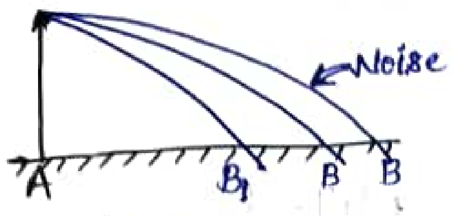
AV241 - Instrumentation and Control Lab

CONTROL LAB-1

19-01-2024

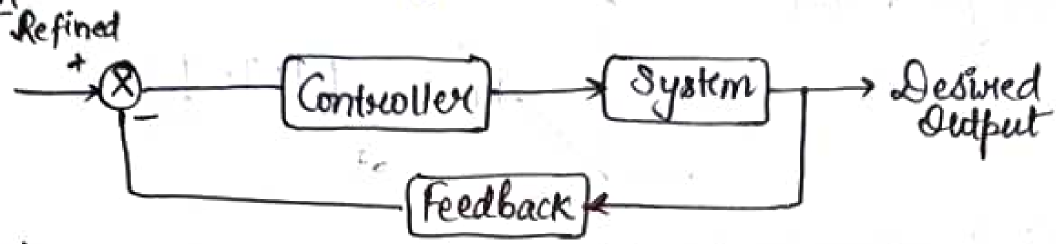
SAURABH KUMAR
SC22B146

#



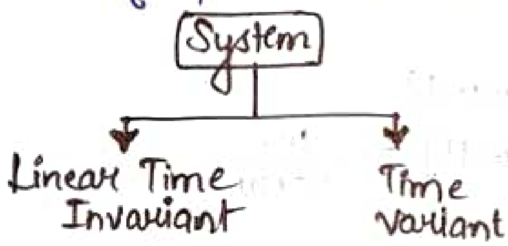
- → Actual output
- → Desired Output

Control system :



Transfer Function

↳ Ratio of ~~linear~~ laplace transform of output to the laplace transform of input when all initial conditions are zero.



Linear System

↳ Obeys superposition and homogeneity theorem.

Superposition : $x_1 \rightarrow \boxed{G} \rightarrow y_1$, $x_2 \rightarrow \boxed{G} \rightarrow y_2$ $\Rightarrow$ $x_1+x_2 \rightarrow \boxed{G} \rightarrow y_1+y_2$

Homogeneity:

$$\alpha x \rightarrow \boxed{G(.)} \rightarrow \alpha y$$

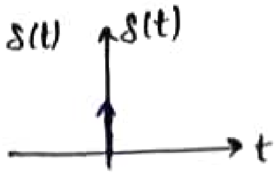
Eg. $C(s) = \dfrac{10(s+1)}{(s+2)(s+3)}$

Poles : Makes function infinite $\longrightarrow -2, -3$

Zeroes : Makes function zero $\longrightarrow -1$

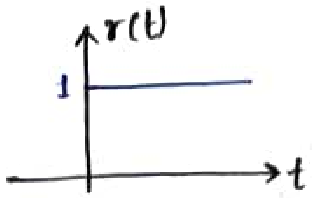
Signals

① Impulse input : $\delta(t)$



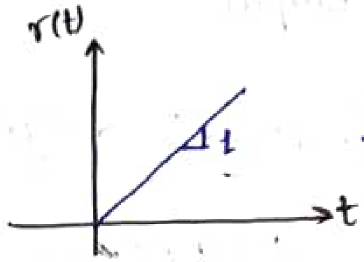
$\mathcal{L}[\delta(t)] = \boxed{1}$

② Unit step input : $r(t)$



$\mathcal{L}[1] = \boxed{\frac{1}{s}}$

③ Ramp/Velocity : $r(t) = t$



$\mathcal{L}[t] = \boxed{\frac{1}{s^2}}$

④ Parabolic/acceleration input :

$r(t) = \frac{t^2}{2}$

$\mathcal{L}[t^2/2] = \boxed{1/s^3}$

⑤ Sinusoidal input :

$r(t) = \sin(\omega_0 t)$

$\mathcal{L}[\sin(\omega_0 t)] = \boxed{\frac{\omega_0}{s^2+\omega_0^2}}$

Order: Total no. of poles.

Type: No. of poles at the origin.

Eg. $\frac{10}{s(s+2)} \longrightarrow$ Order = 2, Type = 1

Second Order System

$$\frac{C(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \longrightarrow \begin{array}{l}\text{Order} = 2 \\ \text{Type} = 0\end{array}$$

ξ : damping ratio

$\xi = 0$: Undamped system

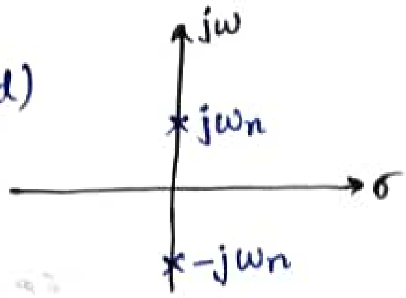
$0 < \xi < 1$: Underdamped system

$\xi = 1$: Critically damped system

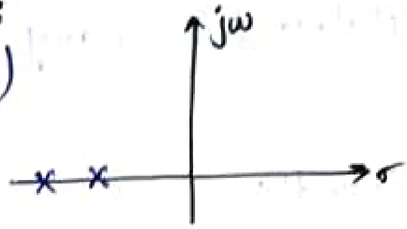
$\xi > 1$: Overdamped system.

S-plane (σ-ω plane):

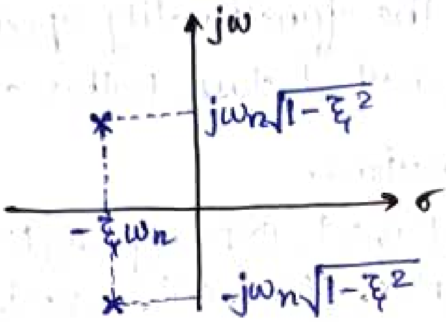
$\xi = 0$:
(Undamped)



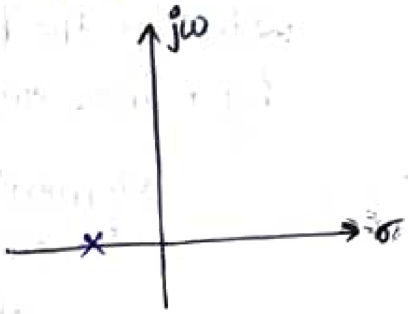
$\xi = 1$:
(Critically damped)

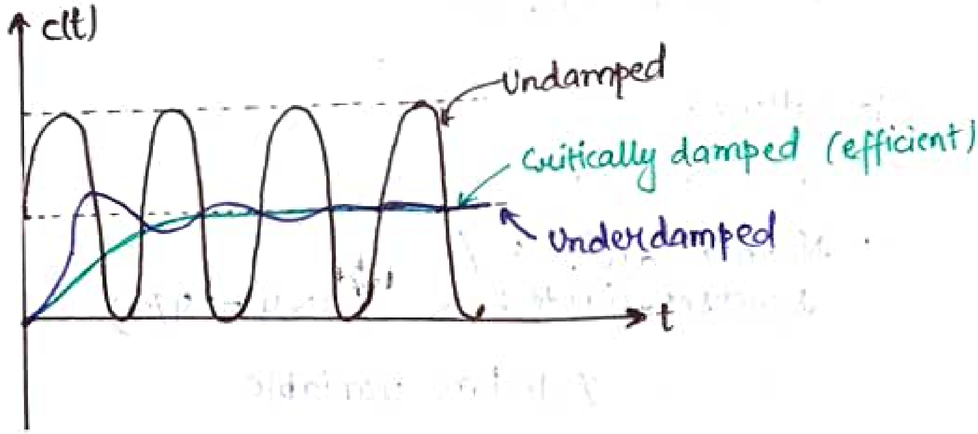


$0 < \xi < 1$:
(Underdamped) $s = -\xi\omega_n \pm j\omega_n\sqrt{1-\xi^2}$



$\xi > 1$:
(Overdamped)





MATLAB

↳ Matrix laboratory

Commands:

→ pwd (present working directory

pwd : displays the current working directory.

s=pwd : returns the current directory in string s.

→ help : displays the help text for the functionality specified by name, such as function, method, class, toolbox or variable.

→ help elfun : Elementary math functions.

↳ sin, sinh, asin, tan, cot, exp, log, log10, sqrt, abs, angle, complex, floor, ceil, round, mod, rem.

→ lookfor : search for keyword in reference page text.

↳ lookfor transfer

Operations:

↳ +, −, *, /, \

↳ Normal divide : /

Reverse divide : \ (3\4 = 4/3)

clc : clears screen.

clear all : clears all workspace variables.

clear abc : clears variable with 'abc' name.

disp : displays word or value

```
a1=3;   % declare variable
a2=4;
r1 = a1/a2;
r2 = a1\a2;   % = a2/a1
```

Plot Graph:

```
a=0:0.01:20;
b=sin(a);
plot(a,b,'r');   % red (plot y vs x graph)
plot(a,b,'--');  % dashed
plot(a,b,'--r'); % dashed red
xlabel('x');     % label x-axis
ylabel('y');     % label y-axis
title('plot of sine); % give title of the plot
```

Two graph on same plot:

```
plot(x,y1); hold on
plot(x,y2);
legend('sin', 'cos')
```

'sin' → for y_1, 'cos' → for y_2

Subplot:

↳ To plot multiple graphs in same window.

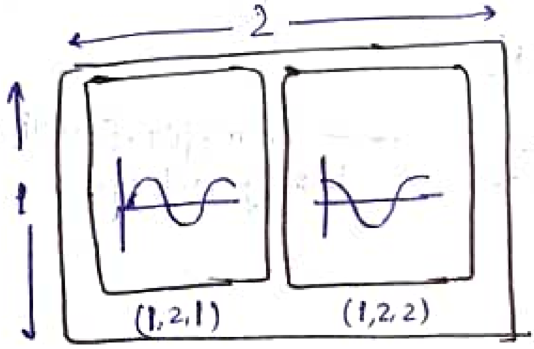
```
x = -pi : pi/20 : pi;
% -pi, -pi + pi/20, -pi + 2pi/20, ... , -pi + 40 pi/20.
y1 = sin(x);
y2 = ~~sin~~ cos(x);
subplot (1, 2, 1);
```

no. of rows

no. of columns

position

```
plot (x, y1);
subplot (1, 2, 2);
plot (x, y2);
```



Matrix:

```
zeros(1,8);  % matrix of 1x8 with all values as zero
ones (1,8);  % matrix of 1x8 with all values as one
eye(4);      % 4x4 identity matrix
inv (B);     % Inverse of matrix B
det (B);     % ~~Inverse of~~ determinant of matrix B

c = A + B    % matrix addition
c = A.B      % matrix multiplication
c = A.*B     % element-by-element matrix multiplication
```

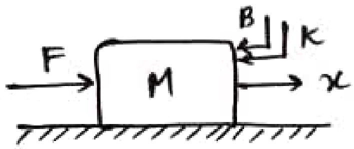
Matrix Declaration:

```
c = [1 5 3];   % declare matrix
size(c);  % --> 1  3
d = [4 4 7; 4 1 8; 3 8 0];
size(d);  % --> 3  3
```

Transpose:

```
tra = d';
```

Eg. Mass system:



$$F - B\dot{x} - Kx = M\ddot{x}$$

LT $\Rightarrow$ $F(s) - Bs\,X(s) - KX(s) = Ms^2X(s)$ [zero initial conditions]

$\Rightarrow$ $F(s) = X(s)\,[Ms^2 + Bs + K]$.

$\Rightarrow$ $G(s) = \dfrac{X(s)}{F(s)} = \dfrac{1}{Ms^2 + Bs + K}$ $\longrightarrow$ Transfer function ↳ open-loop

MATLAB:

```
m = 1;
b = 5;
k = 7;
num = 1;
den = [m b k];
sys = tf(num, den);
% sys =        1
%       ---------------
%        s^2 + 5 s + 7
```

Standard 1st order system : $G(s) = \dfrac{\tau}{s+\tau}$ $\left[\lim_{s \to 0} \dfrac{\tau}{s+\tau} = 1 \Rightarrow \text{dc gain} = 1\right]$

Standard 2nd order system: $G(s) = \dfrac{1}{s^2/\omega_c^2 + s/Q\omega_c + 1}$

AV241 – Instrumentation and Control Lab

CONTROL LAB-2

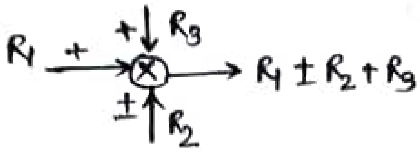
25-01-2024

SAURABH KUMAR
SC22B146

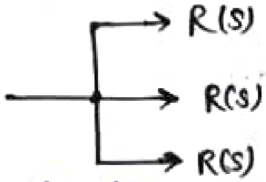
Modelling a System (using a block diagram)

→ : flow of information

□ : transfer function



→ Summing point or Error detector

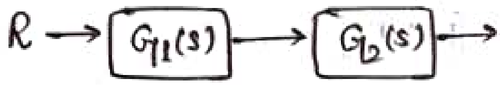


(one i/p, multiple o/p)

→ Takeoff point or Pickoff point

LTI System:

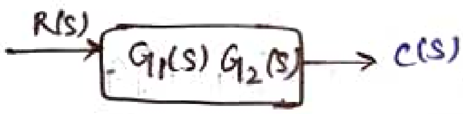
Eg.



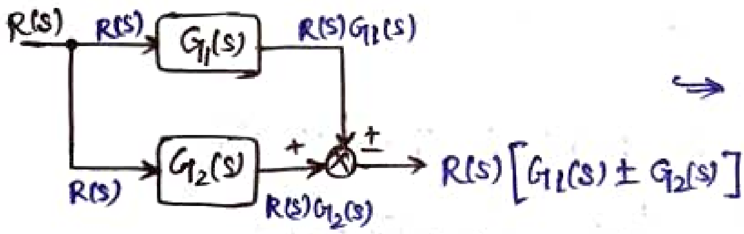
$C(s) = G_1(s)\, G_2(s)\, R(s)$

→ Block in cascade/series

⇒

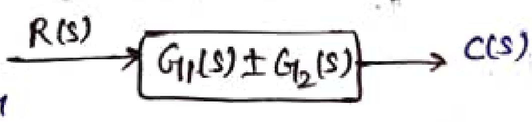


Eg.

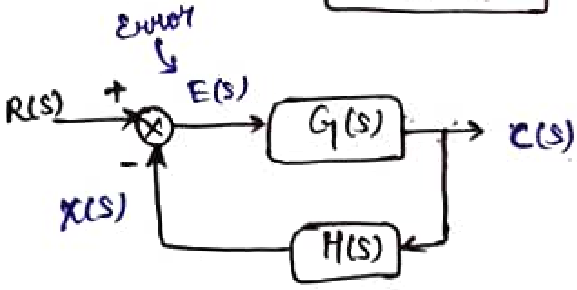


→ Blocks in parallel

⇒



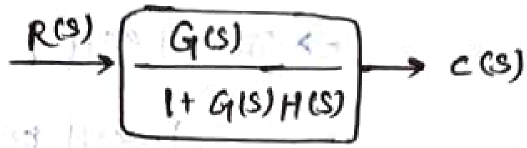
Eg.



→ Blocks in feedback

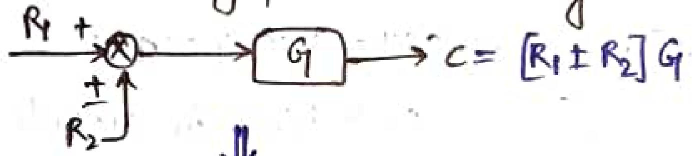
$$E(s) = R(s) - X(s)$$
$$= R(s) - H(s)C(s)$$

$$C(s) = E(s)G(s)$$
$$= [R(s) - H(s)C(s)]\,G(s)$$
$$= G(s)R(s) - G(s)H(s)C(s)$$
$$\Rightarrow C(s)[1 + G(s)H(s)] = G(s)R(s)$$
$$\Rightarrow \boxed{\frac{C(s)}{R(s)} = \frac{G(s)}{1 \pm G(s)H(s)}}$$

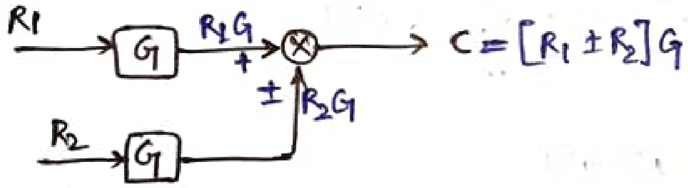
$\Rightarrow$ 

Shifting:

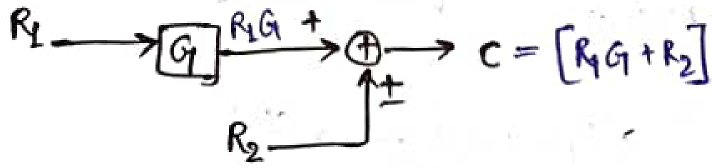
① Shifting the summing point to the right side:

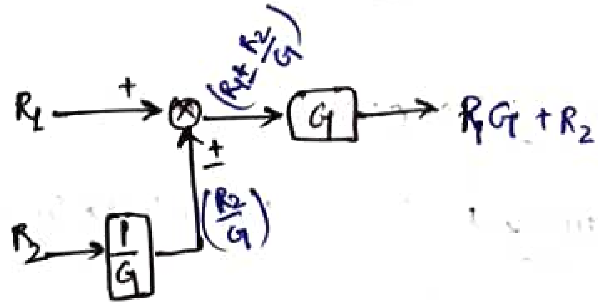


⇓



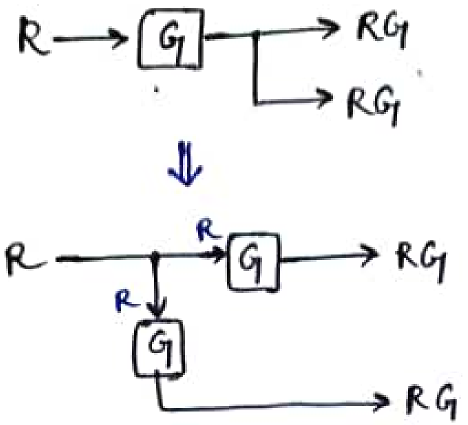
② Shifting the summing point to the left side:



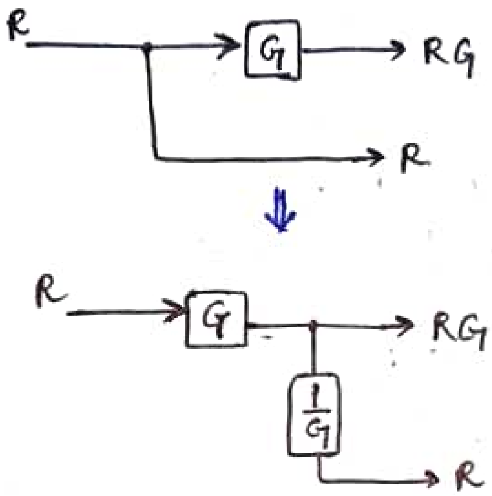


③ Shifting the pickoff point:

(i)



(ii)



Commonly used Blocks:

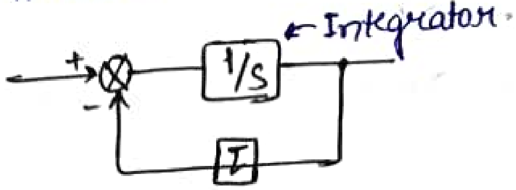
- Continuous: Integrator, differentiator, transfer function.
- Discontinuous: Non-linear blocks.
- Math operation: Adder, subtractor, gain.
- Sources: Input signals like step, ramp, sine; from workspace, from spreadsheet, from file.
- Sinks: To see output.

- Scope: Displays output.
  - → Configuration parameter: ctrl + E
    Solver selection Type: Fixed-step
    Solver: ode 4
  - → Numerical integrator: automatically uses low sample rate
    ↳ Use fixed step
    If bandwidth = 50 Hz ⇒ use sampling frequency = 50 Hz × 50
    ↳ based on bandwidth.

Transfer function block:

↳ Continuous.

Eg. $\frac{1}{S+I}$



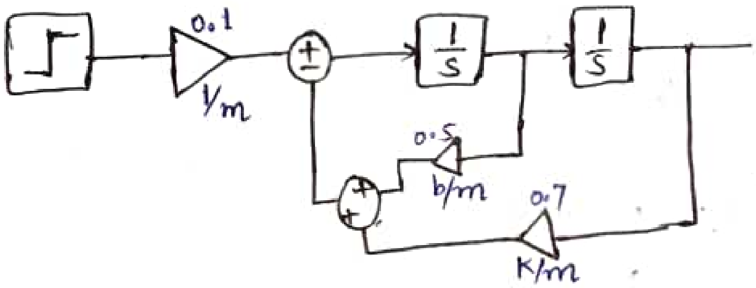
Eg.

$$\frac{X(s)}{F(s)} = \frac{1}{Ms^2 + Bs + K}$$

where $F = M\ddot{x} + B\dot{x} + Kx$

$$\Rightarrow \ddot{x} = \frac{F - B\dot{x} + Kx}{M}$$

$$= \frac{F}{M} - \frac{B}{M}\dot{x} - \frac{K}{M}x$$



AV241 – Instrumentation and Control Lab

CONTROL LAB-3

01-02-2024
SAURABH KUMAR
SC22B146

→ $\dfrac{C(s)}{R(s)} = \dfrac{10}{s^2+2s+1}$ ⇝ Single Input Single output

$\Rightarrow s^2C(s) + 2sC(s) + C(s) = 10R(s)$

$\Rightarrow \ddot{c}(t) + 2\dot{c}(t) + c(t) = 10r(t)$

Let $c(t) = x_1(t)$

$\dot{c}(t) = \dot{x}_1(t) = x_2(t)$

$\ddot{c}(t) = \ddot{x}_2(t)$

$\ddot{c}(t) = \dot{x}_2(t) = -2x_2(t) - x_1(t) + 10r(t)$

$$\underbrace{\begin{bmatrix} \dot{x}_1(t) \\ x_2(t) \end{bmatrix}}_{\dot{x}(t)} = \underbrace{\begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} 0 \\ 10 \end{bmatrix}}_{B} \underbrace{u(t)}_{u}$$

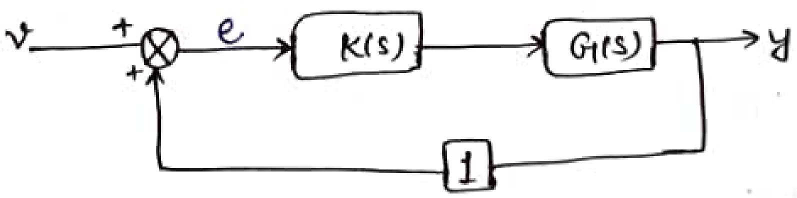
$$y = c(t) = x_1(t)$$

$$= \underbrace{[10]}_{C} \underbrace{\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}}_{x} + \underbrace{[0]}_{D} \underbrace{u}_{u}$$

↳ State-space representation

Open-loop system:

Closed-loop (feedback) system:



MATLAB Commands :

complex → creates complex array

```
C= complex (A,B)  % A+Bi
```

conj (X) → complex conjugate of X

real(x) → Real part of x

imag (X) → Imaginary part of X.

abs (X) → absolute value of element X (magnitude of X)

atan2(y,x)
atan2d(y,x) → in degrees
} Four quadrant inverse tangent of x and y.

```
Eg. atan2d(3,4)  % 36.8699
```

polyval (P,x) → Evaluates polynomial P at x

```
Eg.  P= [3, 2, 1];  % p(x)=3x^2+2x+1
     x = [5, 7, 9];  % at x=5,7,9
     y = polyval(p,x)  % 86 162 262
```

roots (p) → find polynomial roots

Arrays :

ones (x,y) → x-by-y matrix of ones

zeros(x,y) → x-by-y matrix of zeros.

eye (x,y) or eye([x,y]) → x-by-matrix with 1's on the diagonal and zeros elsewhere.

ginput → graphical input from mouse

ginput (2) → gets 2 points from current axes and returns x- and y- coordinates.

norm → vector and matrix norms

conv (A,B) → convolves A and B vectors.

```
Eg. x= 0:0.1:10;
    y = x;
    plot(y,x);
    ginput (2);  % --> for 2 points
```

Transfer Function in Polynomial form

```
% G(s) = [2s^3+5s^2+3s+6] / [s^3+6s^2+11s+6]
num= [2 5 3 6];
den = [1 6 11 6];
sys= tf(num, den);
```

TF in Partial Fraction /Residue form:

```
[r, p, K] = residue (num,den);
% r= [-6, -4, 3],
  p= [-3, -2, -1],
  k=2
```

$\rightarrow \frac{-6}{s+9} + \frac{-4}{s+2} + \frac{3}{s+1} + 2$

TF in Pole-Zero form:

```
[Z, P, K] = tf2zp( num, den);   % zero-pole-gain
 % Z= [-2.3965 + 0i, -0.0518+1.1177i, -0.0518-1.1177i ],
   p= [-3,-2,-1],
   k=2
```

$\hookrightarrow \frac{2(s+2.3965)(s+0.0518-j1.1177)(s+0.0518+j1177)}{(s+3)(s+2)(s+1)}$

Z-P-G Format to Polynomial:

```
[num, den] = zp2tf (z, p, k);
```

Eg.
```
z=[]; p=[-1+2*j; -1-2*j] ; k=10 ;
[num, den] = zp2tf (z, p, k);
sys= tf (num,den)   % T(s)= 10 / [s^2 + 2s + 5]
```

TF to state-space Representation:

```
[A, B, C, D]= tf2ss( num, den);
```

convolution:

```
conv( [1,10], [1,4,16])
```

state-space to TF:

```
[num, den]= ss2tf(A, B, C, D, iu);
```

iu ↳ no. of inputs

Eg. %A = [0 1; -25 -4], B = [1 1; 0 1]; C = [1 0; 0 1]; D = [0 0; 0 0]

```
% obtain the corresponding transfer functions.
A = [0 1; -25 -4];
B = [1 1; 0 1];
C = [1 0; 0 1];
D = [0 0; 0 0];
[num, den] = ss2tf(A, B, C, D, 1);
[num, den] = ss2tf (A, B, C, D, 2);
```

Partial Fraction Expansion

```
% f = 2/[(S+1)(S+2)]
partfrac (2/((s+1)*(s+2)))   % = 2/(s+1) - 2/(s+2)
```

Laplace transform:

```
syms s    % create symbolic variable
f = ilaplace (s/((s+1)*(s+2)))   % inverse laplace transform
 % f = 2*exp(-2*t) - exp(-t)
pretty (f);   % to resemble type-set mathematics
 % f = exp(-2t) 2 - exp(-t)
```

Eg. % obtain the laplace transform of f(t) = sin(3t + 45)

```
syms t
f = sin(3*t + 45);
F = laplace(f);
pretty(F);
%    3 cos(45) + s sin(45)
   --------------------------
           s^2 + 9
```

Eg. % G1 = 10/[S^2 + 2S + 10]; G2 = 5/[S+5]

```
n1 = 10;  d1 = [1 2 10];
n2 = 5;   d2 = [1 5];

%% series
[ns ds] = series (n1, d1, n2, d2);
printsys (ns ds)   % Print system in pretty format
% num/den =          50
             -------------------
             S^3 + 7S^2 + 20S + 50
```

%% Parallel

```
[np, dp] = parallel (n1, d1, n2, d2);
printsys(np, dp)
```

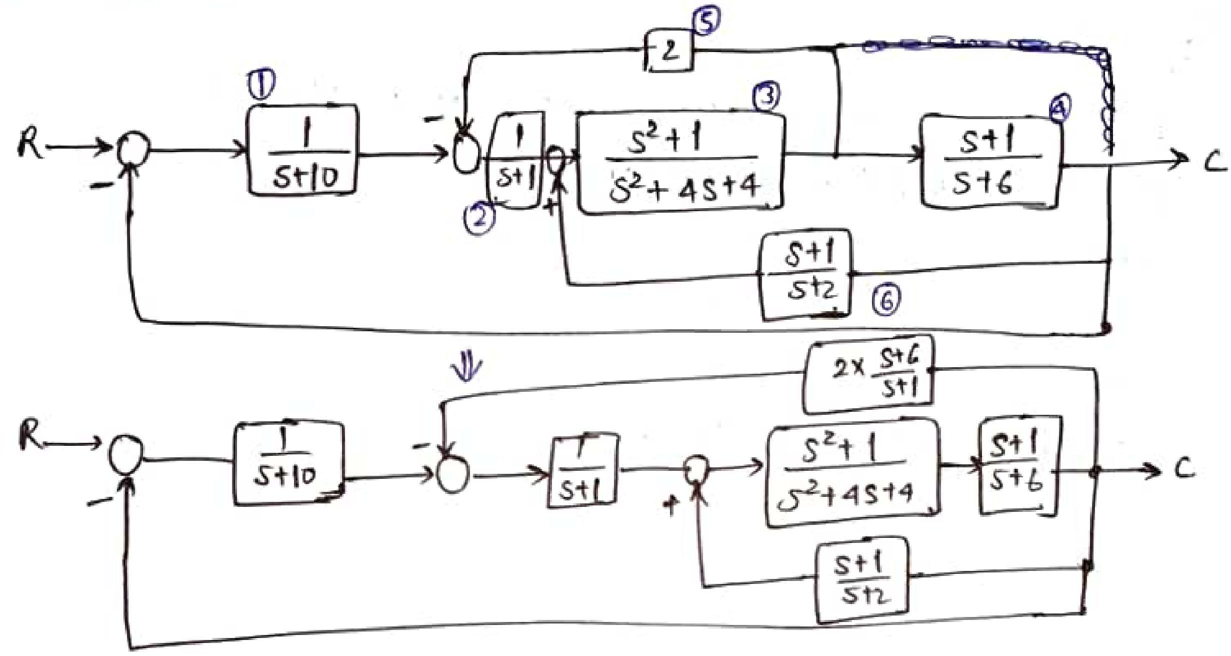
%% Feedback

```
[nf, df] = feedback(n1, d1, n2, d2);
```

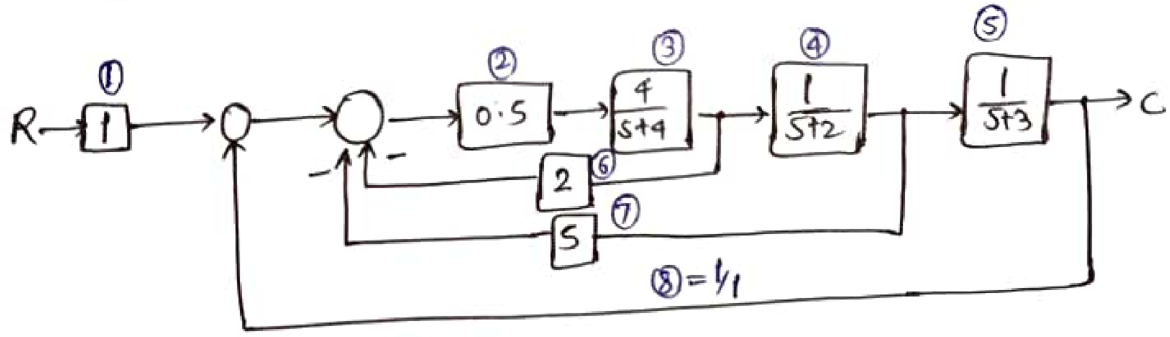
%% Positive Feedback

```
[nfp, dfp] = feedback(n1, d1, n2, d2, 1);
```

Eg.



Eg.




```
n1=1; d1=1;
n2=0.5; d2=1;
n3=4; d3=[1 4];
n4=1; d4=[1 2];
n5=1; d5=[1 3];
n6=2; d6=1;
n7=5; d7=1;
n8=1; d8=1;
nblocks = 8; % no. of blocks
```

Block no. $\quad$ max no. of incoming (internal) signals

$$
q = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
2 & 1 & -6 & -8 & -7 \\
3 & 2 & 0 & 0 & 0 \\
4 & 3 & 0 & 0 & 0 \\
5 & 4 & 0 & 0 & 0 \\
6 & 3 & 0 & 0 & 0 \\
7 & 4 & 0 & 0 & 0 \\
8 & 5 & 0 & 0 & 0
\end{bmatrix}
$$

← i/p's coming from blocks 1, 6, 7, 8

rows = no. of blocks

```
iu=1; iy=5, % input (block 1) to output (block 5)
blkbuild % builds the matrix.
[A B C D] = connect (a, b,c,d,q, iu, iy)
[num, den] = ss2tf (A, B, C, D)
sys= tf (num, den)
% ANS = 2/[s^3 + 13s^2 + 56s + 80]
```

AV241 - Instrumentation and Control lab
CONTROL LAB - 4

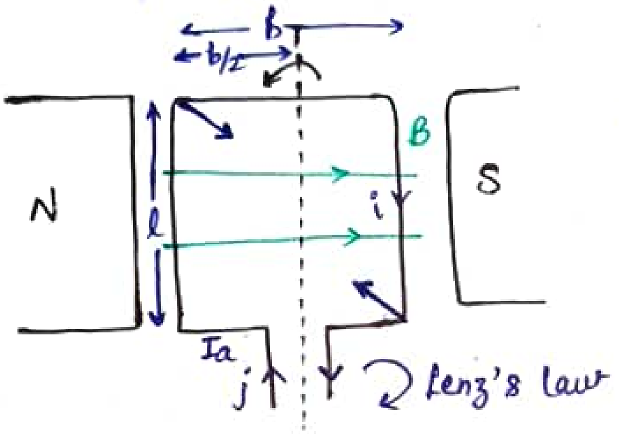
06-02-2024
SAURABH KUMAR
SC22B146

Mathematical Modelling of Electromechanical Actuation System

Electrical input → Mechanical

→ Motor - control system
→ Antenna Control system

Motor: $\underbrace{\text{Armature}}_{\text{field: fixed}}$ - Controlled DC (brushed type)



→ Direction of force given by: Fleming's left hand rule

→ EMF: back emf (e_b)

$$F = \sum B I_a l \sin\theta_j$$

Total force = 2F

Torque, $T = f \times \perp^{r}$ distance

$$= \sum 2BI_a l \sin\theta_j \times b/2$$

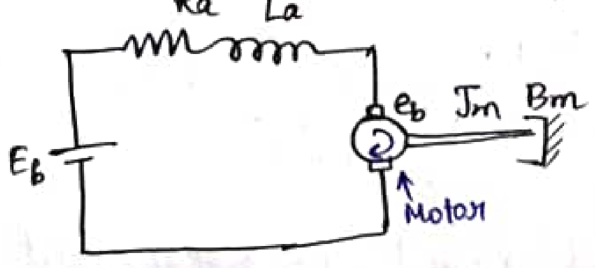
$$T = 2BI_a l \sin\theta_j \times b/2$$
$$= K_T \times I_a$$

Total torque $= \sum K_T I_a$

[Fleming's Right Hand Rule]

Back emf: $e_b = \underbrace{K_b}_{\text{constant}} \omega_m$ [Due to electromagnetic induction]

Generator equation: $E_g = \dfrac{\phi Z P \overset{\text{speed}}{W_m}}{60 A}$, Z: no. of conductors, P: no. of poles



Jm: Moment of inertia of motor

Bm: viscous damping (due to friction)

Electrical eqn: $E_a = I_a R_a + L_a \frac{dI_a}{dt} + e_b$ ...(1)

Mechanical eqn: $T_m = (J_m \ddot{\theta} + B_m \dot{\theta})$: Torque

$$\Rightarrow \boxed{T_m = (J_m s^2 + B_m s)\,\theta(s) = K_T I_a} \quad \ldots (2)$$

LT (1) $\Rightarrow$ $E_a - e_b = I_a R_a + L_a s I_a$

$$\Rightarrow \boxed{E_a - e_b = I_a (R_a + L_a s)} \quad \ldots (1)'$$

and, $\boxed{e_b = K_b \omega_m}$ ...(3)

and, $\omega_m = \dot{\theta} \Rightarrow \boxed{\omega_m = \theta s}$

[Motor
I/P : E_a
O/P : θ]

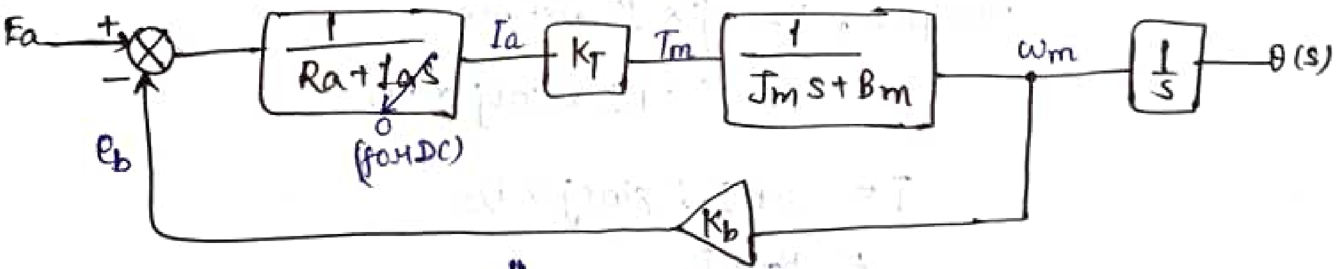
(2) $\Rightarrow$ $K_T I_a = (J_m s^2 + B_m s)\theta = \left(\frac{E_a - e_b}{R_a + L_a s}\right) K_T$

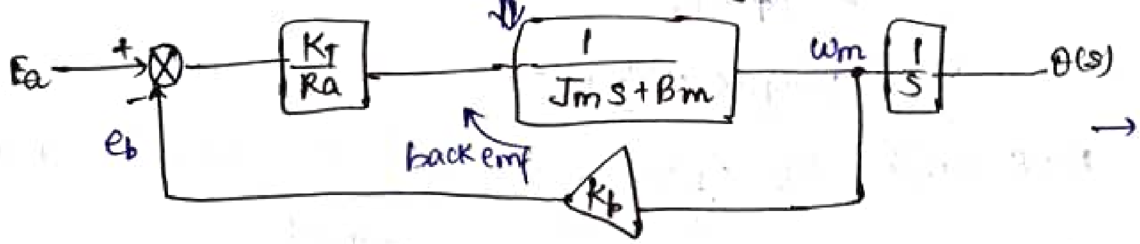
$$\Rightarrow (J_m s^2 + B_m s)\theta = \left(\frac{E_a - K_b \theta s}{R_a + L_a s}\right) K_T$$

$$\Rightarrow \left(J_m s^2 + B_m s + \frac{K_b K_T s}{R_a + \cancel{L_a s}_{\,0}}\right)\theta = \frac{E_a K_T}{R_a + \cancel{L_a s}} \rightarrow 0 \text{ (for DC)}$$

$$\Rightarrow \frac{\theta}{E_a} = \frac{K_T / R_a}{J_m s^2 + \left(B_m + \frac{K_T K_b}{R_a}\right)s}$$

[$L_a = 0$ for DC]





→ Open-loop

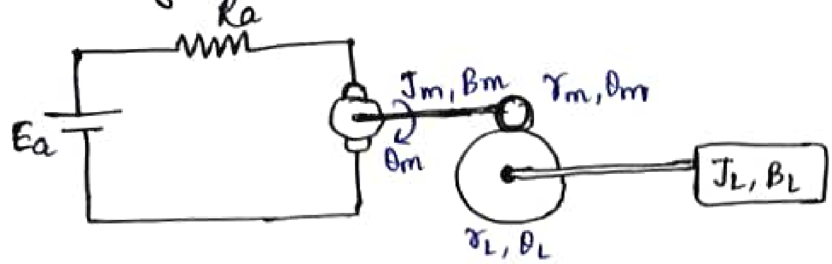
Closed-loop : Speed control using sensor

Open-loop TF : $\frac{\theta(s)}{E_a(s)} = \frac{K_T / R_a}{J_m s^2 + \left(B_m + \frac{K_T K_b}{R_a}\right)s}$, K_b : electromechanical damping

→ Motor : High speed × low torque

For mechanical, we need high torque × low speed.

Controlling Motor :



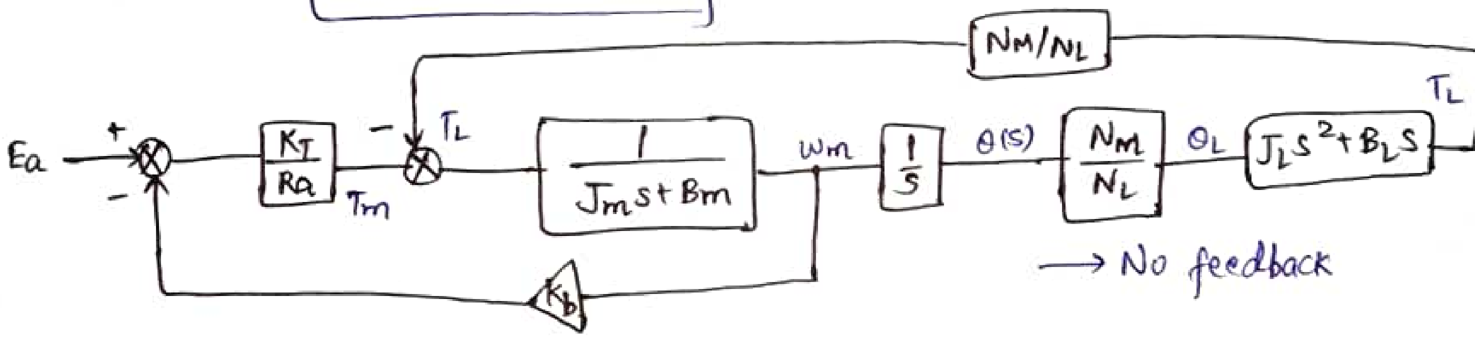
$\left\{ \begin{array}{l} \overset{\text{radius of motor}}{r_m} \propto \overset{\text{no. of teeth}}{N_m} \\ r_L \propto N_L \end{array} \right.$

$$\boxed{T_m \theta_m = \underbrace{T_L \theta_L}_{\text{area covered}}} \rightarrow \text{torque}$$

$$\frac{T_m}{T_L} = \frac{\theta_L}{\theta_m} = \frac{N_L}{N_M}$$

$$\Rightarrow T_L = T_m \left(\frac{N_M}{N_L}\right)$$

$$\boxed{T_L = (J_L s^2 + B_L s)\,\theta_L}$$



→ No feedback

$$\uparrow I_a = \frac{(E_a - e_b)}{R_a}\downarrow$$

AV241 - Instrumentation and control lab

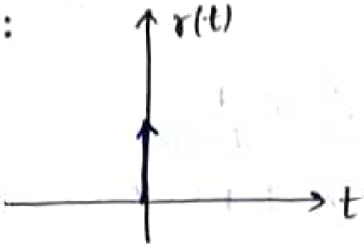
Control LAB-5

16-02-2024

SAURABH KUMAR
SC22B146

Standard Test Signals

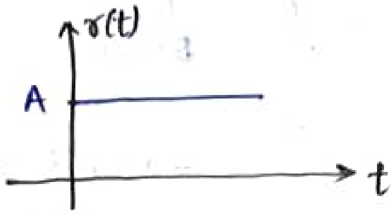
① Impulse input:



$r(t) = \delta(t)$

$\mathcal{L}[\delta(t)] = 1$

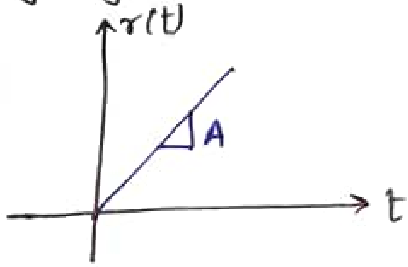
② Step input:



$r(t) = A, \quad t \geq 0$

$R(s) = \frac{A}{s}$

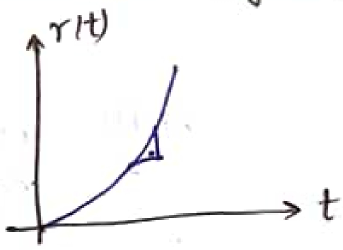
③ Velocity/Ramp input:



$r(t) = At, \quad t \geq 0$
$\quad\;\; = 0, \quad t < 0$

$R(s) = \frac{A}{s^2}$

④ Parabolic/Acceleration input:



$r(t) = \frac{At^2}{2}, \quad t \geq 0$

$R(s) = \frac{A}{s^3}$

⑤ Sinusoidal input:

$r(t) = \sin \omega t$

$R(s) = \frac{\omega}{s^2 + \omega^2}$

Type: Number of poles at origin.

Order: Total number of poles.

Eg. $G(s) = \frac{10}{s+1} \rightarrow$ Type $= 0$, order $= 1$

$G(s) = 50 \rightarrow$ Type $= 0$, order $= 0$

$G(s) = \frac{10}{s^2+s} \rightarrow$ Type $= 1$, order $= 2$

First Order System

$$\frac{C(s)}{R(s)} = \frac{1}{1+sT} \rightarrow \text{Type}=0, \ \text{Order}=1$$

Step response:

$$r(t) = 1 \Rightarrow R(s) = \frac{1}{s}$$

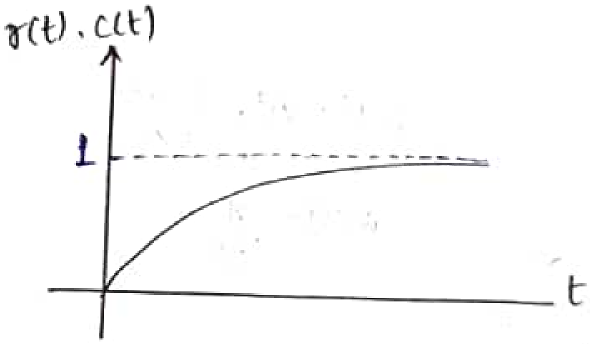
$$C(s) = R(s)\frac{1}{1+sT} = \frac{1}{s}\frac{1}{1+sT} = \frac{A}{s} + \frac{B}{1+sT}$$

$$A = 1, \ B = -T$$

$$\Rightarrow C(s) = \frac{1}{s} - \frac{T}{1+sT} = \frac{1}{s} - \frac{1}{s+1/T}$$

$$\therefore c(t) = 1 - e^{-t/T} \rightarrow \text{Unit step response}$$

T : time constant



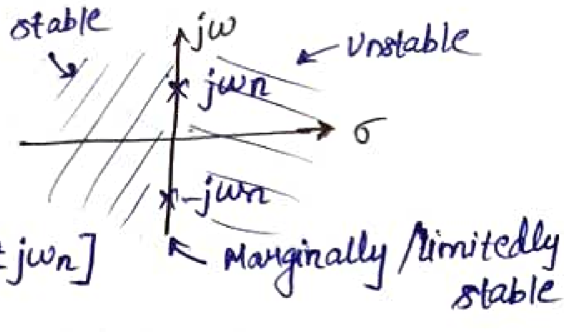
Ramp response : $R(s) = \frac{1}{s^2}$.

Second Order System

$$\frac{C(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

→ $\xi = 0$: Undamped

$$\frac{C(s)}{R(s)} = \frac{\omega_n^2}{s^2+\omega_n^2} \quad [\text{Poles: } s = \pm j\omega_n]$$



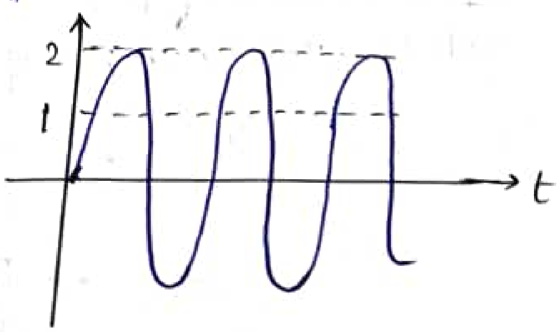
~~Impulse~~ step Response : $R(s) = \frac{1}{s}$ $\quad \therefore C(s) = R(s) \cdot \frac{\omega_n^2}{s^2+\omega_n^2} = \frac{1}{s} \cdot \frac{\omega_n^2}{s^2+\omega_n^2} = \frac{A}{s} + \frac{Bs+C}{s^2+\omega_n^2}$

$$c(t) = 1 - \cos\omega_n t$$

↓

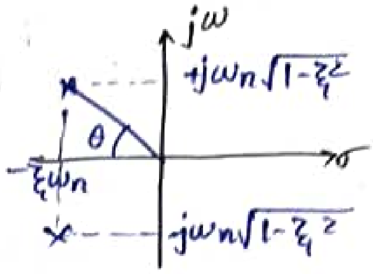
Oscillating

Stability limit



→ $0 < \xi < 1$: Underdamped

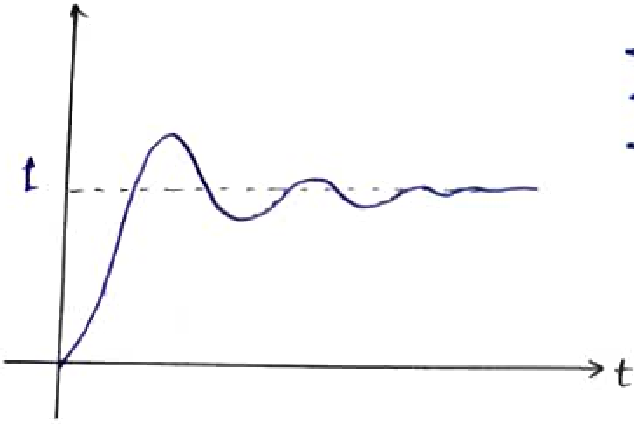
Poles: $s = -\xi\omega_n \pm j\omega_n\sqrt{1-\xi^2}$



$R(s) = 1/s$

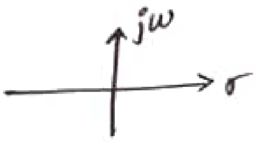
$$\Rightarrow c(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}}\sin(\omega_d t + \theta),$$

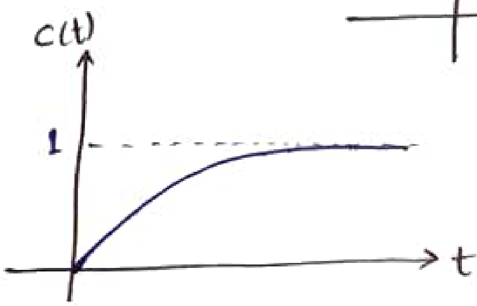
$$\theta = \tan^{-1}\frac{\sqrt{1-\xi^2}}{\xi}$$



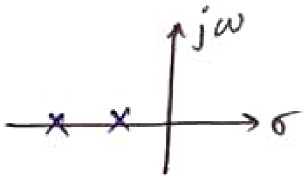
→ overshoot
→ Peak time
→ settling time

→ $\xi = 1$: Critically damped





→ $\xi > 1$: Overdamped



Standard second order system:

Eg. $\frac{25}{s^2+25}$

```
num = 25;
den = [1 0 25];
sys = tf(num, den);
step(sys, 4);    % plot step response for 4 sec.
```

```
t= 0:0.01:10;
x= ones (size(t));    % step input
lsimplot (sys, x, t);  % response of linear system to input (arbitrary)
                                 signal x and t.

X2= 5.* ones (size(t));
lsimplot (sys, X2, t);  % plot step response of amplitude 5
```