Escola Politécnica da Universidade de São Paulo

PME3380 - Modelagem de Sistemas Dinâmicos

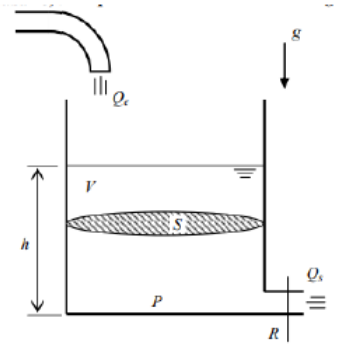
Lista B

Henrique Aquino - 10772543

São Paulo, 2020

Exercício 1: Reservatório Simples

Para esse exercício, modelou-se um reservatório simples que possui uma vazão de entrada, uma vazão de saída, e uma altura inicial que foi escolhida como a altura zero ($h(0) = 0m$).



Com o tanque inicialmente vazio, a modelagem se estendeu por tempo suficiente para que a estabilização da altura h fosse possível, com um passo de integração de $0.5s$.

Utilizando dois métodos numéricos, o Método de Euler e o Runge Kutta de 4ª Ordem, observamos resultados semelhantes: uma estabilização do nível do reservatório em torno de $h(t_{\to\infty}) = 2.1m$, que ocorre por volta de $t = 20.000s$.

Método de Euler

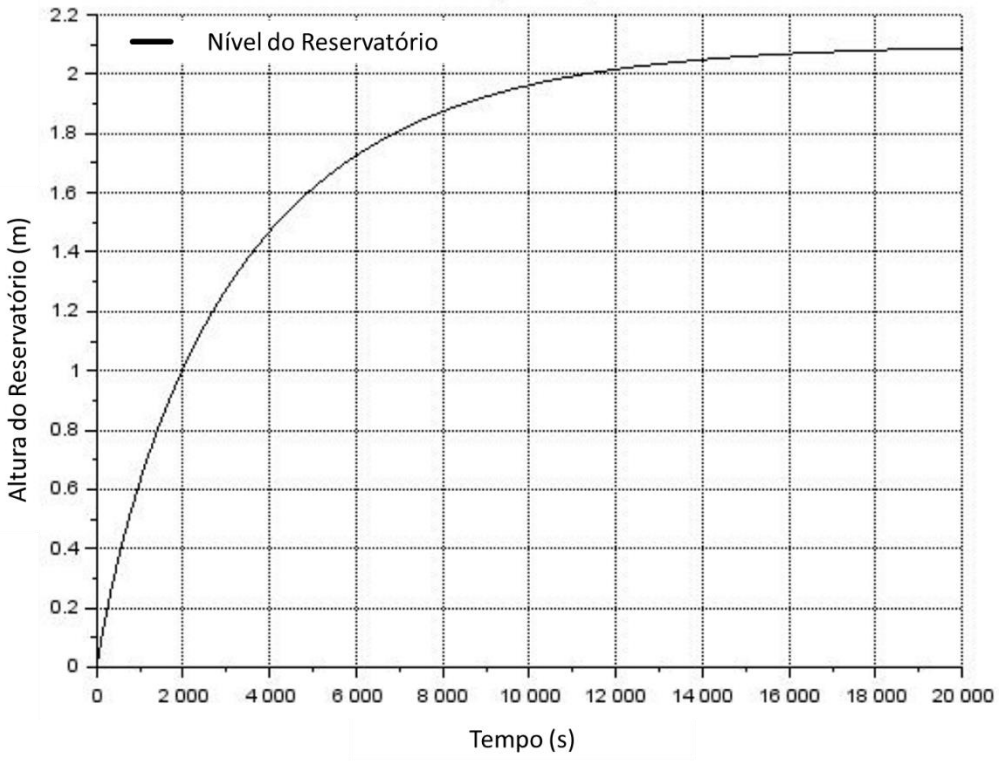


Figura 1: Simulação do Nível do Reservatório em Função do Tempo (Euler)

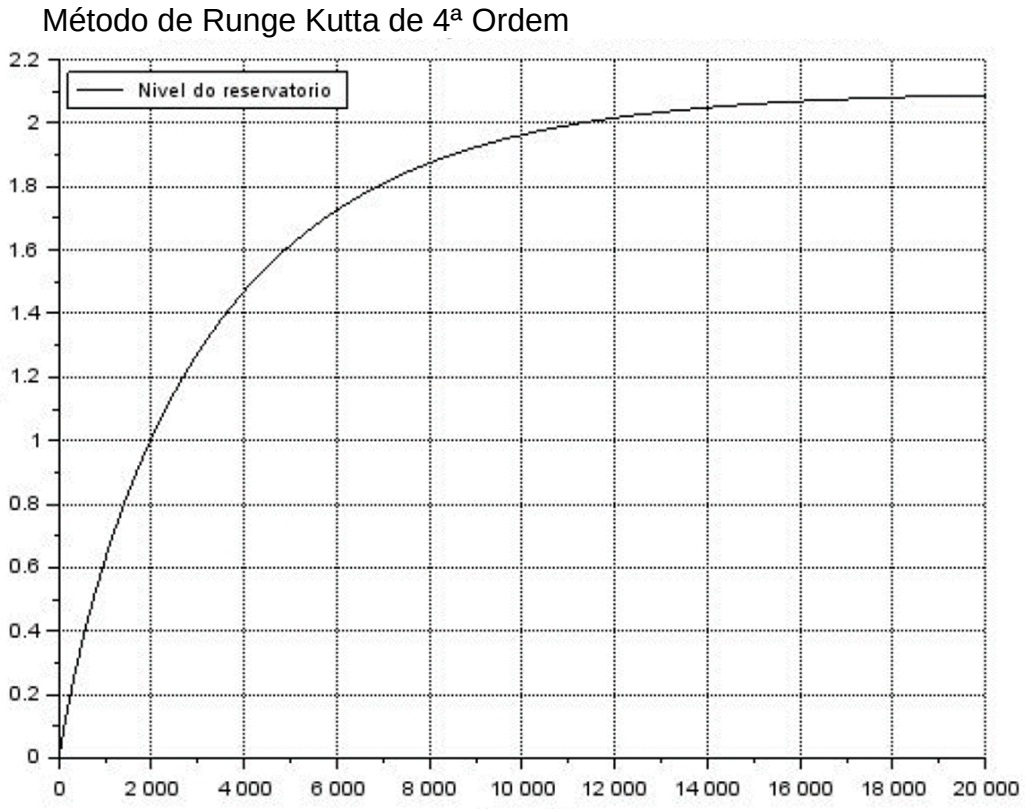
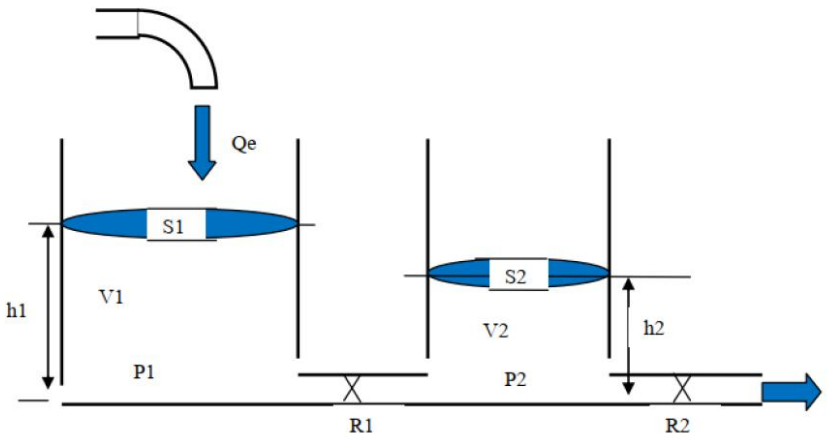


Figura 2: Nível do Reservatório em Função do Tempo (Runge Kutta 4)

Exercício 2: Dois Reservatórios

Para esse exercício, modelaram-se dois reservatórios, um alimentando o outro, de maneira que exista uma vazão de entrada, uma vazão de transferência entre os dois, e uma vazão de saída. Assim, espera-se que haja uma estabilização do nível dos reservatórios, de maneira semelhante ao observado no exercício anterior. A altura inicial para ambos os reservatórios foi escolhida como a altura zero ($h_1(0) = 0m,\ h_2(0) = 0m$). A simulação ocorreu durante $t = 20.000s$ com passo de integração de $0.5s$.



Novamente os dois métodos numéricos a serem empregados para a solução do problema são: Método de Euler e Runge Kutta de 4ª Ordem.

Vemos uma estabilização do nível do Reservatorio 1 (superior) nos mesmos valores encontrados anteriormente: $h_1(t_{\to\infty}) = 2.1m$. O Reservatório 2 (inferior) se estabiliza em um nível de $h_2(t_{\to\infty}) = 4.2m$.

Método de Euler

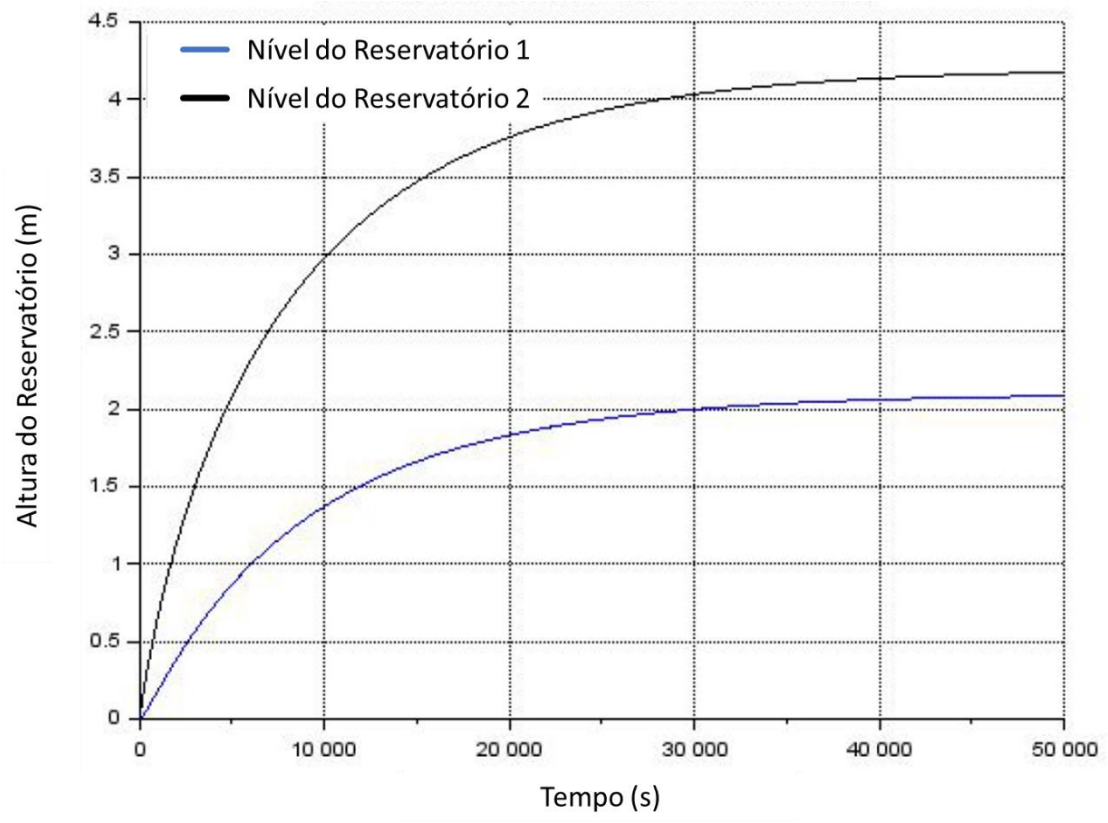


Figura 3: Simulação do Nível dos Reservatórios em Função do Tempo (Euler)

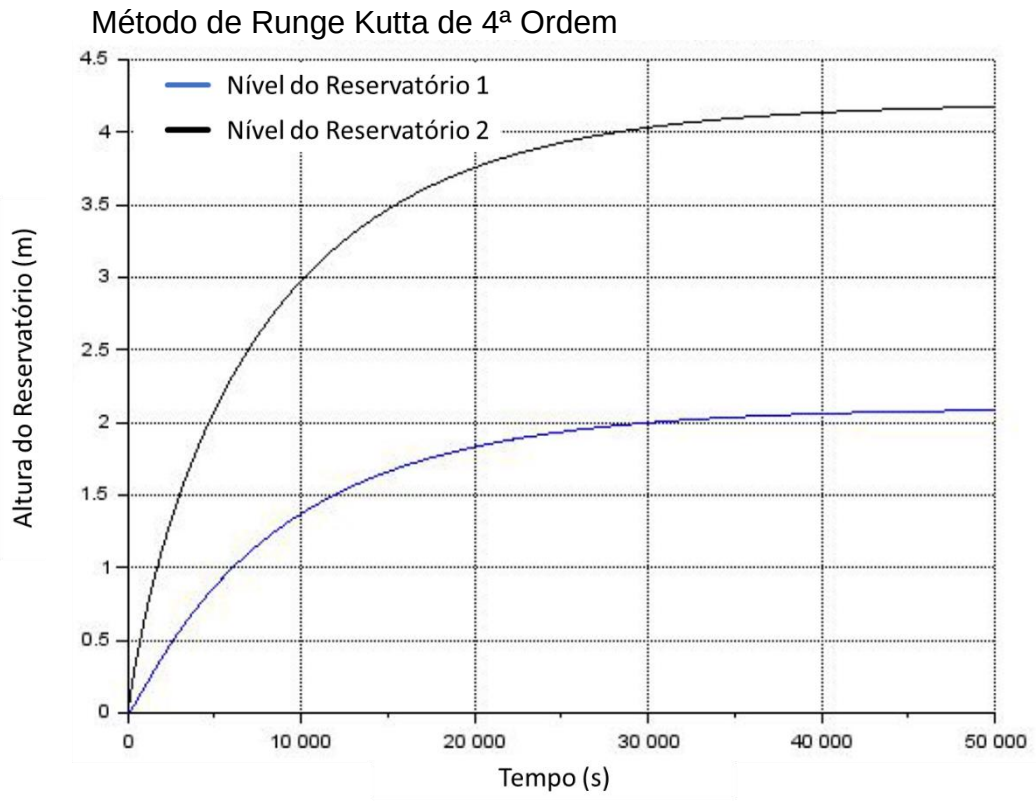


Figura 4: Nível dos Reservatórios em Função do Tempo (Runge Kutta 4)

Códigos para a Modelagem

Exercício 1:

Função Euler:

```
R = 2_10^8
Qe = 0.010247
r = 1000
S = 10
g = 10

function [ hponto ] = altura (h)
hponto = (−sqrt ( r*g*h/R) + Qe) /S ;
endfunction

t(1) = 0;
tf = 20000;
dt = 0.5;

n = (tf - t(1))/dt;

h(1) = 0;

for i = 1 : n

        t ( i+1) = t ( i ) + dt ;
        h( i +1) = h( i ) + dt *altura (h( i ) ) ;
end

set (‘current_figure’,1);
plot2d (t,h,[1 2]);

xgrid (1)
```

Função Runge Kutta de 4ª Ordem:

```
R = 2_10^8
Qe = 0.010247
r = 1000
S = 10
g = 10

function [ hponto ] = altura (h)
hponto = (−sqrt ( r*g*h/R) + Qe) /S ;
endfunction

t(1) = 0;
tf = 20000;
dt = 0.5;

n = (tf - t(1))/dt;

h(1) = 0;
```

```
for i = 1 : n
        t(i+1) = t(i) + dt;
        k1 = dt _ altura (h( i ) ) ;
        k2 = dt _ altura (h( i ) + k1 /2) ;
        k3 = dt _ altura (h( i ) + k2 /2) ;
        k4 = dt _ altura (h( i ) + k3 ) ;
        h( i +1)=h( i ) + ( ( k1+2_k2+2_k3+k4 ) /6) ;
end

set ('current_figure',1);
plot2d (t,h,[1 2]);

xgrid (1)
```

Exercício 2:

Função Euler:

```
Ra = 2_10^8
Rb = 2_10^8
Qe = 0.010247
r = 1000
S1 = 10
S2 = 9
g = 10

function [ hponto ] = altura1 (h1, h2)
hponto = (−sqrt ( r*g*(h1-h2)/Rs) + Qe) /S1 ;
endfunction

function [ hponto ] = altura2 (h1, h2)
hponto = (sqrt ( r*g*(h1-h2)/Ra) - sqrt ( r*g*(h2)/Rs) + Qe) /S2
endfunction

t(1) = 0;
tf = 20000;
dt = 0.5;

n = (tf - t(1))/dt;

h1(1) = 0;
h2(1) = 0;

for i = 1 : n

        t ( i+1) = t ( i ) + dt ;
        h1 ( i+1) = h ( i ) + dt*altura(h1(i),h2(i)) ;
        h2 ( i+1) = h ( i ) + dt*altura(h1(i),h2(i)) ;

end

set ('current_figure',1);
plot2d ([t,t],[h1, h2],[1 2]);

xgrid (1)
```

Função Runge Kutta de 4ª Ordem:

```
Ra = 2_10^8
Rb = 2_10^8
Qe = 0.010247
r = 1000
S1 = 10
S2 = 9
g = 10

function [ hponto ] = altura1 (h1,h2)
hponto = (−sqrt ( r*g*(h1-h2)/Ra) + Qe) /S1 ;
endfunction

function [ hponto ] = altura2 (h1,h2)
hponto = (sqrt ( r*g*(h1-h2)/Ra) - sqrt ( r*g*(h2)/Rs)  /S1 ;
endfunction

t(1) = 0;
tf = 20000;
dt = 0.5;

n = (tf - t(1))/dt;

h(1) = 0;

for i = 1 : n

        t ( i+1) = t ( i ) + dt ;

        k1A = dt _ altura (h1(i), h2(i));
        k1A= dt _ altura (h1(i) + k1A /2, h2(i) + k1A /2);
        k1A = dt _ altura (h( i ) + k1B /2, h2(i) + k1B /2);
        k1A = dt _ altura (h( i ) + k1C /2, h2(i) + k1C /2);
        h1 (i +1)=h1( i ) + ( ( k1A+2*k1B+2*k1C+k1D ) /6) ;

        k2A = dt _ altura (h1(i), h2(i));
        k2A= dt _ altura (h1(i) + k2A /2, h2(i) + k2A /2);
        k2A = dt _ altura (h( i ) + k2B /2, h2(i) + k2B /2);
        k2A = dt _ altura (h( i ) + k2C /2, h2(i) + k2C /2);
        h2 (i +1)=h1( i ) + ( ( k2A+2*k2B+2*k2C+k2D ) /6) ;

end

set ('current_figure',1);
plot2d (t,h,[1 2]);

xgrid (1)
```