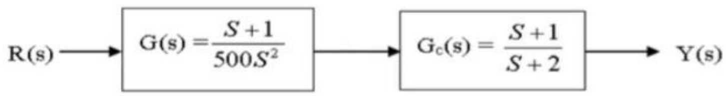
Reduction of Multiple Subsystem

1.

- Find Y(S)



Matlab Code:

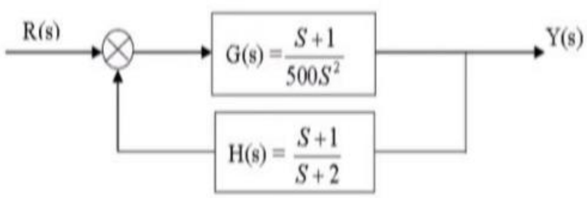
```
>> clear all
>> numg=[1 1];
>> deng=[500 0 0];
>> numh=[1 1];
>> denh=[1 2];
>> [num,den]=feedback(numg,deng,numh,denh,-1);
>> printsys(num,den)
```

Output:

```
num/den =

           s^2 + 3 s + 2
   ----------------------------
   500 s^3 + 1001 s^2 + 2 s + 1
```

1.1

- Find Y(S)



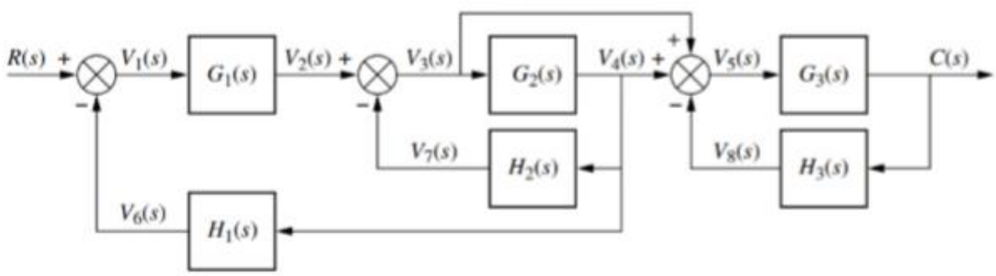
Matlab Code:

```
>> clear all
>> numg=[1 1];
>> deng=[500 0 0];
>> numh=[1 1];
>> denh=[1 2];
>> [num,den]=feedback(numg,deng,numh,denh,-1);
>> printsys(num,den)
```

Output:

```
num/den =

            s^2 + 3 s + 2
   ----------------------------
   500 s^3 + 1001 s^2 + 2 s + 1
```

1.3



Matlab Code:

```
>> clear all
>> g1=tf(1,[1 1]);
>> g2=g1;
>> g3=g1;
>> h1=tf(1,[1 0]);
>> h2=h1;
>> h3=h1;
>> System=append(g1,g2,g3,h1,h2,h3);
>> input=1;
>> output=3;
>> q=[1 -4 0 0 0
       2 1 -5 0 0
       3 2 1 -5 -6
       4 2 0 0 0
       5 2 0 0 0
       6 3 0 0 0];
>> t=connect(System,q,input,output);
>> t=tf(t)
```

Output:

```
t =

                      s^4 + 2 s^3
  ----------------------------------------------------
  s^6 + 3 s^5 + 5 s^4 + 6 s^3 + 4 s^2 + 2 s + 6.901e-17

Continuous-time transfer function.


>> t=minreal(t)

t =

              s^3 + 2 s^2
  -------------------------------------
  s^5 + 3 s^4 + 5 s^3 + 6 s^2 + 4 s + 2

Continuous-time transfer function.
```

Steady State Error

1.

$$G(s) = \frac{1000(s+8)}{(s+7)(s+9)}$$

Determine positional error constant k_p and steady state error e_{ss}

Matlab Code:

```
>> numg=1000*[1 8];
>> deng=poly([-7 -9]);
>> g=tf(numg,deng);
>> kp=dcgain(g);
>> estep=1/(1+kp);
>> kp
```

Output:

```
kp =

  126.9841

>> estep

estep =

    0.0078
```

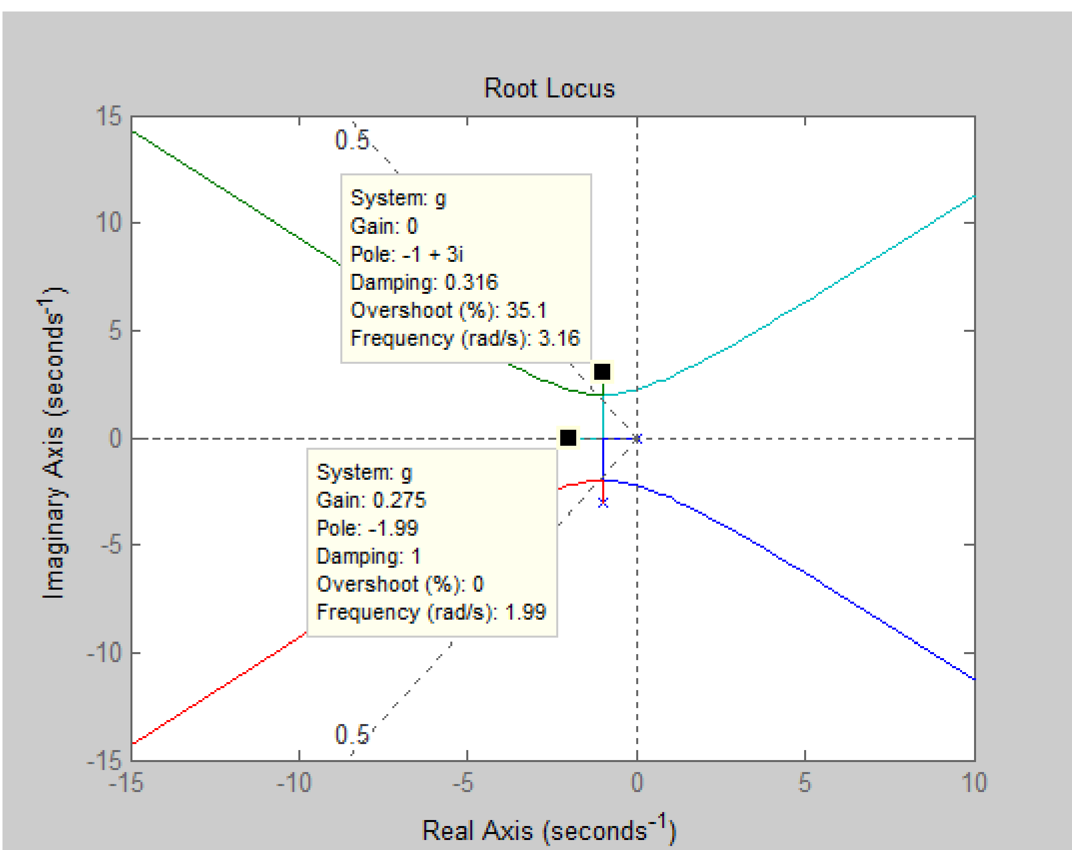
Plot the root locus of the following transfer function using both MATLAB and manually with hand the following transfer function

$$G(s) = \frac{1}{s(s+2)(S^2+2s+10)}$$

Matlab Code:

```
>> clear all
>> num=[1];
>> den=conv([1 2 10],[1 2 0]);
>> g=tf(num,den)

g =

                1
  ---------------------------
  s^4 + 4 s^3 + 14 s^2 + 20 s

Continuous-time transfer function.

>> rlocus(g)
>> z=0.5;
>> sgrid(z,0)
```

Out Put:



Draw the bode plot of the given transfer functions showing minimum stability margins in the figure.

a. $G(s) = \dfrac{10}{s(s+1)(s+2)}$

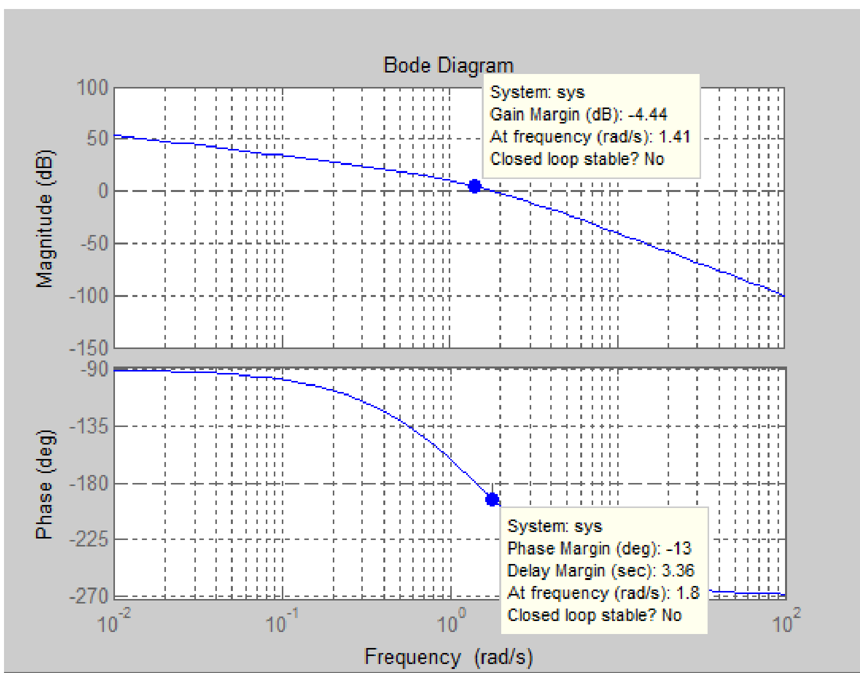
b. $G(s) = \dfrac{1000}{(s+3)(s+4)(s+5)(s+6)}$

c. $G(s) = \dfrac{50(s+3)}{s(s+2)(s+4)}$

A.Matlab Code:

```
>> clear all
>> num=[10];
>> den=poly([0 -1 -2]);
>> bode(num,den)
>> grid on
```
:

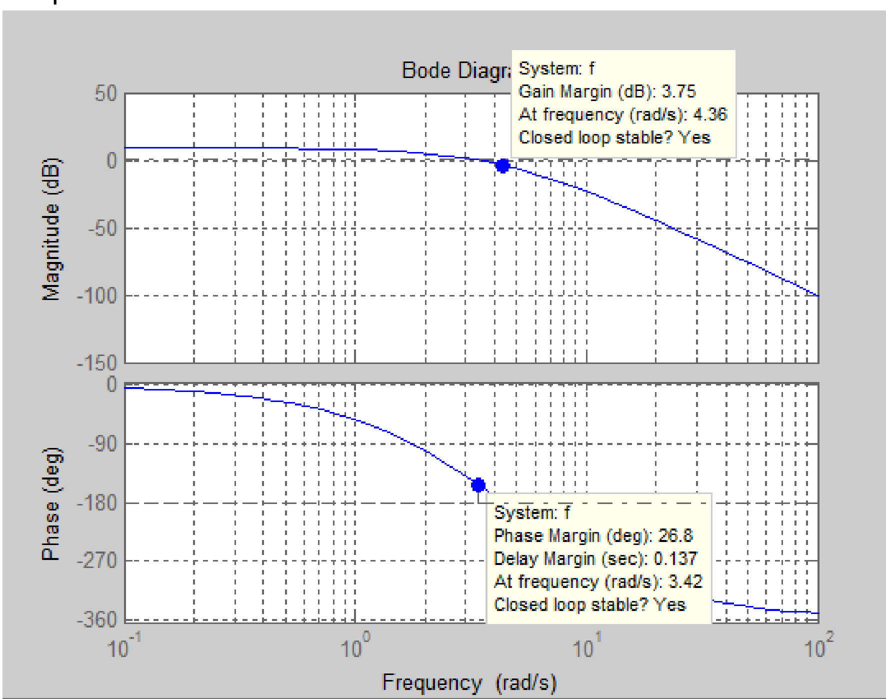
Out put:



B.Matlab Code:

```
>> f=zpk([],[-3 -4 -5 -6],1000)

f =

              1000
   -----------------------
   (s+3) (s+4) (s+5) (s+6)

Continuous-time zero/pole/gain model.

>> bode(f)
>> grid on
```

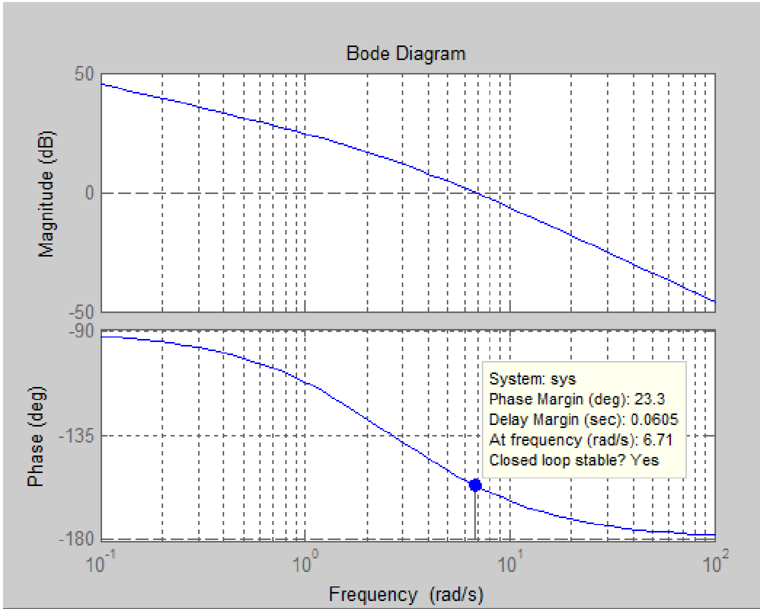
Output:



B. Matlab Code:

```
>> num=50*[1 3];
>> den=poly([0 -2 -4]);
>> bode(num,den)
>> grid on
```

Output:



Determine the Gain Margin, Phase Margin, Gain Cross Over Frequency, Phase Cross Over Frequency of any transfer function using both figure and "margin" command.

a. $G(s) = \dfrac{10}{s(s+1)(s+2)}$

b. $G(s) = \dfrac{1000}{(s+3)(s+4)(s+5)(s+6)}$

c. $G(s) = \dfrac{50(s+3)}{s(s+2)(s+4)}$

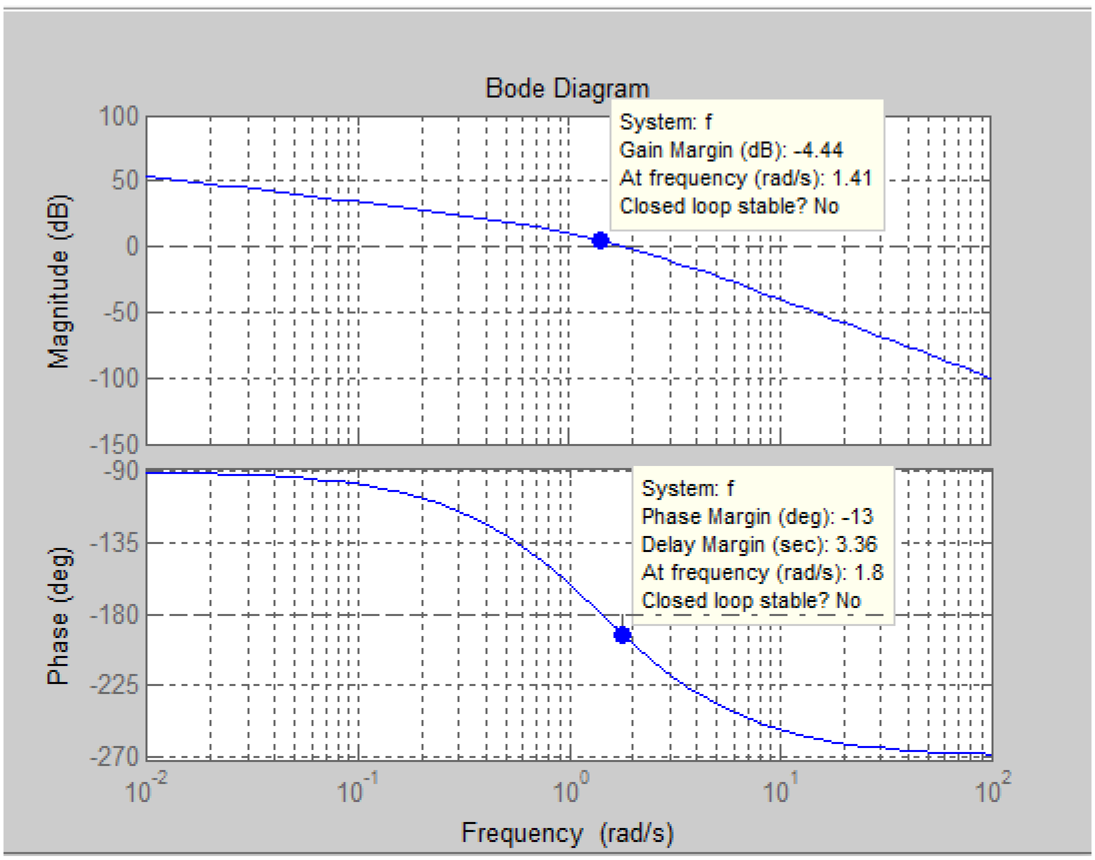
A . Matlab Code:

```
>> num=[10];
>> den=poly([0 -1 -2]);
>> f=tf(num,den);
>> bode(f)
>> [Gm,Pm,Wgc,Wpc]=margin(f);
Warning: The closed-loop system is unstable.
> In warning at 25
  In DynamicSystem.margin at 65
>> Gm_db=20*log10(Gm)
```

Output:

```
Gm =

    0.6000


Pm =

  -12.9919


Wgc =

    1.4142


Wpc =

    1.8020
```

```
Gm_db =

   -4.4370
```

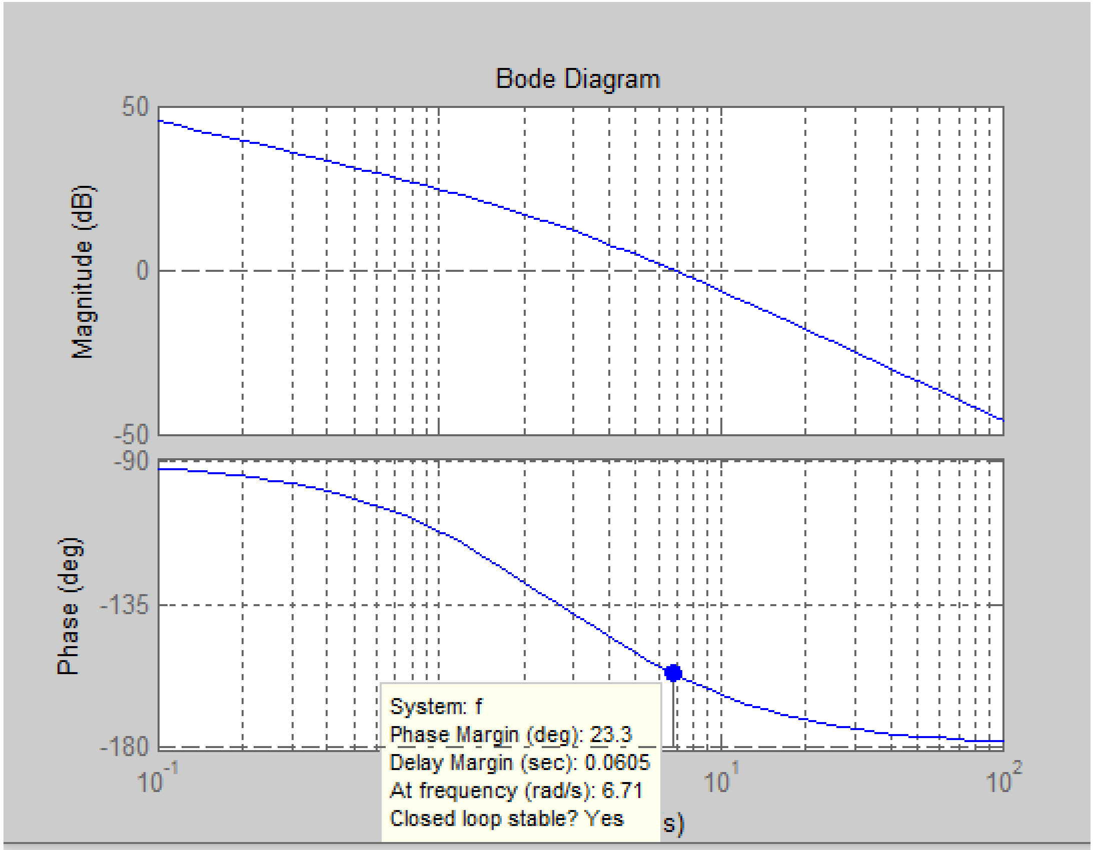


c . Matlab Code:

```
>> f=zpk([-3],[0 -2 -4],50)

f =

     50 (s+3)
  -------------
  s (s+2) (s+4)

Continuous-time zero/pole/gain model.

>> bode(f)
>> grid on
>> [Gm,Pm,Wgc,Wpc]=margin(f);
>> Gm_db=20*log10(Gm);
```

Out put:

```
Gm =

   Inf


Pm =

   23.2941


Wgc =

   Inf


Wpc =

    6.7149

>> Gm_db=20*log10(Gm)

Gm_db =

   Inf
```



Find the bode plot of the following state models showing minimum stability margins in the figure

A.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -25 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 25 \end{bmatrix} u$$

$$y = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

B.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -4 & -1 \\ 3 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u$$

$$y = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

C.

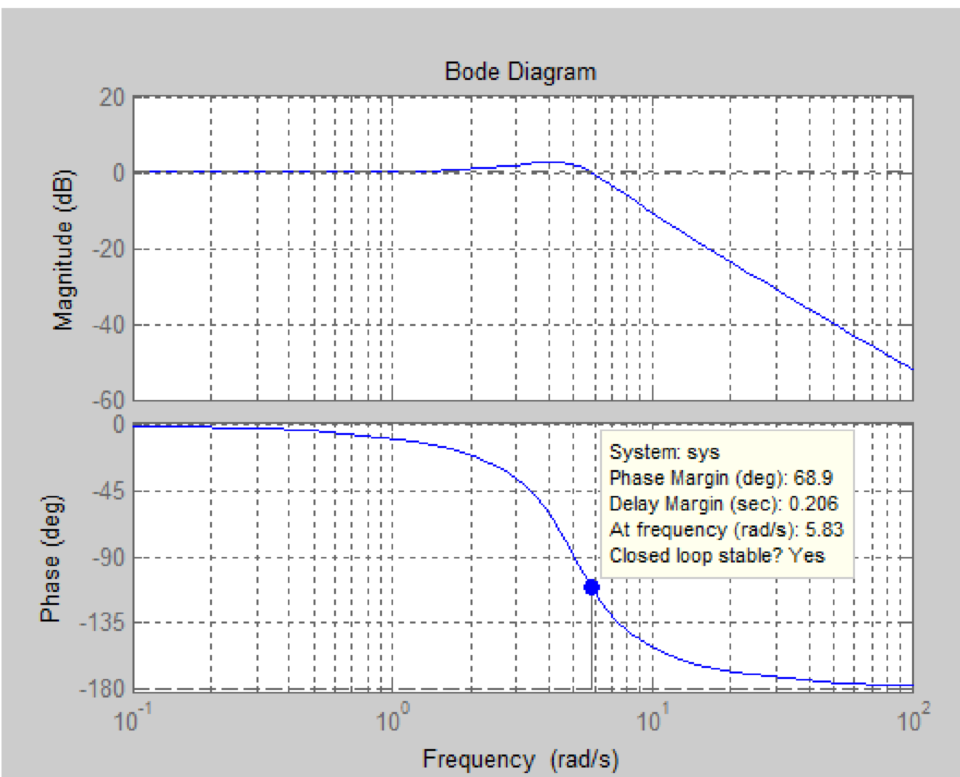
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2 & -4 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

A . Matlab Code:

```
>> clear all
>> A=[0 1;-25 -4];
>> B=[0;25];
>> C=[1 0];
>> D=[0];
>> bode(A,B,C,D)
>> grid on
>> title('Bode Diagram')
```

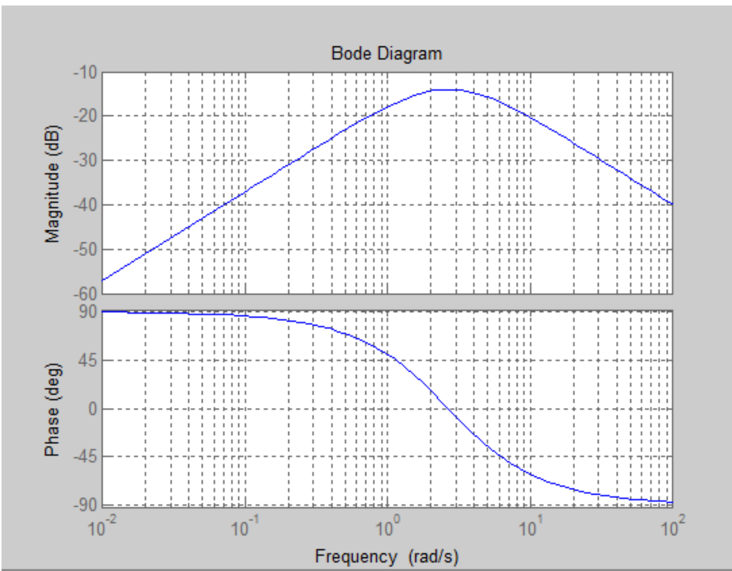
Output:



B .Matlab Code:

```
>> clear all
>> A=[-4 -1;3 -1];
>> B=[1;1];
>> C=[1 0];
>> D=[0];
>> bode(A,B,C,D)
>> grid on
```

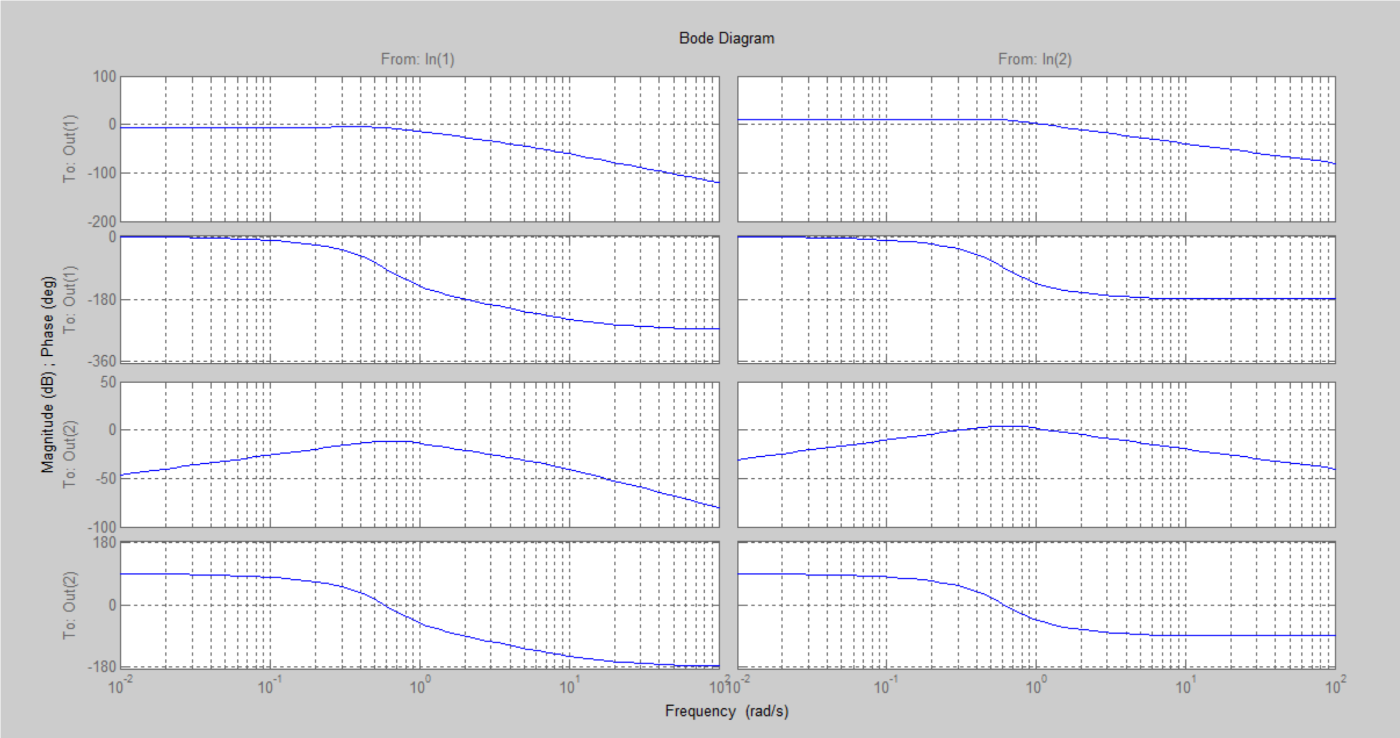
Output:



C . Matlab Code:

```
>> A=[0 1 0;0 0 1;-2 -4 -6];
>> B=[0 0;0 1;1 0];
>> C=[1 0 0;0 1 0];
>> D=[0];
>> bode(A,B,C,D)
>> grid on
```

Output:


Bode Diagram
From: In(1)
From: In(2)
Magnitude (dB) ; Phase (deg)
To: Out(1)
To: Out(2)
Frequency (rad/s)

•Show graphically how the change of damping ratio changes the step response for single transfer function of constant natural frequency of 1 rad/sec which is given by

$$\frac{C(s)}{R(s)} = \frac{1}{s^2 + 2\zeta s + 1}$$

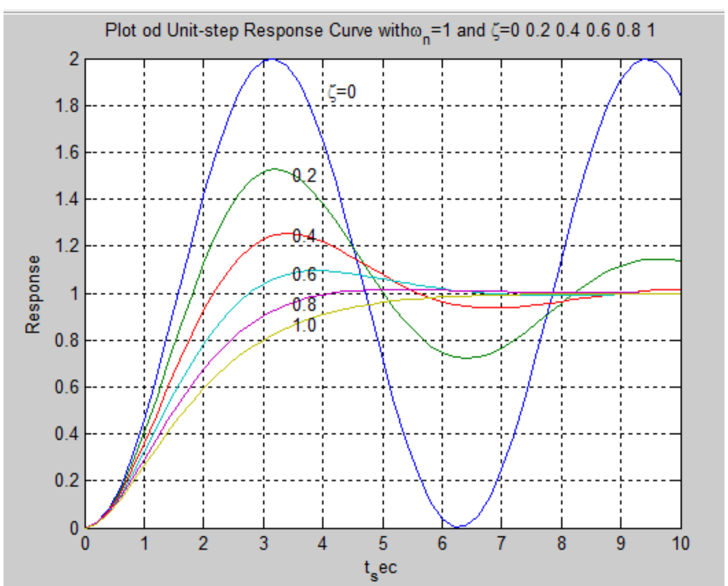
(The undamped natural frequency ω_n is normalized to 1.) Plot unit-step response curves $c(t)$ when ζ assumes the following values:

$$\zeta = 0,\ 0.2,\ 0.4,\ 0.6.\ 0.8,\ 1.0$$

Matlab Code:

```
>> clear all
>> t=0:0.2:10;
>> zeta=[0 0.2 0.4 0.6 0.8 1];
>> for n=1:6;
num=[1];
den=[1 2*zeta(n)*1 1];
[y(1:51,n),x,t]=step(num,den,t);
end
>> plot(t,y)
>> grid
>> title('Plot od Unit-step Response Curve with\omega_n=1 and \zeta=0 0.2 0.4 0.6 0.8 1');
>> xlabel('t_sec')
>> ylabel('Response')
>> text(4.1,1.86,'\zeta=0')
>> text(3.5,1.5,'0.2')
>> text(3.5,1.24,'0.4')
>> text(3.5,1.08,'0.6')
>> text(3.5,0.95,'0.8')
>> text(3.5,0.86,'1.0')
```

Output:



Given the unity feedback system G(S) write a program using MATLAB to determine the value of K for stability.

$$G(s) = \frac{Ks(s+2)}{(s^2 - 4s + 8)(s+3)}$$

Matlab Code:

```
>> K=[0:0.001:200];
>> for i=1:length(K);
den=conv([1 -4 8],[1 3]);
num=[0 K(i) 2*K(i) 0];
dent=num+den;
R=roots(dent);
A=real(R);
B=max(A);
if B<0
R
K=K(i)
break
end
end
```

Output:

```
R =

  -4.0000 + 0.0000i
  -0.0000 + 2.4495i
  -0.0000 - 2.4495i


K =

     5
```

Step Responses of Second order systems according to pole movement

• Prove using MATLAB plot that

1. Frequency of oscillation remains the same for constant imaginary part

2. Envelope remains the same for constant real part

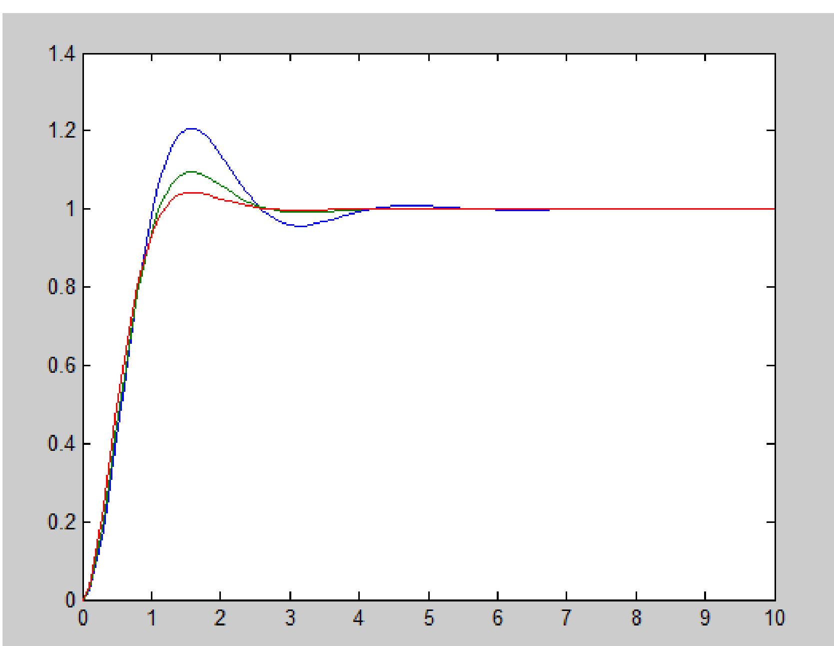
3. Overshoot remains the same for same damping ratio.

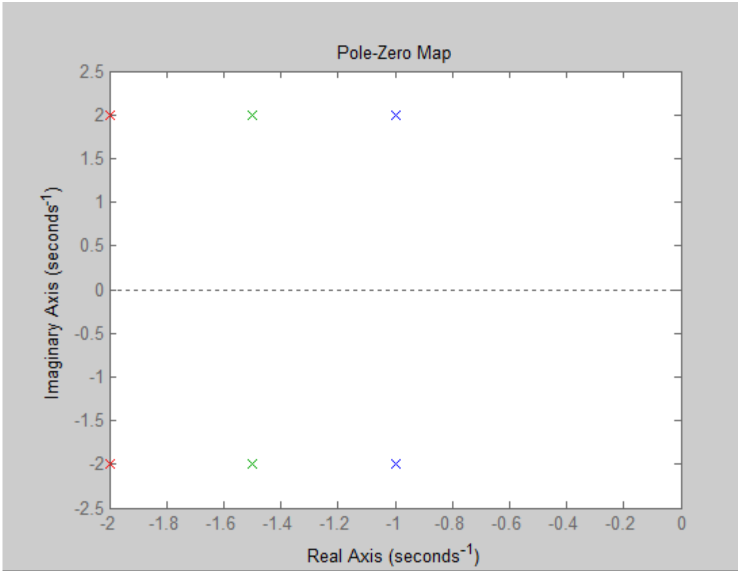
Solution: Frequency of oscillation remains the same for constant imaginary part

1 . Matlab code:

```
>> den1=poly([-1+2i -1-2i]);
>> den2=poly([-1.5+2i -1.5-2i]);
>> den3=poly([-2+2i -2-2i]);
>> num1=den1(3);
>> num2=den2(3);
>> num3=den3(3);
>> f1=tf(num1,den1);
>> f2=tf(num2,den2);
>> f3=tf(num3,den3);
>> pzmap(f1,f2,f3)
>> t=0:0.1:10;
>> c1=step(num1,den1,t);
>> c2=step(num2,den2,t);
>> c3=step(num3,den3,t);
>> plot(t,c1,t,c2,t,c3)
```

Output



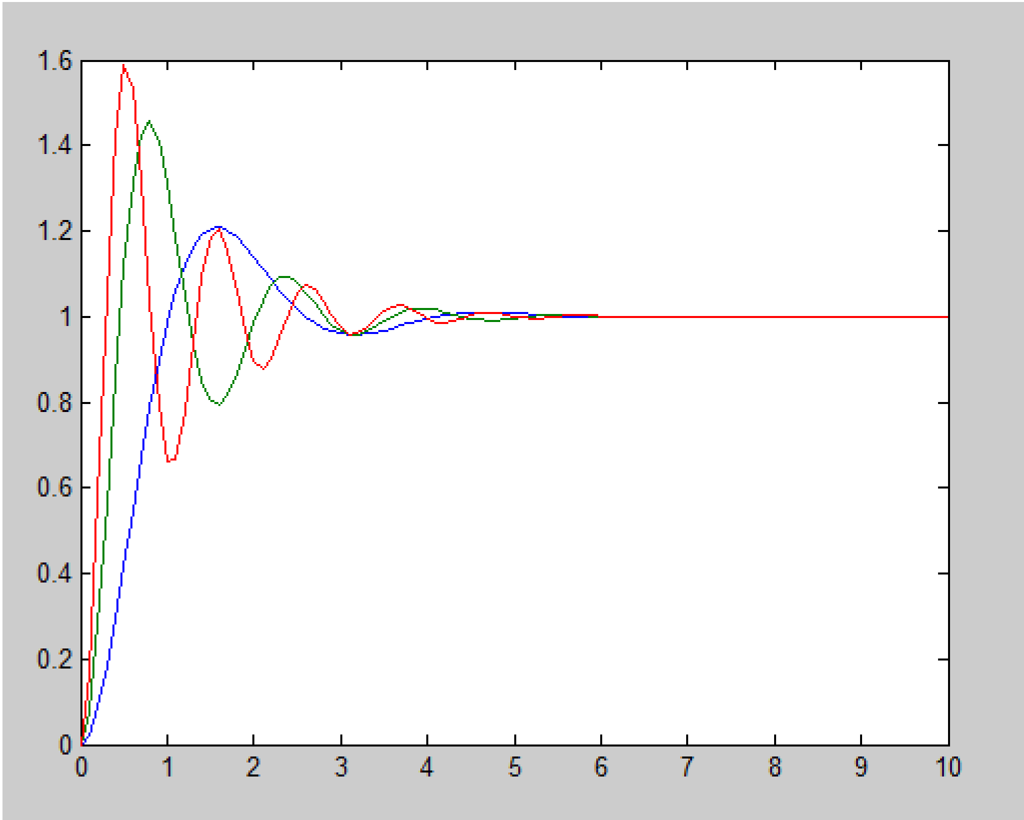


2. Envelope remains the same for constant real part

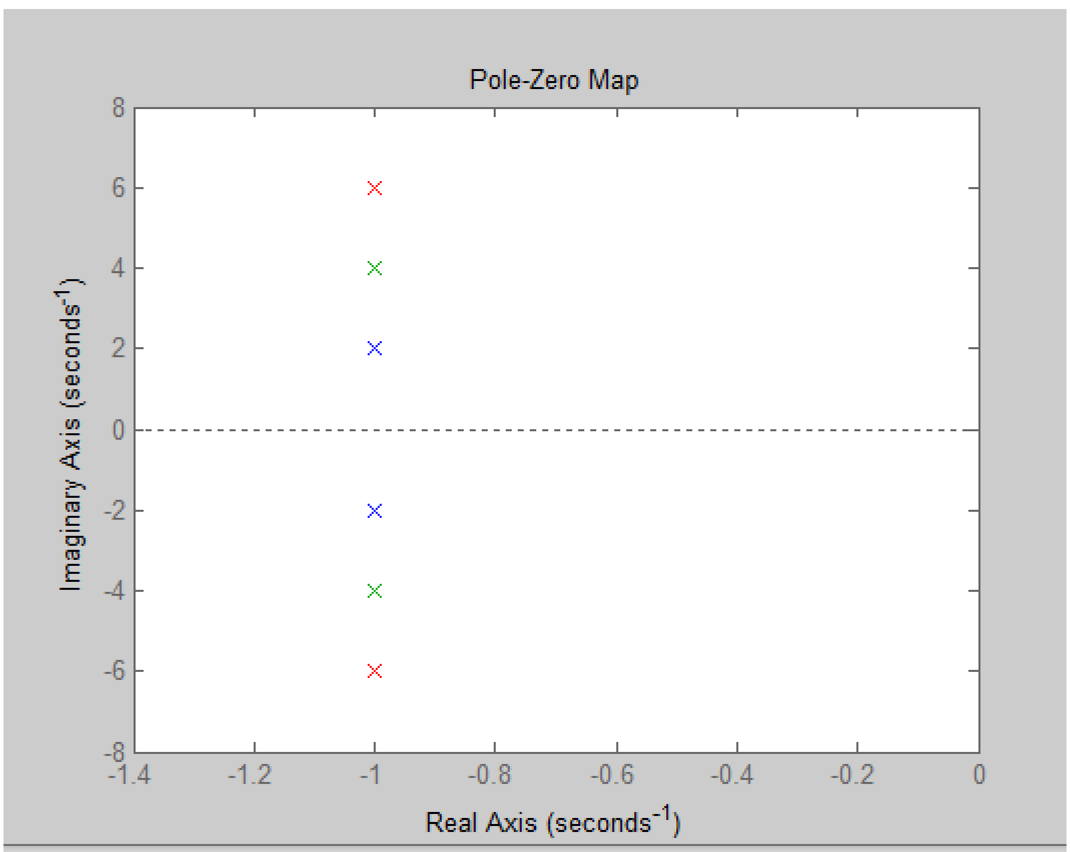
Matlab Code:

```
>> den1=poly([-1+2i -1-2i]);
>> den2=poly([-1+4i -1-4i]);
>> den3=poly([-1+6i -1-6i]);
>> num1=den1(3);
>> num2=den2(3);
>> num3=den3(3);
>> f1=tf(num1,den1);
>> f2=tf(num2,den2);
>> f3=tf(num3,den3);
>> pzmap(f1,f2,f3)
>> figuure
Undefined function or variable 'figuure'.

Did you mean:
>> figure
>> t=0:0.1:10;
>> c1=step(num1,den1,t);
>> c2=step(num2,den2,t);
>> c3=step(num3,den3,t);
>> plot(t,c1,t,c2,t,c3)
>>
```

Output:



1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0 1 2 3 4 5 6 7 8 9 10



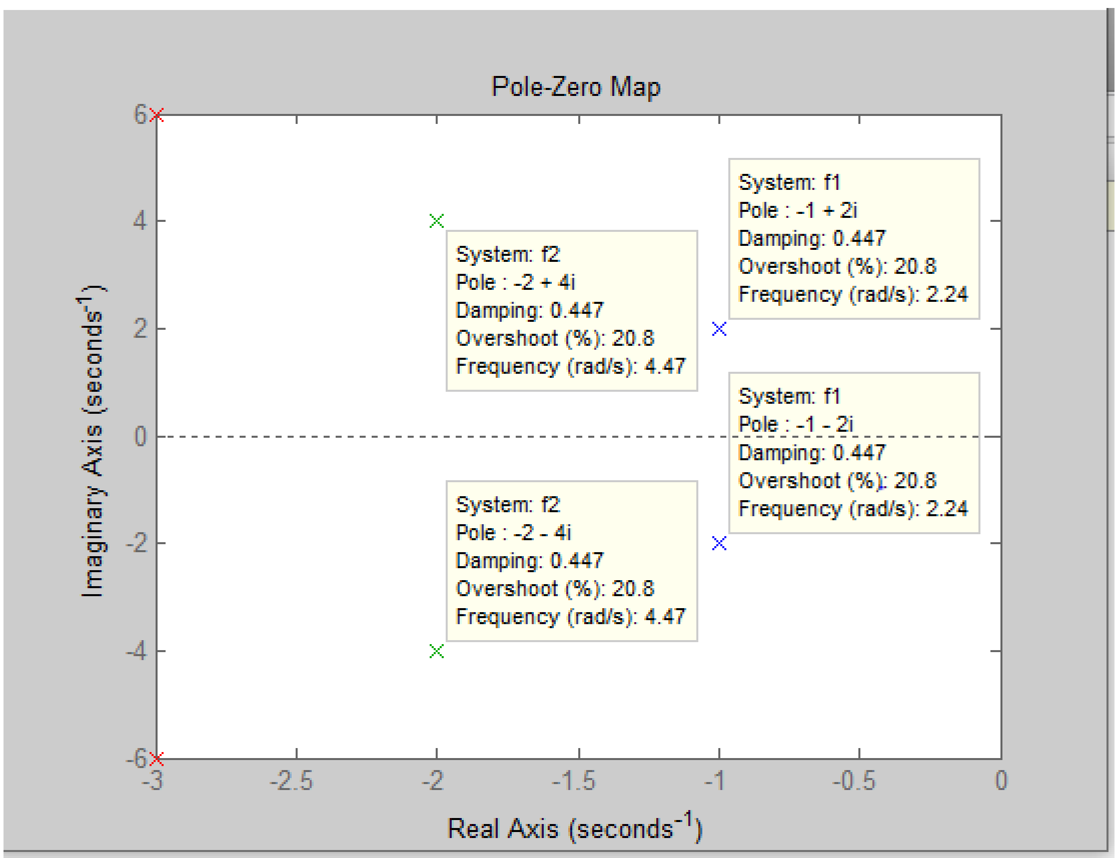
Pole-Zero Map
Imaginary Axis (seconds^{-1})
8
6
4
2
0
-2
-4
-6
-8
-1.4 -1.2 -1 -0.8 -0.6 -0.4 -0.2 0
Real Axis (seconds^{-1})

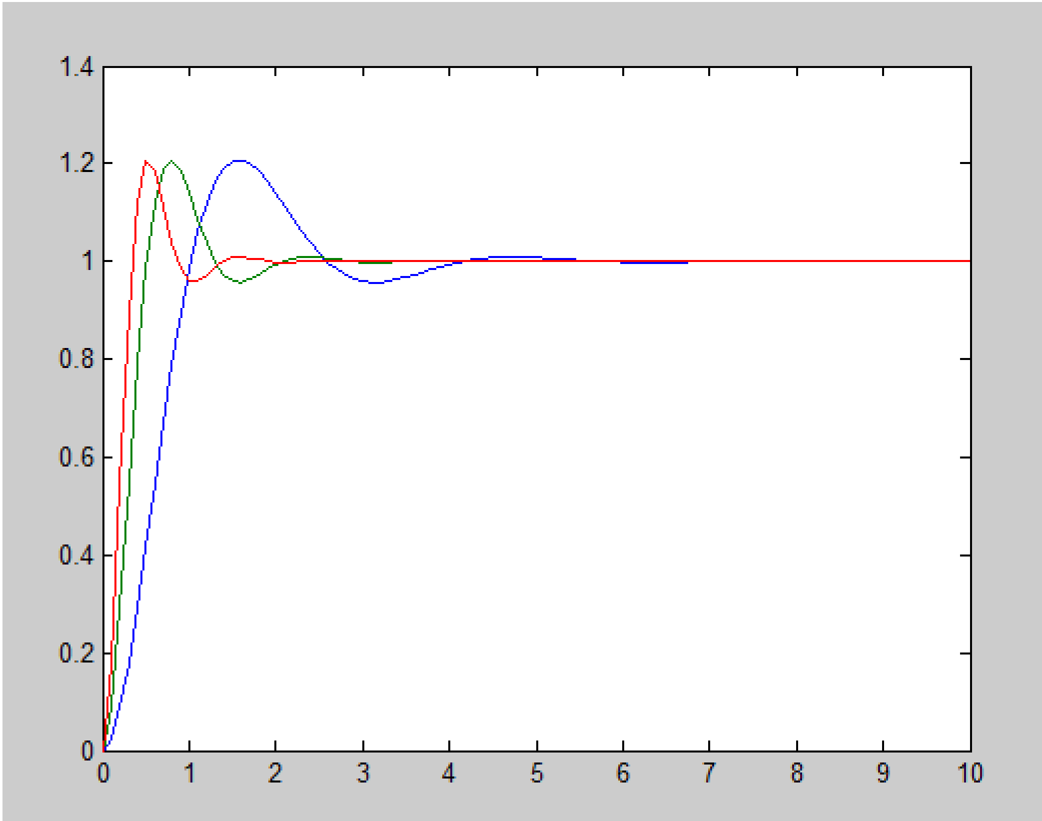
3. Overshoot remains the same for same damping ratio.

Matlab Code:

```
>> den1=poly([-1+2i -1-2i]);
>> den2=poly([-2+4i -2-4i]);
>> den3=poly([-3+6i -3-6i]);
>> num1=den1(3);
>> num2=den2(3);
>> num3=den3(3);
>> f1=tf(num1,den1);
>> f2=tf(num2,den2);
>> f3=tf(num3,den3);
>> pzmap(f1,f2,f3)
>> t=0:0.1:10;
>> c1=step(num1,den1,t);
>> c2=step(num2,den2,t);
>> c3=step(num3,den3,t);
>> plot(t,c1,t,c2,t,c3)
```

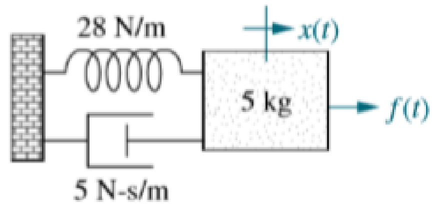
Output:




1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
1
2
3
4
5
6
7
8
9
10

- For the figure given find using MATLAB

1. Rise Time Tr
2. Peak Time Tp
3. Percentage Overshoot %OS
4. Settling Time Ts
5. Step Response



The Transfer function should be determined by hand and included in the lab report. While measuring system parameters consider the denominator of the transfer function only.

Matlab Code:

```
>> w=1/5;
>> a=1;
>> num=w;
>> den=[1 a w];
>> t=tf(num,den);
>> wn=sqrt(w);%natural frequency
>> zeta=a/(2*wn);%Damping Ratio
>> theta=acos(zeta);
>> Ts=4/(zeta*wn);%Settling Time
>> Tp=pi/(wn*sqrt(1-zeta^2));%Peak Time
>> Tr=(pi-theta)/(wn*sqrt(1-zeta^2));%Rise Time
>> OS=exp(-zeta*pi/sqrt(1-zeta^2))*100; %Percentage Overshoot
>> step(t)
```

Output:

```
wn =

    0.4472

>> zeta=a/(2*wn)

zeta =

    1.1180

>> Ts=4/(zeta*wn)

Ts =

     8

>> Tp=pi/(wn*sqrt(1-zeta^2))

Tp =

   0.0000 -14.0496i
```

```
>> Tr=(pi-theta)/(wn*sqrt(1-zeta^2))

Tr =

  -2.1520 -14.0496i

>> OS=exp(-zeta*pi/sqrt(1-zeta^2))*100

OS =

  73.7369 +67.5490i
```