Métodos Numéricos

PARA EQUAÇÕES DIFERENCIAIS ORDINÁRIAS / PROBLEMAS DE VALOR INICIAL

Ana Videira
2015012218 - EICE| ANÁLISE MATEMÁTICA II | ISEC 2020

Índice

1.Introdução

1.1 Enunciado da atividade proposta e interpretação do mesmo

Esta atividade tem como objetivo o aprofundar de conhecimentos sobre métodos numéricos para a resolução de Equações Diferenciais Ordinárias / Problemas de Valor Inicial e também desenvolver competências ao nível da programação em Matlab.

Deste modo, nesta atividade foram implementados os métodos numéricos expostos durante as aulas práticas de análise matemática II , em Matlab. A interface gráfica desenvolvida é uma adaptação da interface disponibilizada pelo professor Arménio Correia, responsável pela unidade curricular. Esta interface possibilita uma fácil utilização dos métodos implementados para conseguir as aproximações às soluções pretendidas e também o erro presente em cada uma dessas soluções.

1.2 Definição de PVI

Um problema que se pode traduzir por uma equação diferencial e o valor dessa função num determinado ponto é um PVI ou Problema de Valor inicial.

Este tipo de problemas tem a seguinte forma $\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases}$

Para a resolução deste tipo de problemas podemos utilizar métodos numéricos , que são métodos discretos, que geram soluções a aproximadas às soluções exatas.

2. Métodos Numéricos para resolução de PVI

2.1 Método de Euler

O método de Euler baseia-se na expansão de um polinómio de Taylor de grau 1 da solução de uma Equação Diferencial Ordinária. Deste método resulta uma solução aproximada à solução exata.

2.1.1 Fórmulas

sendo $\begin{array}{l} y' = f(t,y) \\ y(a) = y_0 \\ t \in [a,b] \end{array}$, $y_{i+1} = y_i + hf(t_i, y_i), \quad i = 0,\ldots,n-1$

2.1.2 Algoritmo/Função

```
Input: f,a,b,n,y0
Output : y

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
```

Função implementada em NEuler.m .

2.2 Método de Euler Melhorado ou Modificado

Erros de arredondamento crescem com a diminuição de passos na integração numérica ocorrendo assim divergência de valores ou mesmo valores errados para a solução procurada. Assim para resolver o problema existente, aumenta-se a ordem do método utilizado passado de método de Euler simples para melhorado ou modificado.

2.2.1 Fórmulas

Sendo $\begin{array}{l} y' = f(t,y) \\ y(a) = y_0 \\ t \in [a,b] \end{array}$, $y_{i+1} = y_i + hf(t_i, y_i), \quad i = 0,\ldots,n-1$

2.1.2 Algoritmo/Função

```
Input: f,a,b,n,y0
Output : y

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
```

Função implementada em NEulerM.m .

2.3 Método de RK2

O método de Runge-Kutta pretende ser uma melhoria do método de Euler. Este parte também da expansão do método de Taylor.

2.3.1 Fórmulas

Sendo $\begin{array}{l} y' = f(t,y) \\ y(a) = y_0 \\ t \in [a,b] \end{array}$, $\begin{array}{l} k_1 = hf(t_i, y_i); \ \ k_2 = hf(t_{i+1}, y_i + k_1) \\ y_{i+1} = y_i + \frac{1}{2}(k_1 + k_2), \quad i = 0, \ldots, n-1 \end{array}$

2.3.2 Algoritmo/Função

```
Input: f,a,b,n,y0
Output : y
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
```

Função implementada em NRK2.m .

2.4 Método de RK4

Tal como o sucedido no método Euler Melhorado também o método Runge-Kutta 4 é semelhante ao Runge-Kutta 2, apenas com mais iterações implementadas.

2.4.1 Fórmulas

Sendo $\begin{array}{l} y' = f(t,y) \\ y(a) = y_0 \\ t \in [a,b] \end{array}$, $\begin{array}{l} k_1 = hf(t_i,y_i);\ k_2 = hf(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_1);\ k_3 = hf(t_i + \frac{h}{2}, y_i + \frac{1}{2}k_2);\ k_4 = hf(t_i + h, y_i + k_3) \\ y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), \quad i = 0,\ldots,n-1 \end{array}$

2.4.2 Algoritmo/Função

Input: f,a,b,n,y0
Output : y

```
h = (b-a)/n;
t = a:h:b;
y(1) = y0;
for i=1:n
   k1 = h*f(t(i),y(i));
   k2 = h*f(t(i) + (0.5*h),y(i)+(0.5*k1));
   k3 = h*f(t(i) + (0.5*h),y(i)+(0.5*k2));
   k4 = h*f(t(i) + h,y(i)+k3);
   y(i+1) = y(i)+((k1+(2*k2)+(2*k3)+k4)/6);
end
```

Função implementada em NRK4.m .

2.5 Função ODE45 do Matlab

A função ode45 é uma função inserida em matlab, que tem como base uma combinação de métodos Runge-Kutta de quarta e quinta ordens.

**Função implementada em ODE45_1ORD.m .*

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

Sendo, $y' = f(t, y), \quad y(a) = y_0, \quad t \in [a, b]$

Condições iniciais do problema : $y' = y + t, \quad y(0) = 1, \quad t \in [0, 3]$

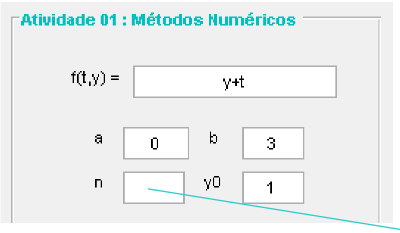


Figura 1 - GUI - Condições iniciais

Resultante de:

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

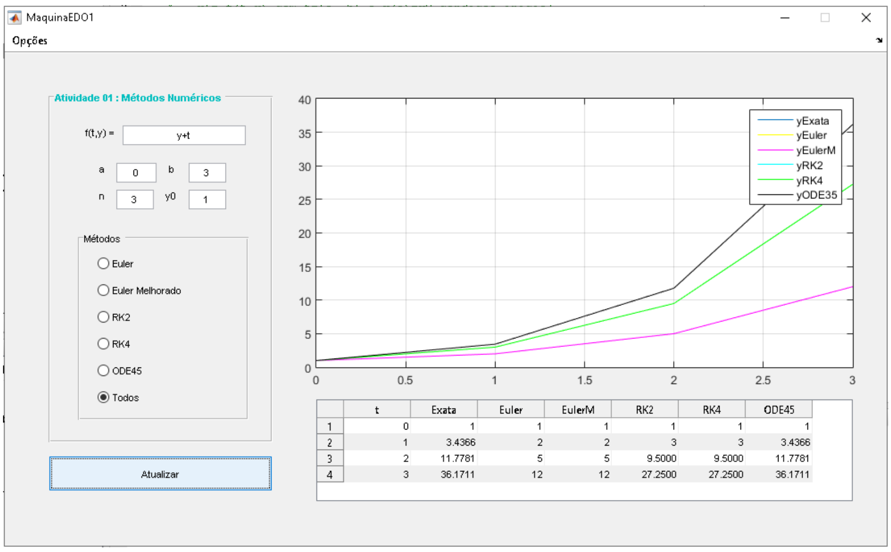
Assim, para conhecer o n necessários:

$$h = 1 \iff h = \frac{b-a}{n} \iff 1 = \frac{3-0}{n} \iff n = 3$$

$$h = 0.5 \iff h = \frac{b-a}{n} \iff 0.5 = \frac{3-0}{n} \iff n = 6$$

$$h = 0.25 \iff h = \frac{b-a}{n} \iff 0.25 = \frac{3-0}{n} \iff n = 12$$

3.1.2 Exemplos de output - GUI com gráfico e tabela



	t	Exata	Euler	EulerM	RK2	RK4	ODE45
1	0	1	1	1	1	1	1
2	1	3.4366	2	2	3	3	3.4366
3	2	11.7781	5	5	9.5000	9.5000	11.7781
4	3	36.1711	12	12	27.2500	27.2500	36.1711

Figura 2 - Todos os métodos para y(3)

Resolução de acordo com as condições iniciais apresentadas:

(a) O método de Euler explícito e:

h	n	$y(3)\ Exata$	$y(3)\ , Euler$	$y(3)\ , Euler\ M$	$Erro\ Euler$	$Erro\ Euler\ M$
1	3	36.1711	12	12	24.1711	24.1711
0.5	6	36.1711	18.7813	18.7813	17.3898	17.3898
0.25	12	36.1711	25.1038	25.1038	11.0672	11.0672

(b) O método de Runge-Kutta de 2ª ordem e:

h	n	$y(3)\ Exata$	$y(3)\ , RK2$	$Erro$
1	3	36.1711	27.2500	8.9211
0.5	6	36.1711	32.8256	3.3454
0.25	12	36.1711	35.1414	1.0297

(c) O método de Runge-Kutta de 4ª ordem e:

h	n	$y(3)\ Exata$	$y(3)\ , RK4$	$Erro$
1	3	36.1711	35.7316	0.4394
0.5	6	36.1711	36.1296	0.0415
0.25	12	36.1711	36.1679	0.0032

(d) Determine, utilizando a função *dsolve*, a solução exata do problema. Construa tabelas como a que se segue e compare a precisão dos resultados obtidos nas alíneas anteriores com o valor exato de $y(3)$

```
>> syms y(t) a
eqn = diff(y,t) == y+t;
cond = y(0) == 1;
ySol(t) = dsolve(eqn,cond)

ySol(t) =

2*exp(t) - t - 1
```

y

Figura 3 - Utilização da função dsolve

Tabela de comparações de métodos com h=1.

			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	1	1	1	1	0	0	0
1	1	3.4366	2	3	3.4167	1.4366	0.4366	0.0199
2	2	11.7781	5	9.5000	11.6701	6.7781	2.2781	0.1080
3	3	36.1711	12	27.2500	35.7316	24.1711	8.9211	0.4394

Através desta tabela de comparação de resultados podemos perceber que o métodos Runge - Kutta de ordem 4 é o mais preciso pois consegue uma gama de resultados mais aproximados à solução exata do problema.

3.2 Problema de aplicação 1

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 $v(5)$.

3.2.1 Modelação matemática do problema

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

$$Como\ v(0) = 0\ , k = 0.125\ , m = 5\ , g = 32 \quad => \quad y' = 32 - \frac{0.125 - v^2}{5} \Leftrightarrow y' = 32 - \frac{v^2}{40}$$

$$se\ t = 5\ => \ t\,[\,0,5\,]\ => \ h = 1 \Leftrightarrow \frac{b-a}{n} = 1 \Leftrightarrow \frac{5-0}{n} = 1 \Leftrightarrow n = 5$$

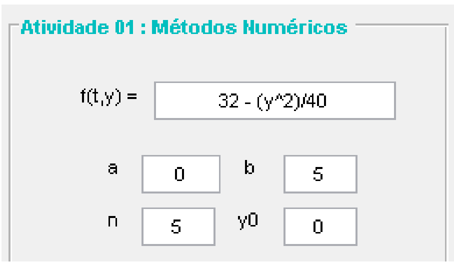
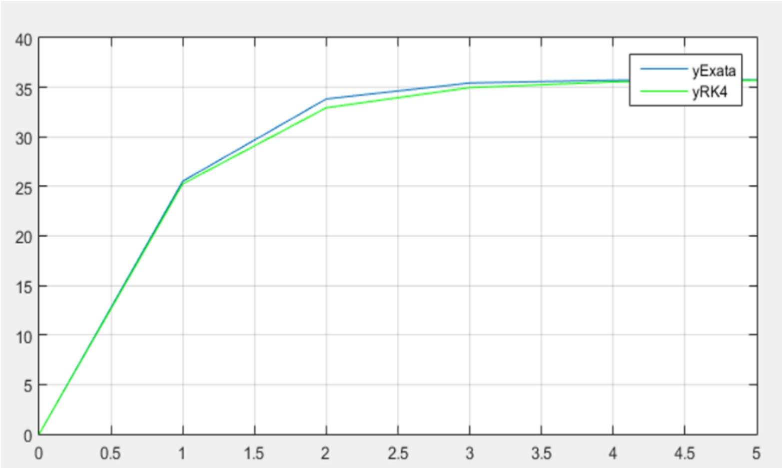


Figura 3 - Dados inseridos na aplicação para resolução do PVI

3.2.2 Resolução através da aplicação criada



	t	Exata	RK4	ErroRK4
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

Figura 4 - Resultados obtidos pelas condições iniciais descritos na figura 3

O método de Runge-Kutta utilizado foi o RK4 que ,como o provado no exercício de aplicação anteriormente exposto, é o que consegue uma maior aproximação aos resultados exatos.

Assim, para o valor de velocidade em t=5 , v(5) , recorrendo á aplicação trabalhada em Matlab, consegue-se os seguintes resultados através dos diversos métodos numéricos:

	t	Exata	Euler	EulerM	RK2	RK4	ODE45
2	1	25.5296	32	32	19.2000	25.2570	25.5305
3	2	33.8322	38.4000	38.4000	24.5588	32.9390	33.8331
4	3	35.4445	33.5360	33.5360	27.5118	34.9772	35.4445
5	4	35.7213	37.4194	37.4194	29.4569	35.5503	35.7212
6	5	35.7678	34.4141	34.4141	30.8457	35.7128	35.7677

4. Conclusão

Os métodos apresentados neste relatório são métodos de implementação simples e produzem soluções eficientes para diversos problemas envolvendo equações diferenciais.

Observa-se que a obtenção de resultados aproximados utilizando estes métodos numéricos é satisfatória, com maior destaque para o método Runge-Kutta de ordem 4 – Este é o método mais preciso e que obtém os melhores resultados como confirmado pelos problemas de aplicação descritos.

Através do método de Euler, embora não tenha sido possível verificar as diferenças entre o método de Euler simples e melhorado, pode também concluir-se que surgem resultados aceitáveis, sendo que estes são também melhores quanto maior for o número de iterações possíveis.