SolveODE[-2xy, (0,2)]`	**(B)**	`SolveODE[-2xy, (-1.5,0.2108)]`
(C)	`NSolveODE[{-2xy}, 0, {2}, 1.5]`	**(D)**	`NSolveODE[{-2xy}, -1.5, {0.2108}, 1.5]`



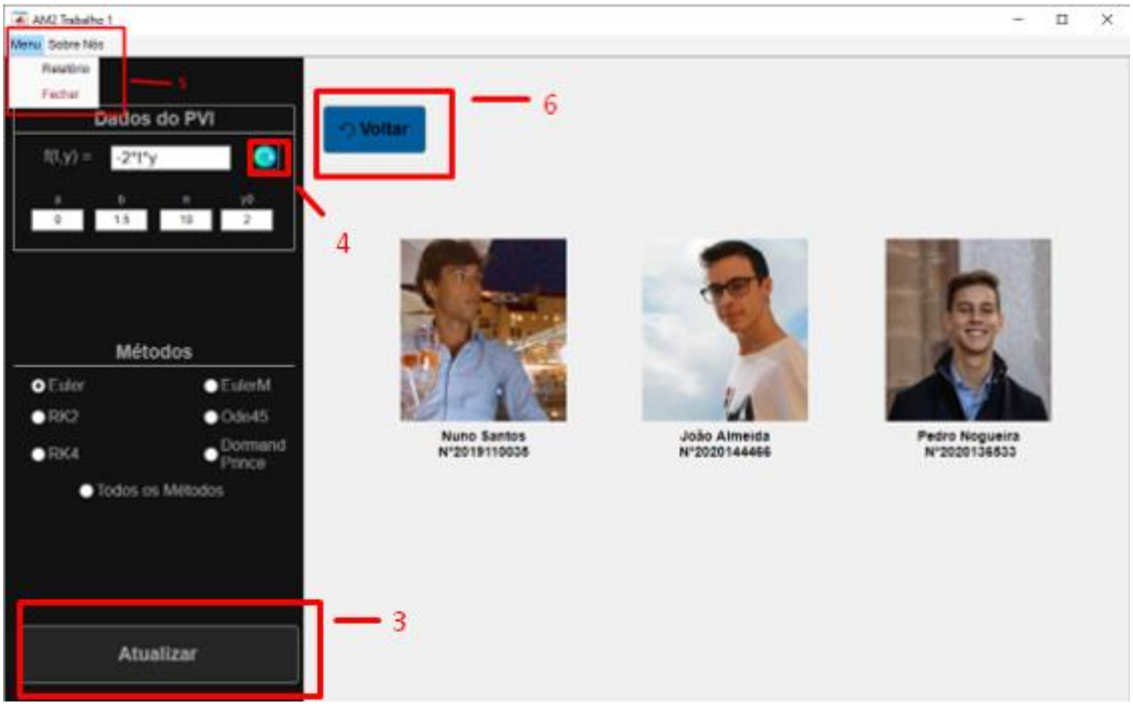
Alínea correta é a (A).

3.2.2. Exemplos de Output – App com Gráfico e Tabela



Ao iniciar, a aplicação automaticamente insere os valores para o problema do farol e calcula o método Euler como predefinido.

A qualquer altura é possível alterar os dados do PVI(1) e os seus métodos(2) de resolução, utilizando os campos próprios para tal. Ao clicar no botão atualizar(3), o gráfico e a tabela são alterados para corresponder ao esperado. Quando feitas alterações indesejadas nos dados do PVI está presente um botão de Reset(4) que introduz os valores do problema do farol nos campos(1).



Na barra superior é possível observar opções extras, no campo Menu(5) é possível abrir o relatório do projeto como também é possível encerrar a aplicação, no campo Sobre Nós é apresentada alguma informação como as fotografias, os nomes e os número dos elementos do grupo. Ao clicar nos botões voltar(6), Atualizar(3) ou no botão de Reset(4) a janela inicial com os gráficos e tabela voltará a ser apresentada.

3.3. Problemas de Aplicação do Livro

3.3.1. Modelação Matemática do Problema

Exercício 1 – O primeiro passo para a resolução deste exercício é olharmos para a informação que nos é dada por hipótese. Todo o exercício 1 se trata de um problema de valor inicial (PVI).

A informação que nos é dada no enunciado pode, então, ser representada por:

$$\begin{cases} m\frac{dv}{dt} = mg - kv^2, k > 0 \\ t \in [0, 5] \\ v(0) = 0 \end{cases}$$

O intervalo a que pertence a t é relativamente ambíguo, desde que inclua o 0 (por ser esse o valor inicial fornecido) e o 5 (por ser esse o valor que pretendemos calcular).

Este problema de valor inicial pode (e deve) ser simplificado, através da simplificação da própria equação diferencial:

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

Atentando à restante informação fornecida no enunciado, obtemos:

$$\begin{cases} \frac{dv}{dt} = g - \frac{kv^2}{m} \ , \ k > 0 \\ m = 5 \ slugs \\ g = 32 \ ft/s^2 \\ k = 0.125 \end{cases} \quad \Leftrightarrow \ \frac{dv}{dt} = 32 - \frac{0.125v^2}{5}$$
$$\Leftrightarrow \ \frac{dv}{dt} = 32 - 0.025v^2$$
$$\Leftrightarrow \ v' = 32 - 0.025v^2$$

Temos, então, o PVI, agora completo e simplificado:

$$\begin{cases} v' = 32 - 0.025v^2 \\ t \ \epsilon \ [0,5] \\ v(0) = 0 \end{cases}$$

Exercício 2 – O exercício 2 trata-se de um PVI com o do exercício 1, embora mais simplificado. A informação que nos é dada traduz-se em:

$$\begin{cases} A' = A(2.128 - 0.0432A) \\ t \ \epsilon \ [0,5] \\ A(0) = 0.24 \ cm^2 \end{cases}$$

Apesar de não nos darem diretamente um intervalo para t, deduzimos que é de 0 a 5, pois o problema vem de uma observação de uma área que é feita diariamente, podendo o seu valor inicial ser traduzido por A(0).

Para introduzirmos esta informação no nosso GUI, falta apenas o valor de n. Geralmente, usamos o **n** para calcular o valor de **h**, através da fórmula:

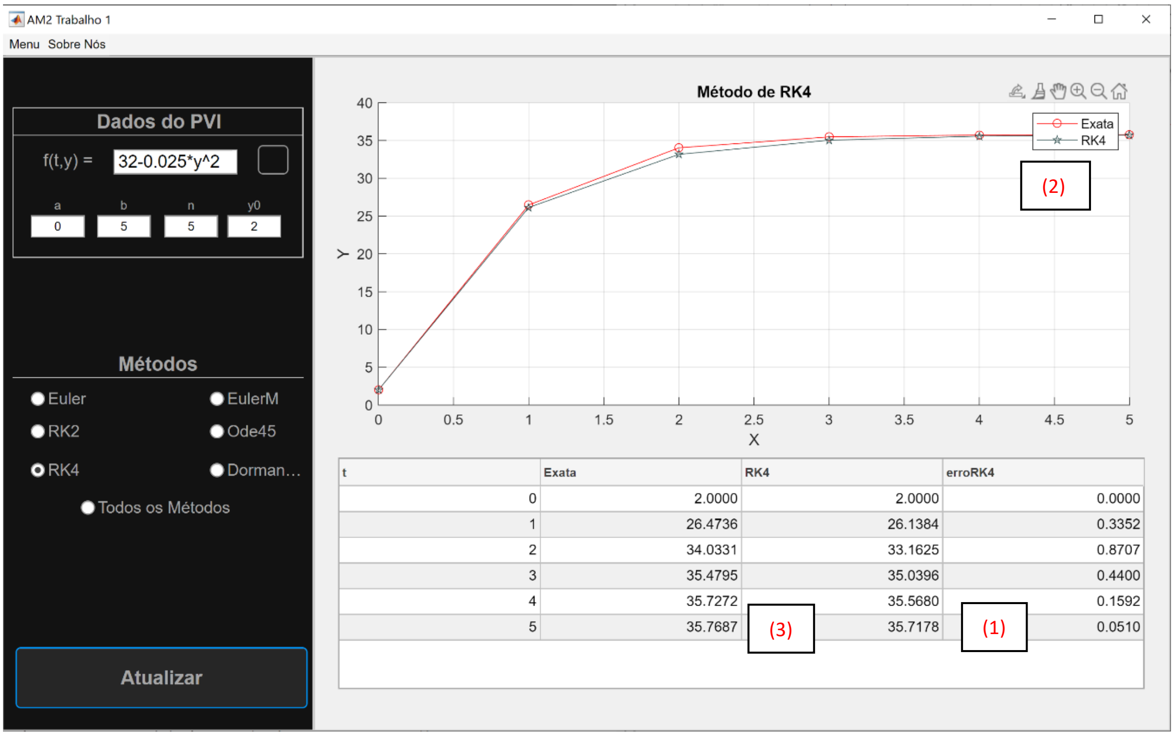
$$h = \frac{b-a}{n}$$

Como nos é dado o valor de **h** no enunciado, teremos que obter o valor do **n**:

$$h = \frac{b-a}{n} \Leftrightarrow 1 = \frac{5-0}{n} \Leftrightarrow n = 5$$

3.3.2. Resolução Através da App Desenvolvida

Exercício 1 – Introduzimos os dados no GUI e atualizamos o mesmo.



Onde:

(1) Resposta ao exercício (a): aproximação pelo método de Runge-Kutta da velocidade da massa em queda em t = 5s.

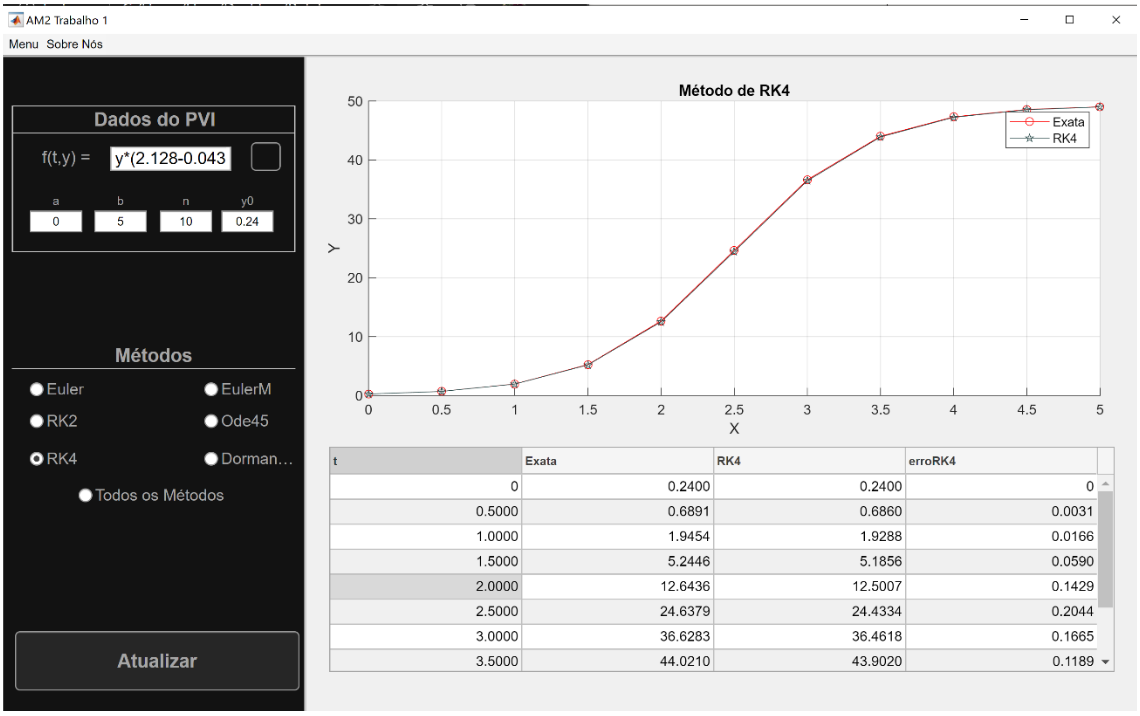
(2) Resposta ao exercício (b): Gráfico da solução do PVI.

(3) Resposta ao exercício (c): Valor real de v(5).

Exercício 2 – Só nos falta calcular o valor de n. Sabemos que h = 0.5, logo, pela mesma lógica do exercício 1:

$$h = \frac{b-a}{n} \Leftrightarrow 0.5 = \frac{5-0}{n} \Leftrightarrow n = 10$$

Introduzindo a informação no GUI…



Valores reais — Valores aproximados pelo método de Runge-Kutta

t	Exata	RK4	
0	0,24	0,24	-> A(0)
0,5	0,689133	0,686014	
1	1,945411	1,928774	-> A(1)
1,5	5,244573	5,185556	
2	12,64356	12,50068	-> A(2)
2,5	24,63789	24,43345	
3	36,6283	36,4618	-> A(3)
3,5	44,02097	43,90204	
4	47,31635	47,23494	-> A(4)
4,5	48,57104	48,52452	
5	49,01958	48,99649	-> A(5)

$t(days)$	1	2	3	4	5
$A(observed)$	2.78	13.53	36.30	47.50	49.40
$A(approximated)$	1.93	12.50	36.46	47.24	49.00
A(valor exato)	1.94	12.64	36.63	47.32	49.02

4. Conclusão

Concluímos, por fim, que os Métodos Numéricos para a resolução de Problemas de Valor Inicial são muito úteis, especialmente quando usados num contexto real e prático, pois originam aproximações com erro mínimo (dependendo do método usado).

Como regra geral, verificamos o esperado: quanto maior for o número de sub-intervalos n, menor é o erro de todos os Métodos.

Quanto à comparação entre os métodos, observamos que os métodos que verificam menor erro e, consequentemente, melhor aproximação ao valor exato, são o método de Runge-Kutta de ordem 4 e o método usando a função ODE45 nativa do MATLAB, que muitas vezes apresentaram erros na ordem apenas das milésimas ou menor. Em contrapartida, temos o método de Euler, cujo erro é especialmente grande comparado com todos os outros métodos implementados.

Com este trabalho, adquirimos várias técnicas, não só de MATLAB como também de pesquisa e compreensão de informação de forma independente.

Referências Bibliográficas

https://www.math.tecnico.ulisboa.pt/~calves/MC/cap5.html

Enviar-Ellison-TCC.pdf (unifap.br)

Capítulo 7 - Equações Diferenciais Ordinárias (ipb.pt)

Wolfram|Alpha: Computational Intelligence (wolframalpha.com)

Dormand–Prince method - Wikipedia

ode45berkley.pdf (auburn.edu)

http://depa.fquim.unam.mx/amyd/archivero/DormandPrince_19856.pdf