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 Taylor de segunda ordem.

Métodos de Runge-Kutta.

Resolução numérica de sistemas de EDOs.

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

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

Método de Taylor de ordem 2

A aproximação

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

leva ao *método de Taylor de ordem 2*:

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

Nota: É claro que se y é um polinómio de segundo grau então o termo $\frac{h^3}{6} y^{(3)}(\theta_i)$ desprezado é nulo e o método de Taylor de ordem 2 fornece a solução exata $\{y(t_0), y(t_1), ..., y(t_n)\}$.

Método de Taylor de ordem 2 - Algoritmo

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

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

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

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

Ao conjunto dos valores $\{y_0, y_1, ..., y_n\}$ 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

$$y(0.2), y(0.4), ..., y(1)$$

usando o método de Taylor de ordem 2?

$$h = 0.2, \quad n = 5, \qquad t_i = 0.2i, \; i = 0, ..., 5$$

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

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

Para $h = 0.2$, o algoritmo do método de Taylor de ordem 2 fica

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

Exemplo

$$t_0 = 0, \quad t_1 = 0.2, \quad t_2 = 0.4, \quad t_3 = 0.6, \quad t_4 = 0.8$$

$$\begin{aligned}
y(0.2) = y(t_1) \approx y_1 &= y_0 + 0.2(y_0 + 1 - t_0^2) + 0.02(y_0 + 1 - 2t_0 - t_0^2) \\
&= 0.5 + 0.2(0.5 + 1) + 0.02(0.5 + 1) = \mathbf{0.83} \\
y(0.4) = y(t_2) \approx y_2 &= y_1 + 0.2(y_1 + 1 - t_1^2) + 0.02(y_1 + 1 - 2t_1 - t_1^2) \\
&= 0.83 + 0.2(0.83 + 1 - 0.04) \\
&\quad +0.02(0.83 + 1 - 0.4 - 0.04) = \mathbf{1.2158} \\
y(0.6) = y(t_3) \approx y_3 &= y_2 + 0.2(y_2 + 1 - t_2^2) + 0.02(y_2 + 1 - 2t_2 - t_2^2) \\
&= 1.2158 + 0.2(1.2158 + 1 - 0.16) \\
&\quad +0.02(1.2158 + 1 - 0.8 - 0.16) = \mathbf{1.652076} \\
y(0.8) = y(t_4) \approx y_4 &= y_3 + 0.2(y_3 + 1 - t_3^2) + 0.02(y_3 + 1 - 2t_3 - t_3^2) \\
&= \mathbf{2.13233272} \\
y(1.0) = y(t_5) \approx y_5 &= y_4 + 0.2(y_4 + 1 - t_4^2) + 0.02(y_4 + 1 - 2t_4 - t_4^2) \\
&= \mathbf{2.64864592}
\end{aligned}$$

Exemplo

O problema

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

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

t	$h = 0.2$, Euler	$h = 0.2$, Taylor 2	y (exato)
0	0.5	0.5	0.5
0.2	0.8	0.83	0.82929862
0.4	1.152	1.2158	1.21408765
0.6	1.5504	1.652076	1.64894060
0.8	1.98848	2.13233272	2.12722954
1.0	2.458176	2.64864592	2.64085909

Métodos de Runge-Kutta de ordem 2

Esta classe pode ser obtida por comparação com o método de Taylor de ordem 2

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

de modo a evitar o cálculo das derivadas parciais de f. Considera-se um esquema da forma

$$\begin{cases} y_{i+1} = y_i + h(\gamma K_1 + \delta K_2), \\ K_1 = f(t_i, y_i), \\ K_2 = f(t_i + \alpha h, y_i + \beta h K_1), \end{cases}$$

onde as constantes α, β, γ e δ serão determinadas de modo que este método seja "semelhante" ao método de Taylor de ordem 2.

Métodos de Runge-Kutta de ordem 2

Se f é de classe C^2, tem-se

$$f(t_i + \alpha h, y_i + \beta h K_1) = \\ = f(t_i, y_i) + \alpha h \frac{\partial f}{\partial t}(t_i, y_i) + \beta h K_1 \frac{\partial f}{\partial y}(t_i, y_i) + \mathcal{O}(h^2).$$

Substitui-se em

$$K_2 = f(t_i + \alpha h, y_i + \beta h K_1)$$

$$y_{i+1} = y_i + h(\gamma K_1 + \delta K_2)$$

e despreza-se o termo $\mathcal{O}(h^2)$, o que dá

$$y_{i+1} = y_i + (\gamma + \delta) h f(t_i, y_i) + \alpha \delta h^2 \frac{\partial f}{\partial t}(t_i, y_i) + \beta \delta h^2 \frac{\partial f}{\partial y}(t_i, y_i) f(t_i, y_i).$$

Métodos de R-K de ordem 2

Comparando

$$y_{i+1} = y_i + (\gamma + \delta) h f(t_i, y_i) + \alpha\delta h^2 \frac{\partial f}{\partial t}(t_i, y_i) + \beta\delta h^2 \frac{\partial f}{\partial y}(t_i, y_i) f(t_i, y_i)$$

com

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

obtém-se

$$\gamma + \delta = 1,\ \alpha\delta = \beta\delta = \frac{1}{2}.$$

Métodos de Runge-Kutta de ordem 2

Das 3 equações anteriores para α, β, γ e δ, resulta

$$\alpha = \beta, \quad \delta = \frac{1}{2\alpha}, \quad \gamma = 1 - \frac{1}{2\alpha}.$$

A família de *métodos de Runge-Kutta de ordem 2* definidos pelo parâmetro α é:

$$\begin{cases} y_0 = y_a, \\ t_i = a + hi, \; i = 0, ..., n, \\ K_1^{(i)} = f(t_i, y_i), \\ K_2^{(i)} = f\left(t_i + \alpha h, y_i + \alpha h K_1^{(i)}\right), \\ y_{i+1} = y_i + h\left[\left(1 - \frac{1}{2\alpha}\right) K_1^{(i)} + \frac{1}{2\alpha} K_2^{(i)}\right], \; i = 0, ..., n-1. \end{cases}$$

Métodos de Runge-Kutta de ordem 2 - método do ponto médio

Corresponde a $\alpha = 1/2$:

$$\begin{cases} y_0 = y_a, \\ t_i = a + hi,\ i = 0, ..., n, \\ K_1^{(i)} = f(t_i, y_i), \\ K_2^{(i)} = f\left(t_i + \dfrac{h}{2}, y_i + \dfrac{h}{2} K_1^{(i)}\right), \\ y_{i+1} = y_i + hK_2^{(i)},\ i = 0, ..., n-1. \end{cases}$$

Métodos de Runge-Kutta de ordem 2 - método de Heun

Corresponde a $\alpha = 1$:

$$\begin{cases} y_0 = y_a, \\ t_i = a + hi, \; i = 0, ..., n, \\ K_1^{(i)} = f(t_i, y_i), \\ K_2^{(i)} = f\left(t_{i+1}, y_i + hK_1^{(i)}\right), \\ y_{i+1} = y_i + \dfrac{h}{2}\left[K_1^{(i)} + K_2^{(i)}\right], \; i = 0, ..., n-1. \end{cases}$$

Método de Runge-Kutta de ordem 4 clássico

$$\begin{cases} y_0 = y_a, \\ t_i = a + hi, \; i = 0, ..., n, \\ K_1 = f(t_i, y_i), \quad K_2 = f\left(t_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}K_1\right), \\ K_3 = f\left(t_i + \dfrac{h}{2}, y_i + \dfrac{h}{2}K_2\right), \quad K_4 = f\left(t_i + h, y_i + hK_3\right), \\ y_{i+1} = y_i + \dfrac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4), \; i = 0, ..., n-1. \end{cases}$$

Sistemas de equações

Resolução numérica de EDOs - Método de Euler

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

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

$$Y(t(i)) \approx Y(:, i)$$

```
function [Yaprox]=Euler_method(f,a,b,Y0,n)
t=linspace(a,b,n+1);
Yaprox=zeros(length(Y0),n+1);
Yaprox(:,1)=Y0;
h=(b-a)/n;
for i=2:n+1
Yaprox(:,i)=Yaprox(:,i-1)+h*f(t(i-1),Yaprox(:,i-1));
end;
end
```

"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), e 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 $R(0) = 2.5$ e $J(0) = 1.5$. Vários casos:

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. $a = -0.01$, $b = 0.05$, $c = -0.03$, $d = 0.01$.

Dados do problema

$$Y0 = \begin{bmatrix} Y0_1 \\ Y0_2 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 1.5 \end{bmatrix}$$

$$F : [t_0, t_f] \times \mathbb{R}^2 \to \mathbb{R}^2$$

$$F(t, Y) = F\left(t, \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}\right) = \begin{bmatrix} aY_1 + bY_2(1 - |Y_2|) \\ cY_1(1 - |Y_1|) + dY_2 \end{bmatrix}$$

Foi criado um script para resolver o problema recorrendo ao programa onde se implementou o método de Euler

```
f_RJ = @(a,b,c,d)(@(t,Y)[a*Y(1)+b*Y(2)*(1-abs(Y(2))); ...
                         c*Y(1)*(1-abs(Y(1)))+d*Y(2)]);
Y0 = [2.5; 1.5];
t_final = 600;
pontos = 0:0.05:t_final;
evol = Euler_method(f_RJ(-0.02,0.05,-0.03,-0.01),...
                              0,t_final,Y0,t_final*20);
evolRomeu = evol(1,:); evolJulieta = evol(2,:);
figure();
hold on;
plot(pontos, evolRomeu, 'Color', 'blue');
plot(pontos, evolJulieta, 'Color', 'red');
legend('Romeu', 'Julieta');
xlabel('tempo'); ylabel('sentimentos')
```

Nota: aqui os valores de a, ...,d referem-se ao Caso 1.

A execução do script anterior produziu o seguinte output:

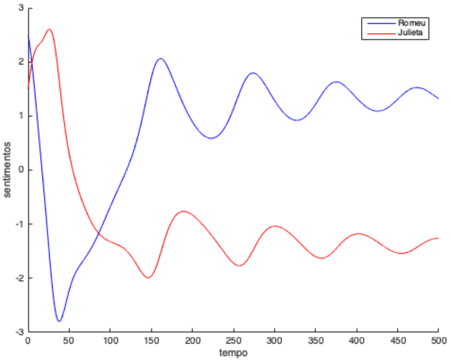

Figura: Simulação do Caso 1.

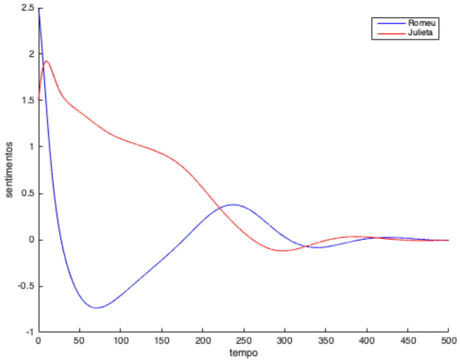

Figura: Caso 2. Convergência (quando $t \to \infty$) para equilíbrio $(R, J) = (0, 0)$.

Note-se que $(R, J) = (0, 0)$ é uma solução estacionária do sistema, mais concretamente, $(R, J) = (0, 0)$ é a solução trivial de

$$\begin{cases} aR + bJ(1 - |J|) = 0, \\ cR(1 - |R|) + dJ = 0. \end{cases}$$

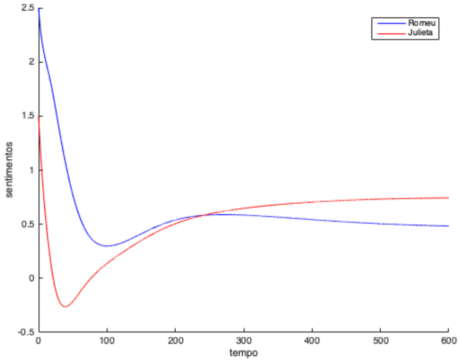

Figura: Caso 3. Convergência (quando $t \to \infty$) para $(R, J) = (0.471732234954834, 0.747602800378054)$.

Aqui temos a situação de equilíbrio que é outra solução do problema estacionário

$$\begin{cases} aR + bJ(1 - |J|) = 0, \\ cR(1 - |R|) + dJ = 0. \end{cases}$$

Para os valores dos parâmetros fornecidos, tem-se

$$\begin{cases} -2R + 5J(1 - |J|) = 0, \\ 3R(1 - |R|) - J = 0. \end{cases}$$

$$\Leftrightarrow \begin{cases} -2R + 15R(1 - |R|)(1 - 3|R(1 - |R|)|) = 0, \\ J = 3R(1 - |R|). \end{cases}$$

A equação

$$-2R + 15R(1 - |R|)(1 - 3|R(1 - |R|)|) = 0$$

além da solução trivial $R = 0$, tem uma solução no intervalo $(0.3, 1)$ a qual pode ser aproximada pelo método da bisseção, Newton, quasi-Newton, etc. Obtém-se $R = 0.471732234954834$ e depois usa-se a equação $J = 3R(1 - |R|)$ para obter $J = 0.747602800378054$.

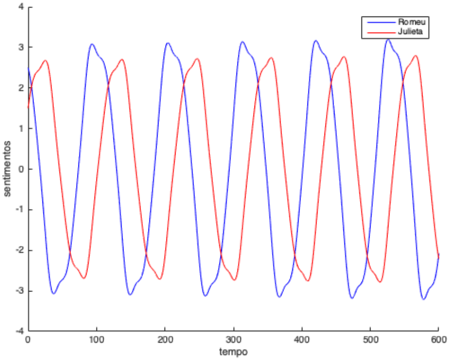

Figura: Caso 4. A solução apresenta um comportamento periódico, traduzindo sentimentos que alternam e se repetem com a mesma frequência ao longo do tempo...