Universidade de São Paulo

Escola Politécnica

PME 3380 – Modelagem de Sistemas Dinâmicos

Relatório- Lista B

Paulo Montijo Bandeira – 9348449

São Paulo, 02 de setembro de 2020.

1) Simulação de um tanque com uma entrada e uma saída

Dados do problema:

S = 10 m² - área da seção transversal

R = $2 \times 10^8 \frac{Pa}{(m^3/s)^2}$ – perda de carga

$\rho = 1000 \frac{kg}{m^3}$ – massa específica da água

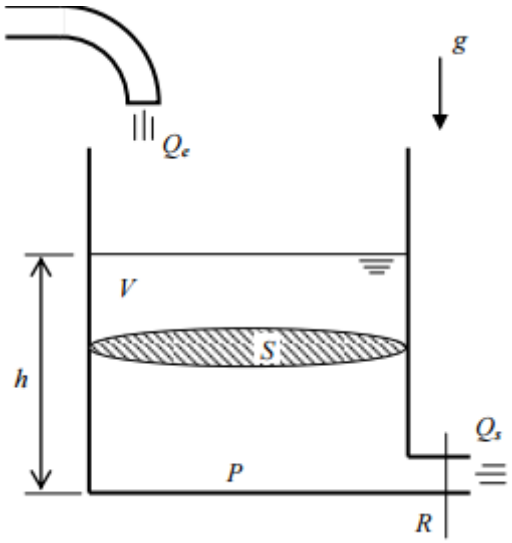
G = $10 \frac{m}{s^2}$ – aceleração da gravidade na superfície da Terra

$Q_e = 0{,}010247 \frac{m^3}{s}$ – vazão de entrada

Admitindo que a água seja um fluido incompressível, utilizando a equação da continuidade e a equação de perda de carga, obtém-se a seguinte equação diferencial ordinária não-linear:

$$\dot{h} = (-\sqrt{\frac{\rho g h}{R}} + Q_e)\frac{1}{S}$$

Figura 1 – Modelo com um reservatório



a) Método de Euler

Código:

```
// Limpeza do console e variáveis
Clear
clc
// Parâmetros:
rho= 1000;
g= 10;
R= 2e8;
S= 0.2;
//Variáveis:
Qe=  0.0010247;
// Instante inicial:
t(1)=0;
// Instante final:
tf=400;
// Condicao inicial:
y(1)=0.5;
// Passo de integração:
h=0.01;
// Calculo de numero de passos):
n=round(tf/h);
// Integracao numerica usando o metodo de Euler:
// Comando for:
function y_ponto=funcao(y)
    y_ponto=(-sqrt(rho*g*y/R)+Qe)/S
endfunction
for i=1:n
// Vetor de tempo:
t(i+1)=t(i)+h;
// Solucao numerica:
y(i+1)=y(i)+h* funcao(y(i));
// Termino do comando for:
end

// Desenhando outro grafico com linhas diferentes:
plot2d(t, y, [5], leg = "h1 (m)")
a = gca()
p1 = a.children(1).children(1);
p1.thickness = 2;
leg = a.children(2)
leg.line_mode = "on"
set(a, "sub_tics", [5 5])
xtitle('Altura y do reservatório vs tempo', 'Tempo (s)', 'Altura (m)')
xgrid()
```

Gráfico:

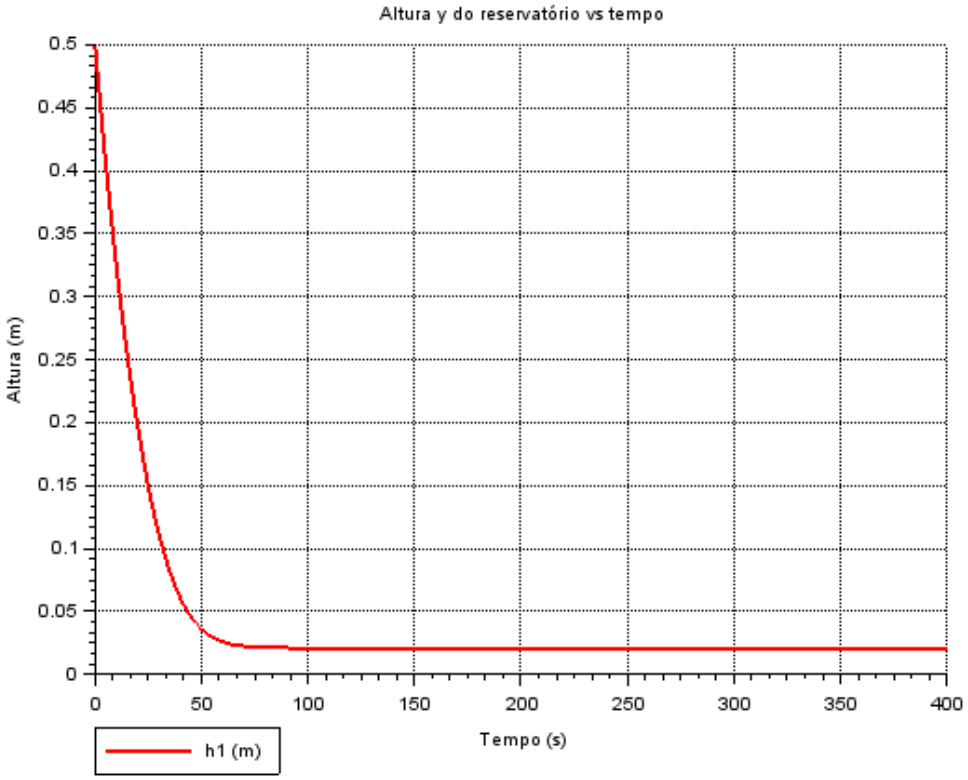


Gráfico 1 – Altura da coluna d'água de um reservatório pelo método de Euler

b) Método de Runge Kutta

Código:

```
clc
clear

ti = 0; tf = 400; h = 0.01
t = [ti:h:tf]
hr = zeros(t);

// parâmetros do problema
S = 0.2;
R = 2D8;
ro = 1000;
g = 10;
Qe = 0.0010247;
C = ro*g/R

// valor inicial
hr(1) = 0.5;

// aplicação do Runge-Kutta:
for i = 1:(length(t) - 1)
k1 = (h/(2*S))*(Qe - sqrt((C*(hr(i)))));
k2 = (h/(2*S))*(Qe - sqrt((C*((hr(i) + k1/2)))));
k3 = (h/(2*S))*(Qe - sqrt((C*((hr(i) + k2/2)))));
k4 = (h/(2*S))*(Qe - sqrt((C*((hr(i) + k3)))));
hr(i + 1) = hr(i) + (k1 + 2*k2 + 2*k3 + k4)/6; // solução numérica de h1
end
```

```
plot2d(t, hr, [5], leg = "h1 (m)")
a = gca()
p1 = a.children(1).children(1);
p1.thickness = 2;
leg = a.children(2)
leg.line_mode = "on"
set(a, "sub_tics", [5 5])
xtitle('Altura hr do reservatório vs tempo', 'Tempo (s)', 'Altura (m)')
xgrid()
```

Gráfico:

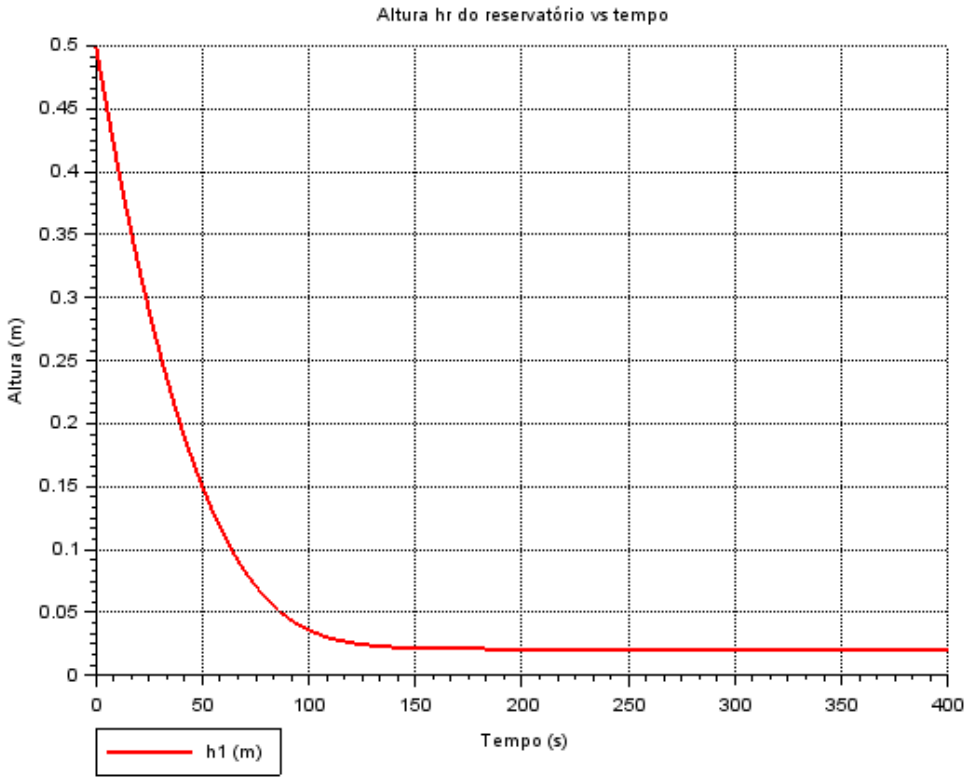


Gráfico 2 – Altura de coluna d'água de um reservatório pelo método de Runge Kutta

2) Simulação de dois tanques interligados, com uma entrada e uma saída

Modelo do sistema de 2 reservatórios (considerando a entrada constante e perdas de carga não lineares como no caso do ex. de 1 tanque):

$$\dot{h_1} = [Q_e - \sqrt{\frac{\rho g}{R_a}(h_1 - h_2)}]\frac{1}{S_1}$$

$$\dot{h_2} = [\sqrt{\frac{\rho g}{R_a}(h_1 - h_2)} - \sqrt{\frac{\rho g}{R_s}h_2}]\frac{1}{S_2}$$

a) Método de Euler

Código:

```
// Limpeza do console e variáveis;
Clear
clc
// Parâmetros:
rho= 1000;
g= 10;
Ra= 2e8;
Rb= 2e8;
Sa= 2;
Sb= 3;
//Variáveis:
Qe=  0.0010247;
// Instante inicial:
t(1)=0;
// Instante final:
tf=3600;
// Condicao inicial:
y(1,1)=1.5;
y(1,2)=1;
// Passo de integracao (experimente alterar o passo):
h=0.01;
// Calculo de numero de passos):
n=round(tf/h);
// Integracao numerica usando o metodo de Euler:
// Comando for:
function y_pontoA=funcaoA(y)
   y_pontoA=(Qe - sqrt(rho*g*(y(1)-y(2))/Ra))/Sa
endfunction
function y_pontoB=funcaoB(y)
   y_pontoB=(sqrt(rho*g*(y(1)-y(2))/Ra)-sqrt(rho*g*y(2)/Rb))/Sb
endfunction
for i=1:n
// Vetor de tempo:
t(i+1)=t(i)+h;
// Solucao numerica:

y(i+1,1)=y(i,1)+h* funcaoA(y(i,:));
y(i+1,2)=y(i,2)+h* funcaoB(y(i,:));
// Termino do comando for:
end

plot2d(t, [y(:,1)' y(:,2)'], [2 5], leg = "h1 (m) @ h2 (m)")
a = gca()
p1 = a.children(1).children(1);
p1.thickness = 2;
p2 = a.children(1).children(2);
p2.thickness = 2;
leg = a.children(2)
leg.line_mode = "on"
set(a, "sub_tics", [5 5])
xtitle('Alturas h1 e h2 dos reservatórios vs tempo', 'Tempo (s)', 'Altura (m)')
xgrid()
```

Gráfico:

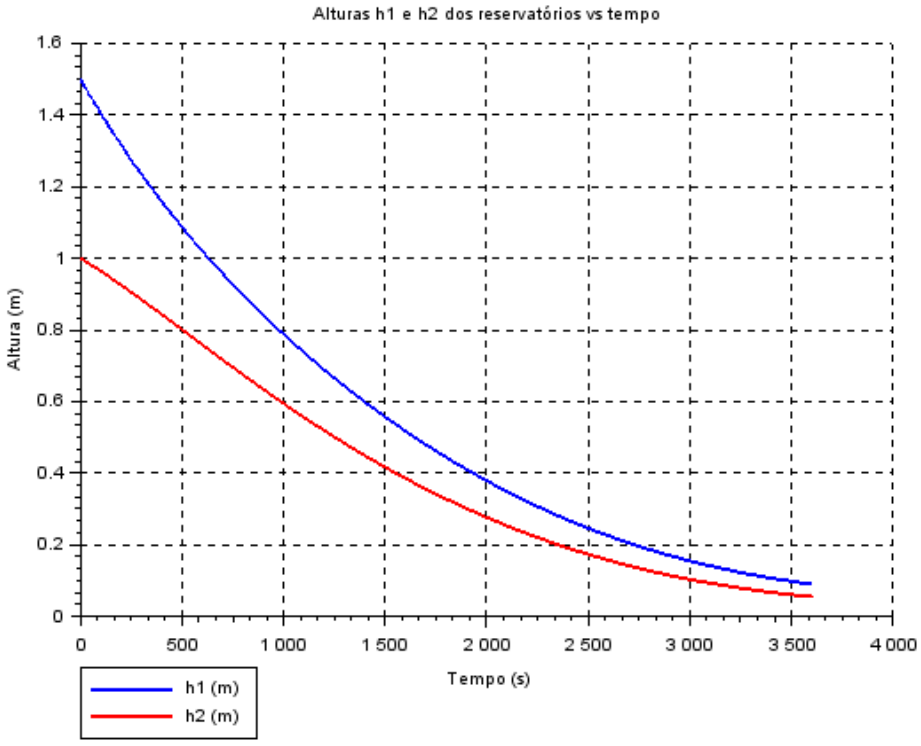


Gráfico 3 – Resolução de dois reservatórios pelo método de Euler

b) Método de Runge Kutta

Código:

```
//Limpeza do console e variáveis
clc
clear

//parâmetros de integração
ti = 0; tf = 3600; h = 0.01
t = [ti:h:tf]
h1 = zeros(t); h2 = zeros(t);

// parâmetros do problema
S1 = 2;
S2 = 3;
R = 2D8;
ro = 1000;
g = 10;
Qe = 0.0010247;
C = ro*g/R

// valores iniciais
h1(1) = 1.5; h2(1) = 1

// aplicação do Runge-Kutta:
for i = 1:(length(t) - 1)
k1 = (h/(2*S1))*(Qe - sqrt((C*(h1(i) - h2(i)))));
q1 = (h/(2*S2))*(sqrt((C*(h1(i) - h2(i)))) - sqrt(C*h2(i)));
```

```
k2 = (h/(2*S1))*(Qe - sqrt((C*((h1(i) + k1/2) - (h2(i) + q1/2)))));
q2 = (h/(2*S2))*(sqrt((C*((h1(i) + k1/2) - (h2(i) + q1/2)))) - sqrt(C*(h2(i) + q1/2)));
k3 = (h/(2*S1))*(Qe - sqrt((C*((h1(i) + k2/2) - (h2(i) + q2/2)))));
q3 = (h/(2*S2))*(sqrt((C*((h1(i) + k2/2) - (h2(i) + q2/2)))) - sqrt(C*(h2(i) + q2/2)));
k4 = (h/(2*S1))*(Qe - sqrt((C*((h1(i) + k3) - (h2(i) + q3)))));
q4 = (h/(2*S2))*(sqrt((C*((h1(i) + k3) - (h2(i) + q3/2)))) - sqrt(C*(h2(i) + q3)));
h1(i + 1) = h1(i) + (k1 + 2*k2 + 2*k3 + k4)/6; // solução numérica de h1
h2(i + 1) = h2(i) + (q1 + 2*q2 + 2*q3 + q4)/6; // solução numérica de h2
end

//plotagem do gra´fico
plot2d(t, [h1' h2'], [2 5], leg = "h1 (m) @ h2 (m)")
a = gca()
p1 = a.children(1).children(1);
p1.thickness = 2;
p2 = a.children(1).children(2);
p2.thickness = 2;
leg = a.children(2)
leg.line_mode = "on"
set(a, "sub_tics", [5 5])
xtitle('Alturas h1 e h2 dos reservatórios vs tempo', 'Tempo (s)', 'Altura (m)')
xgrid()
```

Gráfico:

Gráfico 4 – Resolução de dois reservatórios pelo método de Runge Kutta