Gabriel Jenner de Faria Orsi
Nº USP 10772800

Lista B - Modelagem de Sistemas Dinâmicos

Brasil

2020

Gabriel Jenner de Faria Orsi
Nº USP 10772800

Lista B - Modelagem de Sistemas Dinâmicos

Apresentação da Lista A da disciplina PME3380 - Modelagem de Sistemas Dinâmicos

Universidade de São Paulo

Escola Politécnica

PME3380 - Modelagem de Sistemas Dinâmicos

Orientador: Prof. Dr. Agenor de Toledo Fleury e Prof. Dr. Decio Crisol Donha

Brasil

2020

Lista de ilustrações

Sumário

1 Exemplo 1

1.1 Código

O primeiro exemplo trata da resolução de uma equação diferencial por integração numérica, utilizando o método de Euler. Segue o código da equação diferencial de $\dot{y}$ em função de y:

```
//função de y ponto
function [ydot]=funcao(y)
    ydot=(1-y)/2;
endfunction
```

Código da integração numérica e das plotagens feitas.

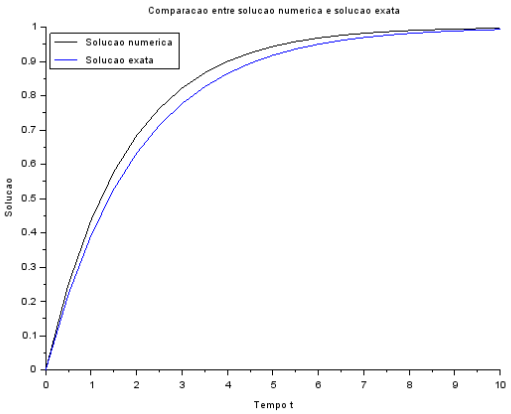
```
//Parametros
t(1)=0;// Instante inicial
tf=10;// Instante final
y(1)=0;// Condicao inicial
ye(1)=0;// Valor inicial da solucao exata
h=0.5;// Passo de integracao (experimente alterar o passo)
n=round(tf/h);// Calculo de numero de passos)

//integração numérica por Euler
for i=1:n
    t(i+1)=t(i)+h;
    y(i+1)=y(i)+h*funcao(y(i));
    ye(i+1)=1-%e^(-t(i+1)/2);
end

// Plotando solucao numerica y versus vetor de tempo t e solucao exata ye
   versus vetor de tempo t:
plot2d([t,t],[y,ye],[-1 -2]);
legends(["Solucao numerica","Solucao exata"],[-1,-2],4)// Colocando uma
   legenda na parte inferior direito da figura (parametro 4)
xtitle("Comparacao entre solucao numerica e solucao exata","Tempo t","
   Solucao")
// Colocando um titulo na figura e nomeando os eixos
set("current_figure",1);// Abrindo uma nova janela de graficos
plot2d([t,t],[y,ye],[1 2]);// Desenhando outro grafico com linhas
   diferentes
T=list("Comparacao entre solucao numerica e solucao exata","Tempo t","
   Solucao","Solucao numerica","Solucao exata");
legends([T(4),T(5)],[1,2],2);// Colocando uma legenda na parte superior
   esquerda da figura (parametro 2)
```

```
25 xtitle(T(1),T(2),T(3));// Colocando um titulo na figura e nomeando os eixos
```

1.2 Plotagens

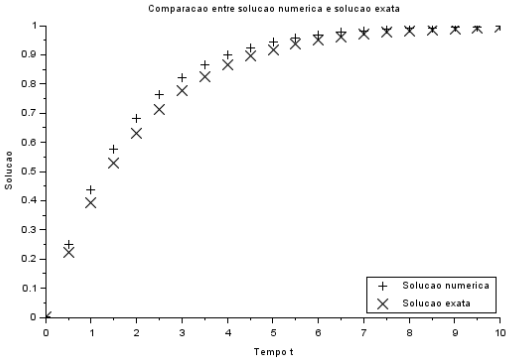
O código exposto anteriormente resultou em duas plotagens com tipos de traço diferentes. Os gráficos mostram a solução exata y_e em comparação com a solução numérica obtida pelo método de Euler - y.

Figura 1 – Função y ao longo do tempo - solução numérica (método de Euler) e exata



Fonte: autoria própria

Figura 2 – Função y ao longo do tempo - solução numérica (método de Euler) e exata



Fonte: autoria própria

2 Exemplo 2

O exemplo 2 trata da mesma equação diferencial do exemplo anterior, porém resolvida numericamente pelo método de Runge Kutta de 4ª ordem, e sem declaração da função previamente, como feito no exemplo 1, mas com a conta feita no próprio laço *for* da integração numérica.

2.1 Código

Segue o código da integração por Runge Kutta e das plotagens do exemplo 2.

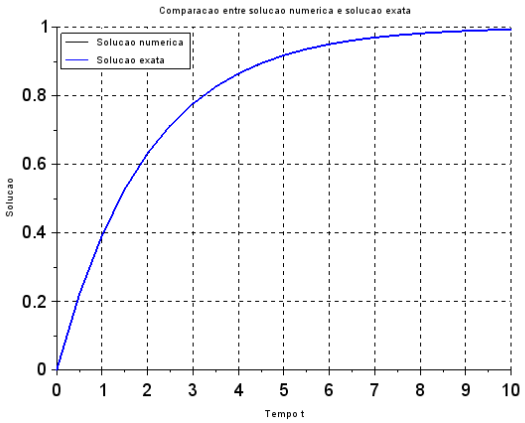
```
clc()
clear()

t(1)=0;// Instante inicial
tf=10;// Instante final
y(1)=0;// Condicao inicial
ye(1)=0;// Valor inicial da solucao exata:
h=0.5;// Passo de integracao (experimente alterar o passo):
n=round((tf-t(1))/h);// Calculo de numero de passos):

// Integracao numerica usando o metodo de Runge Kutta:
for i=1:n
    t(i+1)=t(i)+h;
    k1=h*(1-(y(i)))/2;
    k2=h*(1-(y(i)+k1/2))/2;
    k3=h*(1-(y(i)+k2/2))/2;
    k4=h*(1-(y(i)+k3))/2;
    y(i+1)=y(i)+((k1+2*k2+2*k3+k4)/6);
    ye(i+1)=1-%e^(-t(i+1)/2);// Solucao exata
end
// Plotando solucao numerica y versus vetor de tempo t e solucao exata ye
    versus vetor de tempo t:
plot2d([t,t],[y,ye],[-1 -2]);
// Colocando uma legenda na parte inferior direito da figura (parametro 4):
legends(["Solucao numerica","Solucao exata"],[-1,-2],4)
// Colocando um titulo na figura e nomeando os eixos:
xtitle("Comparacao entre solucao numerica e solucao exata","Tempo t","
    Solucao")
// Abrindo uma nova janela de graficos:
set("current_figure", 1);
// Aumentando a espessura das linhas:
xset("thickness",2)
// Aumentando o tamanho da fonte:
```

```
xset('font size',4)
// Desenhando outro grafico com linhas diferentes:
plot2d([t,t],[y,ye],[1 2]);
// Usando a variavel do tipo 'lista':
T=list("Comparacao entre solucao numerica e solucao exata","Tempo t","
    Solucao","Solucao numerica","Solucao exata");
// Diminuindo a espessura das linhas:
xset('thickness',1)
// Colocando uma legenda na parte superior esquerda da figura (parametro 2)
    :
legends([T(4),T(5)],[1,2],2);
// Colocando um titulo na figura e nomeando os eixos:
xtitle(T(1),T(2),T(3));
// Colocando uma grade no grafico:
xgrid(1)
```

2.2 Plotagens

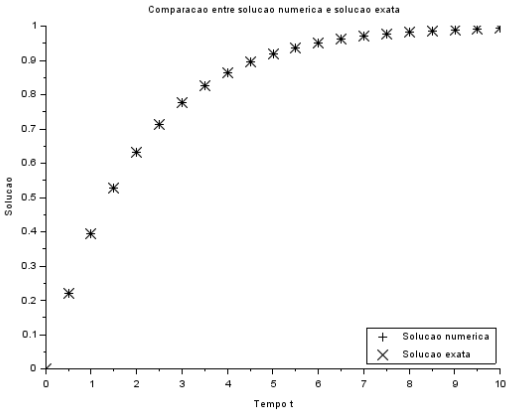
As plotagens são análogas ao exemplo 1, dois tipos de linhas diferentes em gráficos que comparam a solução exata com a numérica. Nota-se que o método de Runge Kutta é consideravelmente mais preciso que o Euler para este caso.

Figura 3 – Função y ao longo do tempo - solução numérica (método de Runge Kutta) e exata



Fonte: autoria própria

Figura 4 – Função y ao longo do tempo - solução numérica (método de Runge Kutta) e exata



Fonte: autoria própria

3 Exercícios

Segue os exercícios e suas resoluções por Runge Kutta e Euler.

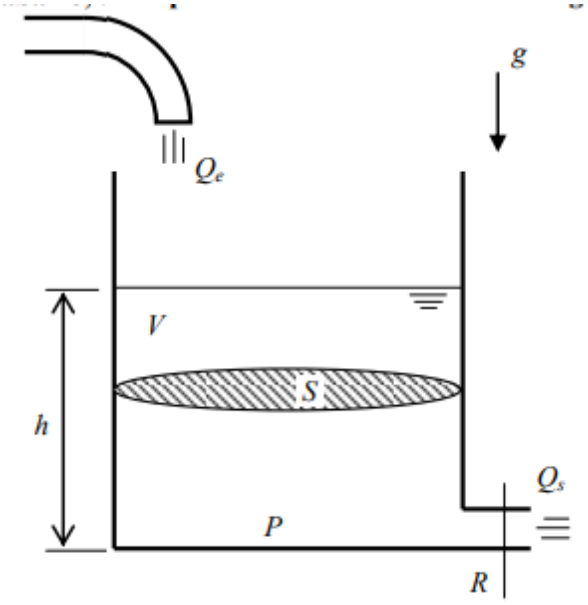
3.1 1 reservatório

O problema consiste num reservatório com vazões de saída e de entrada. Os parâmetros que foram usados para modelar são os seguintes:

- $S = 10m^2$ - área da seção transversal (constante)
- $R = 2 * 108Pa/(m^3/s)^2$ - parâmetro que relaciona vazão com queda de pressão (perda de carga)
- $\rho = 1000kg/m^3$ - massa específica da água
- $G = 10m/s^2$ - aceleração da gravidade na superfície da terra
- $Q_e = 0,010247m^3/s$ - vazão de entrada
- $h(0) = 8m$ - altura inicial estimada do nível de água do tanque (condição de contorno)

O esboço do problema está ilustrado na figura 5:

Figura 5 – Problema de 1 reservatório com vazão de entrada e saída



A equação que modela a variação da altura do reservatório em relação ao tempo $\dot{h}$ é:

$$\dot{h} = \left(-\sqrt{\frac{\rho G h}{R}} + Q_e \right) \frac{1}{S} \tag{3.1}$$

3.1.1 Código

Segue o código que modela a função $\dot{h}$:

```
//função de h ponto
function [hdot]=funcao(h)
    hdot = (-((rho*G*h)/R)^0.5 + Qe)*(1/S)
endfunction
```

Segue o código completo da integração por Euler e por Runge Kutta do problema de 1 reservatório.

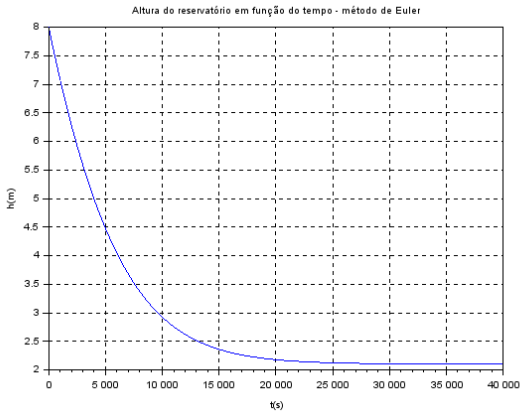
```
clc()
clear()

//Parametros
S = 10 //área da seção transversal (m^2)
R = 2*(10^8)//par metro que relaciona vazão com queda de pressão (perda de
    carga)
rho = 1000//- massa específica da água (kg/m^3)
G = 10//aceleração da gravidade na superfície da terra (m/s^2)
Qe = 0.010247//vazão de entrada (m^3/s)

//Condição inicial
he(1) = 8 //altura inicial estimada (m)
hr(1) = 8
t(1) = 0 //intante inicial de tempo
tf = 40000 //instante final de tempo
p = 0.1 //passo
n = round(tf/p)// Calculo de numero de passos)

//integração por Euler
for i=1:n
    t(i+1)=t(i)+p;
    he(i+1)=he(i)+p*funcao(he(i)); //h ponto por Euler
end

//integração por Runge Kutta de 4 ordem
for i=1:n
    t(i+1)=t(i)+p;
    k1=p*funcao(hr(i));
    k2=p*funcao(hr(i)+k1/2);
```

```
30      k3=p*funcao(hr(i)+k2/2);
31      k4=p*funcao(hr(i)+k3);
32      hr(i+1)=hr(i)+((k1+2*k2+2*k3+k4)/6); //h ponto por Runge Kutta
33  end
34
35  f1 = scf(1)
36  plot(t,he,'b')
37  xtitle('Altura do reservatório em função do tempo - método de Euler', 't(s)
        ','h(m)')
38  xgrid(1)
39
40  f2 = scf(2)
41  plot(t,hr,'r')
42  xtitle('Altura do reservatório em função do tempo - método de Runge Kutta',
        't(s)','h(m)')
43  xgrid(1)
44
45  f3 = scf(3)
46  plot(t,(he-hr),'g')
47  xtitle('Diferença das soluções de Euler e Runge Kutta ponto a ponto', 't(s)
        ', 'Diferença entre soluções(m)')
48  xgrid(1)
```

3.1.2 Plotagens

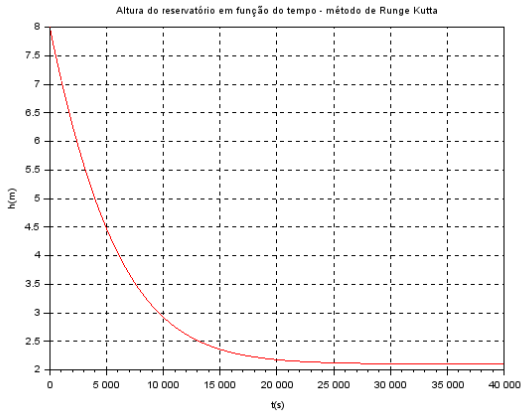
O código exposto anteriormente gerou gráficos da solução do problema por método de Euler (figura 6) e de Runge Kutta (figura 7), além de mostrar também a diferença entre os métodos, ponto a ponto (figura 8).

Figura 6 – Altura do reservatório em função do tempo - método de Euler



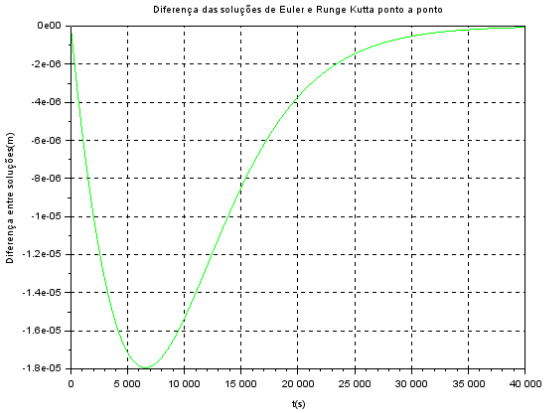
Fonte: autoria própria

Figura 7 – Altura do reservatório em função do tempo - método de Runge Kutta



Fonte: autoria própria

Figura 8 – Diferença entre métodos de integração por instante de tempo



Fonte: autoria própria

É possível perceber que a diferença entre os métodos para este caso é muito pequena, da ordem de 10^{-6}.

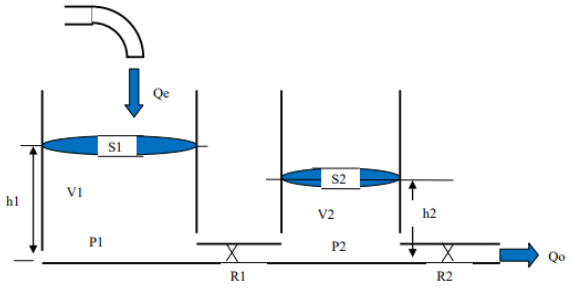
3.2 2 reservatórios

Neste problema há 2 reservatórios ligados por um duto, com uma vazão de entrada e uma de saída como mostra a figura 9. Os parâmetros do tanque 1 são os mesmos do tanque do problema de 1 reservatório (sendo $h1(0) = 8m$ a altura inicial do tanque 1 e $R_a = R$). Os parâmetros do tanque 2 são os seguintes:

- $S_2 = 8m^2$ - área da seção transversal do tanque 2

- $R_s = 1.5 * 10^8$ - parâmetro que relaciona vazão com queda de pressão (perda de carga) do tanque 2

Figura 9 – Problema de 2 reservatórios com vazão de entrada e saída



A densidade, aceleração da gravidade e vazão de entrada são iguais aos adotados no problema de 1 reservatório.

A equação que modela o problema é a seguinte:

$$\dot{h}_1 = \left[-\sqrt{\frac{\rho G \left(h_1 - h_2\right)}{R_a}} + Q_e \right] \frac{1}{S_1} \tag{3.2}$$

$$\dot{h}_2 = \left[\sqrt{\frac{\rho G \left(h_1 - h_2\right)}{R_a}} - \sqrt{\frac{\rho G h_2}{R_s}} \right] \frac{1}{S_2} \tag{3.3}$$

Sendo h_1 a altura do primeiro tanque e h_2 a do segundo.

3.2.1 Código

Segue o código das funções de $\dot{h}_1$ e $\dot{h}_2$ (equações 3.2 e 3.3)

```
//função de h ponto do tanque 1
function [h1dot]=funcao(h1,h2)
    h1dot = (-((rho*G*(h1-h2))/Ra)^0.5 + Qe)*(1/S1)
endfunction

//função de h ponto do tanque 2
function [h2dot]=funcao2(h1,h2)
    h2dot = (((rho*G*(h1-h2))/Ra)^0.5 - (rho*G*h2/Rs)^0.5)*(1/S2)
endfunction
```

Segue o código da integração e plotagens.

```
clc()
clear()

```

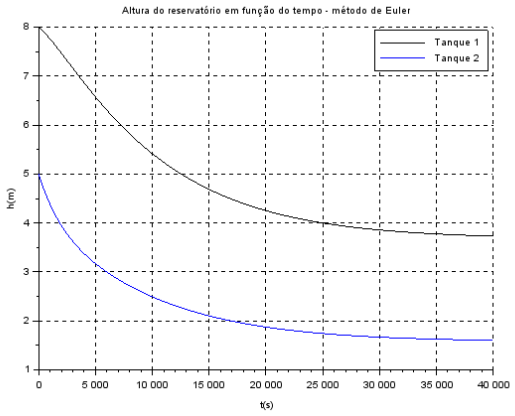
```
G = 10//aceleração da gravidade na superfície da terra (m/s^2)
rho = 1000//- massa específica da água (kg/m^3)
Qe = 0.010247//vazão de entrada (m^3/s)

//Par metros tanque 1
S1 = 10 //área da seção transversal (m^2)
Ra = 2*(10^8)//par metro que relaciona vazão com queda de pressão (perda
    de carga)

//Par metros tanque 2
S2 = 8 //área da seção transversal (m^2)
Rs = 1.5*(10^8)//par metro que relaciona vazão com queda de pressão (perda
     de carga)


//Condição inicial
h1e(1) = 8 //altura inicial estimada do tanque 1 (m)
h1r(1) = 8 //altura inicial estimada do tanque 1 (m)
h2e(1) = 5 //altura inicial estimada do tanque 2 (m)
h2r(1) = 5 //altura inicial estimada do tanque 2 (m)
t(1) = 0 //intante inicial de tempo
tf = 40000 //instante final de tempo
p = 0.1 //passo
n = round(tf/p)// Calculo de numero de passos)

//integração por Euler
for i=1:n
    t(i+1)=t(i)+p;
    h1e(i+1)=h1e(i)+p*funcao(h1e(i),h2e(i)); //h ponto por Euler do tanque
    1
    h2e(i+1)=h2e(i)+p*funcao2(h1e(i),h2e(i)); //h ponto por Euler do tanque
     2
end

//integração por Runge Kutta de 4 ordem
for i=1:n
    t(i+1)=t(i)+p;
    k1=p*funcao(h1r(i),h2r(i));
    k2=p*funcao(h1r(i)+k1/2,h2r(i)+k1/2);
    k3=p*funcao(h1r(i)+k2/2,h2r(i)+k2/2);
    k4=p*funcao(h1r(i)+k3,h2r(i)+k3);

    m1=p*funcao2(h1r(i),h2r(i));
    m2=p*funcao2(h1r(i)+m1/2,h2r(i)+m1/2);
    m3=p*funcao2(h1r(i)+m2/2,h2r(i)+m2/2);
    m4=p*funcao2(h1r(i)+m3,h2r(i)+m3);

```

```
47      h1r(i+1)=h1r(i)+((k1+2*k2+2*k3+k4)/6); //h ponto do tanque 1 por Runge
    Kutta
48      h2r(i+1)=h2r(i)+((m1+2*m2+2*m3+m4)/6); //h ponto do tanque 2 por Runge
    Kutta
49  end
50
51  f1 = scf(1)
52  plot2d([t,t],[h1e,h2e],[1  2])
53  xtitle('Altura do reservatório em função do tempo - método de Euler', 't(s)
    ','h(m)')
54  legend(["Tanque 1","Tanque 2"],[1  2],1)
55  xgrid(1)
56
57  f2 = scf(2)
58  plot([t,t],[h1r,h2r],[1  2])
59  xtitle('Altura do reservatório em função do tempo - método de Runge Kutta',
     't(s)','h(m)')
60  legend(["Tanque 1","Tanque 2"],[1  2],1)
61  xgrid(1)
```

3.2.2 Plotagens

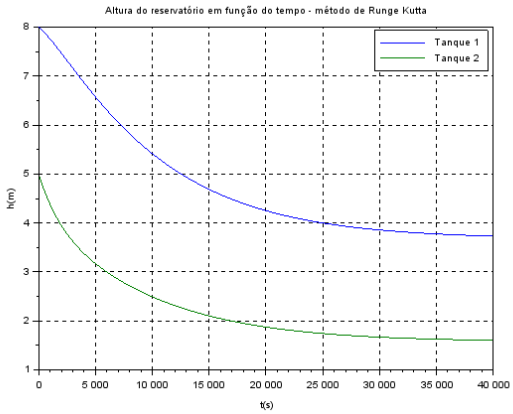
Segue os gráficos gerados da solução numérica de Euler e de Runge Kutta.

Figura 10 – Altura dos reservatórios em função do tempo - método de Euler



Fonte: autoria própria

Figura 11 – Altura dos reservatórios em função do tempo - método de Runge Kutta



Fonte: autoria própria