EE324 CONTROL SYSTEMS LAB

Problem Sheet 3

Yatish Vaman Patil | 190070076

Question 1: Pole Zero Cancellation

Part A)

$$Gs = \frac{s + 5 + a}{s^2 + 11s + 30}$$

In this section, we have to vary the value of parameter 'a' from -1 to 1 in the step of '0.1'. When the value of 'a' is '0', the numerator becomes 's+5.'So the zero at -5 is cancelled with the pole at -5.

$$Gs = \frac{s + 5}{(s + 5)(s + 6)} = \frac{1}{s + 6}$$

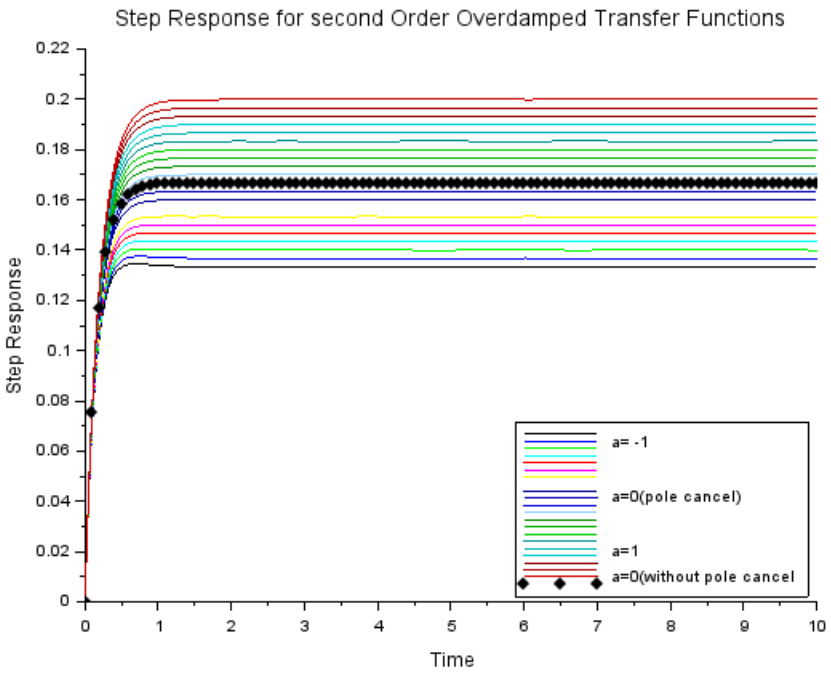


Figure: Pole Zero Cancellation

Scilab code for the same:

```
s=%s;
a=-1;
for i=1:21
    Ns(i)= s+5+a;
    Ds(i)= s^2+11*s+30;
    a=a+0.1;
end
[Ns1,Ds1]=simp(Ns,Ds);
Hs=syslin('c',Ns1,Ds1);
time=0:0.01:10;
Ps=csim('step',time,Hs);
plot2d(time,Ps');
time1=0:0.1:10
Ps0=csim('step',time1,syslin('c',(s+5)/(s^2+11*s+30)));
plot2d(time1,Ps0,-4) ;
leg1=legend('a= -1','','','','','','','','','','a=0(pole
cancel)','','','','','','','','','','a=1','a=0(without pole cancel',4)
xtitle('Step Response for second Order Overdamped Transfer
Functions','Time','Step Response');
```

Part B)

$$Gs = \frac{1}{s^2 - s - 6}$$

In this section, we have to introduce the RHP zero to the given transfer function, such that it cancels the RHP pole at 's=3.'First, we introduce zero at s=3. So the transfer function becomes

$$Gs = \frac{s - 3}{(s - 3)(s + 2)} = \frac{1}{s + 2}$$

Now, if we try to add zero slightly away from 3. Lets newly introduced zero be at s=3.01, transfer function becomes

$$Gs = \frac{s - 3.01}{(s - 3)(s + 2)}$$

In this case, pole and zero are not cancelling.

If we take the step response of the system, we get

$$Cs = \frac{s - 3.01}{s(s-3)(s+2)}$$

By taking inverse Laplace Transform, we get

$$c(t) = -\frac{e^{3t}}{1500} + \frac{301}{600} - \frac{501e^{-2t}}{1000}$$

So in this step response function, we can see that as the RHP pole is not cancelled altogether, it has appeared in the form of exponential increasing function. Due to this, the system is unstable.

Hence in practice, it is almost impossible to introduce the zero so accurately. Thus this system cant be made stable by pole-zero cancellation

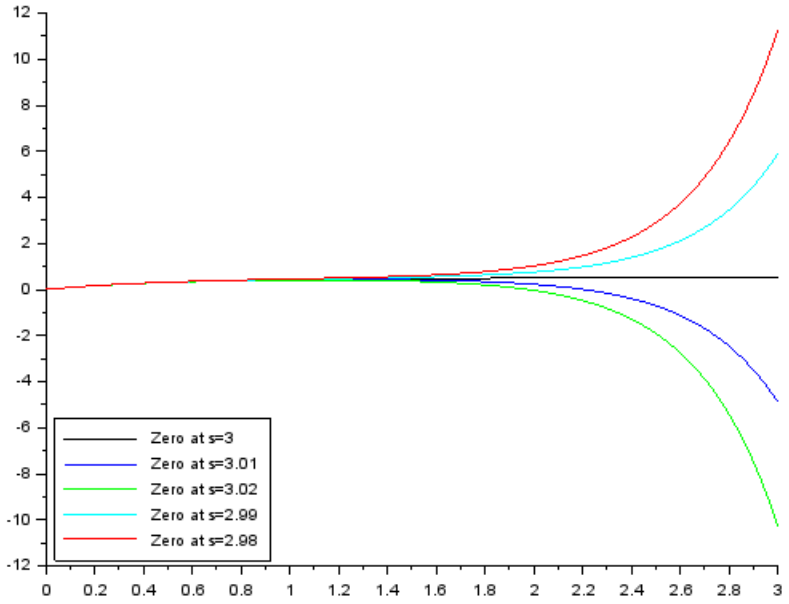


Figure: Pole Zero Cancellation for Unstable Plant

Scilab code for the same:

```
s=%s;
Gs1=1/(s^2-s-6);
time=0:0.01:3;

Gs2=(s-3);
Gs_final=syslin('c',Gs1*Gs2);
Ps2=csim('step',time,Gs_final);
```

```
Gs3=(s-3.01);
Gs_final1=syslin('c',Gs1*Gs3);
Ps3=csim('step',time,Gs_final1);

Gs4=(s-3.02);
Gs_final2=syslin('c',Gs1*Gs4);
Ps4=csim('step',time,Gs_final2);

Gs5=(s-2.99);
Gs_final3=syslin('c',Gs1*Gs5);
Ps5=csim('step',time,Gs_final3);

Gs6=(s-2.98);
Gs_final4=syslin('c',Gs1*Gs6);
Ps6=csim('step',time,Gs_final4);

plot2d(time,[Ps2',Ps3',Ps4',Ps5',Ps6']);

leg2=legend('Zero at s=3','Zero at s=3.01','Zero at s=3.02','Zero at
s=2.99','Zero at s=2.98',3)
```

Question 2: Second-Order Approximation

Part A)

$$Gs = \frac{85}{s^3 + 7s^2 + 27s + 85}$$

This section makes a second-order approximation for a higher-order system(here, 3rd order). Poles of the above transfer function are

Poles: s=-5

s= -1-4i

s=-1+4i

The real part of imaginary poles=-1

The real pole =-5

For taking second-order approximation, the real part of the imaginary pole(a) should be tiny than the real pole(b). In general, we consider the following relation

|B|>|5a|

In this case, the real Pole is 5 times that of the real part of imaginary poles. Hence we can make a second-order approximation.

Now my Gs will become

$$Gs = \frac{5 * 17}{(s+5)(s^2+2s+17)}$$

We can approximate this Transfer function as following

$$Gs = \frac{17}{s^2+2s+17}$$

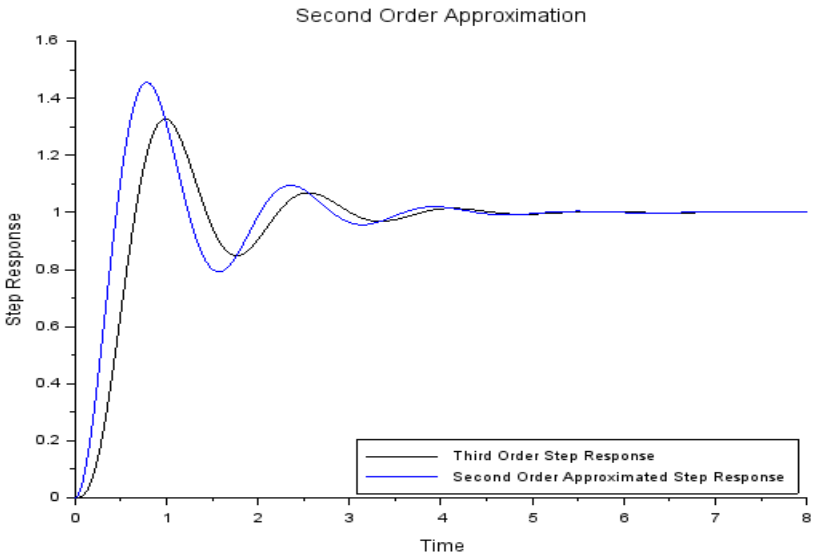


Figure: Second-Order Approximation

Scilab code for the same:

```
s=%s;
Gs1=85/(s^3+7*s^2+27*s+85);
Gs2=17/(s^2+2*s+17);
Gs1=syslin('c',Gs1);
Gs2=syslin('c',Gs2);
time=0:0.01:8;
Ps1=csim('step',time,Gs1);
Ps2=csim('step',time,Gs2);

plot2d(time,[Ps1',Ps2']);

leg1=legend('Third Order Step Response','Second Order Approximated
Step Response',4);
xtitle('Second Order Approximation','Time','Step Response');
```

Part B)

$$Gs = \frac{s + 0.01}{s^3 + \left(\frac{101}{50}\right)s^2 + \left(\frac{126}{25}\right)s + 0.1}$$

In this section, we have to find criteria for Pole zero cancellation. After simplifying the above transfer function becomes

$$Gs = \frac{(s + 0.01)}{(s + 0.02)(s^2 + 2s + 5)}$$

We can make pole-zero cancellation when the residue of the pole closest to zero is negligible compare to the residue of other poles at the same zero. We can find the residue by taking the inverse Laplace transform of the Response function.

$$\text{Response Function=Cs=}\frac{(s+0.01)}{s(s+0.02)(s^2+2s+5)} = \frac{0.1}{s} + \frac{0.1}{s+0.02} - \frac{0.2s+0.4}{s^2+2s+5}$$

$$C(t) = -\frac{(24651\sin(2t) + 49802\cos(2t))e^{-t}}{248020} + \frac{1}{10} + \frac{1250e^{-\frac{t}{50}}}{12401}$$

The above equation shows that pole residue at 's=0.02' (0.1) is comparable to other residues. Hence we conclude that pole-zero cancellation is not possible. But if we try following are the results

$$Cs2 = \frac{0.1}{s} - \frac{0.2s+0.4}{s^2+2s+5}$$

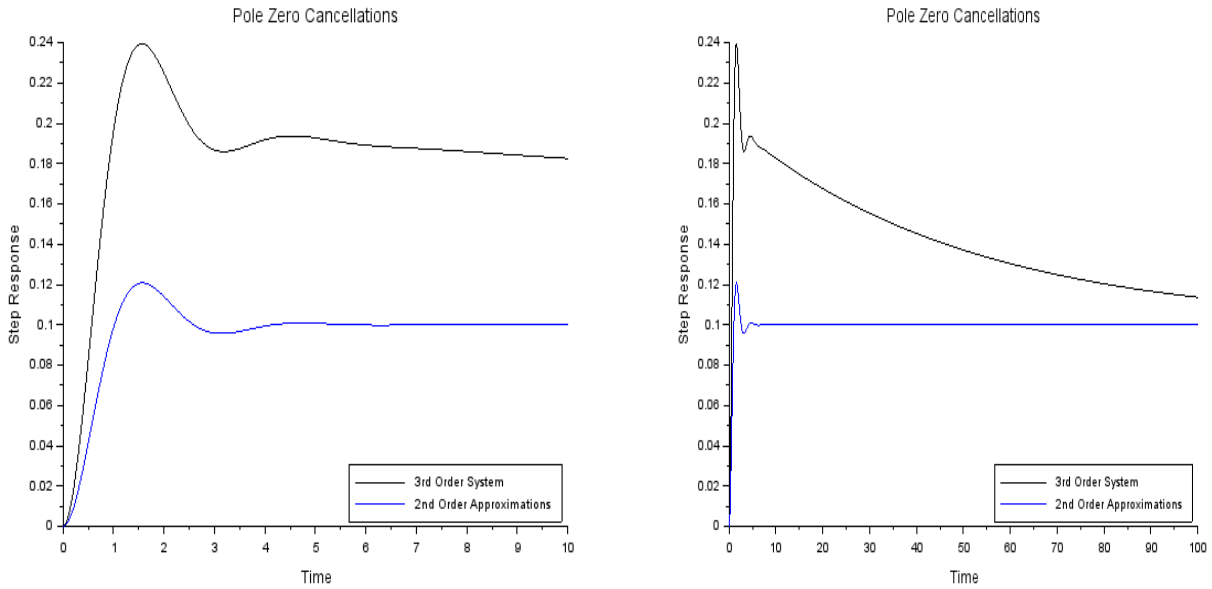


Figure: Pole Zero Cancellation

Scilab code for the same:

```
s=%s;
Gs1=(s+0.01)/(s^3+(101/50)*s^2+(126/25)*s+0.1);
Gs2=0.5/(s^2+2*s+5);
Gs1=syslin('c',Gs1);
Gs2=syslin('c',Gs2);
time=0:0.01:10;
Ps1=csim('step',time,Gs1);
Ps2=csim('step',time,Gs2);

plot2d(time,[Ps1',Ps2']);

leg1=legend('3rd Order System','2nd Order Approximations',4);
xtitle('Pole Zero Cancellations','Time','Step Response');
```

Question 3: Effect of Additional Poles and Zeros

Part A)

$$Gs = \frac{9}{s^2 + 2s + 9}$$

Let step response of the above function be Cs

$$Cs = \frac{1}{s} \frac{9}{s^2 + 2s + 9}$$

Now we have to add zero to the above transfer function and then analyse its step response. Let the zero to be added is at 3. Transfer function becomes

$$Gs = \frac{9\left(\frac{s}{3} + 1\right)}{s^2 + 2s + 9}$$

We can do the following operations on the transfer function

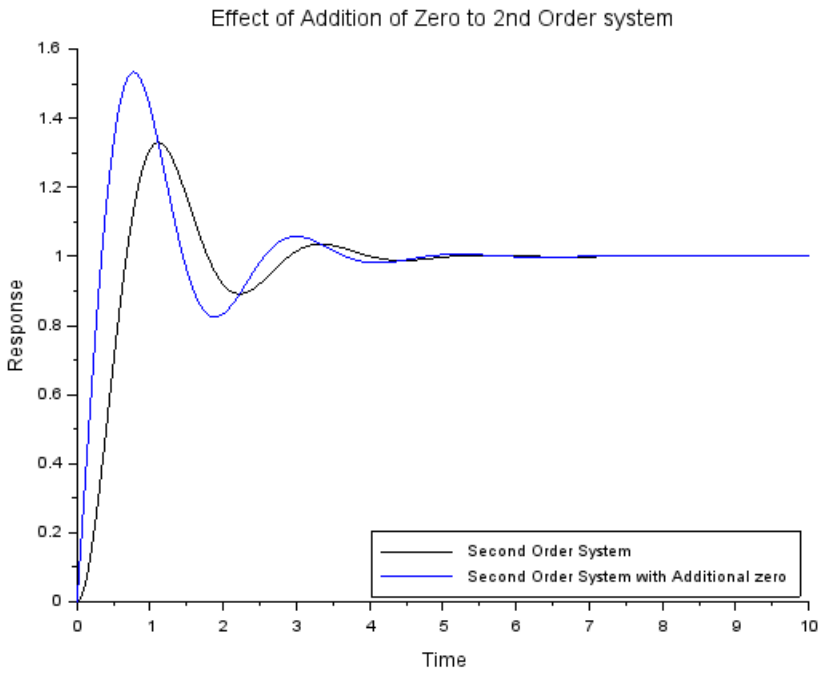
$$Gs = \left(\frac{s}{3}\right)\left(\frac{9}{s^2 + 2s + 9}\right) + \frac{9}{s^2 + 2s + 9}$$

Step Response will be

$$Ys = Cs + \left(\frac{s}{3}\right)Cs$$

$$y(t) = c(t) + \frac{c'(t)}{3}$$

Here we can observe that the derivative of the step response is added to the step response of the original function with a weight of zero. Hence graph for a system with an additional pole will be steeper; therefore, shorter rise time and higher percentage overshoot.




```
--> exec('C:\Users\YATISH\Documents\SEMESTER 5\EE324-Control Systems Lab\Lab 03\Q3_A.sce', -1)
Normal 2nd Order System:
 Rise Time=0.455000
 Percent Overshoot=32.932141
Underdamped System with Additional zero:
 Rise Time=0.266000
 Percent Overshoot=53.305841
--> clear
```

Scilab code for the same:

```
function Tr=Rise_time(x2)
    i=1;
    while x2(i)<0.1
        i=i+1
    end
    j=1;
    while x2(j)<0.9
        j=j+1;
    end
    Tr=(j-i)/1000;
endfunction

function OS=Percent_OS(x5, gainK)
    [m6,n6]=max(x5);
```

```
    OS=((m6-gainK)/gainK)*100;
endfunction

s=%s;
time=0:0.001:10;
Gs1=syslin('c',9/(s^2+2*s+9));
Gs2=syslin('c',9*(s/3+1)/(s^2+2*s+9));
Ps1=csim('step',time,Gs1);
Ps2=csim('step',time,Gs2);
plot2d(time,[Ps1',Ps2']);
leg1=legend('Second Order System','Second Order System with Additional
zero',4);
xtitle('Effect of Addition of Zero to 2nd Order
system','Time','Response');

tr1=Rise_time(Ps1);
OS1=Percent_OS(Ps1,1);
tr2=Rise_time(Ps2);
OS2=Percent_OS(Ps2,1);

printf('Normal 2nd Order System:\n Rise Time=%f',tr1);
printf('\n Percent Overshoot=%f',OS1);
printf('\nUnderdamped System with Additional zero:\n Rise
Time=%f',tr2);
printf('\n Percent Overshoot=%f',OS2);
```

Part B)

$$Gs = \frac{9}{s^2 + 2s + 9}$$

Let step response of the above function be Cs

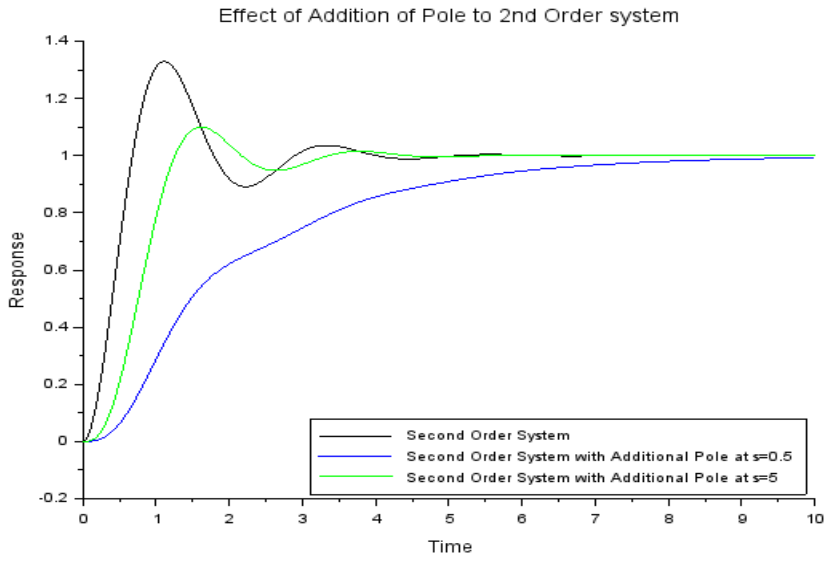
$$Cs = \frac{1}{s} \frac{9}{s^2 + 2s + 9}$$

We have to add two Poles one by one to the above transfer function and then analyse its step response. Let the Poles to be added are at 5 and 0.5. Transfer function becomes

$$Gs = \frac{9}{(s^2 + 2s + 9)(\frac{s}{5} + 1)}$$

$$Gs = \frac{9}{(s^2 + 2s + 9)(\frac{s}{0.5} + 1)}$$

We can observe that step response does not overshoot when adding poles closer to the origin than imaginary poles. When adding a pole away from the origin, step response becomes slow, i.e. with lower percentage overshoot and more considerable rise time.




```
--> exec('C:\Users\YATISH\Documents\SEMESTER 5\EE324-Control Systems Lab\Lab 03\Q3_B.sce', -1)
Normal 2nd Order System:
 Rise Time=0.455000
 Percent Overshoot=32.932141
Underdamped System with Additional Pole at s=0.5:
 Rise Time=4.189000
Underdamped System with Additional Pole at s=5:
 Rise Time=0.751000
 Percent Overshoot=9.961722
```

Scilab code for the same:

```
function Tr=Rise_time(x2)
    i=1;
    while x2(i)<0.1
        i=i+1
    end
    j=1;
    while x2(j)<0.9
        j=j+1;
    end
    Tr=(j-i)/1000;
endfunction

function OS=Percent_OS(x5, gainK)
    [m6,n6]=max(x5);
    OS=((m6-gainK)/gainK)*100;
endfunction
```

```
s=%s;
time=0:0.001:10;
Gs1=syslin('c',9/(s^2+2*s+9));
Gs2=syslin('c',9/((s/0.5+1)*(s^2+2*s+9)));
Gs3=syslin('c',9/((s/2+1)*(s^2+2*s+9)));
Ps1=csim('step',time,Gs1);
Ps2=csim('step',time,Gs2);
Ps3=csim('step',time,Gs3);
plot2d(time,[Ps1',Ps2',Ps3']);
leg1=legend('Second Order System','Second Order System with Additional
Pole at s=0.5','Second Order System with Additional Pole at s=5',4);
xtitle('Effect of Addition of Pole to 2nd Order
system','Time','Response');

tr1=Rise_time(Ps1);
OS1=Percent_OS(Ps1,1);
tr2=Rise_time(Ps2);
OS2=Percent_OS(Ps2,1);
tr3=Rise_time(Ps3);
OS3=Percent_OS(Ps3,1);

printf('Normal 2nd Order System:\n Rise Time=%f',tr1);
printf('\n Percent Overshoot=%f',OS1);
printf('\nUnderdamped System with Additional Pole at s=0.5:\n Rise
Time=%f',tr2);
printf('\nUnderdamped System with Additional Pole at s=5:\n Rise
Time=%f',tr3);
printf('\n Percent Overshoot=%f',OS3);
```

Part C)

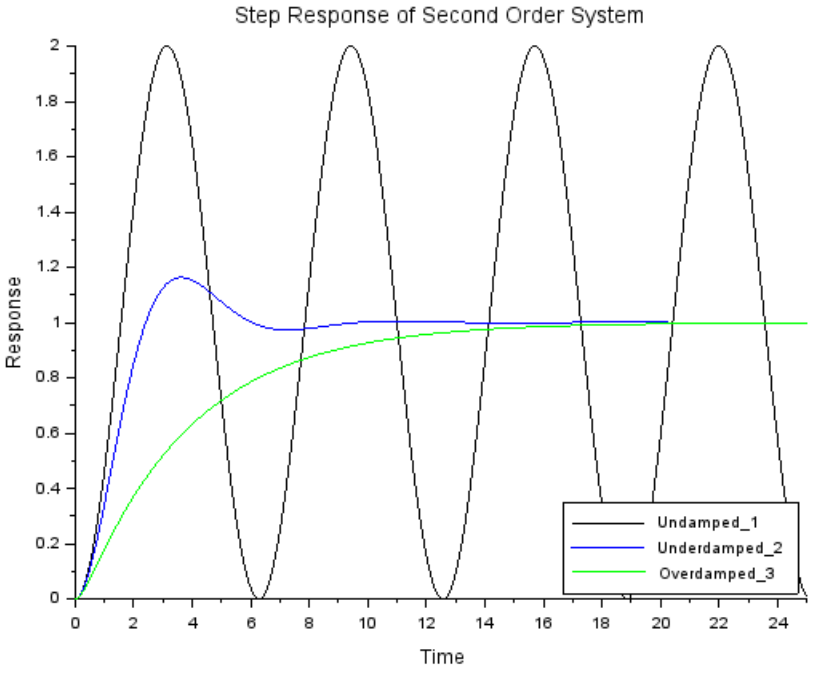
- When we add zero to the second-order system, percentage overshoot increases and rise time increases. Peak time also decreases
- When we add pole close to the origin than the original poles of the second-order system, we get overdamped like the situation with 0% Overshoot and rise time also increases
- When we add further from origin than the original poles of the system, step response is still slower than original response, i.e. low %OS and high rise time. But as the introduced pole goes further from origin, step response approaches the original response

Question 4: Different Second-Order Systems:

Part A)

In this part, we have to consider three types of second-order systems Undamped, Underdamped, and Overdamped, respectively.

- Undamped System = $\frac{1}{s^2+1}$
- Underdamped System = $\frac{1}{s^2+s+1}$
- Overdamped System = $\frac{1}{s^2+4s+1}$




```
--> exec('C:\Users\YATISH\Documents\SEMESTER 5\EE324-Control Systems Lab\Lab 03\Q4_A.sce', -1)
Result for Undamped System is:
 Peak Time=3.141593
Result for Underdamped System is:
 Peak Time=3.629000
Delay Time=1.295000
Rise Time=1.638000
Settling Time=8.077000
Percentage OS=16.303352
Result for Overdamped System is:
 Delay Time=2.866000
Rise Time=8.229000
Set Time=14.878000
```

Scilab code for the same:

```
function Td=Delay_time(x1)
    y1=x1-0.5*ones(x1);
    a1=abs(y1);
    [m1,n1]=min(a1);
    Td=n1/1000;
endfunction

function Tr=Rise_time(x2)
    y2=x2-0.1*ones(x2);
    a2=abs(y2);
    [m2,n2]=min(a2);

    y3=x2-0.9*ones(x2);
    a3=abs(y3);
    [m3,n3]=min(a3);

    Tr=(n3-n2)/1000;
endfunction

function Tsu=Set_time(x3)
    i=25000
    while abs(1-(x3(i)))<0.02
        i=i-1;
    end
    Tsu=i/1000;

endfunction

function Tp=Peak_time(x4)
    [m5,n5]=max(x4);
    Tp=n5(1)/1000;
endfunction

function Tp_1=Peak_time_undamped(wn0)
    Tp_1=%pi*wn0;

endfunction

function OS=Percent_OS(x5, gainK)
    [m6,n6]=max(x5);
    OS=((m6-gainK)/gainK)*100;
endfunction

s=%s;
time=0:0.001:25;
Gs_undamped=syslin('c',1/(s^2+1));            //zeta=0    wn=1
```

```
Gs_underdamped=syslin('c',1/(s^2+s+1));        //zeta=0.5  wn=1
Gs_overdamped=syslin('c',1/(s^2+4*s+1));       //zeta=2    wn=1

Ps_undamped=csim('step',time,Gs_undamped);
Ps_underdamped=csim('step',time,Gs_underdamped);
Ps_overdamped=csim('step',time,Gs_overdamped);

plot2d(time,[Ps_undamped',Ps_underdamped',Ps_overdamped']);
leg1=legend('Undamped_1','Underdamped_2','Overdamped_3',4);
xtitle('Step Response of Second Order System','Time','Response');

//For Undamped System
tp1=Peak time undamped(1);

//For Underdamped System
tp2=Peak time(Ps_underdamped);
td2=Delay time(Ps_underdamped);
tr2=Rise time(Ps_underdamped);
ts2=Set time(Ps_underdamped);
OS2=Percent OS(Ps_underdamped,1);

//For Overdamped System
td3=Delay time(Ps_overdamped);
tr3=Rise time(Ps_overdamped);
ts3=Set time(Ps_overdamped);

printf('Result for Undamped System is:\n Peak Time=%f',tp1);
printf('\nResult for Underdamped System is:\n Peak Time=%f',tp2);
printf('\nDelay Time=%f',td2);
printf('\nRise Time=%f',tr2);
printf('\nSettling Time=%f',ts2);
printf('\nPercentage OS=%f',OS2);
printf('\nResult for Overdamped System is:\n Delay Time=%f',td3);
printf('\nRise Time=%f',tr3);
printf('\nSet Time=%f',ts3);
```