Escola Politécnica da Universidade de São Paulo

PME3380 - Modelagem de Sistemas Dinâmicos

José Arthur Siqueira Guerrero NºUSP: 10791767

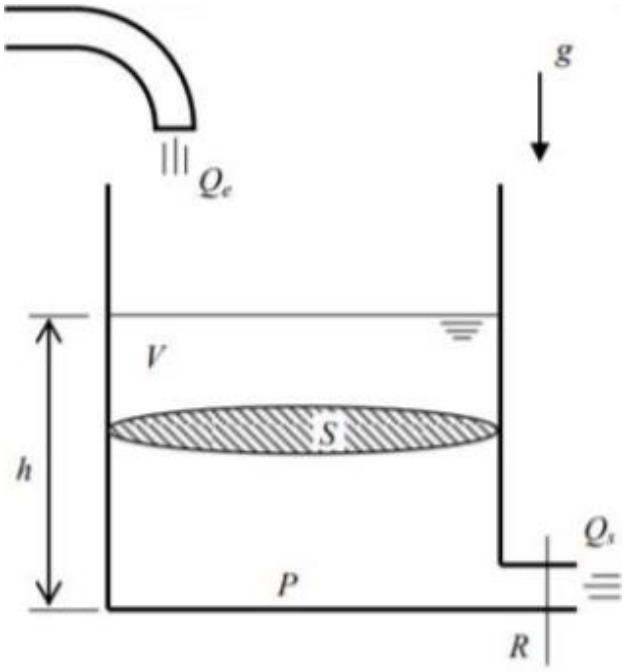
São Paulo - SP

2020

Todos os códigos utilizados para as simulações deste trabalho estão disponíveis no final do mesmo.

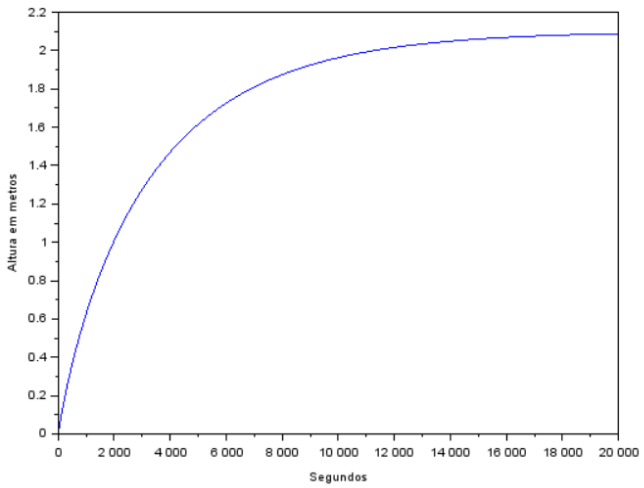
Parte 1 – Simulação de um reservatório

Para a lista B é pedido que se desenvolva um programa capas de modelar o sistema que consiste em um reservatório que tem uma vazão de entrada de água assim como uma certa vazão de saída, como é mostrado na imagem abaixo:



1.1 – Método de Euler

Altura da água no Reservatório

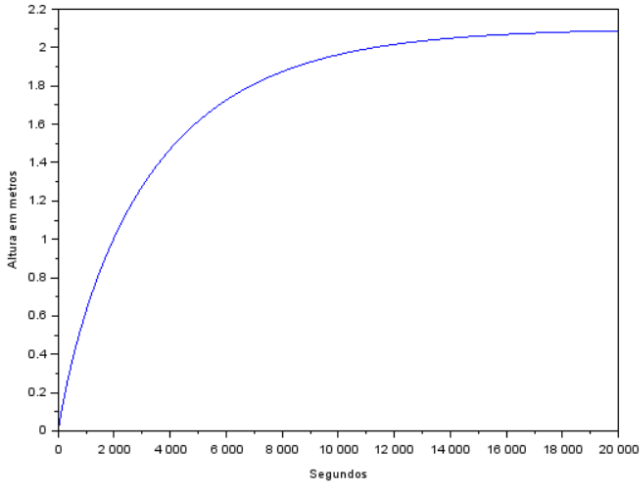


Durante as simulações previamente feitas, foi possível identificar que a altura do nível da água no reservatório tende a se estabilizar próximo ao momento de 20000 segundos, por isso essa foi a escolha de temporal da simulação mostrada no gráfico acima.

1.2 – Método de Runge Kutta

Assim como no método de Euler, foi possível identificar que a altura do nível da água no reservatório tende a se estabilizar próximo ao momento de 20000 segundos, por isso essa foi a escolha de temporal da simulação mostrada no gráfico abaixo.

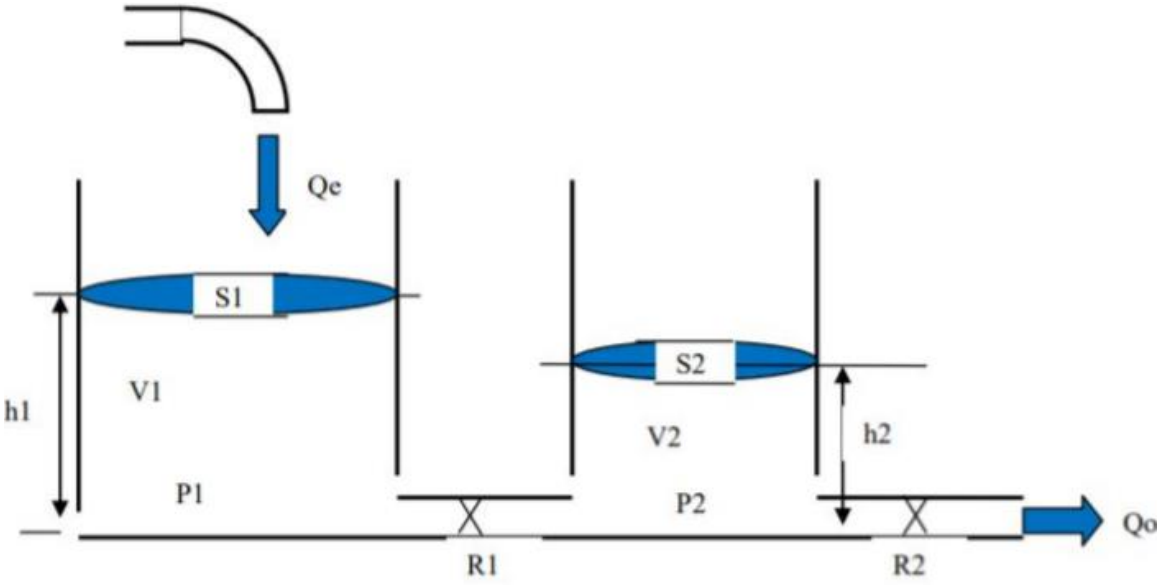
Altura da água no Reservatório



Analisando os gráficos podemos observar que não tivemos diferença nos resultados entre o método de Euler e o Metodo de Runge Kutta, sendo os dois bons modos de modelar o problema que nos foi fornecido.

Parte 2 - Simulação de dois reservatórios

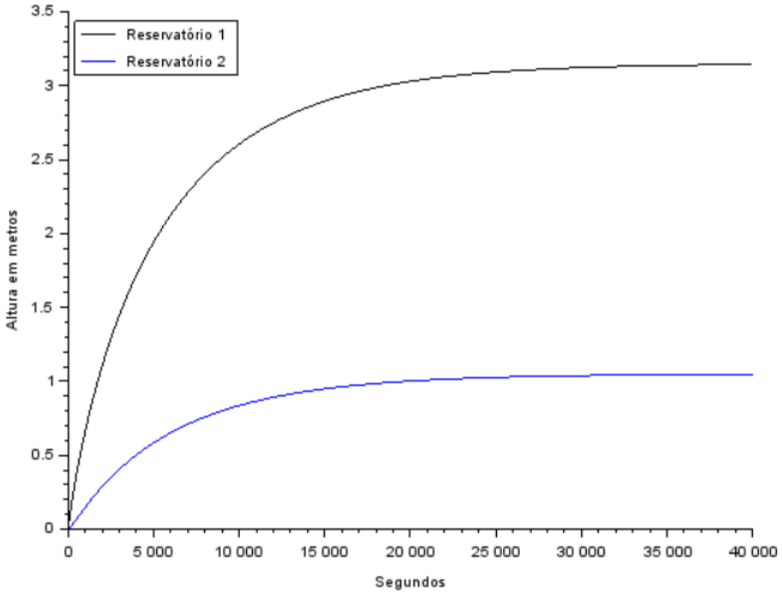
Agora é pedido que se desenvolva um programa capas de modelar o sistema que consiste em dois reservatórios que tem uma vazão de entrada de água assim como uma certa vazão de saída, como é mostrado na imagem abaixo:



2.1 – Método de Euler

Assim como na primeira parte, foi possível identificar que a altura do nível da água no reservatório tende a se estabilizar próximo ao momento de 40000 segundos, por isso essa foi a escolha de temporal da simulação mostrada no gráfico abaixo.

Altura da água no Reservatório

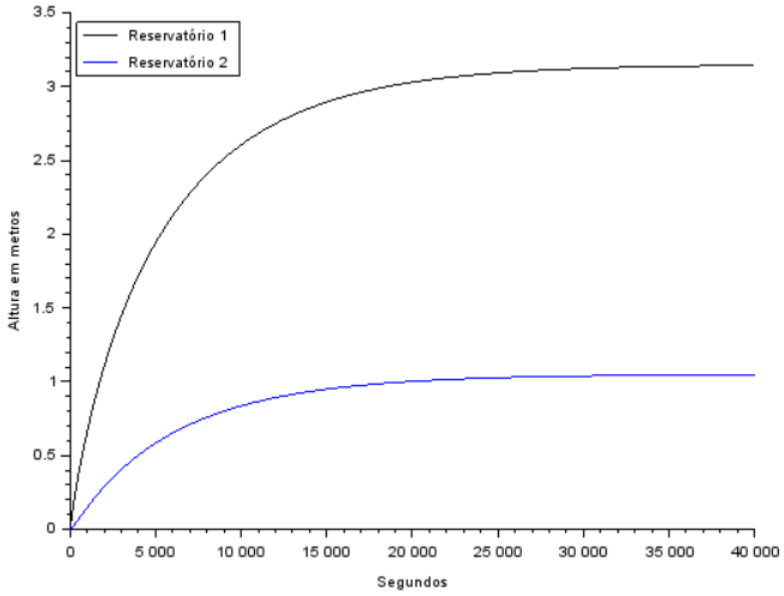


Observamos que com 2 reservatórios a altura da água no primeiro destes foi bem maior do que durante a simulação na parte 1, e tivemos um período de tempo maior até que estes reservatórios encontrassem o “equilíbrio” em relação a primeira parte.

2.2 – Método de Runge Kutta

Assim como no resto deste trabalho, foi possível identificar que a altura do nível da água no reservatório tende a se estabilizar próximo ao momento de 40000 segundos, por isso essa foi a escolha de temporal da simulação mostrada no gráfico abaixo.

Altura da água no Reservatório



Analisando os gráficos podemos observar que não tivemos diferença nos resultados entre o método de Euler e o Metodo de Runge Kutta, sendo os dois bons modos de modelar o problema que nos foi fornecido, sendo esta a mesma conclusão que chegamos na parte 1.

Parte 3 – Código das Simulação

1.1

```
function [Euler1]=a(h, rho, G, R, Qe, S)
   Euler1 = (Qe - sqrt(rho*G*h/R))/S;
endfunction
S = 10;
R = 2e+8;
rho = 1000;
G = 10;
Qe = 0.010247;
ti = 0;
tf = 20000;
n = 20000;
h = (tf-ti)/n;
ta = zeros(n+1,1)
for i = 2:n+1
   ta(i,1) = ta(i-1,1) + h
end
H = zeros(n+1,1);
for i = 2:n+1
   H(i,1) = H(i-1,1) + h*a(H(i-1,1), rho, G, R, Qe, S)
end
plot(ta,H);
xlabel("Segundos", "fontsize", 1)
ylabel("Altura em metros", "fontsize", 1)
```

1.2

```
function [Kutta1]=b(h, rho, g, R, Qe, S)
   Kutta1 = (Qe - sqrt(rho*g*h/R))/S;
endfunction
S = 10;
R = 2e+8;
rho = 1000;
G = 10;
Qe = 0.010247;
ti = 0;
tf = 20000;
n = 20000;
h = (tf-ti)/n;
ta = zeros(n+1,1)
for i = 2:n+1
   ta(i,1) = ta(i-1,1) + h
end
```

```
H = zeros(n+1,1);
for i = 2:n+1
   k1 = b(H(i-1,1), rho, G, R, Qe, S);
   k2 = b(H(i-1,1) + (1/2)*k1*h, rho, G, R, Qe, S);
   k3 = b(H(i-1,1) + (1/2)*k2,*h rho, G, R, Qe, S);
   k4 = b(H(i-1,1) + h*k3, rho, G, R, Qe, S);
   H(i,1) = H(i-1,1) + (h/6)*(k1+2*k2+2*k3+k4);
end
plot(ta,H);
xlabel("Segundos", "fontsize", 1)
ylabel("Altura em metros", "fontsize", 1)
```

2.1

```
function [Euler1]=c(h1, h2, Qe, rho, g, Ra, S1)
   Euler1 = (Qe-sqrt(rho*g*(h1-h2)/Ra))/S1;
endfunction
function [Euler2]=d(h1, h2, rho, g, Ra, Rs, S2)
   Euler2 = (sqrt(rho*g*(h1-h2)/Ra)-sqrt(rho*g*h2/Rs))/S2;
endfunction
S1 = 10;
R1 = 2e+8;
S2 = 8;
R2 = 1e+8;
rho = 1000;
G = 10;
Qe = 0.010247;
ti = 0;
tf = 40000;
n = 20000;
h = (tf-ti)/n;
ta = zeros(n+1,1)
for i = 2:n+1
   ta(i,1) = ta(i-1,1) + h;
end
h1 = zeros(n+1,1);
h2 = zeros(n+1,1);
for i = 2:n+1
   h1(i,1) = h1(i-1,1) + h*c(h1(i-1,1), h2(i-1,1), Qe, rho, g, R1, S1);
   h2(i,1) = h2(i-1,1) + h*d(h1(i-1,1), h2(i-1,1), rho, g, R1, R2, S2)
end
plot2d([ta,ta],[h1,h2],[1 2]);
legends(["Reservatório 1", "Reservatório 2"],[1,2],2)
xlabel("Segundos", "fontsize", 1)
ylabel("Altura em metros", "fontsize", 1)
```

2.2

```
function [Kutta1]=e(h1, h2, Qe, rho, g, Ra, S1)
  Kutta1 = (Qe-sqrt(rho*g*(h1-h2)/Ra))/S1;
endfunction
function [Kutta2]=f(h1, h2, rho, g, Ra, Rs, S2)
   Kutta2 = (sqrt(rho*g*(h1-h2)/Ra)-sqrt(rho*g*h2/Rs))/S2;
endfunction
S1 = 10;
R1 = 2e+8;
S2 = 8;
R2 = 1e+8;
rho = 1000;
G = 10;
Qe = 0.010247;
ti = 0;
tf = 40000;
n = 20000;
h = (tf-ti)/n;
ta = zeros(n+1,1)
for i = 2:n+1
   ta(i,1) = ta(i-1,1) + h;
end
h1 = zeros(n+1,1);
h2 = zeros(n+1,1);
for i = 2:n+1
   k1 = e(h1(i-1,1), h2(i-1,1), Qe, rho, G, R1, S1);
   k2 = e(h1(i-1,1) + (1/2)*k1*h, h2(i-1,1) + (1/2)*k1*h, Qe, rho, G, R1, S1);
   k3 = e(h1(i-1,1) + (1/2)*k2*h, h2(i-1,1) + (1/2)*k2*h, Qe, rho, G, R1, S1);
   k4 = e(h1(i-1,1) + h*k3, h2(i-1,1) + h*k3, Qe, rho, G, R1, S1);
   h1(i,1) = h1(i-1,1) +(h*(1/6))*(k1+2*k2+2*k3+k4);
   k1 = f(h1(i-1,1), h2(i-1,1), rho, G, R1, R2, S2);
   k2 = f(h1(i-1,1) + (1/2)*k1*h, h2(i-1,1) + (1/2)*k1*h, rho, G, R1, R2, S2);
   k3 = f(h1(i-1,1) + (1/2)*k2*h, h2(i-1,1) + (1/2)*k2*h, rho, G, R1, R2, S2);
   k4 = f(h1(i-1,1) + h*k3, h2(i-1,1) + h*k3, rho, G, R1, R2, S2);
   h2(i,1) = h2(i-1,1) +(h*(1/6))*(k1+2*k2+2*k3+k4);
end
plot2d([ta,ta],[h1,h2],[1 2]);
legends(["Reservatório 1", "Reservatório 2"],[1,2],2)
xlabel("Segundos", "fontsize", 1)
ylabel("Altura em metros", "fontsize", 1)
```