Matemática Computacional
MEBiol, MEBiom e MEFT
Aula 18 - Métodos numéricos para Equações Diferenciais Ordinárias

Ana Leonor Silvestre

Instituto Superior Técnico, $1^{\underline{o}}$ *Semestre, 2020/2021*

Sumário da Aula 18

Cap. 6 - Resolução Numérica de Equações Diferenciais Ordinárias

Método de Euler. Análise de erros.

Método de Taylor de segunda ordem.

Métodos de Runge-Kutta.

Resolução numérica de equações diferenciais

No Mathematica: a rotina **NDSolve** permite a **resolução numérica de equações diferenciais ordinárias**.

No MATLAB: `ode23`, `ode45`, ...

No Python: `odeint` disponível no pacote `scipy.integrate`.

Exemplo: "Dynamical Models of Love"

De acordo com um modelo de dinâmica de relações, a evolução dos sentimentos entre duas pessoas, digamos Romeu (R) e Julieta (J), é descrita por um sistema de duas EDOs não-lineares

$$\begin{cases} R'(t) &= aR(t) + bJ(t)(1 - |J(t)|), \\ J'(t) &= cR(t)(1 - |R(t)|) + dJ(t). \end{cases}$$

Neste sistema, $R(t)$ e $J(t)$ representam o nível de satisfação de Romeu e Julieta com a relação (no dia t).
Os sinais dos coeficientes a, b, c, d definem os seus estilos românticos. Por exemplo, Romeu ansioso/ávido ($a > 0$ e $b > 0$), narcisista ($a > 0$ e $b < 0$), cauteloso ($a < 0$ e $b > 0$) ou tímido ($a < 0$ e $b < 0$). A dinâmica é impulsionada apenas por interação entre os estados emocionais de Romeu e Julieta e consequentes reações.

"Dynamical Models of Love"

Pretende-se obter gráficos que ilustrem a dinâmica da relação de Romeu e Julieta durante (pelo menos) 1 ano, partindo de uma situação descrita pelos valores $R(0) = 5.5$ e $J(0) = 4.5$.
Vários casos a estudar:

1. $a = -0.02$, $b = 0.05$, $c = -0.03$, $d = 0.01$;
2. $a = -0.02$, $b = 0.05$, $c = -0.03$, $d = -0.01$;
3. $a = -0.02$, $b = 0.05$, $c = 0.03$, $d = -0.01$;
4. $a = -0.01$, $b = 0.05$, $c = -0.03$, $d = 0.01$.

Resolução em Mathematica

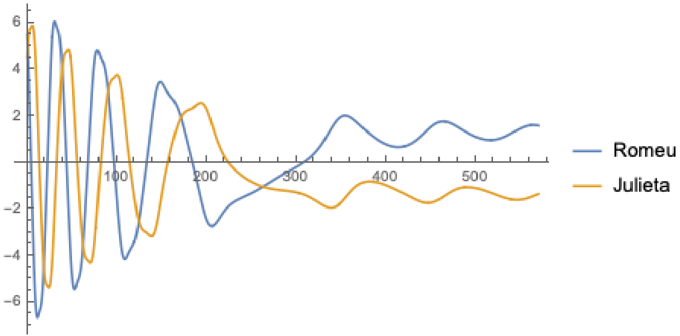
```
In[1]:=
a=-0.02;b=0.05;c=-0.03;d=0.01;
sl=NDSolve[
    {R'[t]==a*R[t]+b*J[t]*(1-Abs[J[t]]),
        J'[t]==c*R[t]*(1-Abs[R[t]])+d*J[t],
        R[0]==5.5,J[0]==4.5},{R,J},{t,0,570}]

Out[1]:=
R->InterpolatingFunction[Domain: 0.,570.Output: scalar],
J->InterpolatingFunction[Domain: 0.,570. Output: scalar]

In[2]:=
Plot[Evaluate[{R(t), J(t)}/. sl], {t, 0, 570},
    PlotStyle → Automatic, PlotLegends → {Romeu, Julieta}]
```

Gráfico

Out[2]:=



Romeu seguro $a = -0.02$, $b = 0.05$,
Julieta segura $c = -0.03$, $d = 0.01$.

Algumas questões...

- O que está na base da implementação do comando

  `NDSolve`?

- Por que razão a solução é fornecida em termos de

  `InterpolatingFunction`?

O Teorema de Picard-Lindelöf

Fornece

- Existência e unicidade de solução (local) para o *problema de valor inicial*

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

- Um processo iterativo para aproximar a solução de $(\mathcal{P})$.

Teorema (Picard-Lindelöf) Seja $G \subset \mathbb{R}^{d+1}$ um domínio e $f : G \to \mathbb{R}^d$ uma função contínua satisfazendo uma *condição de Lipschitz*

$$\exists L_f \geq 0 \, : \|f(t, y) - f(t, z)\| \leq L_f \|y - z\|, \, \forall (t, y), (t, z) \in G.$$

Então para cada par de dados iniciais $(t_0, y_0) \in G$, existe um intervalo $[t_0 - a, t_0 + a]$, $a > 0$, tal que o problema de Cauchy $(\mathcal{P})$ tem solução única.

O Teorema de Picard-Lindelöf

Em alguns casos, é possível estabelecer a seguinte *versão global* do teorema anterior:

Teorema Seja $I = [a, b]$ um intervalo de $\mathbb{R}$ e $f : I \times \mathbb{R}^d \to \mathbb{R}^d$ uma função contínua satisfazendo uma condição de Lipschitz

$$\exists L_f \geq 0 \, : \|f(t, y) - f(t, z)\| \leq L_f \|y - z\|, \, \forall (t, y), (t, z) \in I \times \mathbb{R}^d.$$

Então para cada par de dados iniciais $(t_0, y_0) \in I \times \mathbb{R}^d$, a solução $y = y(t)$ do problema ($\mathcal{P}$) existe, é única e pertence a $C^1(I)$.

Resolução numérica de EDOs - Método de Euler

Supondo $y \in C^2([a,b])$ e $t, t+h \in [a,b]$, pela *fórmula de Taylor*, tem-se

$$y(t+h) = y(t) + hy'(t) + \frac{h^2}{2} y''(\xi(t,h)),\ t < \xi(t,h) < t+h,$$

donde

$$y'(t) = \frac{y(t+h) - y(t)}{h} - \frac{h}{2} y''(\xi(t,h)) = \frac{y(t+h) - y(t)}{h} + \tau(t,h).$$

Ao termo

$$\boxed{\tau(t,h) = -\frac{h}{2} y''(\xi(t,h))}$$

chama-se erro de truncatura local.

Resolução numérica de EDOs - Método de Euler

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

Consideremos $h = (b-a)/n$ e os nós igualmente espaçados

$$t_i = a + ih,\ i = 0, ..., n.$$

Supondo $y \in C^2([a, b])$

$$y'(t_i) = \frac{y(t_i + h) - y(t_i)}{h} + \tau(t_i, h).$$

Usando a equação diferencial, temos

$$y'(t_i) = f(t_i, y(t_i))$$

e portanto

$$f(t_i, y(t_i)) = \frac{y(t_i + h) - y(t_i)}{h} + \tau(t_i, h).$$

Método de Euler

Agora, desprezando o erro de truncatura local $\tau(t_i, h)$ em

$$\frac{y(t_{i+1}) - y(t_i)}{h} = f(t_i, y(t_i)) - \tau(t_i, h)$$

obtém-se a aproximação

$$y(t_{i+1}) \approx y(t_i) + hf(t_i, y(t_i)),\ i = 0, ..., n-1.$$

É claro que

$$y(t_0) = y(a) = y_a =: y_0.$$

Para $i = 0$ tem-se

$$y(t_1) \approx y(t_0) + hf(t_0, y(t_0)) = y_0 + hf(t_0, y_0) =: y_1.$$

Para $i = 1$ tem-se

$$y(t_2) \approx y(t_1) + hf(t_1, y(t_1)) \approx y_1 + hf(t_1, y_1) =: y_2.$$

Método de Euler - Algoritmo

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

$$h = (b-a)/n, \qquad t_i = a + ih, \, i = 0, ..., n.$$

$$\begin{cases} y_0 = y_a, \\ t_i = a + ih, \, i = 0, ..., n, \\ y_{i+1} = y_i + hf(t_i, y_i), \, i = 0, ..., n-1 \end{cases}$$

$$y(t_i) \approx y_i, \, i = 1, ..., n$$

Ao conjunto dos valores $\{y_0, y_1, ..., y_n\}$ chama-se solução numérica do problema $(\mathcal{P})$ obtida pelo método de Euler.

Exemplo

$$\begin{cases} y'(t) = y(t) + 1 - t^2, \; t \in [0,1], \\ y(0) = 0.5 \end{cases}$$

Aproximações para $y(0.2), y(0.4), ..., y(1)$?

$0.2 - 0 = 0.4 - 0.2 = ... = 1 - 0.8 = h = 0.2 \Rightarrow n = 5$

$t_i = 0.2i, \; i = 0, ..., 5$

$y(t_0) = y(0) = 0.5.$

Pretendemos $y(t_i) \approx y_i, \; i = 1, ..., 5$. Para escrever o algoritmo do método de Euler, começamos por identificar

$$f(t, y(t)) = y(t) + 1 - t^2$$

e daqui:

$$f(t, y) = y + 1 - t^2 \Rightarrow f(t_i, y_i) = y_i + 1 - t_i^2.$$

Exemplo

Assim, as aproximações para $y(0.2), y(0.4), ..., y(1)$ são obtidas através de

$$\begin{cases} y_0 = 0.5, \\ t_i = 0.2i, \; i = 0, ..., 5, \\ y_{i+1} = y_i + 0.2(y_i + 1 - t_i^2), \; i = 0, ..., 4. \end{cases}$$

Temos então

$$y(0.2) = y(t_1) \approx y_1 = y_0 + 0.2(y_0 + 1 - t_0^2) = 0.5 + 0.2(0.5 + 1 - 0^2) = 0.8$$

$$y(0.4) = y(t_2) \approx y_2 = y_1 + 0.2(y_1 + 1 - t_1^2) = 0.8 + 0.2(0.8 + 1 - 0.2^2) = 1.152$$

$$y(0.6) = y(t_3) \approx y_3 = y_2 + 0.2(y_2 + 1 - t_2^2) = 1.152 + 0.2(1.152 + 1 - 0.4^2) = 1.5504$$

$$y(0.8) = y(t_4) \approx y_4 = y_3 + 0.2(y_3 + 1 - t_3^2) = 1.5504 + 0.2(1.5504 + 1 - 0.6^2) = 1.98848$$

$$y(1.0) = y(t_5) \approx y_5 = y_4 + 0.2(y_4 + 1 - t_4^2) = 1.98848 + 0.2(1.98848 + 1 - 0.8^2) = 2.458176$$

Método de Euler - Implementação em Matlab

Método de Euler

$$\begin{cases} y_0 = y_a, \\ t_i = a + ih,\ i = 0 : n, \\ y_{i+1} = y_i + hf(t_i, y_i),\ i = 0 : n-1 \end{cases}$$

Em MATLAB

$$t(i) = a + h(i-1),\ i = 1 : n+1$$

$$y(t(i)) \approx yaprox(i)$$

```
function  yaprox = met_Euler(a,b,f,y0,n)
h=(b-a)/n;
t=linspace(a,b,n+1);
yaprox=zeros(1,n+1);
yaprox(1)=y0;
for i=1:n
   yaprox(i+1)=yaprox(i)+h*f(t(i),yaprox(i));
end;
end
```

Exemplo

$$y'(t) = y(t) + 1 - t^2, \quad y(0) = 0.5$$

Aproximações para $y(0.2), y(0.4), ..., y(1)$?

$h = 0.2$ ou $n = 5$

$t_i = 0.2i,\ i = 0 : 5$

$y(t_i) \approx y_i,\ i = 0 : 5$

Em MATLAB

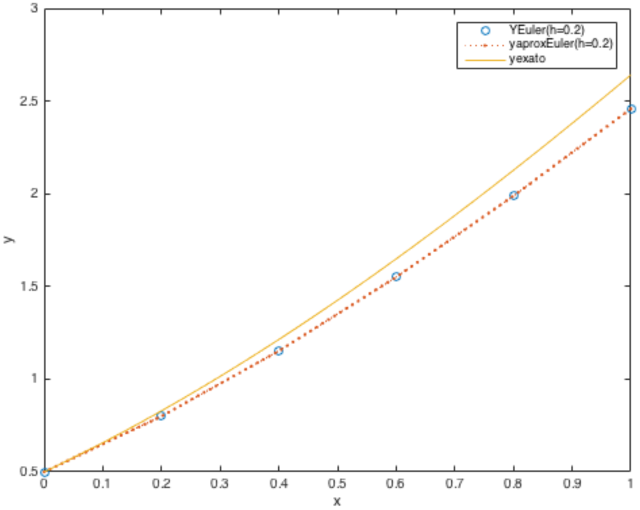
$t(i) = 0.2(i-1),\ i = 1 : 6$

$y(t(i)) \approx yEuler(i)$

```
>> yEuler = met_Euler(0,1,@(t,y)1-t^2+y,0.5,5)'

yEuler =
                    0.5
                    0.8
                  1.152
                 1.5504
                1.98848
               2.458176
```

Exemplo



YEuler(h=0.2)
yaproxEuler(h=0.2)
yexato
y
x
3
2.5
2
1.5
1
0.5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Exemplo

$h = 0.1$

$t_i = 0.1i,\ i = 0:10$

$y(t_i) \approx y_i,\ i = 0:10$

Em MATLAB

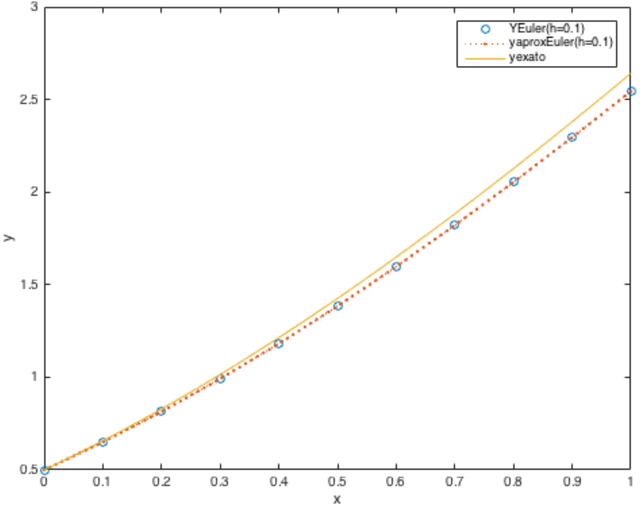
$t(i) = 0.1(i-1),\ i = 1:11$

$y(t(i)) \approx yEuler(i)$

```
>> yEuler = met_Euler(0,1,@(t,y)1-t^2+y,0.5,10)'
yEuler =

              0.5
             0.65
            0.814
           0.9914
          1.18154
         1.383694
        1.5970634
       1.82076974
      2.053846714
     2.2952313854
    2.54375452394
```

Exemplo



YEuler(h=0.1)
yaproxEuler(h=0.1)
yexato
y
x
0.5
1
1.5
2
2.5
3
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Exemplo

t	y_h $(h = 0.2)$	y_h $(h = 0.1)$	y (exato)
0	0.5	0.5	0.5
0.2	0.8	0.814	0.829298620919915
0.4	1.152	1.18154	1.21408765117936
0.6	1.5504	1.5970634	1.64894059980475
0.8	1.98848	2.053846714	2.12722953575377
1.0	2.458176	2.54375452394	2.64085908577048

solução exata: $y(t) = 1 + 2t + t^2 - 0.5\,\exp(t)$

Majoração dos erros no método de Euler

Teorema

Seja $\{y_1, y_2, ..., y_n\}$ a solução numérica gerada pelo método de Euler para a aproximação de

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

em pontos equidistantes $t_1, ..., t_n$. Se f é uma função nas condições do Teorema de Picard-Lindelöf e se $y \in C^2([t_0, t_n])$ com $\max_{t \in [t_0, t_n]} |y''(t)| \leq M$, então o erro global satisfaz

$$|y(t_i) - y_i| \leq \frac{hM}{2L_f}(e^{(t_i - t_0)L_f} - 1), \; i = 1, 2, ..., n.$$

Resolução numérica de EDOs - Mét. de Taylor de ordem 2

Supondo $y \in C^3([a,b])$, pela *fórmula de Taylor*, tem-se

$$y(t_{i+1}) = y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(t_i) + \frac{h^3}{6}y^{(3)}(\theta_i),\ t_i < \theta_i < t_{i+1}.$$

Usando a equação diferencial, temos

$$y'(t_i) = f(t_i, y(t_i))$$

e

$$\begin{aligned} y''(t_i) &= \frac{\partial f}{\partial t}(t_i, y(t_i)) + \frac{\partial f}{\partial y}(t_i, y(t_i))y'(t_i) \\ &= \frac{\partial f}{\partial t}(t_i, y(t_i)) + \frac{\partial f}{\partial y}(t_i, y(t_i))f(t_i, y(t_i)) \end{aligned}$$

e desprezando o termo $\frac{h^3}{6}y^{(3)}(\theta_i)$, obtém-se a aproximação

$$\begin{aligned} y(t_{i+1}) &\approx y(t_i) + hf(t_i, y(t_i)) \\ &\qquad + \frac{h^2}{2}\left[\frac{\partial f}{\partial t}(t_i, y(t_i)) + \frac{\partial f}{\partial y}(t_i, y(t_i))f(t_i, y(t_i))\right]. \end{aligned}$$

Método de Taylor de ordem 2

A aproximação

$$y(t_{i+1}) \approx y(t_i) + hf(t_i, y(t_i)) \\ + \frac{h^2}{2}\left[\frac{\partial f}{\partial t}(t_i, y(t_i)) + \frac{\partial f}{\partial y}(t_i, y(t_i)) f(t_i, y(t_i))\right]$$

leva ao *método de Taylor de ordem 2*:

$$y_{i+1} = y_i + hf(t_i, y_i) + \frac{h^2}{2}\left[\frac{\partial f}{\partial t}(t_i, y_i) + \frac{\partial f}{\partial y}(t_i, y_i) f(t_i, y_i)\right].$$

Método de Taylor de ordem 2 - Algoritmo

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

$$h = (b-a)/n, \qquad t_i = a + ih,\ i = 0, \dots, n$$

As aproximações $y(t_i) \approx y_i$, $i = 1, \dots, n$ são obtidas recursivamente por

$$\begin{cases} y_0 = y_a, \\ t_i = a + ih,\ i = 0, \dots, n, \\ y_{i+1} = y_i + hf(t_i, y_i) + \\ \qquad\qquad \dfrac{h^2}{2}\left[\dfrac{\partial f}{\partial t}(t_i, y_i) + \dfrac{\partial f}{\partial y}(t_i, y_i) f(t_i, y_i)\right],\ i = 0, \dots, n-1 \end{cases}$$

Ao conjunto dos valores $\{y_0, y_1, \dots, y_n\}$ chama-se solução numérica do problema $(\mathcal{P})$ obtida pelo método de Taylor de ordem 2.

Exemplo

$$\begin{cases} y'(t) = y(t) + 1 - t^2, \ t \in [0, 1], \\ y(0) = 0.5 \end{cases}$$

Aproximações para $y(0.2), y(0.4), ..., y(1)$?

$$0.2 - 0 = 0.4 - 0.2 = ... = 1 - 0.8 = h = 0.2 \Rightarrow n = 5$$

$$t_i = 0.2i, \ i = 0, ..., 5$$

$$y(t_0) = y(0) = 0.5.$$

Pretendemos $y(t_i) \approx y_i, \ i = 1, ..., 5$. Para escrever o algoritmo do método de Taylor de ordem 2, começamos por lembrar

$$f(t, y(t)) = y(t) + 1 - t^2$$

e daqui:

$$f(t, y) = y + 1 - t^2, \quad \frac{\partial f}{\partial t}(t, y) = -2t, \quad \frac{\partial f}{\partial y}(t, y) = 1$$

Exemplo

$$\begin{cases} y_0 = 0.5, \\ t_i = 0.2i, \; i = 0, ..., 5, \\ y_{i+1} = y_i + 0.2(y_i + 1 - t_i^2) + \\ \qquad\qquad 0.02 \left[-2t_i + y_i + 1 - t_i^2 \right], \; i = 0, ..., 4 \end{cases}$$