

Vitória Menino Campos

Agenor de Toledo Fleury

PME 3380 - Modelagem de Sistemas Dinâmicos

02 de setembro de 2020

LISTA B

EXERCÍCIO 1:

Foi implementado um programa no Scilab que resolve a equação diferencial que modela o sistema de um tanque. Para o desenvolvimento do exercício foi utilizado um tempo que permite observar a estabilidade do nível do reservatório. Os gráficos a seguir mostram a solução numérica pelo método de Euler e pelo método de Runge-Kutta.

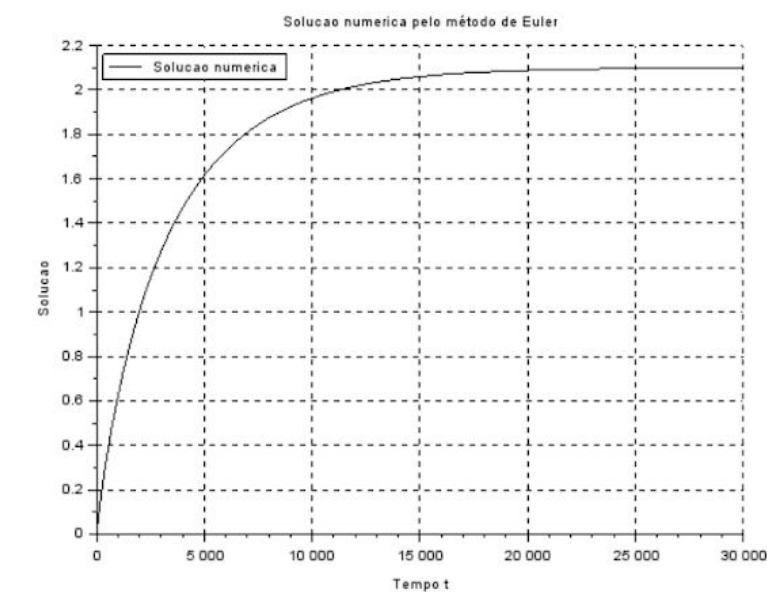


Figura 1: Gráfico solução numérica pelo método de Euler um tanque

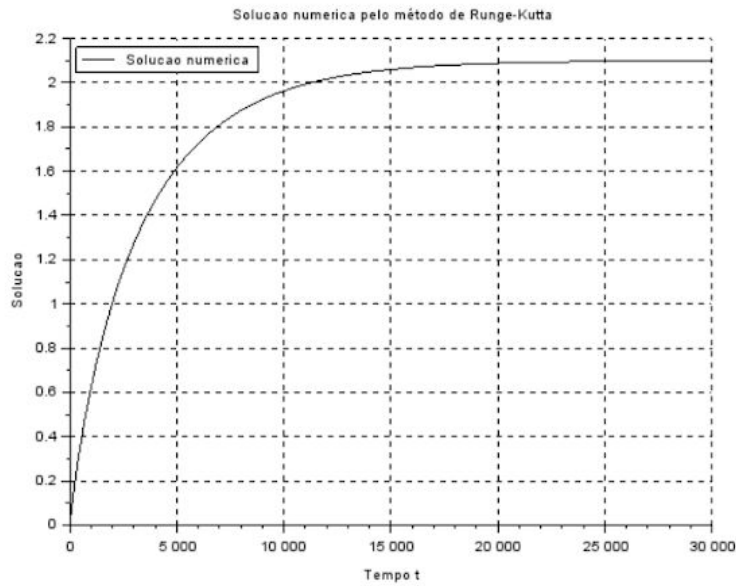


Figura 2: Gráfico solução numérica pelo método de Runge Kutta um tanque

A solução exata não é conhecida e nem pode ser comparada nos gráficos, pois não conseguimos isolar $y(t)$ na seguinte equação solução:

$$c_1 - \frac{t}{S} = \frac{2 \left(Q \log \left(Q - \sqrt{C y(t)} \right) + \sqrt{C y(t)} \right)}{C}$$

EXERCÍCIO 2:

Foi implementado um programa no Scilab que resolve as equações diferenciais que modela o sistema de dois tanques. Para o desenvolvimento do exercício foi utilizado um tempo que permite observar a estabilidade do nível do reservatório, também, como não estava especificado no enunciado, as áreas das seções transversais S_1 e S_2 foram consideradas $10 m^2$ e $8 m^2$

respectivamente e os valores de R_a e R_b foram considerados iguais ao R do primeiro exercício, $2 \times 10^8 Pa/(m^3/s)^2$. Os gráficos a seguir mostram a solução numérica pelo método de Euler e pelo método de Runge-Kutta.

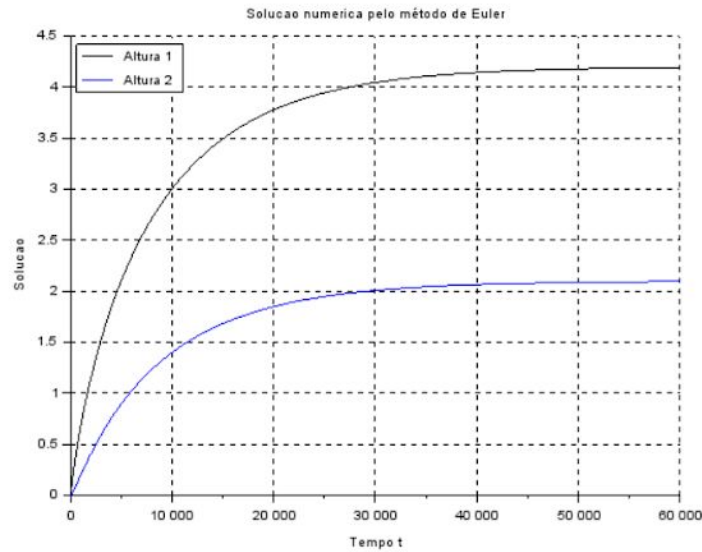


Figura 3: Gráfico solução numérica pelo método de Euler dois tanques

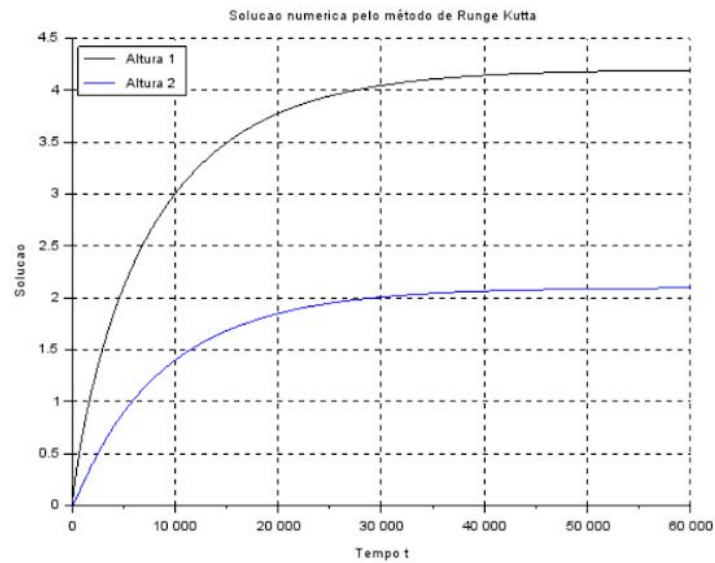


Figura 4: Gráfico solução numérica pelo método de Euler para dois tanques

CÓDIGOS:

- FUNÇÃO PARA EULER EXERCÍCIO 1:

```

1 function [ydot]=exel(y)
2     ydot=(-((C*y)^(1/2))+Q)/S;
3 endfunction
4 save('exel.sci','exel');
5

```

- EULER EXERCÍCIO 1:

```

1 //Conjunto de comandos para solucao numerica de equacao diferencial dada pela funcao funcao.sci
2 //Apagando dados anteriores:
3 clear
4 //Carregando a equacao diferencial:
5 //Carregue a função usando o comando Load do Scilab
6 load('exel.sci','exel');
7 //Instante inicial:
8 C=(1000*10)/(2*10^8);
9 Q=0.010247;
10 S=10;
11 t(1)=0;
12 //Instante final:
13 tf=30000;
14 //Condicao inicial:
15 y(1)=0;
16 //Valor inicial da solucao exata:
17 ye(1)=0;
18 //Passo de integracao (experimente alterar o passo):
19 h=0.5;
20 //Calculo de numero de passos:
21 n=round(tf/h);
22 //Integracao numerica usando o metodo de Euler:
23 //Comando for:
24 for i=1:n
25 //Vetor de tempo:
26     t(i+1)=t(i)+h;
27 //Solucao numerica:
28     y(i+1)=y(i)+h*exel(y(i));
29 //Termino do comando for:
30 end
31 //Desenhando outro grafico com linhas diferentes:
32 plot2d([t],[y],[1 2]);
33 //Usando a variavel do tipo 'lista':
34 T=list("Solucao numerica pelo método de Runge-Kutta","Tempo t","Solucao","Solucao numerica");
35 //Colocando uma legenda na parte superior esquerda da figura (parametro 2):
36 legends(T(4),[1,2],2);
37 //Colocando um titulo na figura e nomeando os eixos:
38 xtitle(T(1),T(2),T(3));
39 xgrid(1)
40

```

- RUNGE KUTTA EXERCÍCIO 1:

```

3 clear
4 //variáveis:
5 C=(1000*10)/(2*10^8);
6 Q=0.010247;
7 S=10;
8 //Instante inicial:
9 t(1)=0;
10 //Instante final:
11 tf=30000;
12 //Condição inicial:
13 y(1)=0;
14 //Valor inicial da solução exata:
15 ye(1)=0;
16 //Passo de integração (experimente alterar o passo):
17 h=0.5;
18 //Cálculo de número de passos:
19 n=round((tf-t(1))/h);
20 //Integração numérica usando o método de Runge-Kutta:
21 //Comando for:
22 for i=1:n
23 //Vetor de tempo:
24 ... t(i+1)=t(i)+h;
25 //Solução numérica:
26 ... k1=h*(-(C*(y(i)))^(1/2)+Q)/S; ...
27 ... k2=h*(-(C*(y(i)+k1/2))^(1/2)+Q)/S; ...
28 ... k3=h*(-(C*(y(i)+k2/2))^(1/2)+Q)/S; ...
29 ... k4=h*(-(C*(y(i)+k3))^(1/2)+Q)/S; ...
30 ... y(i+1)=y(i)+((k1+2*k2+2*k3+k4)/6);
31 //Termino do comando for:
32 end
33 //Desenhando outro gráfico com linhas diferentes:
34 plot2d(t,y);
35 //Usando a variável do tipo 'lista':
36 T=list("Solução numérica pelo método de Euler","Tempo t","Solução","Solução numérica");
37 //Diminuindo a espessura das linhas:
38 xset('thickness',1)
39 //Colocando uma legenda na parte superior esquerda da figura (parametro 2):
40 legends(T(4),[1,2],2);
41 //Colocando um título na figura e nomeando os eixos:
42 xtitle(T(1),T(2),T(3));
43 //Colocando uma grade no gráfico:
44 xgrid(1)

```

- FUNÇÃO PARA EULER EXERCÍCIO 2:

```

1 function[hdot]=exe2A(h1,h2)
2 ... hdot=(Q-(C*(h1-h2))^(1/2))/S1;
3 endfunction
4 save('exe2A.sci','exe2A');
5
6 function[hdot]=exe2B(h1,h2)
7 ... hdot=((C*(h1-h2))^(1/2)-(W*h2)^(1/2))/S2;
8 endfunction
9 save('exe2B.sci','exe2B');
10
11

```

- EULER PARA EXERCÍCIO 2:

```

1 //Conjunto de comandos para solucao numerica de equacao diferencial dada pela funcao funcao.sci
2 //Apagando dados anteriores:
3 clear
4 load('exe2A.sci','exe2A');
5 load('exe2B.sci','exe2B');
6 //variáveis:
7 S1=10;
8 S2=8;
9 p=1000;
10 g=10;
11 Ra=2*10^8;
12 Rs=3*10^8;
13 Q=0.010247;
14 C=p*g/Ra
15 W=p*g/Rs
16 //Instante inicial:
17 t(1)=0;
18 //Instante final:
19 tf=60000;
20 //Condicao inicial:
21 h1(1)=0;
22 h2(1)=0;
23 ye(1)=0;
24 P=1;
25 n=round(tf/P);
26 for i=1:n
27 //Vetor de tempo:
28 ... t(i+1)=t(i)+P;
29 ... h1(i+1)=h1(i)+P*exe2A(h1(i),h2(i));
30 ... h2(i+1)=h2(i)+P*exe2B(h1(i),h2(i));
31 end
32 //Desenhando outro grafico com linhas diferentes:
33 plot2d([t],[h1,h2],[1,2]);
34 //Usando a variavel do tipo 'lista':
35 T=list("Solucao numerica pelo método de Euler","Tempo-t","Solucao","Altura-1","Altura-2");
36 //Colocando uma legenda na parte superior esquerda da figura (parametro 2):
37 legends([T(4),T(5)],[1,2],2);
38 //Colocando um titulo na figura e nomeando os eixos:
39 xtitle(T(1),T(3),T(3));
40 xgrid(1)
41

```

- RUNGE_KUTTA EXERCÍCIO 2:

```

3 clear
4 load('ene2A.sci','ene2A');
5 load('ene2B.sci','ene2B');
6 //variáveis:
7 S1=10
8 S2=8
9 p=1000
10 g=10
11 Ra=2*(10^8)
12 Rs=2*(10^8)
13 Q=0.010247
14 C=p*g/Ra;
15 W=p*g/Rs;
16 // Instante inicial:
17 t(1)=0;
18 // Instante final:
19 tf=50000;
20 // Condicao inicial:
21 h1(1)=0;
22 h2(1)=0;
23
24 // Passo de integracao (experimente alterar o passo):
25 l=0.5;
26 // Calculo de numero de passos:
27 n=round((tf-t(1))/l);
28 // Integracao numerica usando o metodo de Runge-Kutta:
29 // Comando for:
30 for i=1:n
31 // Vetor de tempo:
32 ... t(i+1)=t(i)+l;
33 // Solucao numerica:
34 ... k1=1*ene2A(h1(i),h2(i));
35 ... k2=1*ene2A(h1(i)+k1/2,h2(i)+k1/2);
36 ... k3=1*ene2A(h1(i)+k2/2,h2(i)+k2/2);
37 ... k4=1*ene2A(h1(i)+k3,h2(i)+k3);
38
39 ... p1=1*ene2B(h1(i),h2(i));
40 ... p2=1*ene2B(h1(i)+p1/2,h2(i)+p1/2);
41 ... p3=1*ene2B(h1(i)+p2/2,h2(i)+p2/2);
42 ... p4=1*ene2B(h1(i)+p3,h2(i)+p3);
43
44 ... h1(i+1)=h1(i)+((k1+2*k2+2*k3+k4)/6);
45 ... h2(i+1)=h2(i)+((k1+2*k2+2*k3+k4)/6);
46
47 // Termina do comando for:
48 end
49 // Desenhando outro grafico com linhas diferentes:
50 plot2d([t,t],[h1,h2],[1,2]);
51 // Usando a variavel do tipo 'lista':
52 T=list("Solucao numerica pelo metodo de Runge-Kutta", "Tempo t", "Solucao", "Solucao numerica");
53 // Diminuindo a espessura das linhas:
54 set('thickness',1)
55 // Colocando uma legenda na parte superior esquerda da figura (parametro 2):
56 legends(T(4),[1,2],2);
57 // Colocando um titulo na figura e nomeando os eixos:
58 title(T(1),T(2),T(3));
59 // Colocando uma grade no grafico:

```