Instituto Superior de Engenharia de Coimbra

Análise Matemática II – Ano Letivo 2017/2018

Matlab: Atividade 2

Problemas de Valor inicial

Ana Rita Videira (21250074) – 18 de maio de 2018

Índice

1. Introdução

A atividade 2, descrita futuramente ao longo desde relatório, é um trabalho sugerido pela unidade curricular de Análise Matemática II. Pretende-se com esta atividade possibilitar mais uma oportunidade para desenvolvimento da linguagem Matlab como também para adquirir, aprofundar e consolidar conhecimentos sobre métodos numéricos.

Este relatório terá a seguinte estrutura: Iniciará com a descrição do enunciado da atividade e sua respetiva interpretação. Seguidamente irá ser dava uma explicação sobre a resolução do enunciado proposto e, numa parte final irá ser resolvido um exercício presente no Teste Intercalar A de 2015/2016 e um exercício de aplicação.

1.1 Enunciado da atividade proposta e interpretação do mesmo

Tendo em vista os ficheiros disponibilizados no moodle da unidade curricular, devem ser implementados os seguintes métodos numéricos:

- Euler;
- Euler Melhorado/Modificado;
- Runge-Kutta de ordem 2;
- Runge-Kutta de ordem 4;
- Função Ode 45;

Deverá ser criada uma interface gráfica que utilize os métodos numérico descritos acima. Assim, com a interface criada, que deve ser simples e funcional, deverá ser possível resolver um problema de valor inicial através dos vários métodos numéricos.

1.2 Definição de PVI

Equações que contém derivadas, chamadas de equações diferenciais. Uma equação diferencial que contém um conjunto de condições iniciais é chamada de **problema de valor inicial (PVI)**. São equações evolutivas, ou seja, dadas as condições inicias referidas anteriormente, evoluem com o passar do tempo.

Um PVI é uma equação diferencial na forma:

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases}$$

Assim, uma solução para este problema de valor inicial seria uma função f que satisfaça a condição:

$$y(t_0) = y_0 .$$

2. Métodos Numéricos para resolução de PVI

Para a resolução de problemas de valor inicial podemos utilizar diferentes métodos numéricos, alguns dos quais irão ser especificados seguidamente. A precisão dos resultados do PVI pode variar consoante o método utilizado e também o número de iterações.

2.1 Método de Euler

O Método de Euler é o tipo mais de método básico de integração numérica para equações diferenciais ordinárias com um valor inicial dado. Assim é o método mais básico para a resolução de problemas de valor inicial.

2.1.1 Fórmulas

$$y_{i+1} = y_i + hf(x_i, y_i)$$

i =0 ,1 ,2 , ….. n-1

2.1.2 Algoritmo/Função

```
% N_euler -  Método de Euler

%   y = N_euler(f,a,b,n,y0) Método numérico para a resolução de um PVI
%   y'= f(t,y) com t=[a, b] e y(a)=y0 condição inicial

%INPUT:
%   f - função do 2.º membro da Equação Diferencial
%   [a, b] - extremos do intervalo da variável independente t
%   n - número de subintervalos ou iterações do método
%   y0 - condição inicial t=a -> y=y0

%OUTPUT:
%   y - vector das aproximações discretas da solução exacta
%   y(i+1) = y(i)+h*f(t(i),y(i)) , i =0,1,...,n-2

% 18/03/13 - ArménioCorreia .: armenioc@isec.pt
% 27/04/18 - Ana Videira .: a21250074@isec.pt


function y = N_euler(f,a,b,n,y0)
h    = (b-a)/n;
t(1) = a;
y(1) = y0;

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

2.2 Método de Euler Melhorado ou Modificado

O método de Euler Melhorado ou Modificado é semelhante ao Método de Euler, mas encontra um resultado mais preciso.

2.2.1 Fórmulas

$$y_{i+1}^{*} = y_i + h * f(t_i, y_i), i = 0,1,2, \dots, \mathrm{n} - 1$$

$$y_{i+1} = y_i + h * \frac{f(t_i, y_i) + f(t_{i+1}, y_{i+1}^{*})}{2}, i = 0,1,2, \dots, n - 1$$

2.1.2 Algoritmo/Função

```
% N_eulerM - Método de Euler Melhorado

%   y = N_eulerM(f,a,b,n,y0) Método numérico para a resolução de um PVI
%   y'= f(t,y) com t=[a, b] e y(a)=y0 condição inicial

%INPUT:
%   f - função do 2.º membro da Equação Diferencial
%   [a, b] - extremos do intervalo da variável independente t
%   n - número de subintervalos ou iterações do método
%   y0 - condição inicial t=a -> y=y0

%OUTPUT:
%   y - vector das aproximações discretas da solução exacta
%   y(i+1) = y(i)+h*f(t(i),y(i)) , i =0,1,...,n-2

% 27/04/18 - Ana Videira .: a21250074@isec.pt
```

```
function y = N_EulerM(f,a,b,n,y0)
h    = (b-a)/n;
t(1) = a;
y(1) = y0;

for i=y0:h:10.1
    y(i+1)=y(i)+(y(i)+y(i+1))*(h/2);
    t(i+1)=t(i)+(h);
end
```

2.3 Método de RK2

O método Runge-Kutta pode ser visto com uma melhoria do método de Euler.

2.3.1 Fórmulas

$$y_{i+1} = y_i + \frac{h}{2}(f(x_i, y_i) + f(x_i + h, y_i + hf(x_i, y_i)))$$

2.3.2 Algoritmo/Função

```
%INPUT:
%   f - função do 2º membro da equação diferencial
%   [a, b] - extremos do intervalo da variável independente t
%   n - número de iterações do método
%   y0 - condição inicial t=a -> y=y0

%OUTPUT:
%   y - vector das aproximações discretas da solução exacta
%   y(i+1) = y(i)+(1/2)*(k1 + k2), i =0,1,...,n-1

function y = N_RK2(f,a,b,n,y0)

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+1),y(i)+k1);
    y(i+1) = y(i)+0.5*(k1+k2);

end
```

2.4 Método de RK4

O Runge-Kutta 4 é o método mais preciso de todos os descritos neste relatório, isto porque utiliza 4 valores de K diferentes para cada iteração.

2.4.1 Fórmulas

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

$$k_1 = f(x_i, y_i) \qquad k_3 = f(x_i + h/2, y_i + hk_2/2)$$

$$k_2 = f(x_i + h/2, y_i + hk_1/2) \qquad k_4 = f(x_i + h, y_i + hk_3)$$

2.4.2 Algoritmo/Função

```
%RK4  Método de Runge-Kutta de ordem 4

%   Método numérico para a resolução de um
%   problema de valor inicial (PVI):
%                        y = N_RK4(f,a,b,n,y0)

%INPUT:
%   f - função do 2º membro da equação diferencial
%   [a, b] - extremos do intervalo da variável independente t
%   n - número de iterações do método
%   y0 - condição inicial t=a -> y=y0

%OUTPUT:
%   y - vector das aproximações discretas da solução exacta
%   y(i+1) = y(i)+((k1+(2*k2)+(2*k3)+k4)/6), i =0,1,...,n-1


function y = N_RK4(f,a,b,n,y0)

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
```

2.5 Função ODE45 do Matlab

ode45(odefun,tspan,y0) -> Permite resolver sistemas de equações do tipo: $y' = f(t, y)$

Sendo que:
odefun -> função f
tspan -> intervalo [a,b]
y0 -> condição inicial

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

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

f(t,y) = y+t
f(t,y) = -2ty
y(0) = 3

a=0
b=2
n=1

h=(2-0)/1=2

3.2 Problema de aplicação

Devido a problemas relativos à implementação de funções/interface, não foi possível a resolução deste ponto.

4. Conclusão

Através desta atividade foi possível praticar o desenvolvimento de código em linguagem Matlab, aprofundar e consolidar conhecimentos.

Os métodos indicados para implementação são relativamente simples e produzem soluções para diversos problemas de valor inicial, com níveis de precisão diferentes.

Para finalizar, pode-se concluir, que o método de Euler é o simples menos preciso em comparação a todos os outros descritos neste relatório. O método de Runge-Kutta 4 é o mais preciso visto que é o que utiliza a maior quantidade (4) de valores k diferentes para cada iteração.