# 107506 – Laboratório de Controle de Processos

## Aula: Simulação de sistemas dinâmicos no *Matlab & Simulink*

Prof. Eduardo Stockler Tognetti

Departamento de Engenharia Elétrica
Universidade de Brasília – UnB

2º Semestre 2016

# Introdução

## Simulação de sistemas dinâmicos

1. Sistema não-linear

$$\begin{cases} \frac{d}{dt}x(t) & = f(x(t), u(t)) \\ y(t) & = g(x(t), u(t)) \end{cases} \quad \text{ou} \quad \begin{cases} \dot{x} & = f(x, u) \\ y & = g(x, u) \end{cases} \tag{1}$$

2. Sistema linear

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{2}$$

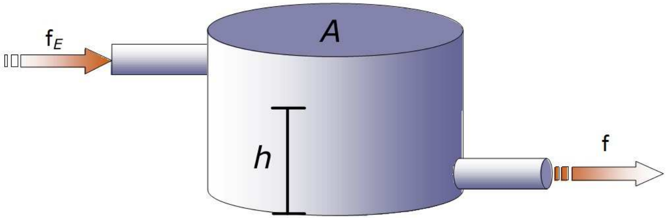
3. Função de transferência

$$Y(s) = H(s)U(s) \tag{3}$$

## Ferramentas computacionais

1. Simulink
2. Matlab & Simulink
3. Matlab

# Tanque de Nível




- Modelo matemático

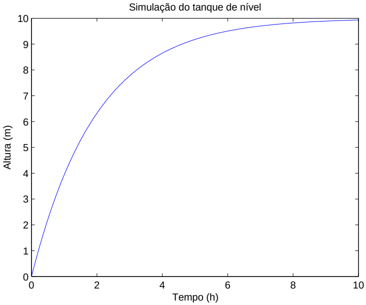
$$\frac{dh(t)}{dt} = \frac{1}{A}(f_E(t) - f(t)), \qquad h(t=0) = h_0 \tag{4}$$

- Considerando $f(t)$ proporcional à altura da coluna de líquido e inversamente proporcional a uma resistência ao escoamento $(R)$, $f(t) = \dfrac{h(t)}{R}$,

$$\frac{dh(t)}{dt} = \frac{1}{A}\left(f_E(t) - \frac{h(t)}{R}\right), \qquad h(t=0) = h_0 \tag{5}$$

# Tanque de Nível - Simulação da resposta analítica

```
% Definição das constantes do modelo
R = 1;   % h/m2
A = 2; % m2
Fe = 10; % m3/h

% Tempo de simulação
t = 0.0 : 0.01 : 10.0;   % h

% Simulação da altura de líquido
h = R*Fe*(1 - exp(-t/(R*A)));   % m

%  Visualização da simulação
plot(t,h);
title('Simulação do tanque de nível');
xlabel('Tempo (h)');
ylabel('Altura (m)');
```



# Solução numérica de EDOs

Considere a EDO

$$\begin{cases} \dfrac{dx_1}{dt} = x_1(1 - x_2^2) - x_2 \\ \dfrac{dx_2}{dt} = x_1, \end{cases} \qquad x(0) = \begin{bmatrix} 0 & 0.25 \end{bmatrix}'$$

$\leadsto$ Crie o arquivo `dxdt.m`:

```
function dx = dxdt(t,x)
dx1 = x(1)*(1-x(2)^2)-x(2);
dx2 = x(1);
dx = [dx1 dx2] ';
```

$\leadsto$ Crie o script:

```
x0 = [0 0.25];
TSPAN = [0 20];
[t,x] = ode23('dxdt',TSPAN,x0);
plot(t,x(:,1), ' *',t,x(:,2), 'o')
title('x vs. Time');
legend('x1', 'x2',0);
xlabel('time'); ylabel('x');
```

$\leadsto$ Solvers para edo's (ode45, ode23, etc):

```
[T,Y] = solver(odefun,tspan,y0,options)
```

- Problemas Não-Stiff: $y' = f(t,y)$, $y(t_0) = y_0$
  - **ode45** (Runge-Kutta, passo simples)
  - **ode23** (Runge-Kutta, passo simples)
  - **ode113** (Adams-Bashforth-Moulton, passo múltiplo)
- Problemas Stiff:
  - **ode15s** (numerical differentiation formulas (NDFs))
  - **ode23s** (Rosenbrock, passo único)
  - **ode23t** (Trapezoide)
  - **ode23tb** (Runge-Kutta)
  - **ode15i** ($f(t,y,y') = 0$)

Obs.: Também é possível usar

```
[t,x] = ode23(@dxdt,TSPAN,x0);
```

# Tanque de Nível - Simulação da EDO (Matlab)

## Parâmetros na função que contém a EDO (I)

Arquivo solve_edo.m :

```
function output = solve_edo

global R A Fe  % Def. das ctes do modelo

R = 1;  % h/m2
A = 2; % m2
Fe = 10; % m3/h

% Simulação da altura de líquido
h0 = 0; t = 0.0:0.01:10.0;
options = odeset('RelTol',1e-4,'AbsTol',1e-9);
[t,h] = ode45('dhdt',t, h0, options);

% Visualização da simulação
plot(t,h);
```

Arquivo dhdt.m :

```
function dh = dhdt(t,h)

global R A Fe

dh = (Fe-(h/R))/A;
```

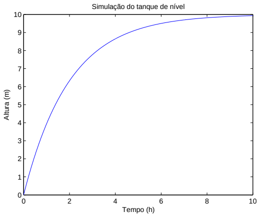
# Tanque de Nível - Simulação da EDO (Matlab)

## Parâmetros na função que contém a EDO (II)

```
function output = tnivel3(R,A,Fe)

if nargin == 0
% Def. das ctes do modelo
R = 1;  % h/m2
A = 2; % m2
Fe = 10; % m3/h
end

% Simulação da altura de líquido
h0 = 0; t = 0.0:0.01:10.0;
[t,h] = ode45('dhdt',t, h0,[],[R A Fe]);

% Visualização da simulação
plot(t,h);
```

Arquivo dhdt.m :

```
function dh = dhdt(t,h,flag,par)
R = par(1);
A = par(2);
Fe = par(3);
dh = (Fe-(h/R))/A;
```



# Entrada variante no tempo

**Simulação com entrada variante**

- Como considerar uma entrada que varia no tempo ou dependente de outras variáveis (estado, parâmetros) ?

**Ferramentas computacionais**

1. Matlab
2. Simulink

# Tanque de Nível - Simulação com Distúrbio de Entrada (Matlab)

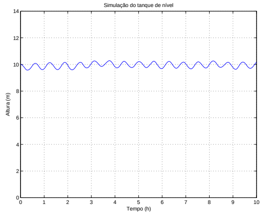
```
function output = tnivel5

% Simulação da altura de líquido
h0 = 10; t = 0.0:0.01:10.0;
[t,h] = ode45(@dhdt2,t,h0,[]);

% Visualização da simulação
plot(t,h);
```

Mesmo arquivo (tnivel5.m):

```
function dh = dhdt2(t,h,flag,par)
R = 1;  % h/m2
A = 2; % m2
dh = (Fe(t)-(h/R))/A;
```

Mesmo arquivo (tnivel5.m):

```
function fe = Fe(t)
% função que descreve a entrada Fe
fe1 = 10*(1-0.5*sin(10*t));
fe =  awgn(fe1,10,0);
```



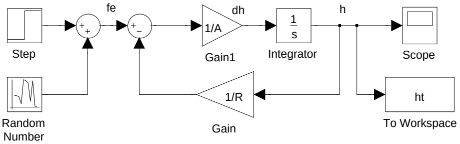




Diagrama Simulink correspondente.

# Representação na forma de espaço de estados

- Observe que o sistema (5) pode ser escrito na forma

$$\begin{aligned} \dot{x} &= \mathscr{A}x + \mathscr{B}u \\ y &= \mathscr{C}x + \mathscr{D}u \end{aligned} \tag{6}$$

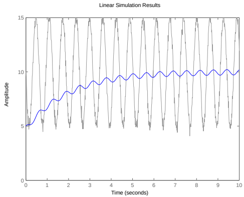
com

$$\mathscr{A} = -1/(RA); \qquad \mathscr{B} = 1/A; \qquad \mathscr{C} = 1; \qquad \mathscr{D} = 0. \tag{7}$$

Comandos matlab:

```
% Sistema
As = -1/(R*A); Bs = 1/A;
Cs = 1; Ds = 0;
sys = ss(As,Bs,Cs,Ds);

% Simulação à condição inicial
h0 = 10;
initial(sys,h0)

% Simulação ao degrau unitário (h0=0)
step(sys)
```

```
% Simulação ao distúrbio (h0=5)
t = 0.0:0.01:10.0;
fe1 = 10*(1-0.5*sin(10*t));
fe =  awgn(fe1,10,0);
lsim(sys,fe,t,h0)
```



# Representação na forma de função de transferencia

- Representando o sistema (5) no domínio-s

$$H(s) = \frac{K}{\tau s + 1}, \qquad h(0) = 0 \tag{8}$$

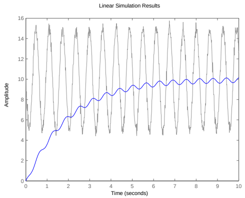
com

$$\tau = AR \quad \text{e} \quad K = R. \tag{9}$$

Comandos matlab:

```
% Sistema
[num,den] = ss2tf(As,Bs,Cs,Ds,1);
sys = tf(num,den);

% ou
sys=tf([R],[A*R 1])

% Simulação ao degrau unitário (h0=0)
step(sys)
```

```
% Simulação ao distúrbio (h0=0)
t = 0.0:0.01:10.0;
fe1 = 10*(1-0.5*sin(10*t));
fe =  awgn(fe1,10,0);
lsim(sys,fe,t)
```



# Tanque de nível não-linear

- Considere a situação mais realística no qual o fluxo de saída é dado por

$$f(t) = C_v\sqrt{h(t)}, \qquad Cv \triangleq \frac{1}{R}. \tag{10}$$

Tem-se

$$\frac{dh(t)}{dt} = \frac{1}{A}(f_E(t) - \frac{\sqrt{h(t)}}{R}), \qquad h(t=0) = h_0. \tag{11}$$

- Em regime permanente,

$$\overline{h} = (R\overline{f}_E)^2. \tag{12}$$

- Linearizando em torno de $(\overline{f}_E, \overline{h})$, tem-se

$$\frac{d\tilde{h}(t)}{dt} = \frac{1}{A}\tilde{f}_E(t) - \frac{1}{2AR\sqrt{\overline{h}}}\tilde{h}(t) = \frac{1}{A}\tilde{f}_E(t) - \frac{1}{\tau}\tilde{h}(t), \tag{13}$$

onde

$$\tilde{h}(t) = h(t) - \overline{h}, \qquad \tilde{f}_E(t) = f_E(t) - \overline{f}_E \tag{14}$$

- No domínio-s,

$$\frac{\tilde{H}(s)}{\tilde{F}_E(s)} = \frac{K}{\tau s + 1}, \qquad K = \frac{\tau}{A}, \qquad \tau = 2AR\sqrt{\overline{h}}. \tag{15}$$

# Simulação Matlab - Sistema Não-Linear

Comandos matlab:

```
[t,h] = ode23('dhdt_NL',ts,h0,
 opts,[R A fe0 fe1]);
function dh = dhdt_NL(t,h,flag,par)
% Sistema não-linear
R = par(1);
A = par(2);
fe0 = par(3:4); % degrau de entrada
dh = (f_Fe(t,fe0)-sqrt(h)/R)/A;
```

```
function fe = f_Fe(t,fe0)
% Dist. de entrada (m3/h)
if t < 3 || t > 30
    fe = fe0(1);
else
    fe = fe0(2);
end
```

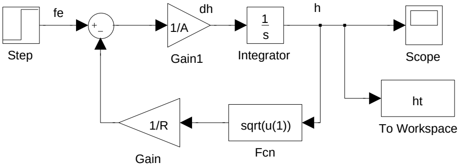


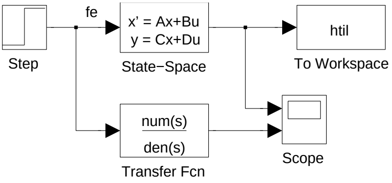

Figura: Diagrama Simulink do sistema não-linear.

# Simulação Matlab - Sistema Linearizado

Comandos matlab:

```
[t,htil] = ode45('dhdt_NL_lin',
  ts, h0-hlin, opts, [R A
  fe0-felin fe1-felin hlin]);
function dhtil =
 dhdt_NL_lin(t,htil,flag,par)
R = par(1);
A = par(2);
fetil = par(3:4);
hlin = par(5);
dhtil = (f_Fe(t,fetil)
  -1/(2*R*sqrt(hlin))*htil)/A;
```

```
% Espaço de Estados:
tau = 2*A*R*sqrt(hlin);
Alin = -1/tau;
Blin = 1/A;
Clin = 1;
Dlin = 0;
% Função Transferência:
numlin = tau/A;
denlin = [tau 1];
% Simulação:
[T,H] =
 sim('tnivel_simNL_lin',tmax);
```



## Exercício 1

$\leadsto$ Obtenha a solução analítica (apresenta a expressão) e numérica (plote a resposta) para as seguintes equações diferenciais:

$$(a) \qquad \frac{dy}{dx} = 5x\cos^2 y; \qquad y(0) = \pi/4$$

$$(b) \qquad \frac{d^2x}{dt^2} + 7\frac{dx}{dt} + 5x = 8u(t-5); \qquad t \geq 0$$

$(i)$ $\quad x(t)$ quando todas as condições inicias são nulas

$(ii)$ $\quad x(t)$ quando $x(0) = 1, \quad \dot{x}(0) = 2.$

Obs.: $u(t-5)$ é o sinal degrau unitário atrasado de 5 segundos.

# Linearização via Jacobiana

```
% model variables (Area = a
syms a rho g K;
% state variables
syms phi_i A_v h m p phi_o;
% state vectors
u = [phi_i; A_v];
x = [m];
y = [h; m; p; phi_o];
% non-linear system, dx = f(x,u,t)
F1 = rho * phi_i - rho * K * A_v *
sqrt((g / a) * m);
F = [F1];
% non-linear system, y(t) = g(x,u,t)
G2 = m;
G1 = rho * a * G2;
G3 = (g * G2) / a;
G4 = K * A_v * sqrt(G3);
G = [G1; G2; G3; G4];
% compute jacobian
A.symbolic = jacobian(F, x);
B.symbolic = jacobian(F, u);
C.symbolic = jacobian(G, x);
D.symbolic = jacobian(G, u);
```

```
% Operating point
g = 9.81; % gravitational force [m/s^2]
rho = 980; % mass density [kg/m^3]
a = 1; % area [m^2]
K = 0.01; % valve coefficient [-]
A_v = 0.01; % valve cross-sectional area [m^2]
phi_o_0 = 0.001; % initial flow out
m_0 = (phi_o_0 / (K * A_v))^2 * a / g; % initial
mass [kg]

% compute matrices A, B, C, D
A.algebraic = simplify(subs(A.symbolic, {A_v a
rho g K m}, [0.01 1 980 9.81 0.01 m_0]));
B.algebraic = simplify(subs(B.symbolic, {A_v a
rho g K m}, [sym(0.01) sym(1) sym(980) sym(9.81)
sym(0.01) m_0]));
C.algebraic = simplify(subs(C.symbolic, {A_v a
rho g K m}, [A_v a rho g K m_0]));
D.algebraic = simplify(subs(D.symbolic, {A_v a
rho g K m}, [A_v a rho g K m_0]));

% compute numerical values
A.eval = eval(A.algebraic);
B.eval = eval(B.algebraic);
C.eval = eval(C.algebraic);
D.eval = eval(D.algebraic);

% linearized system
linsys = ss(A.eval, B.eval, C.eval, D.eval);
```

# Linearização via fsolve – Exemplo I

```
function JACOB = exemplo_fsolve
% Condição inicial de calculo
V0 = [0 0];
% Ponto de linearizacao
global lin
lin = 'x'; % x(forneço u calculo
x), u(forneço x calculo u)

if lin == 'x'
% Forneço u calculo x
global u1 u2
u1 = 2; u2 = 1;
else
% Forneço x calculo u
global x1 x2 x1 = 1; x2 = 0.25;
end
% fsolve: dx/dt = F(X)
= 0 (chute inicial V0)
[X,FVAL,exitflag,output,JACOB]=fsolve(@dxdt_jacob,V0);
```

```
% function F = dxdt_jacob(t,x)
function F = dxdt_jacob(V)

global lin
if lin == 'x'
global u1 u2
x1 = V(1);
x2 = V(2);
else
global x1 x2
u1 = V(1);
u2 = V(2);
end
dx(1) = x1*(1-x2^2)-x2+0.5*u1-u2;
dx(2) = x1-0.1*u2;
F=[dx(:)];
```

# Linearização via fsolve – Exemplo II

```
function [A,B] = exemplo_fsolve

% Forneço ubar calculo xbar; A e B
em torno de (xbar,ubar)
[xbar,A] = jacob_calc('x',[2 1]);
[ubar,B] = jacob_calc('u',xbar);
linsys = ss(A,B,eye(2),zeros(2));
% Forneço xbar calculo ubar; A e B
em torno de (xbar,ubar)
[ubar,B] = jacob_calc('u',[1
0.25]);
[xbar,A] = jacob_calc('x',ubar);
linsys = ss(A, B, eye(2),
zeros(2));
```

```
function [X,JACOB] = jacob_calc(var,v)
global lin
lin = var;
if lin == 'x' % Forneço u calculo x
global u1 u2
u1 = v(1);
u2 = v(2);
else % Forneço x calculo u
global x1 x2
x1 = v(1);
x2 = v(2);
end
% fsolve: dx/dt = F(X) =
0 0 (chute inicial [0 0])
[X,FVAL,exitflag,output,JACOB]=
fsolve(@dxdt_jacob,[0 0]);
```

# Exercício

## Exercício 2

1. Encontre o ponto de operação em regime permanente, $\bar{h}$ e $\bar{T}$, a partir dos dados fornecidos.
2. Linearize o sistema em torno do ponto de operação em regime permanente $(\bar{h}, \bar{T})$.
3. Ache a representação no espaço de estados (A,B,C,D) considerando como entradas $F_e(t)$ e $T_e(t)$, e saídas $h(t)$ e $T(t)$
4. Ache as funções de transferência $\dfrac{H(s)}{F_e(s)}$, $\dfrac{T(s)}{F_e(s)}$, e $\dfrac{T(s)}{T_e(s)}$

- Parâmetros e dados em estado estacionário:

R = 1; % h/m2, A = 2; % m2
$\rho C_p = 750$; % kJ/m3.K
U = 150; % kJ/(m2.s.K)
$\bar{F}_e = 10$; % m3/h
$\bar{T}_e = 530$; % K
$\bar{T}_h = 540$; % K

- EDOs:

$$\frac{dh(t)}{dt} = \frac{1}{A}\left(F_e(t) - \frac{h(t)}{R}\right)$$

$$\frac{dT(t)}{dt} = \frac{1}{h(t)}\left[\left(\frac{F_e(t)T_e(t)}{A}\right) - T(t)\left(\frac{F_e(t)}{A} + \frac{U}{\rho Cp}\right)\right]$$

# Simulação Matlab - Sistema Linearizado

### Simulação de $h(t)$

- Simulação de $h(t)$ ao invés de $\tilde{h}(t)$
- Simulação da resposta degrau
- Linearização em outros pontos de operação

### Ferramentas computacionais

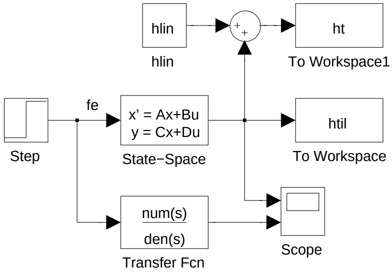
1. Matlab
2. Simulink

# Simulação Matlab - Sistema Linearizado

Comandos matlab:

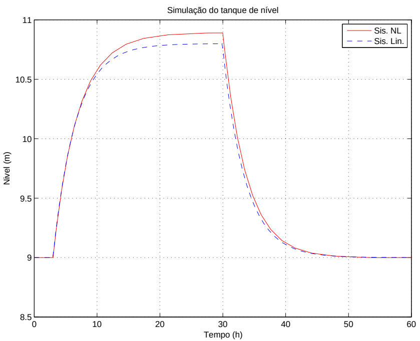
```
ht = htil +
 hlin*ones(length(htil),1);
plot(t,ht);

[t,ht2] = ode45('dhdt_NL_lin2',
 ts,h0, opts,[R A
 fe0 fe1 felin hlin]);
```

```
function dh =
 dhdt_NL_lin2(t,h,flag,par)
R = par(1);
A = par(2);
fe0 = par(3:4);
felin = par(5);
hlin = par(6);
fe = [fe0(1)-felin fe0(2)-felin];
dh = (f_Fe(t,fe)-
 1/(2*R*sqrt(hlin))*(h-hlin))/A;
```



# Resposta ao degrau

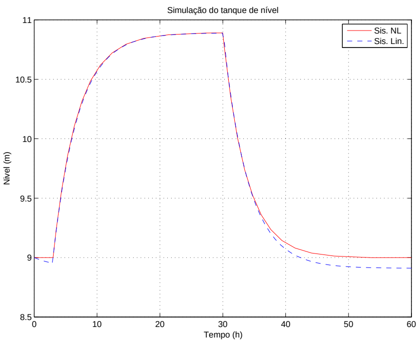
## Sistema Não-linear *versus* Linearizado




- Regime Permanente $(f_e(0),\ h(0)) = (10.0,\ 9.0)$
- Linearizado em $(\bar{f}_e,\ \bar{h}) = (10.0,\ 9.0)$
- Simulação: Condição inicial $h(0) = 9.0$; Distúrbio $f_e(t) = 10.0 \implies 11.0$

# Resposta ao degrau

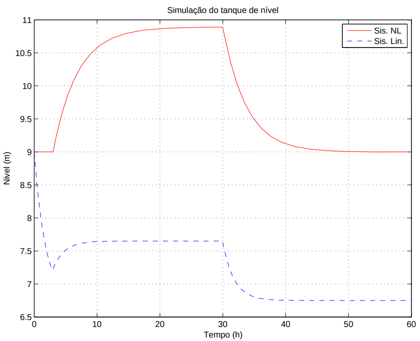
## Sistema Não-linear *versus* Linearizado




- Regime Permanente $(f_e(0),\ h(0)) = (10.0,\ 9.0)$
- Linearizado em $(\overline{f}_e,\ \overline{h}) = (11.0,\ 10.9)$
- Simulação: Condição inicial $h(0) = 9.0$; Distúrbio $f_e(t) = 10.0 \Longrightarrow 11.0$

# Resposta ao degrau

## Sistema Não-linear *versus* Linearizado




- Regime Permanente $(f_e(0),\ h(0)) = (10.0,\ 9.0)$
- Linearizado em $(\overline{f}_e,\ \overline{h}) = (5.0,\ 2.3)$
- Simulação: Condição inicial $h(0) = 9.0$; Distúrbio $f_e(t) = 10.0 \implies 11.0$

# Exercício

## Exercício 3

Utilize os dados fornecidos no Exercício 2.

1. Escreva um script matlab e simule a dinâmica usando ode45 ou ode23. Ex.:

   ```
   [t,x]=ode45('dxdt',tempo de simulação,condições iniciais);
   ```

2. Plote a resposta à condição inicial $(h(0), T(0)) = (5/A, T_h)$
3. Plote a resposta de $h(t)$ à variação de da entrada $F_e$ de 10 para 15 m3/h no instante $t = 10$s. Coloque num mesmo gráfico a resposta do sistema não-linear e linearizado (função de transferência). Obs.: $h(t) = \bar{h}$ no intervalo $0 \leq t < 10$.
4. Plote a resposta de $T(t)$ à variação de da entrada $F_e$ de 10 para 15 m3/h no instante $t = 10$s. Coloque num mesmo gráfico a resposta do sistema não-linear e linearizado (função de transferência). Obs.: $T(t) = \bar{T}$ no intervalo $0 \leq t < 10$.

- EDOs:

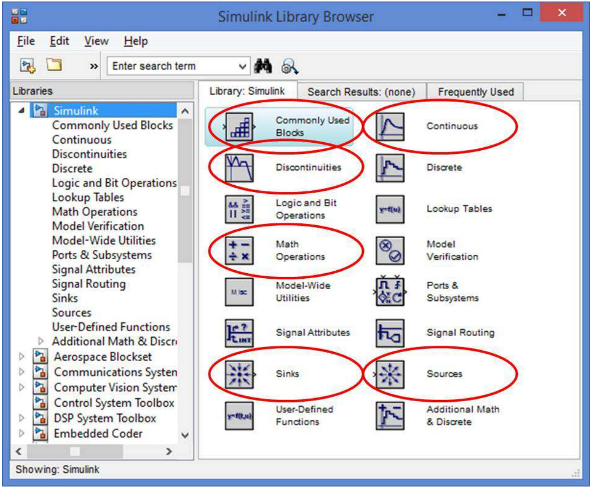
$$\frac{dh(t)}{dt} = \frac{1}{A}\left(F_e(t) - \frac{h(t)}{R}\right), \qquad \frac{dT(t)}{dt} = \frac{1}{h(t)}\left[\left(\frac{F_e(t)T_e}{A}\right) - T(t)\left(\frac{F_e(t)}{A} + \frac{U}{\rho Cp}\right)\right]$$

# Simulink

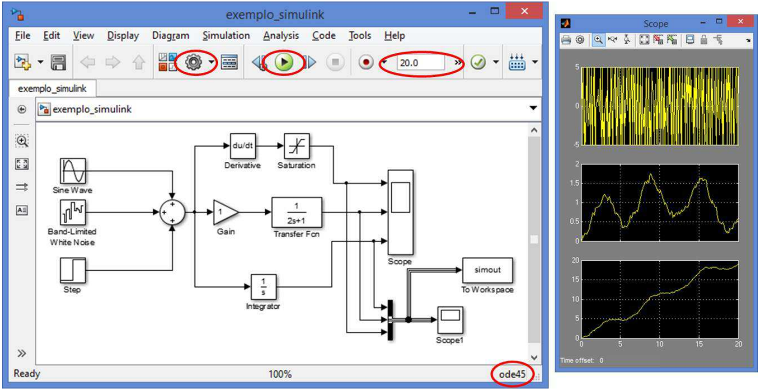
Simulink

# Simulink

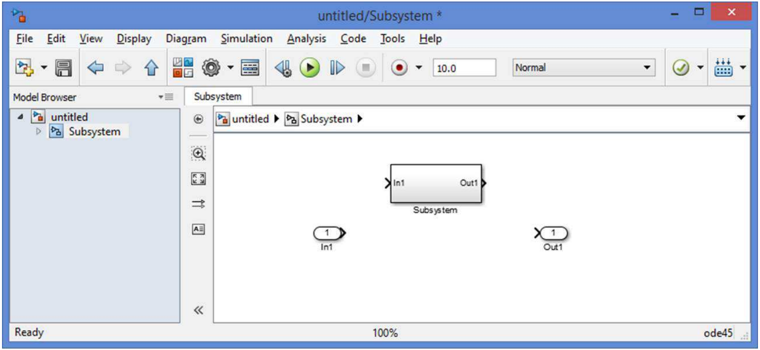
## Funções Frequentes



## Exemplo


exemplo_simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
20.0
Sine Wave
Band-Limited White Noise
Step
du/dt
Derivative
Saturation
Gain
1
2s+1
Transfer Fcn
1
s
Integrator
Scope
simout
To Workspace
Scope1
Ready
100%
ode45
Scope
Time offset: 0

## Subsistemas




- Bloco *Subsystem* (Biblioteca: Simulink/Commonly Used Blocks)

## Equações diferenciais

- Observe que integradores podem ser utilizados para a construção de sistemas dinâmicos no Simulink

### Exemplo: equação diferencial no Simulink

- Circuito elétrico RLC série relacionando a tensão fonte $V$ (entrada) com a tensão no capacitor $V_c$ (saída)

$$V_c(s) = \frac{25}{s^2+6s+25}V(s)$$

Aplicando a transformada inversa de Laplace

$$\frac{d^2v_c(t)}{dt^2} + 6\frac{dv_c(t)}{dt} + 25v_c(t) = 25v(t)$$

$$\frac{d^2v_c(t)}{dt^2} = -6\frac{dv_c(t)}{dt} - 25v_c(t) + 25v(t)$$

$\rightsquigarrow$ As condições iniciais podem ser inseridas no bloco de integração.

## Equações diferenciais

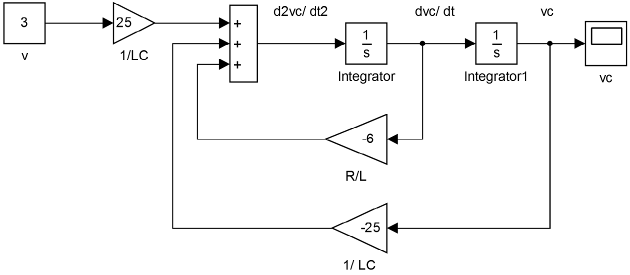



Diagrama Simulink da resposta de $v_c(t)$ ao degrau de 3 volts em $v(t)$ do circuito RLC série.

# Simulink

## Exemplos

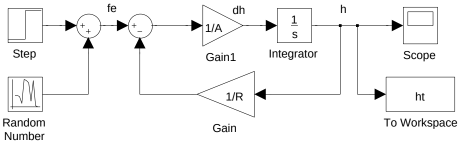



Diagrama Simulink – sistema linear.

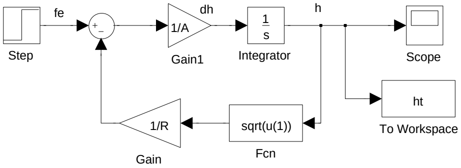



Diagrama Simulink – sistema não-linear.

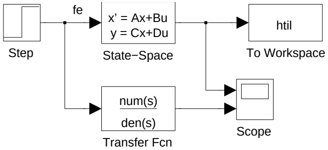



Diagrama Simulink de sistema linear – saída em termos de variáveis de desvio.

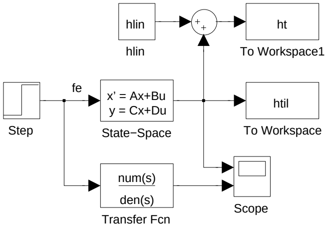
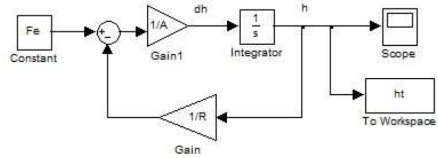


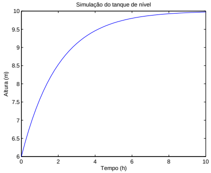

Diagrama Simulink de sistema linear – saída em termos de variáveis absolutas.

# Tanque de Nível - Simulação (Simulink)

```
% Def. das ctes do modelo
R = 1;   % h/m2
A = 2; % m2
Fe = 10; % m3/h

% Tempo de simulação
tf = 10.0;   % h

% Condição inicial
h0 = 6;
```

```
% Simulação da altura de líquido
[T,H] = sim('tnivel1_sim',tf);

% Saida
t = ht.time;
h = ht.signals.values;

%  Visualização da simulação
plot(t,h);
```




- Observação:

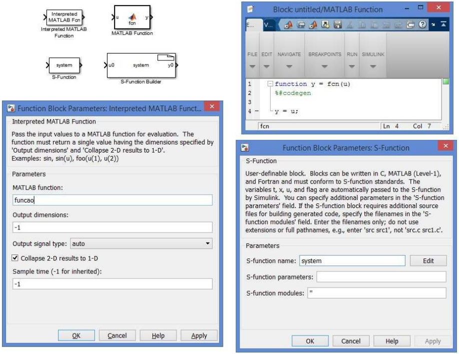
Para executar um arquivo *Simulink* via comando `sim(.)` de dentro de uma função, é necessário carregar os parâmetros da simulação usando os comandos

```
options = simset('SrcWorkspace','current'); % ou evalin('base','h0=10');
[T,H] = sim('tnivel_sim2',tf,options);
```

# Simulando funções no Simulink

- Para simular funções definidas pelo usuário no *Simulink* as principais possibilidades são

1. Interpreted Matlab Function
2. MATALB function
3. S-function

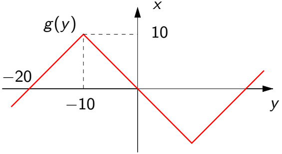



⇝ Pode-se utilizar esses blocos para a definição de EDOs de sistemas não-lineares.

# Simulando funções no Simulink

- Seja o sistema não linear

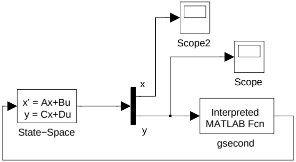
$$\begin{cases} \dot{x} = -\beta x - y \\ \dot{y} = x - g(y) \end{cases} \qquad (18)$$

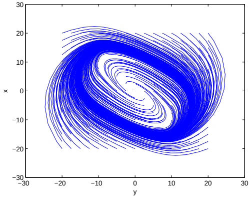



onde $g(y)$ é uma função linear por partes.

Simulação:

```
function y=gsecond(u)
% Implementa g(y)
if u < -10
    y=(u+20);
else if u > 10
        y=(u-20);
    else
        y=-u;
    end
end
```



# Sistema em malha fechada

## Controle do nível pela válvula do fluxo de entrada

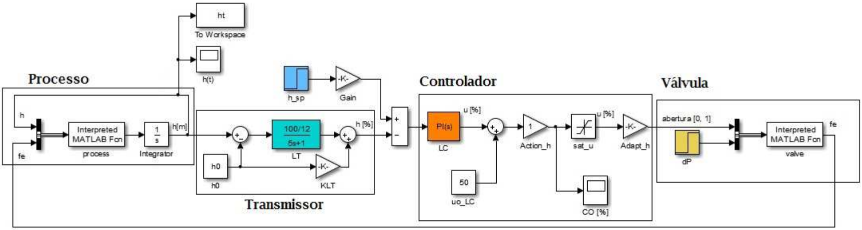
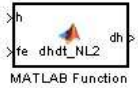



Figura: Sistema em malha fechada: sistema não-linear, transmissor de nível, controlador PI (ação direta) e válvula de controle na entrada do tanque.


ou




```
function dh = dhdt_NL2(h,fe)
% modelo não-linear do tq. de aquecimento
dh = (fe-sqrt(h)/R)/A;
```

# Sistema em malha fechada

## Controle do nível pela válvula do fluxo de saída do tanque

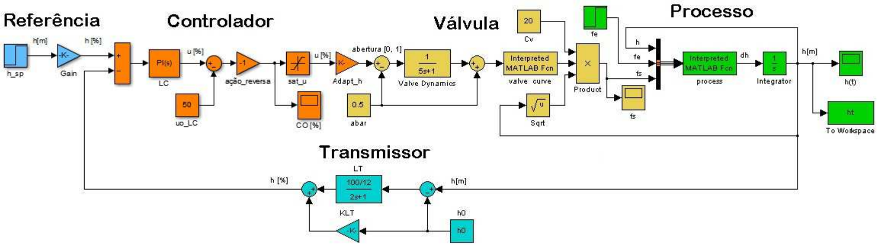



Figura: Sistema em malha fechada: sistema não-linear, transmissor de nível, controlador PI (ação reversa) e válvula de controle na saída do tanque.

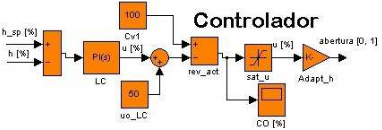



Figura: Implementação alternativa para ação reversa do controlador.

# Representação de processos no *Simulink*

## Observações

- O processo deve estar inicialmente em estado estacionário em um dado ponto de operação, ou seja, para entradas fixas a saída deve permanecer constante até que um degrau ou uma perturbação aconteça (geralmente as entradas e saídas são não nulas em estado estacionário);
- A referência e a saída do processo devem ser expressas em variáveis absolutas com unidades de engenharia definidas;
- O nível pode ser expresso em termos percentuais;
- Em cada parte do diagrama deve ficar claro a variável envolvida e sua correspondente unidade ou significado;
- Os ganhos também devem ter unidades de engenharia definida (ex.: ganho de um conversor I/P $[\mathrm{psi}/\mathrm{mA}]$);
- Recomenda-se que o ganho do controlador seja expresso de forma adimensional;
- A saída do transmissor pode ser expressa em $\mathrm{mA}$, $\mathrm{V}$ ou %;
- É usual que o sinal de controle seja expresso na faixa de 0 a 100 % com saturação em 100 %.

## Exercício 4

Realize o que é solicitado no Exercício 3 no Simulink, ou seja,

1. Plote a resposta à condição inicial $(h(0), T(0)) = (5/A, T_h)$
2. Plote a resposta de $h(t)$ à variação de da entrada $F_e$ de 10 para 15 m3/h no instante $t = 10$s. Coloque num mesmo gráfico a resposta do sistema não-linear e linearizado (função de transferência). Obs.: $h(t) = \bar{h}$ no intervalo $0 \leq t < 10$.
3. Plote a resposta de $T(t)$ à variação de da entrada $F_e$ de 10 para 15 m3/h no instante $t = 10$s. Coloque num mesmo gráfico a resposta do sistema não-linear e linearizado (função de transferência). Obs.: $T(t) = \bar{T}$ no intervalo $0 \leq t < 10$ .

- EDOs:

$$\frac{dh(t)}{dt} = \frac{1}{A}\left(F_e(t) - \frac{h(t)}{R}\right), \qquad \frac{dT(t)}{dt} = \frac{1}{h(t)}\left[\left(\frac{F_e(t)T_e}{A}\right) - T(t)\left(\frac{F_e(t)}{A} + \frac{U}{\rho Cp}\right)\right]$$