Análise Matemática II

Atividade 01 – Métodos Numéricos para EDO/PVI

Docente: Arménio Correia

Carlos Miguel Parreira Pais nº 2010017171 – LEI

José Fernando Esteves de Almeida nº 2019129077 – LEICE

Pedro Henrique de Sousa Rodrigues nº2019136525 – LEICE

Coimbra, 28 de Maio de 2020

Índice

1. Introdução

1.1 Enunciado da atividade proposta e interpretação do mesmo

Matlab .: Atividade01Trabalho » MNuméricos para EDO/PVI

Pretende-se com esta atividade que os alunos adquiram conhecimentos sobre os métodos numéricos para resolução de EDO/PVI, programem esses métodos e ao mesmo tempo que desenvolvam competências algorítmicas e de programação em Matlab.

Com base nos ficheiros existentes nas sub-pastas de » ficheiros de suporte à Atividade01 e/ou ficheiros implementados nas aulas práticas sobre este assunto » <u>**implementem os métodos numéricos:**</u>

- **Euler** (<u>versão beta com "alocação" de memória</u>)
- **Euler melhorado ou modificado;**
- **Runge-Kutta de ordem 2** (RK2);
- **Runge-Kutta de ordem 4** (RK4);
- Função que utilize o **ODE45** do Matlab;
- Pesquisa de outro método.

<u>**1ª Parte da atividade**</u>: [até 15 de abril]
Com base nos ficheiros da pasta de suporte a esta atividade e/ou outros implementados nas aulas, implementar e acrescentar os métodos anteriores e ajustar a interface de texto para esta atividade com validação dos parâmetros de entrada.

<u>**2ª Parte da atividade**</u>: [até 30 de abril]
Implementar interfaces gráficas GUI com o utilizador.
» Na sub-pasta at01_MN_PVI_GUI, encontram um esquiço para implementação de uma GUI com radio buttons e outros objetos que será construída nas próximas aulas Práticas a distância.

<u>**3ª Parte da atividade**</u>: [até 30 de abril]
Elaboração de um relatório que aborde pelo menos os seguintes pontos:

1. Introdução
1.1 Enunciado da actividade proposta e interpretação do mesmo
1.2 Definição de PVI

2. Métodos Numéricos para resolução de PVI
2.1 Método de Euler
2.1.1 Fórmulas
2.1.2 Algoritmo/Função
2.2 Método de Euler Melhorado ou Modificado
2.2.1 Fórmulas
2.1.2 Algoritmo/Função
2.3 Método de RK2
2.3.1 Fórmulas
2.3.2 Algoritmo/Função
2.4 Método de RK4
2.4.1 Fórmulas
2.4.2 Algoritmo/Função
2.5 Função ODE45 do Matlab

3. Exemplos de aplicação e teste dos métodos
3.1 Exercício 4 do um teste A de 2015/2016
3.1.1 PVI - Equação Diferencial de 1ª ordem e Condições Iniciais
3.2.2 Exemplos de output - GUI com gráfico e tabela
3.2 Problema de aplicação » https://moodle.isec.pt/moodle/mod/page/view.php?id=125256
3.2.1 Modelação matemática do problema
3.2.2 Resolução através da aplicação criada

4. Conclusão

Este trabalho surge do âmbito da unidade curricular de Análise Matemática 2 do curso de Engenharia Informática do Instituto Superior de Engenharia de Coimbra.

O seu foco consiste no estudo de Métodos Numéricos para a resolução de Equações Diferenciais Ordinárias (EDOs) e de Problemas de Valor inicial (PVIs), e na implementação desses Métodos através do desenvolvimento de uma aplicação em linguagem de programação MATLAB.

Tem, também, como objetivo principal, o fomentar da investigação, da aquisição de conhecimentos, do desenvolvimento de competências algorítmicas, da aprendizagem da programação em MATLAB e da experimentação da criação de Interface Gráfica do Utilizador (GUI's).

Além disso, é pretendida a comparação dos resultados obtidos nos diversos Métodos Numéricos através da realização de vários exercícios práticos (utilizando a aplicação anteriormente citada).

O trabalho está dividido da seguinte forma:

- 1ª Parte – Definição de Problema de Valor Inicial (PVI);
- 2ª Parte - Descrição de vários Métodos Numéricos e seus respetivos algoritmos e funções;
- 3ª Parte – Exemplos de aplicação e teste dos Métodos Numéricos.

1.2 Definição de P.V.I.

Um P.V.I. (Problema de Valor Inicial) trata-se de uma equação diferencial que é acompanhada do valor da função num determinado ponto, chamado de valor inicial ou condição inicial. Muitas vezes é associado a problemas reais, com aplicação em muitas áreas científicas, sendo que geralmente a equação diferencial dada é uma equação evolutiva que descreve como o sistema irá evoluir ao longo do tempo, caso as condições iniciais se verifiquem.

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

$$\begin{cases} y' = f(t, y) & (1) \\ t \in [a, b] & (2) \\ y(a) = y_0 & (3) \end{cases}$$

Onde:

(1) é a Equação Diferencial
(2) é o Intervalo Pretendido
(3) é a Condição Inicial (valor inicial)

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

2. Métodos Numéricos para resolução de PVI

2.0.1 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 subintervalos no intervalo original [a, b], e pode ser calculado da seguinte forma:

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

onde:

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

Importante: Quanto menor o valor de ***h***, mais subintervalos irão existir dentro de um dado intervalo e melhor será a aproximação ao valor real.

2.1 Método de Euler

2.1.1 Fórmulas

O método de Euler é um procedimento numérico de primeira ordem *(y')* para aproximar a solução da equação diferencial $\boxed{y' = f(t,y)}$ que satisfaz a condição inicial: $\boxed{y(t_0) = y_0}$.

O Método de Euler para resolver um PVI é dado pelas seguintes equações:

Fórmula Geral:

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

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 subintervalo (passo);
- $f(t_i, y_i)$ → Valor da equação em t_i e y_i;

2.1.2 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*** de 1 a ***n***, fazemos o cálculo do método de Euler para a iésima iteração.

Função (MATLAB):

```
function y = N_Euler(f,a,b,n,y0)
    h = (b-a)/n;                        % Valor de cada subintervalo (passo)

    t = a:h:b;                          % Alocação de memória
    y = zeros(1, n+1);                  % Alocação de memória

    y(1) = y0;                          % O primeiro valor de y é sempre y0

    for i=1:n                           % O número de iterações vai ser igual a n
        y(i+1)=y(i)+h*f(t(i),y(i));     % Aproximação do método de Euler para a iésima iteração
    end
end
```

2.2 Método de Euler Melhorado ou Modificado (Método de Heun)

2.2.1 Fórmulas

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

O Método de Heun para resolver um PVI é dado pelas seguintes equações:

Fórmula Geral:

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

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 subintervalo (passo);

Cálculo de k1:

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

- k_1 → Inclinação no início do intervalo
- $f(t_i, y_i)$ → Valor da equação em t_i e y_i;

Cálculo de k2:

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

- k_2 → Inclinação no fim do intervalo;
- t_i → Valor da abcissa atual;
- h → Tamanho de cada subintervalo (passo);
- y_i → Valor aproximado da solução do problema original na abcissa atual;
- k_1 → Inclinação no início do intervalo.

2.1.2 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 iésima iteração.

Função (MATLAB):

```
function y = N_Heun(f,a,b,n,y0)
    h = (b-a)/n;                    % Valor de cada subintervalo (passo)

    t = a:h:b;                      % Alocação de memória
    y = zeros(1, n+1);              % Alocação de memória

    y(1) = y0;                      % O primeiro valor de y é sempre y0

    for i=1:n                       % O número de iterações vai ser igual a n
        k1 = f(t(i),y(i));          % Inclinação no início do intervalo
        k2 = f(t(i+1), y(i) + k1*h); % Inclinação no fim do intervalo
        k = 0.5*(k1+k2);            % Cálculo da média das inclinações
        y(i+1)=y(i)+h*k;            % Aproximação do método de Heun para a iésima iteração
    end
end
```

2.3 Método de Runge-Kutta de Ordem 2

2.3.1 Fórmulas

É um método de passo simples que requer apenas derivadas de primeira ordem e pode fornecer aproximações precisas.

O Método de RK2 para resolver um PVI é dado pelas seguintes equações:

Fórmula Geral:

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

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;

Cálculo de k1:

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

- k_1 → Inclinação no início do intervalo
- h → Valor de cada subintervalo (passo);
- $f(t_i, y_i)$ → Valor da equação em x_i e y_i;

Cálculo de k2:

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

- k_2 → Inclinação no fim do intervalo;
- t_i → Valor da abcissa atual;
- h → Tamanho de cada subintervalo (passo);
- y_i → Valor aproximado da solução do problema original na abcissa atual;
- k_1 → 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 método de RK2 para a iésima iteração.

Função (MATLAB):

```
function y = N_RK2(f,a,b,n,y0)
    h = (b-a)/n;                        % Valor de cada subintervalo (passo)

    t = a:h:b;                          % Alocação de memória
    y = zeros(1, n+1);                  % Alocação de memória

    y(1) = y0;                          % O primeiro valor de y é sempre y0

    for i=1:n                           % O número de iterações vai ser igual a n
        k1 = h * f(t(i), y(i));         % Inclinação no início do intervalo
        k2 = h * f(t(i + 1), y(i) + k1);  % Inclinação no fim do intervalo

        y(i + 1)=y(i) + (k1 + k2)/2;    % Aproximação do método de RK2 para a iésima iteração
    end
end
```

2.4 Método de Runge-Kutta de Ordem 4

2.4.1 Fórmulas

O método de Runge-Kutta de ordem 4 não necessita do cálculo de qualquer derivada de f, mas depende de outra função que é definida avaliando f em diferentes pontos.

O método RK4 para resolver um PVI é dado pelas seguintes equações:

Fórmula Geral:

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

onde:

- y_{i+1} → Aproximação pelo método RK4 de $y(x_{n+1})$;
- y_i → Valor de y na iésima iteração;
- h → Valor de cada subintervalo (passo).

e também:

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

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.

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

Média ponderada das inclinações:

2.4.2 Algoritmo/Função

Algoritmo:

1. Definir o passo *h*;
2. Criar um vetor y para guardar a solução e atribuir *y(1) = y0*;
3. Atribuir o primeiro valor de *y*;
4. Cálculo da 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 iésima iteração.

Função (MATLAB):

```
function y = N_RK4(f,a,b,n,y0)
    h = (b-a)/n;                                    % Valor de cada subintervalo (passo)

    t = a:h:b;                                      % Alocação de memória
    y = zeros(1, n+1);                              % Alocação de memória

    y(1) = y0;                                      % O primeiro valor de y é sempre y0

    for i=1:n                                       % O número de iterações vai ser igual a n
        k1 = h*f(t(i), y(i));                       % Inclinação no início do intervalo
        k2 = h*f(t(i) + h/2, y(i) + 0.5*k1);        % Inclinação no ponto médio do intervalo
        k3 = h*f(t(i) + h/2, y(i) + 0.5*k2);        % Inclinação (novamente) no ponto médio do intervalo
        k4 = h*f(t(i+1), y(i) + k3);                % Inclinação no final do intervalo

        y(i + 1) = y(i) + (k1 + 2*k2 + 2*k3 + k4)/6;    % Aproximação do método de RK4 para a iésima iteração
    end
end
```

2.5 Função ODE45 do Matlab

2.5.1 Fórmulas

A função ODE45 é uma das funções nativas do MATLAB, e é baseada num método de Runge-Kutta. Para resolver um PVI com uma EDO de ordem 2, pode ser chamada da seguinte forma:

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

Onde:

- t → Vetor das abcissas;
- f → Equação diferencial em ***t*** e em ***y***;
- y_0 → Valor inicial do PVI (condição inicial);

2.5.2 Algoritmo/Função

Algoritmo:

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

Função (MATLAB):

```
function y = N_ODE45(f,a,b,n,y0)
    h = (b-a)/n;                    % Valor de cada subintervalo (passo)

    t = a:h:b;                      % Alocação de memória

    [~,y] = ode45(f, t, y0);        % Aproximação através da função ODE45 e colocação dos valores no vetor y
    y = y';                         % Mudança da orientação do vetor
end
```

2.6 Método do Ponto Médio (Midpoint Method)

2.6.1 Fórmulas

O Método do Ponto-Médio é um método numérico para resolver Equações Diferenciais Ordinárias (ODE). É também conhecido como Método de Euler modificado.

O método do Ponto-Médio para resolver um PVI é dado pelas seguintes equações:

Fórmulas Gerais:

PM Explícito: $$\boxed{y_{i+1} = y_i + h * f(t_i + \frac{h}{2}, y_i + h * k1)}$$

PM Implícito: $$\boxed{y_{i+1} = y_i + h * f(t_i + \frac{h}{2}, \frac{1}{2}(y_i + y_{i+1}))}$$

onde:

- y_{t+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 subintervalo (passo);
- t_i → Valor de t na inésima iteração.

2.6.2 Algoritmo/Função

Algoritmo:

1. Definir o passo *h*;
2. Cálculo do método do Ponto-Médio <u>Explícito</u> para a iésima iteração;
3. Cálculo do método do Ponto-Médio <u>Implícito</u> para a iésima iteração;

Função (MATLAB):

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

    h = (b-a)/n;                                        % Valor de cada subintervalo (passo)

    t = a:h:b;                                          % Alocação de memória
    y = zeros(1,n+1);                                   % Alocação de memória

    y(1) = y0;

    for i=1:n                                           % O número de iterações vai ser igual a n
        k1 = 0.5 * f(t(i), y(i));                       % variavel auxiliar
        y(i+1) = y(i) + h*f(t(i) + h/2, y(i) + h*k1);   % ponto médio explícito
        y(i+1) = y(i) + h*f(t(i) + h/2, 0.5*(y(i) + y(i+1)));  % ponto médio implícito
    end
```

3. Exemplos de aplicação e teste dos métodos

3.1. Exercício 4 do Teste A de 2015/2016

3.1.1 PVI – Equação Diferencial de 1ª ordem e Condições Iniciais: Exercícios

Resolução do Exercício 4 do Teste A de 2015/2016

4.

[0.75] **(a)** Qual é o valor lógico da seguinte afirmação? Justifique analiticamente e graficamente a sua resposta.

A equação diferencial, de menor ordem possível, que possui a família de curvas $y = c \times \exp(-x^2)$ como integral geral é dada por $y' = -2xy$ cujo campo direcional é dado pela figura 6 e o gráfico da solução geral pela figura 7.

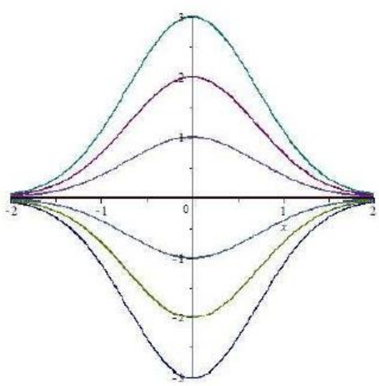


Figura 5

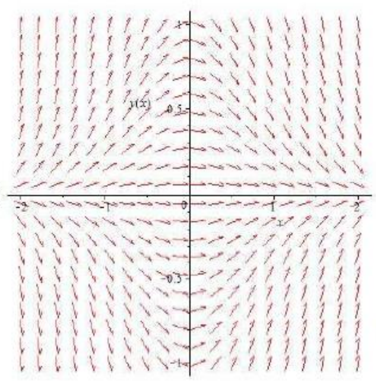


Figura 6

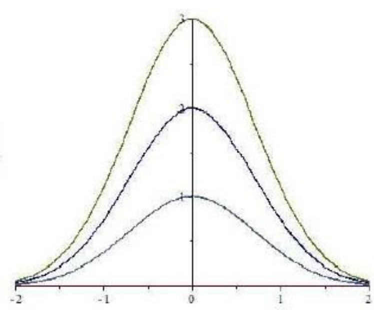


Figura 7

Alínea a)

<u>Para analisar esta afirmação, temos de a dividir e analisar em 3 partes:</u>

1. A equação diferencial, de menor ordem possível, que possui a família de curvas $y = c * \exp(-x^2)$ como integral geral é dada por $y' = -2xy$.
2. O campo direcional de $y' = -2xy$ é dado pela figura 6.
3. O gráfico da solução geral é dado pela figura 7.

Se as 3 afirmações forem verdadeiras, temos então que a afirmação do enunciado é verdadeira.

1. <u>Comecemos por calcular o integral geral de $y' = -2xy$.</u>

Para isso, precisamos de saber se a equação se trata de uma equação diferencial de variáveis separáveis. Temos:

$$y' = \frac{dy}{dx} \quad \Leftrightarrow -2xy = \frac{dy}{dx}$$

$$\Leftrightarrow (-2xy)dx - dy = 0$$

Equação Diferencial de Variáveis Separáveis

$$\Leftrightarrow y(-2x)dx - (1)dy = 0$$

Equação Diferencial de Variáveis Separadas

$$\Leftrightarrow (-2x)dx - (\frac{1}{y})dy = 0$$

A solução passa agora por calcular o integral da expressão obtida.

$$\int(-2x)dx - \int(\frac{1}{y})dy = \int(0)dx \ \Leftrightarrow -2\frac{x^2}{2} - \ln|y| = c, \ c \in \mathbb{R}$$

$$\Leftrightarrow \ln|y| = -x^2 - c, \ c \in \mathbb{R}$$

$$\Leftrightarrow |y| = e^{-x^2-c}, \ c \in \mathbb{R}$$

É também uma constante

$$\Leftrightarrow |y| = \boxed{\frac{1}{e^c}} e^{-x^2}, \ c \in \mathbb{R}$$

$$\Leftrightarrow |y| = c_2 e^{-x^2}, \ c_2 \in \mathbb{R}$$

Constantes

$$\Leftrightarrow y = \boxed{c_2} e^{-x^2} \vee y = \boxed{-c_2} e^{-x^2}, \ c_2 \in \mathbb{R}$$

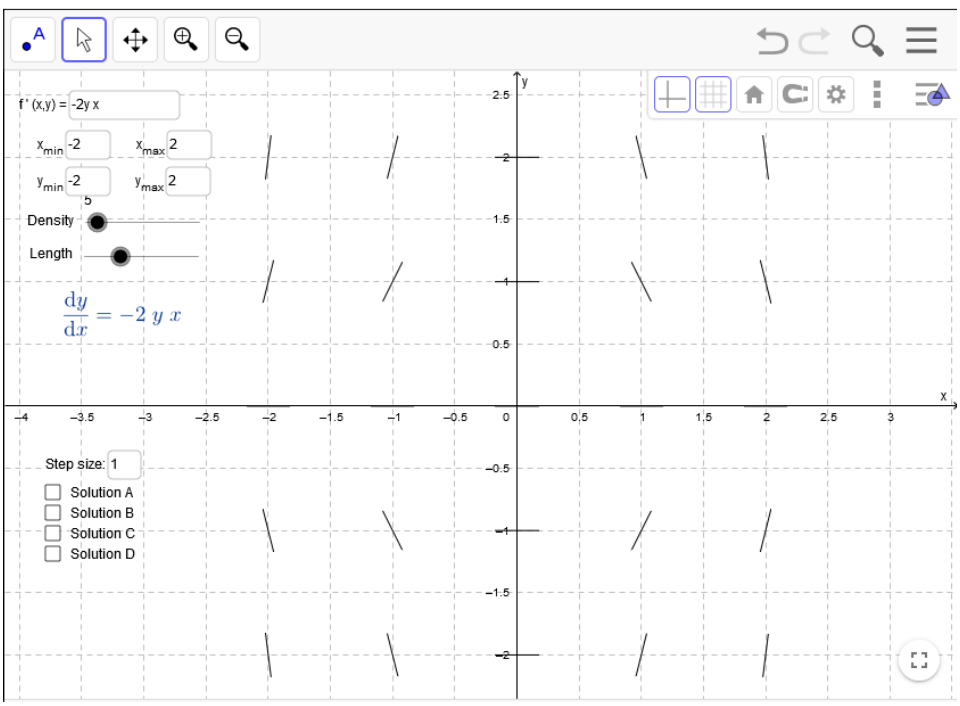
$$\Leftrightarrow y = c * e^{-x^2}, \ c \in \mathbb{R}$$

Assim, provamos que a parte 1. da afirmação está correta.

2. O campo direcional mostra-nos, através do gráfico, a inclinação ou declive da reta tangente em cada um dos pontos do gráfico da equação diferencial. Para obtermos o campo direcional de $y' = -2xy$, podemos ir dando valores a x e y e representá-los num gráfico. Vamos obter, por exemplo, os seguintes valores:

y \ x	-2	-1	0	1	2
-2	$y' = -8$	$y' = -4$	$y' = 0$	$y' = 4$	$y' = 8$
-1	$y' = -4$	$y' = -2$	$y' = 0$	$y' = 2$	$y' = 4$
0	$y' = 0$	$y' = 0$	$y' = 0$	$y' = 0$	$y' = 0$
1	$y' = 4$	$y' = 2$	$y' = 0$	$y' = -2$	$y' = -4$
2	$y' = 8$	$y' = 4$	$y' = 0$	$y' = -4$	$y' = -8$

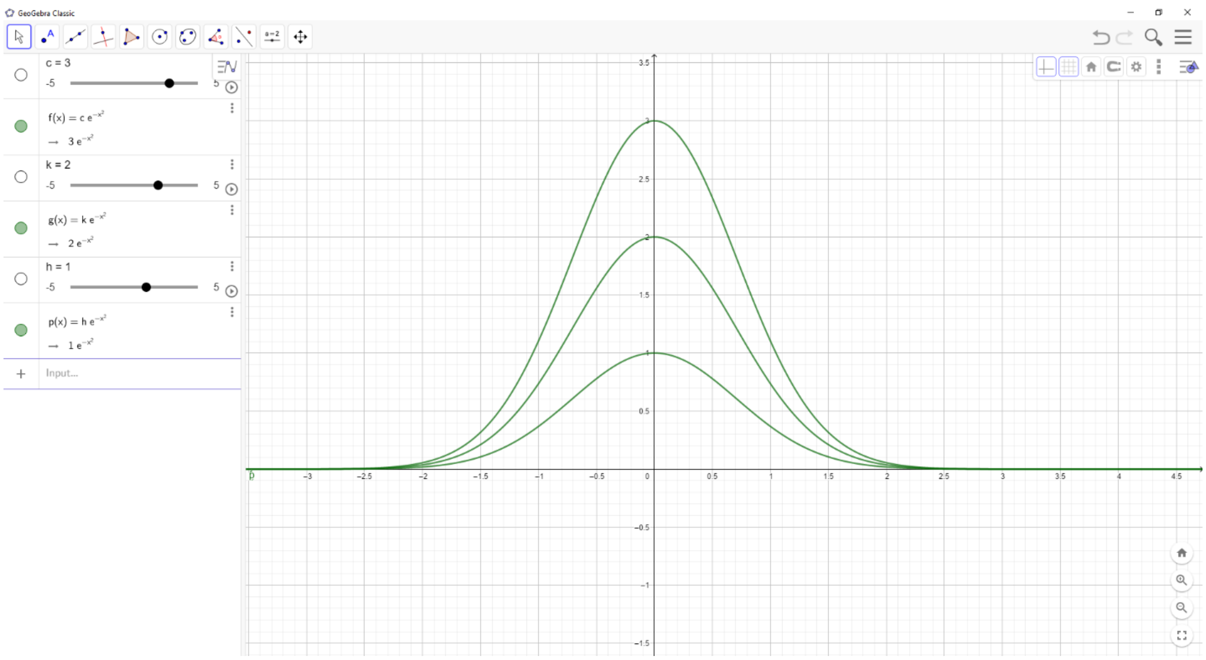
Podemos representar estes resultados num referencial:



(Feito com uma ferramenta do GEOGebra)

Se continuássemos a adicionar pontos, ficaríamos com um gráfico equivalente ao da figura 6. Logo, a parte 2. da afirmação está também correta.

3. Por fim, temos de obter o gráfico da solução geral e verificar se é equivalente ao da figura 7. Para isto, podemos também usar o GEOGebra.



Como podemos ver, dependendo do valor de ***c***, o gráfico adapta-se exatamente ao da figura 7. Logo, a parte 3. da afirmação é também correta.

Assim, depois de confirmar tudo isto, concluímos que a afirmação do enunciado tem o valor lógico de verdade.

[0.25] **(b)** Verifique que $y(t) = 3\exp(-t^2)$ é a solução exata do problema de valor inicial seguinte

$$y' + 2ty = 0, \quad y(0) = 3, \quad t \in [0, 2]$$

Alínea b)

Podemos identificar o seguinte PVI:

$$\begin{cases} y' = -2ty \\ t \in [0,2] \\ y(0) = 3 \end{cases}$$

Vamos resolver este PVI analiticamente.

1º Passo: Determinação da solução ou integral da E.D. Este passo é exatamente igual ao passo em que calculamos o integral da E.D. da alínea anterior. Portanto, sabemos que a solução é $y = c * e^{(-t^2)}$.

2º Passo: Calcular o valor de ***c***, através da condição inicial $y(0) = 3$.

$$\begin{cases} y(t) = c * e^{(-t^2)} \\ y(0) = 3 \end{cases} \Leftrightarrow c * e^{(-0^2)} = 3$$

$$\Leftrightarrow c = 3$$

3º Passo: Substituir o valor de ***c*** na solução geral $y = c * e^{(-t^2)}$.

Ficamos, então, com $y = 3 * e^{(-t^2)}$, como pretendíamos demonstrar.

Alínea c)

[2.0] **(c)** Relativamente ao PVI da alínea anterior, complete a tabela seguinte e interprete os resultados obtidos.

			Aproximações			Erros		
i	t_i	$y(t_i)$ Exata	y_i Euler	y_i RK2	y_i RK4	$\lvert y(t_i)-y_i\rvert$ Euler	$\lvert y(t_i)-y_i\rvert$ RK2	$\lvert y(t_i)-y_i\rvert$ RK4
0	0	3				0	0	0
1				2.25	2.3359		0.0864	0.0005
2	1	1.1036		1.125	1.1041			
3					0.3350		0.2463	0.0188
4	2	0.0549		0.4219				0.0358

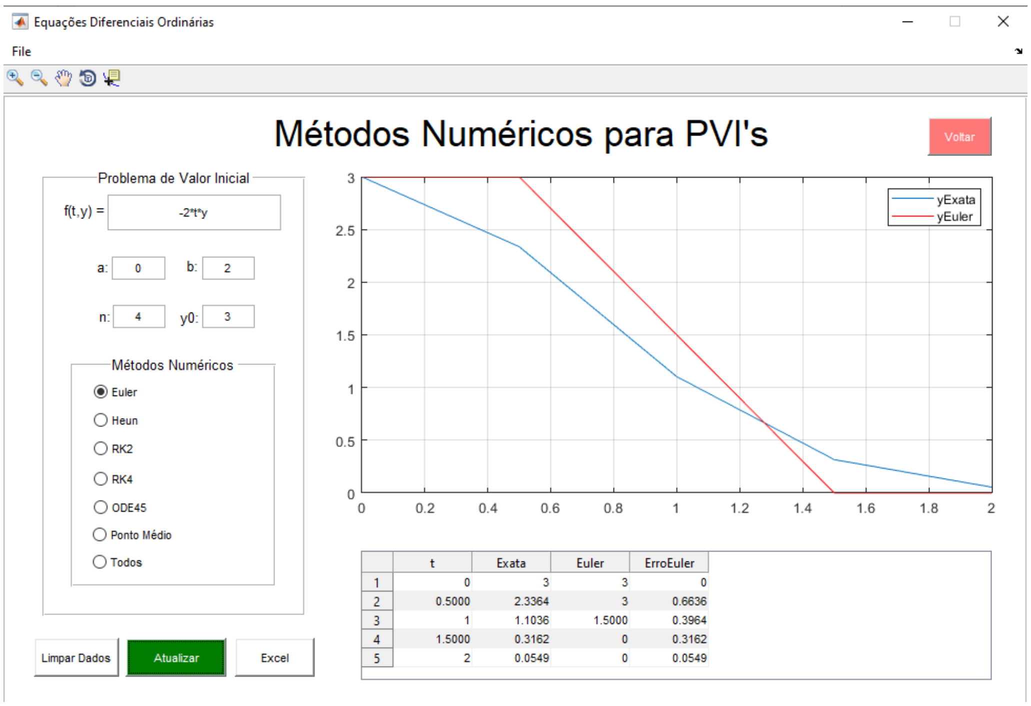
Podemos, nesta alínea, aplicar o nosso GUI para obter os valores em falta da tabela. A tabela indica que precisamos de um valor ***h*** = 0.5. Obtemos, então, o valor de ***n***:

$$h = \frac{b-a}{n} \Leftrightarrow 0.5 = \frac{2-0}{n} \Leftrightarrow n = 4$$

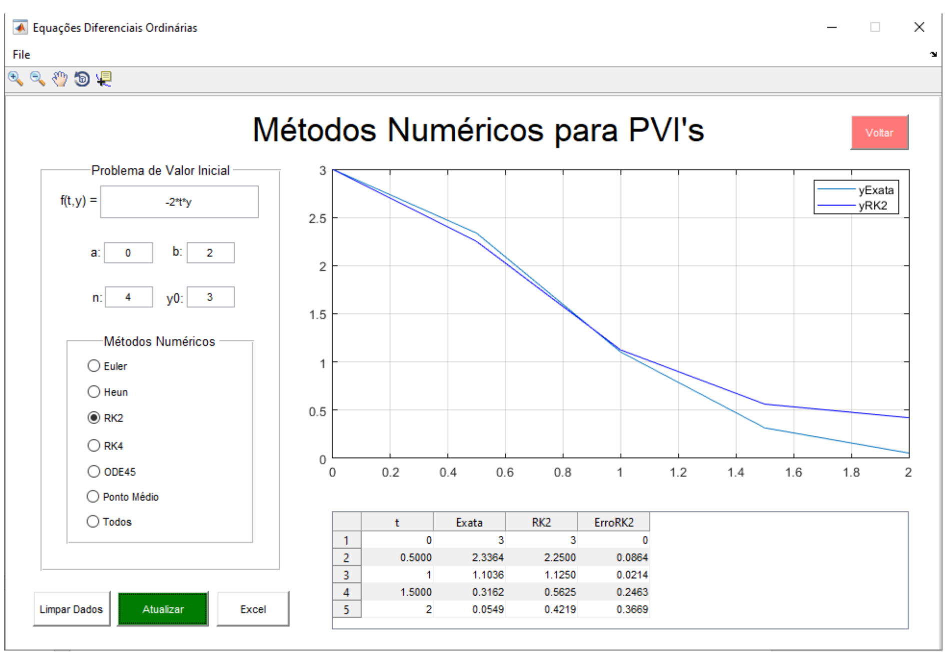
3.1.2 Exemplos de GUI com gráfico e tabela

Introduzindo os dados no GUI

Euler:



Runge-Kutta de ordem 2:



Runge-Kutta de ordem 4:

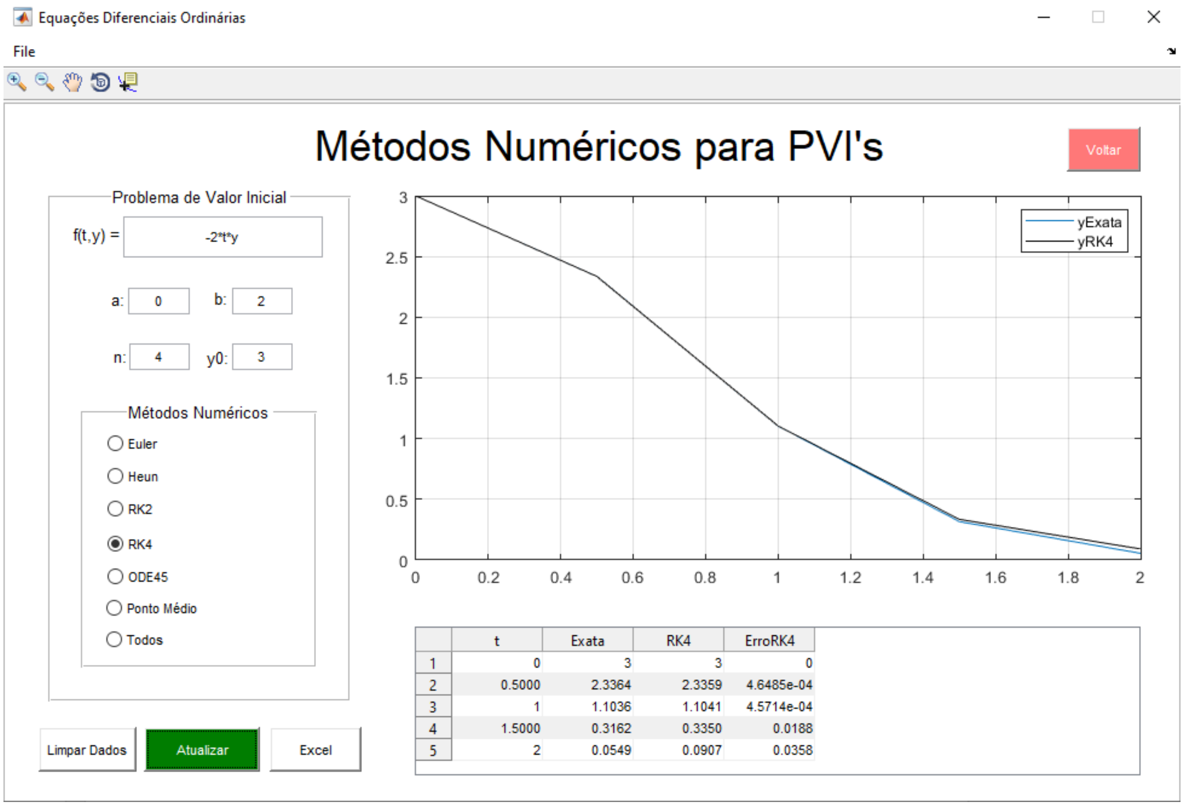


Tabela preenchida:

			Aproximações			Erros		
i	t_i	$y(t_i)$ Exata	y_i Euler	y_i RK2	y_i RK4	$\lvert y(t_i) - y_i\rvert$ Euler	$\lvert y(t_i) - y_i\rvert$ RK2	$\lvert y(t_i) - y_i\rvert$ RK4
0	0	3	3	3	3	0	0	0
1	0.5	2.3364	3	2.25	2.3359	0.6636	0.0864	0.0005
2	1	1.1036	1.5	1.125	1.1041	0.3964	0.0214	0.0005
3	1.5	0.3162	0	0.5625	0.3350	0.3162	0.2463	0.0188
4	2	0.0549	0	0.4219	0.0907	0.0549	0.3669	0.0358

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

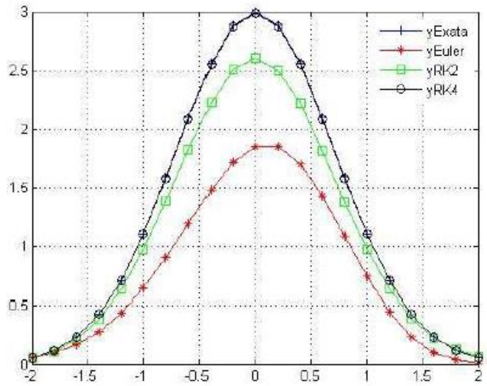


Figura 8

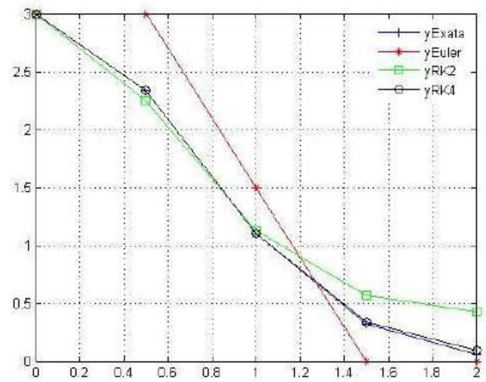


Figura 9

Alínea d)

Como podemos ver pelos gráficos obtidos no GUI na alínea anterior, a figura 9 é a que representa corretamente uma solução do PVI dado.

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

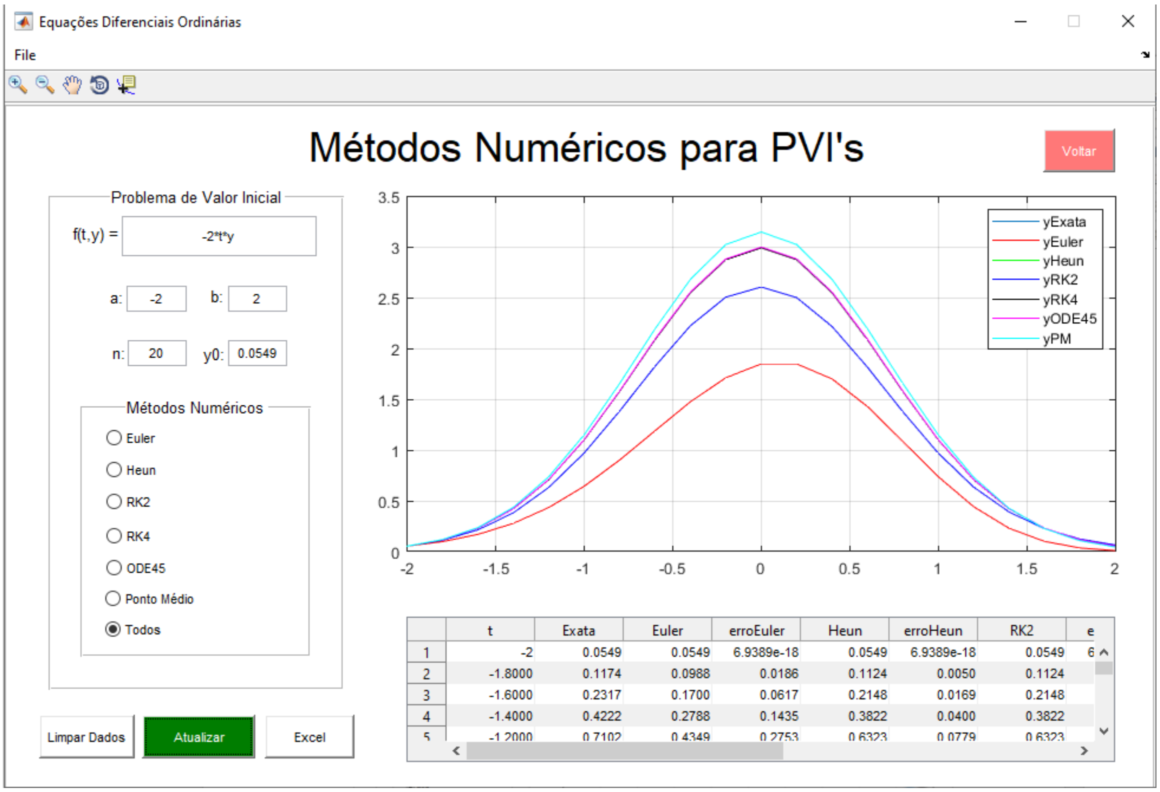
Alínea e)

O gráfico da figura 8 vem da mesma Equação Diferencial, mas no seu PVI tem um intervalo de valores ***t*** diferente (que podemos considerar [-2 , 2]), assim como o número de subintervalos (que podemos considerar 20). A condição inicial é um valor próximo de zero, vamos considerar então $y(-2) = y(2) = 0.0549$ (aproximação obtida a partir da tabela da alínea c)).

Obtemos, assim, o PVI:

$$\begin{cases} y' = -2ty \\ t \in [-2, 2] \\ y(0) = 0.0549 \end{cases}$$

Podemos, para confirmar, introduzir os dados no GUI. Resultado:



Que coincide com a figura 8.

[1.25] **(f)** Complete as funções e acrescente comentários para explicar o algoritmo/regras que lhes estão associadas.

Alínea f)

```
function y = NEuler(f,a,b,n,y0)
h=(b-a)/n; % Tamanho de um sub-intervalo
t=a:h:b;    % O vetor dos valores t vai de a até b com passo h
y=zeros(1,n+1); % Alocação de memória para acelerar o algoritmo
y(1)=y0;    % Condição inicial do PVI ocupa sempre o primeiro valor do vetor das soluções
for i=1:n
y(i+1)=y(i)+h*f(t(i),y(i)); %Fórmula do Método de Euler: popular o vetor das soluções. Cada solução é obtida à custa da anterior.
end
```

```
function y = NRK2(f,a,b,n,y0)
h=(b-a)/n; % Tamanho de um sub-intervalo
t= a:h:b;   % O vetor dos valores t vai de a até b com passo h
y= zeros(1,n+1); % Alocação de memória para acelerar o algoritmo
y(1)=y0;    % Condição inicial do PVI ocupa sempre o primeiro valor do vetor das soluções
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
```

[1.25] (g) A *script* seguinte traduz corretamente a resolução em MATLAB do PVI dado? Justifique a sua resposta, corrigindo se for esse o caso os erros existentes.

```
clear;
clc;

strF = '2/t*y'
f     = @(t,y) vectorize(eval(strF));
a   = 2;
b   = 3;
n   = 3;
y0 = 1;

yEuler = NEuler(f,a,b,n,y0);
yRK2    = NRK2(f,a,b,n,y0);
yRK4    = NRK4(f,a,b,n,y0);

t = b:-(b-a)/n:a;
sExata = dsolve(['Dy=',strF],['y(',a,')=',num2str(0)]);
yExata = vectorize(eval(char(sExacta)));

plot(t,yExata,'-kd')
hold on
plot(t,yEuler,'-bo')
plot(t,yRK2,'-g*')
plot(t,yRK4,'-r+')
grid on
legend('RK4','RK2','Euler','Exata')
hold off
erroEuler = abs(yRK4-yEuler);
erroRK2    = abs(yRK4-yRK2);
erroRK4    = abs(yExata-yRK4);
tabela     = [t.',yExata.',yEuler.',yRK2.',yRK4.',…
              erroEuler.',erroRK2.',erroRk4.'];
disp(tabela);
```

Alínea g)

O algoritmo apresentado é muito parecido com o algoritmo para a resolução do PVI dado, embora com alguns erros:

- O vetor $\boldsymbol{t}$ é definido ao contrário, o que irá causar problemas com o resto do algoritmo. O correto seria $t = a:(b-a)/n:b$;

- O valor inicial não é usado. Para corrigir, substituiríamos o 0 em num2str(0) por y0, para ficarmos com num2str(y0).

- Erro de nome de variável: char(sExacta) deveria ser char(sExata), pois foi sExata que foi definida mais cedo e não a outra.

- Legenda do gráfico foi colocada ao contrário. o correto seria legend('Exata','Euler','RK2,'RK4');

- Valores do erro do Euler e RK2 mal calculados. O correto seria, respetivamente,

erroEuler = abs(yExata-yEuler); e erroRK2 = abs(yExata-yRK2);

- Na definição da tabela, está escrito erroRk4 em vez de erroRK4 (o MATLab é case-sensitive).

- No fim do algoritmo deveríamos incluir a instrução shg; para também mostrar o gráfico.

3.2.Problema de aplicação

3.2.1 Modelação matemática do problema

Atividade 02 – Problema de Aplicação

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:

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

O intervalo a que pertence ***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:

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

$$\Leftrightarrow \ \frac{dv}{dt} = 32 - 0.025v^2$$

$$\Leftrightarrow \ v' = 32 - 0.025v^2$$

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

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

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:

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

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:

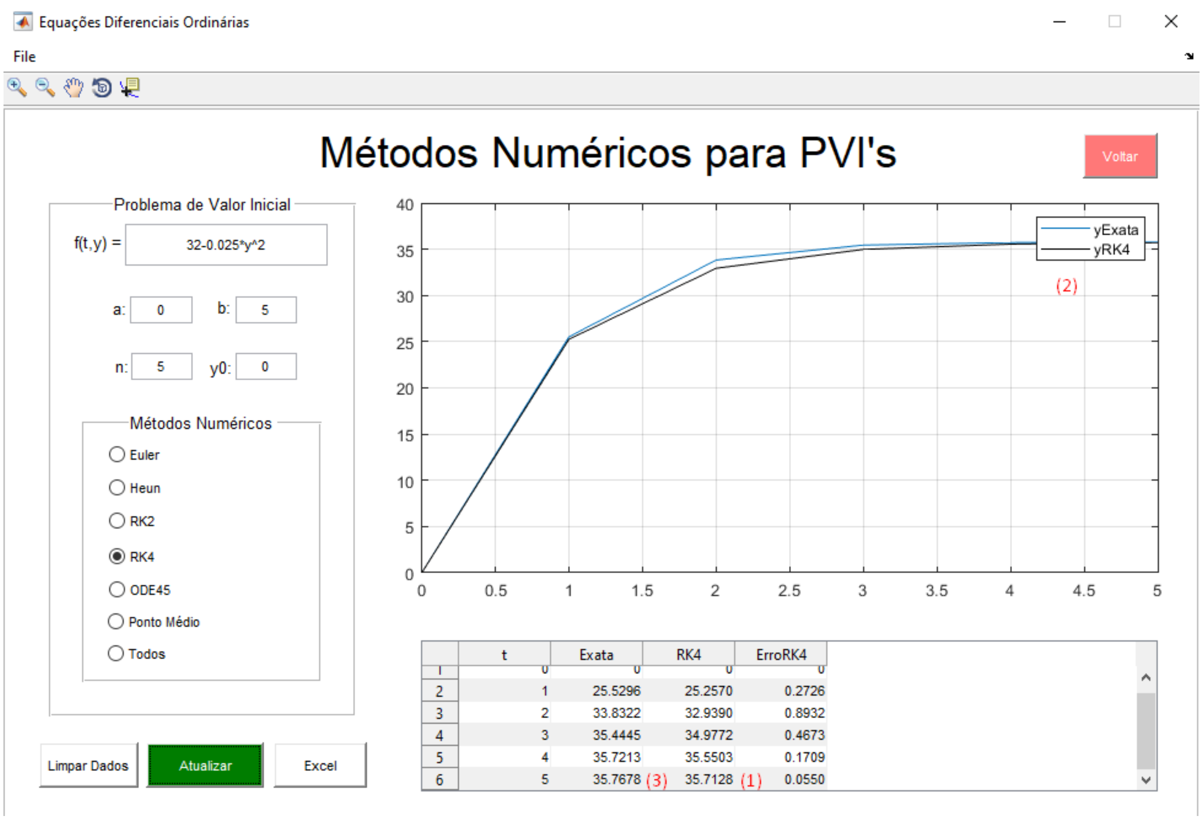
$$h = \frac{b-a}{n}$$

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

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

3.2.1 Resolução através da aplicação criada

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



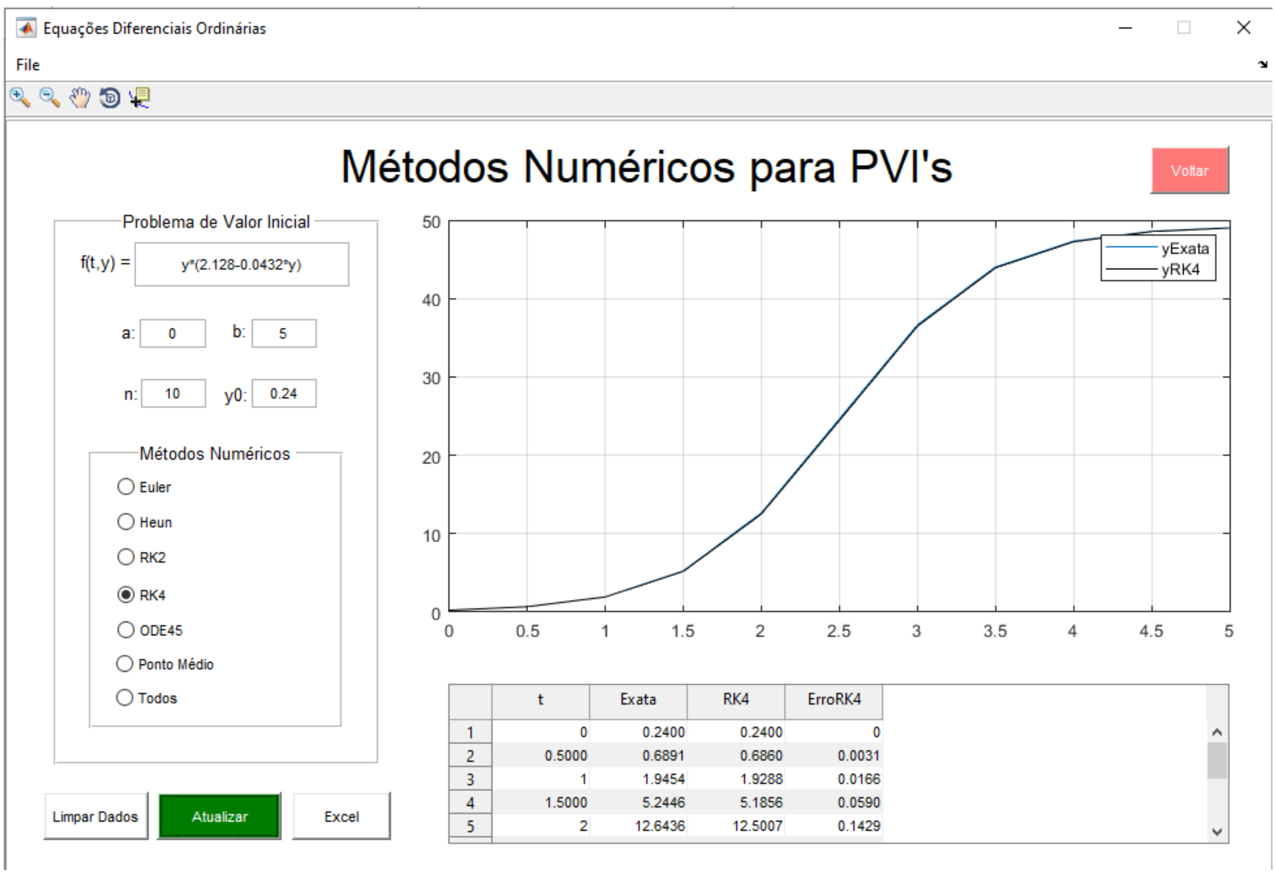
Onde:

(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…



Podemos fazer uso do botão para gerar e abrir o ficheiro Excel com os resultados em forma de tabela, se decidirmos que a tabela não é muito visível no GUI em si.

Valores reais → Exata

Valores aproximados pelo método de Runge-Kutta → RK4

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 subintervalos ***n***, menor é o erro de todos os Métodos.

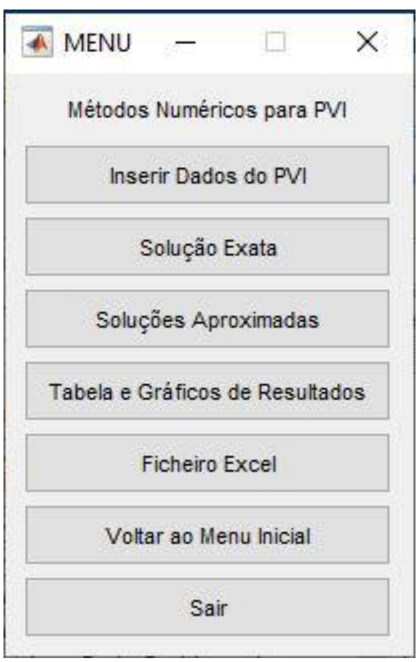
Quanto à comparação entre os métodos, observamos que os métodos que verificam menor erro e, consequentemente, melhor aproximação ao valor exato, são o método de Runge-Kutta de ordem 4 e o método usando a função ODE45 nativa do MATLAB, que muitas vezes apresentaram erros na ordem apenas das milésimas 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.

5.Anexos

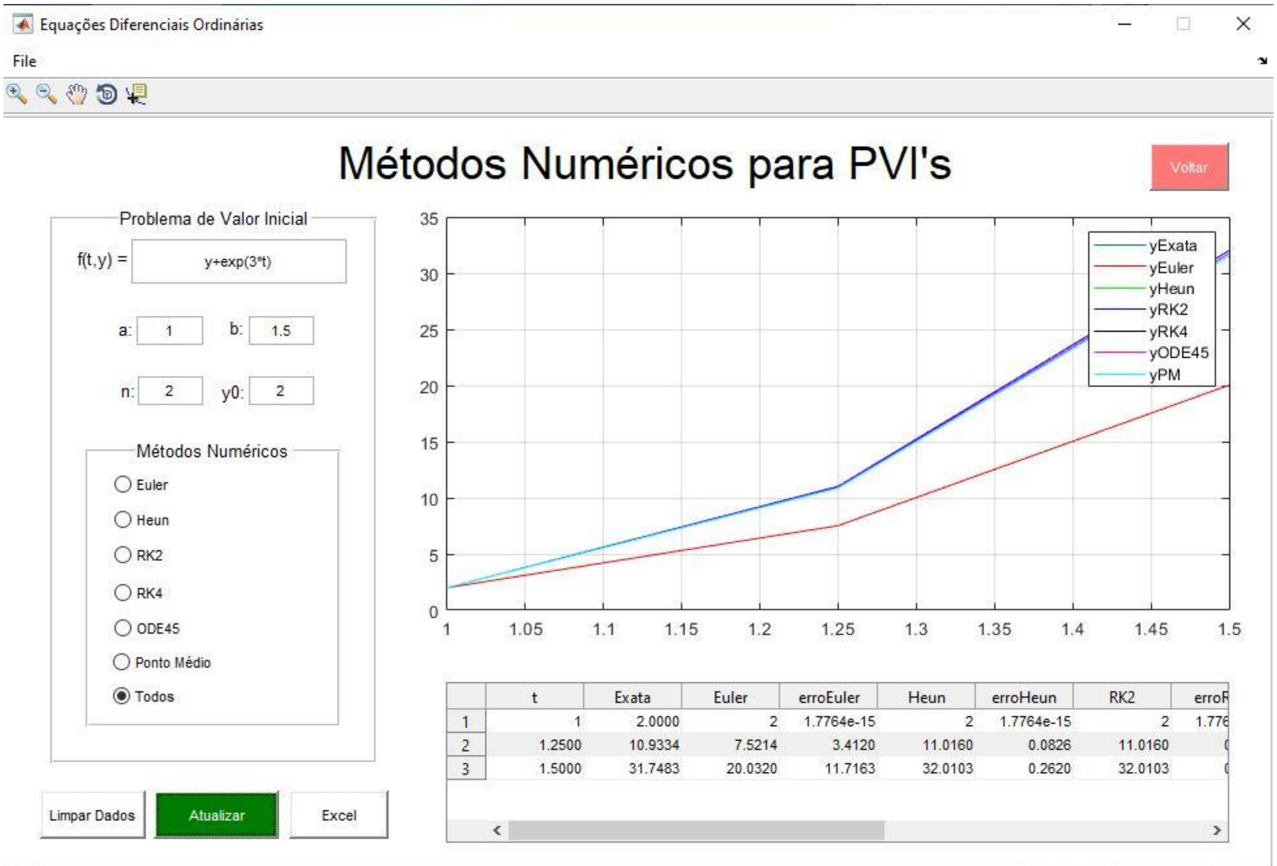
Início do Programa

Menu da Interface de Texto



Menu de Métodos Numéricos

GUI Métodos Numéricos para PVIs

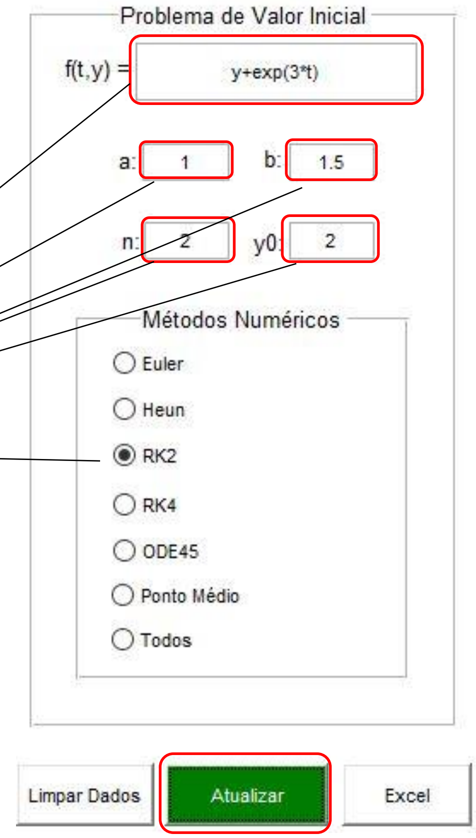


GUI Autores

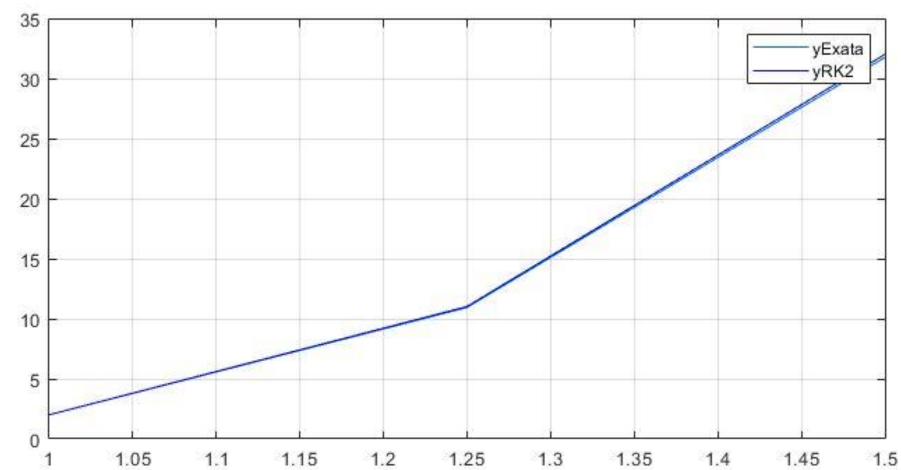
Breve Explicação do Programa

Os valores da **função *f(t,y)***, de ***a***,***b***,***n*** e ***y0*,** são enviados das <u>edit texts</u> correspondentes para a função do Método selecionado (neste caso, para RK2).

Métodos Numéricos
Método de Euler
Método de Heun
Método RK2
Método RK4
Método utilizando ODE45
Método do Ponto Médio



Os resultados da função são guardados em tabelas e enviados de volta para o programa, onde vai mostrar a tabela e o gráfico dos métodos selecionados.



	t	Exata	RK2	ErroRK2
1	1	2.0000	2	1.7764e-15
2	1.2500	10.9334	11.0160	0.0826
3	1.5000	31.7483	32.0103	0.2620

6.Bibliografia

- http://cee.uma.pt/edu/acn/docs/acn_formul5.pdf
- http://www.mat.uc.pt/~alma/aulas/anem/sebenta/cap6.pdf
- http://www.mat.uc.pt/~alma/aulas/matcomp/documentos/IntroducaoaMatlabParte203.pdf
- http://www.mat.uc.pt/~amca/MPII0607/folha3.pdf
- https://en.wikipedia.org/wiki/Heun%27s_method
- https://en.wikipedia.org/wiki/Midpoint_method
- https://pt.qwe.wiki/wiki/Heun%27s_method
- https://pt.wikipedia.org/wiki/M%C3%A9todo_de_Euler
- https://pt.wikipedia.org/wiki/M%C3%A9todo_de_Runge-Kutta#O_m%C3%A9todo_Runge%E2%80%93Kutta_cl%C3%A1ssico_de_quarta_ordem
- https://www.codecogs.com/latex/eqneditor.php?lang=en-us
- https://www.ime.unicamp.br/~valle/Teaching/MS211/Aula21.pdf
- https://www.respondeai.com.br/conteudo/calculo-numerico/resolucao-numerica-de-equacoes-diferenciais/metodo-de-euler/1355
- https://www.youtube.com/watch?v=HuRzoQjkZEs