Lab 2

Control Systems

Name	ID	Section
Nada Tarek Mowafi	20012094	3
Yasmin Yasser Mohamed Abo El Fetouh	20012216	4

Design of Satellite-Tracking Antenna:

a. Evaluate the closed loop transfer function:

$$J\theta + B\theta = K(\theta r - \theta)$$

$$T(s) = \theta / \theta r = \frac{k}{Js^2 + Bs + k}$$

b. Use MATLAB to generate the state-space representation for the closed loop system for *K* = 1:

- Here is the code:

```
J = 600000 ;
B = 20000 ;
K = 1 ;
numerator = K ;
denominator = [J B K] ;
trans_func = tf(numerator , denominator) ;
state_space = ss(trans_func)
```

- The state space representation:

```
>> lab2

state_space =

  A =
                x1          x2
   x1   -0.03333   -0.001707
   x2  0.0009766           0

  B =
              u1
   x1    0.03125
   x2          0

  C =
              x1         x2
   y1          0    0.05461

  D =
        u1
   y1    0
```

c. What is the maximum value of K that can be used if you wish to have a stable closed loop system?

1. The **characteristic equation** for the system is:

$$Js^2 + Bs + K = 0$$

2. Stability conditions:
   - For a stable system, all coefficients of the characteristic equation must have the same sign.
   - Here, $J > 0$, $B > 0$ and $K > 0$. As a result, the roots of the characteristic equation will always lie in the left-half of the complex plane for any $K > 0$.
3. Behavior of Roots with Increasing K:
   - The roots are:

$$s = \frac{-B \pm \sqrt{B^2 - aJK}}{2J}$$

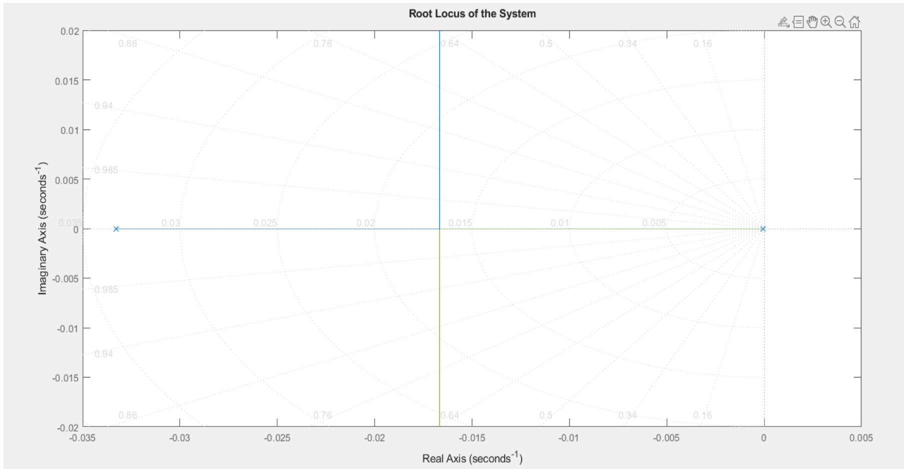
   - As $K \to \infty$, the dominant root moves further left in the complex plane ($s \to -\infty$), and the system remains stable.

So, No maximum value exists for **K** , as the system remains stable for all $K>0$.

- Using matlab:

```
%% c
J = 600000;
B = 20000;
numerator = 1;              % K = 1
denominator = [J B numerator];
trans_func = tf(numerator, denominator);

% Plot root locus
figure;
rlocus(trans_func);
title('Root Locus of the System');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;

% Display a message
disp('The root locus shows the stability of the system for varying K.');
disp('If all poles remain in the left-half plane for all K > 0, the system is stable.');
```

➢ The output:



d. What is the maximum value of K that can be used if you wish to have an overshoot Mp < 10%?

- Rearrange the overshoot formula to determine the required ζ:

$$Mp < 0.1 \implies \zeta > \mathbf{0.591}$$

- For a second-order system, the damping ratio ζ is calculated as:

$$\boldsymbol{\zeta = \frac{B}{2\sqrt{J \cdot k}}}$$

- Here, the 2k comes from the characteristic equation:

$$Js^2 + Bs + K = 0$$

- Solve for K such that ζ>0.591:

$$\boldsymbol{\zeta} = \frac{\boldsymbol{B}}{\boldsymbol{2\sqrt{J\,.2k}}} > \mathbf{0.591}$$

$$\boldsymbol{\sqrt{J\,.2k}} < \frac{\boldsymbol{B}}{\mathbf{2\ \cdot 0.591}}$$

$$\boldsymbol{K} < \frac{\boldsymbol{B^2}}{\boldsymbol{4J \cdot 0.591^2}}$$

$$\boldsymbol{k} = \frac{\mathbf{20000^2}}{\mathbf{4\ \cdot 600000 \cdot\ 0.591}} = \text{ 476.5.}$$

Thus, the **maximum value of K** for Mp<10% is approximately 476.5.

- Using matlab:

```
%% d
J = 600000;
B = 20000;

% Required damping ratio for M_p < 10%
zeta_min = 0.591;

% Maximum value of K for M_p < 10%
K_max = (B^2) / (4 * J * zeta_min^2);

% Display the result
disp(['Maximum value of K for M_p < 10%: ', num2str(K_max)]);

```

Command Window

```
Maximum value of K for M_p < 10%: 477.1707
```

e. What values of K will provide a rise time of less than 80 sec? (Ignore the Mp constraint.)

For a standard second-order system, the rise time tr is approximately given by:

$$tr \approx \frac{\pi - \theta}{wn\sqrt{1-\zeta^2}}$$

However, a more practical approximation for the rise time is:

$$tr \approx \frac{1.8}{wn}$$

where ωn is the natural frequency of the system. The natural frequency ωn is related to the system parameters J, B, and K by the following equation:

$$wn = \sqrt{\frac{K}{J}}$$

Substitute tr <80 seconds:

$$80 > \frac{1.8}{wn}$$

$$wn > 0.0225 \text{ rad/s.}$$

Then ,

$$k > 303.75$$

For the system to have a rise time tr <80 seconds, K must be greater than **303.75**.

- Using matlab:

```
%% e
j = 600000;
b = 20000;
RiseTime_max = 80;
for k = 1:0.1:400
trans=tf(k,[j b k]);
state_space=ss(trans);
info=stepinfo(state_space);
% Check if rise time is less than the desired value
if info.RiseTime < RiseTime_max
break;
end
end
% Display the max of K
disp(['Maximum value of K for Rise Time < 80 sec:',num2str(k)]);
```

Command Window

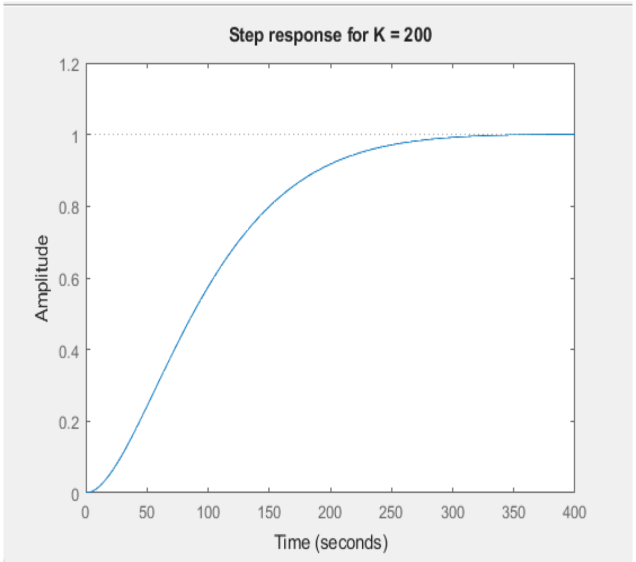
```
Maximum value of K for Rise Time < 80 sec:380.7
```

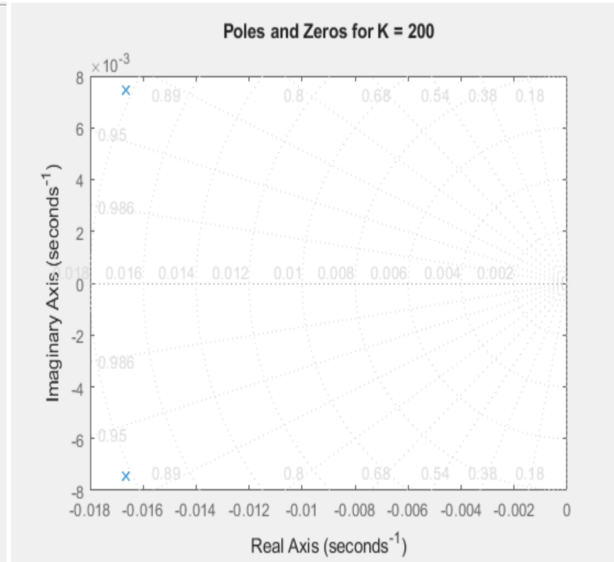
f. **Use MATLAB to plot the step response of the antenna system for K = 200, 400, 1000, and 2000. Find the overshoot and rise time of the four step responses by examining your plots. Do the plots to confirm your calculations in previous parts?**
g. **Use MATLAB to plot the zeros and poles locations for each value of K in part (e). comment on the effect of K on the closed loop zeros and poles.**
h. **for each value of K in part (e). find the steady state error.**

- **The code:**

```
%% f , g , h
K = [200, 400, 1000, 2000];
J = 600000;
B = 20000;
for i = 1:4
    sys_tf = tf(K(i), [J, B, K(i)]);
    figure;
    step(sys_tf);
    title(sprintf('Step response for K = %i', K(i)));
    [stepresponse, time] = step(sys_tf);
    stat = ss(sys_tf);

    % (Rise time and Overshoot)
    info = stepinfo(stat, 'RiseTimeThreshold', [0 1]);
    Overshoot = info.Overshoot;
    Risetime = info.RiseTime;
    fprintf('Overshoot for K = %i is: %f%%\n', K(i), Overshoot);
    fprintf('Risetime for K = %i is: %f seconds\n', K(i), Risetime);

    % steady-state error
    sterror = abs(1 - stepresponse(end));
    fprintf('Steady state error for K = %i is: %f\n', K(i), sterror);

    % Pole-Zero plot
    figure;
    h = pzplot(sys_tf);
    title(sprintf('Poles and Zeros for K = %i', K(i)));
    grid on;
end
```

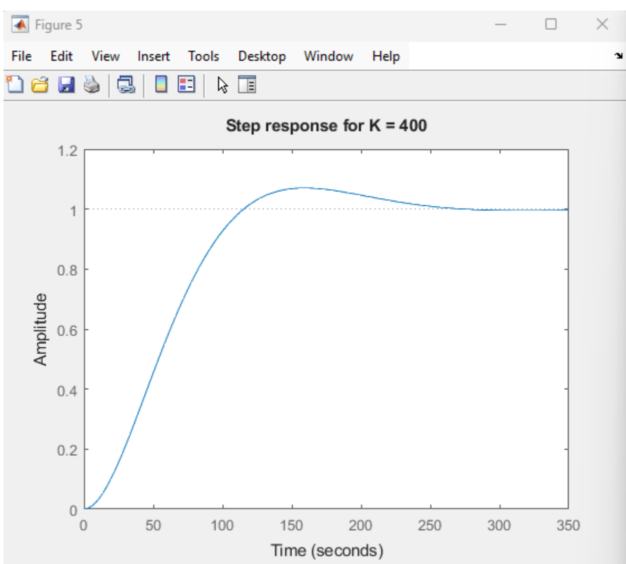
- **For k = 200:**

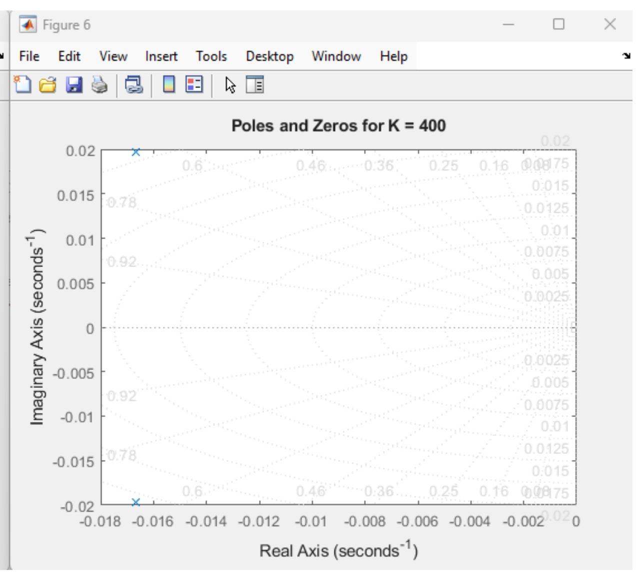





```
Overshoot for K = 200 is: 0.088930%
Risetime for K = 200 is: 365.082228 seconds
Steady state error for K = 200 is: 0.000514
```

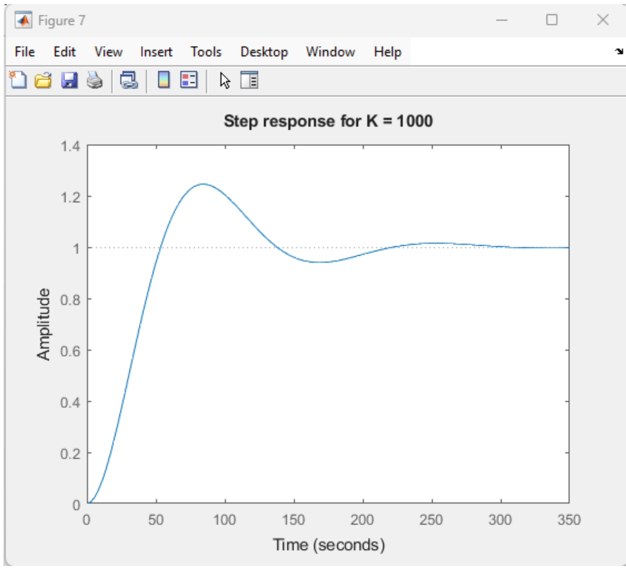
- For k = 400:

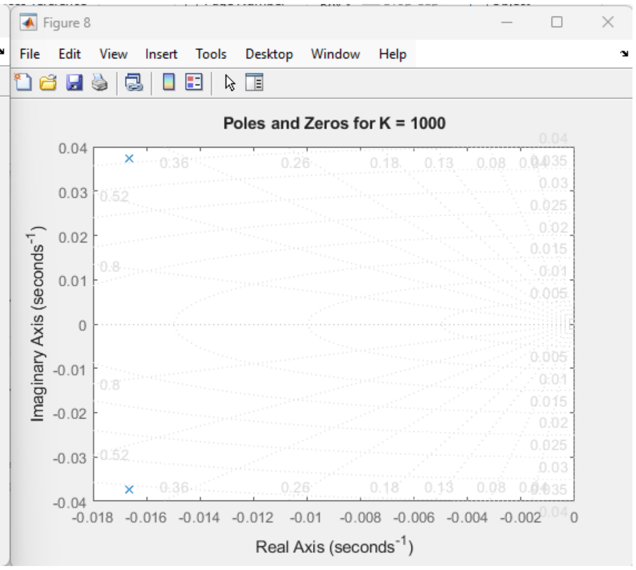





```
Overshoot for K = 400 is: 7.026866%
Risetime for K = 400 is: 115.261380 seconds
Steady state error for K = 400 is: 0.004918
```

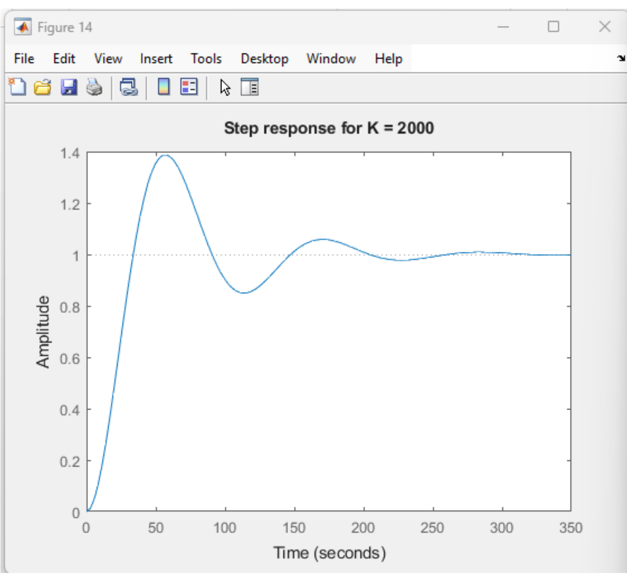
- **For k = 1000:**

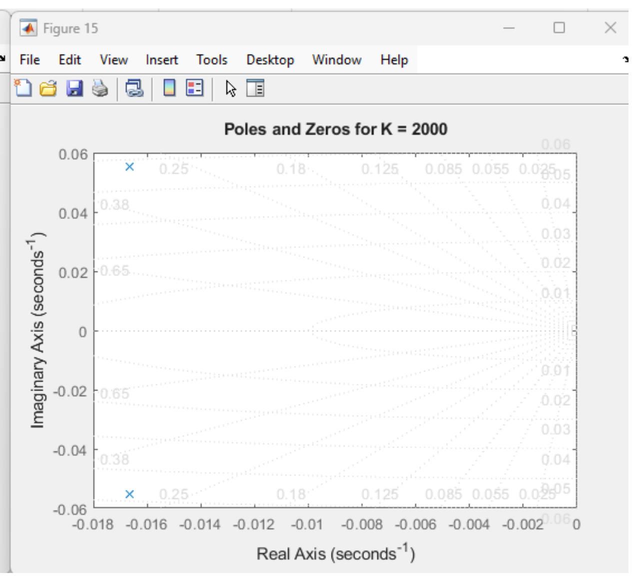





```
Overshoot for K = 1000 is: 24.500456%
Risetime for K = 1000 is: 53.462051 seconds
Steady state error for K = 1000 is: 0.003308
```

- For k = 2000:






```
Overshoot for K = 2000 is: 38.691002%
Risetime for K = 2000 is: 33.736331 seconds
Steady state error for K = 2000 is: 0.000872
```

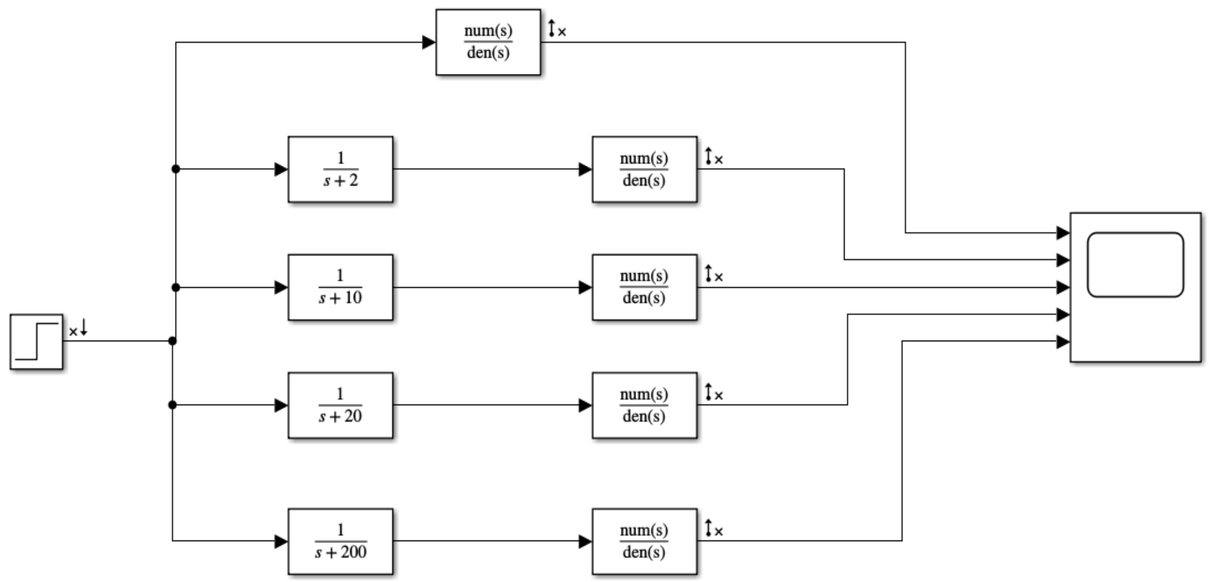
➢ The comment on closed loop zeros and poles:

- Increasing the value of K increases the poles value and they tend to move further left in the complex plane. This means the system becomes more stable. With the poles moving left, the system responds more quickly, reducing oscillations and settling faster. Essentially, increasing k makes the system less likely to "overshoot" and more likely to reach a steady state quickly.

- The zeros, which represent points where the system’s response is zero, are also influenced by K. When K increases, the zeros can move closer to the imaginary axis or, in some cases, shift toward infinity. The exact movement depends on the details of the system. In general, higher K values tend to make the system more responsive but can also impact how the system behaves in terms of stability and overshoot.

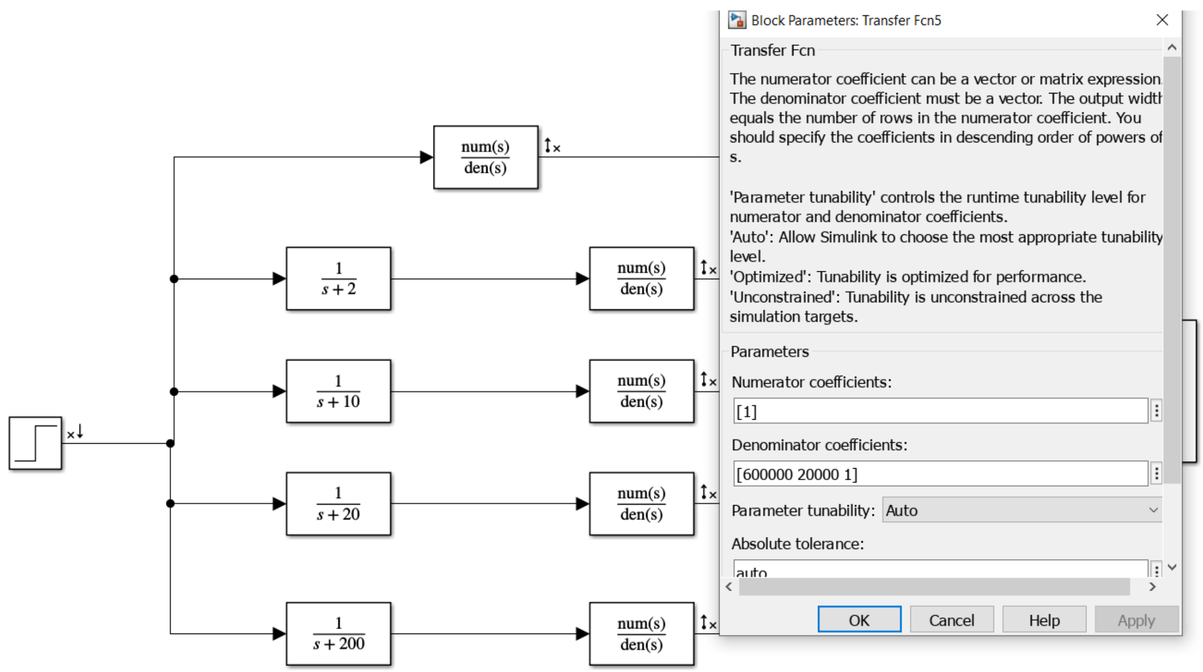
i. **Using Simulink, add a pole to the second-order system at -2, -10, -20, -200.**

Block diagram:

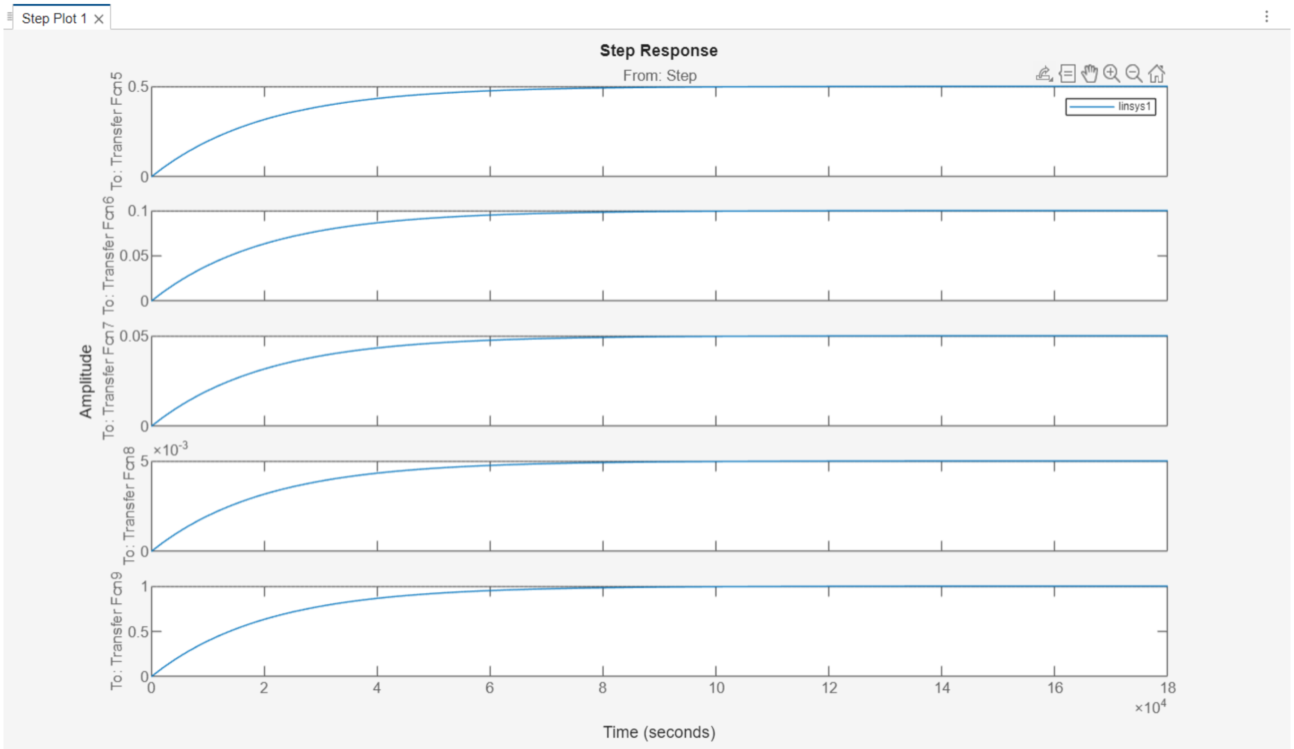


Coefficients of the numerator and denominator of the transfer functions [num(s)/den(s)]:

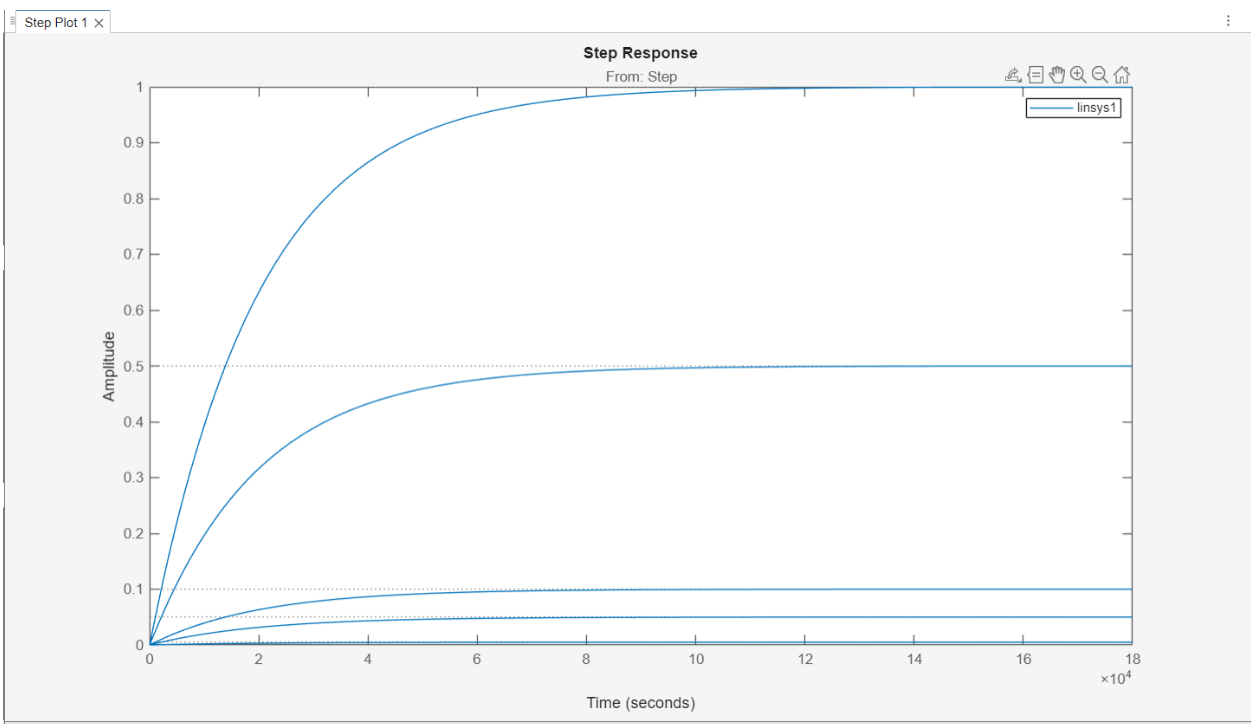
- k=1, j=600,000 and B=20,000

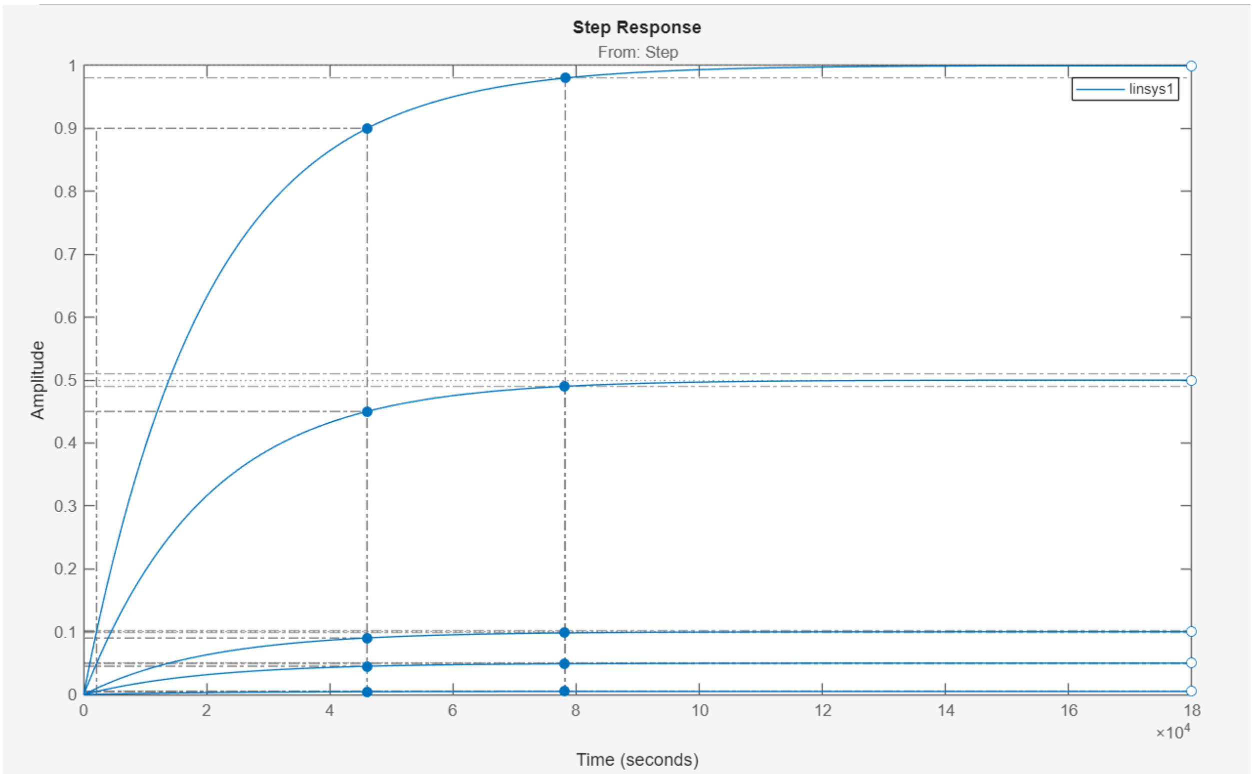


Plot the step responses of the system:

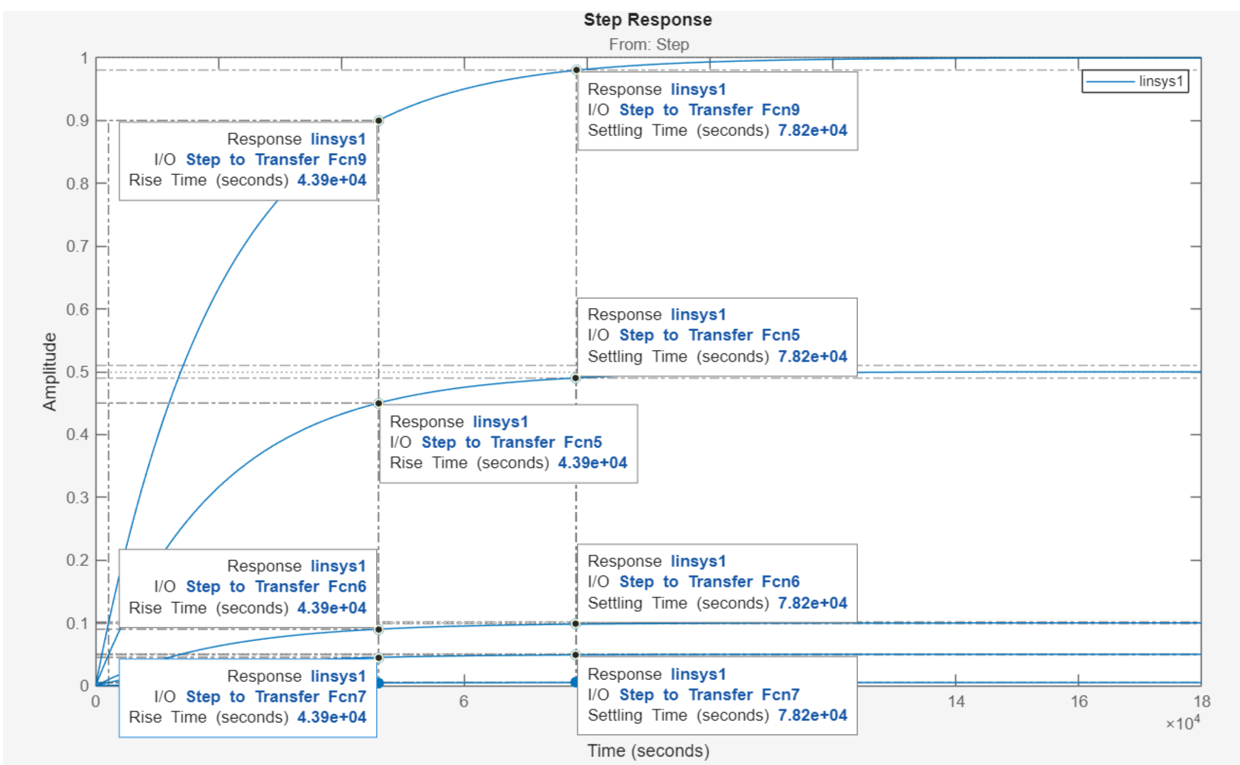


Make the plots on a single graph:



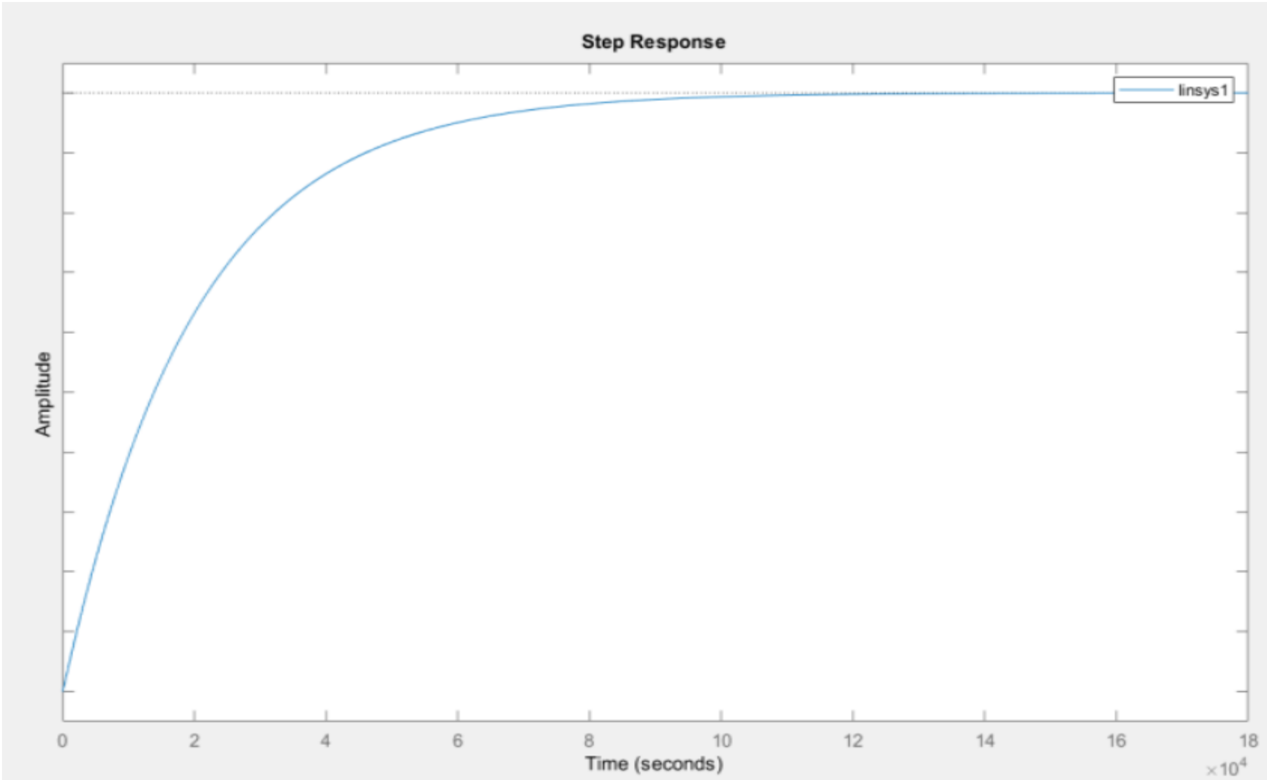


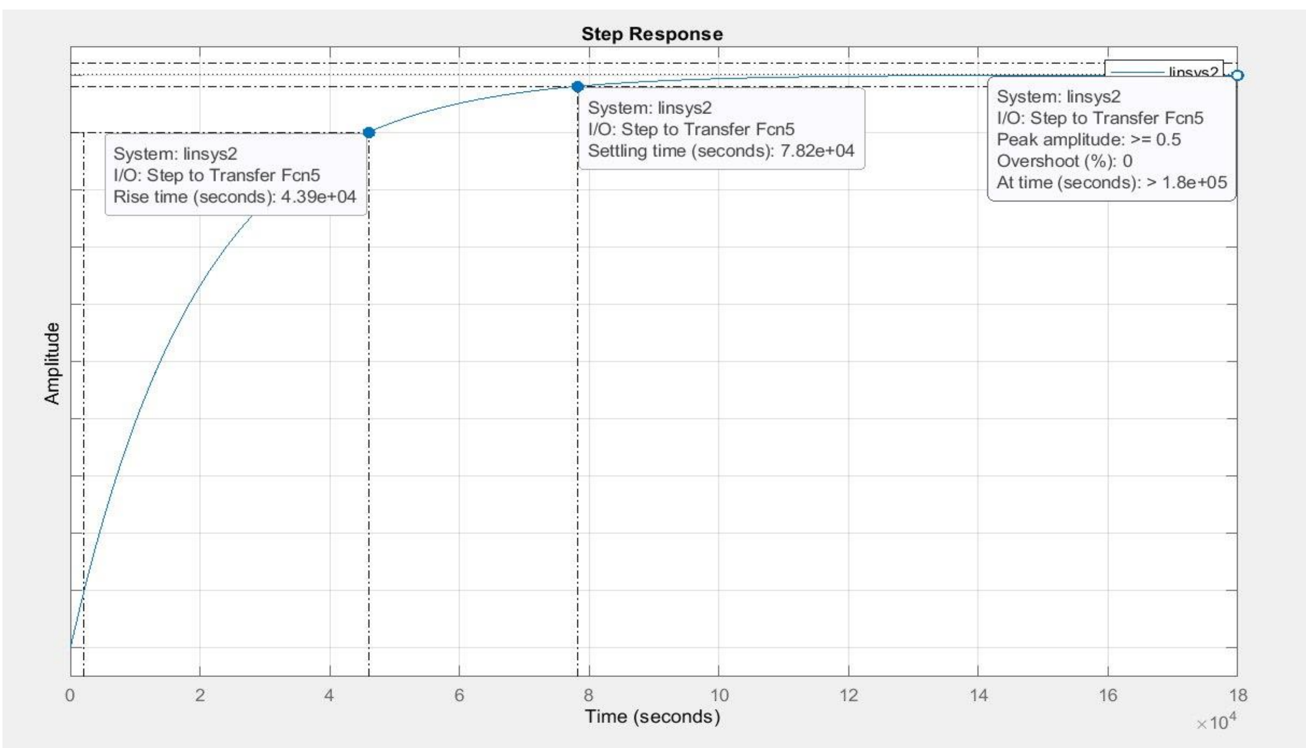
Step Response
From: Step
linsys1
Amplitude
Time (seconds)
×10^4



Step Response
From: Step
linsys1
Response linsys1
I/O Step to Transfer Fcn9
Rise Time (seconds) 4.39e+04
Response linsys1
I/O Step to Transfer Fcn9
Settling Time (seconds) 7.82e+04
Response linsys1
I/O Step to Transfer Fcn5
Settling Time (seconds) 7.82e+04
Response linsys1
I/O Step to Transfer Fcn5
Rise Time (seconds) 4.39e+04
Response linsys1
I/O Step to Transfer Fcn6
Rise Time (seconds) 4.39e+04
Response linsys1
I/O Step to Transfer Fcn6
Settling Time (seconds) 7.82e+04
Response linsys1
I/O Step to Transfer Fcn7
Rise Time (seconds) 4.39e+04
Response linsys1
I/O Step to Transfer Fcn7
Settling Time (seconds) 7.82e+04
Amplitude
Time (seconds)
×10^4

The step response after normalization:





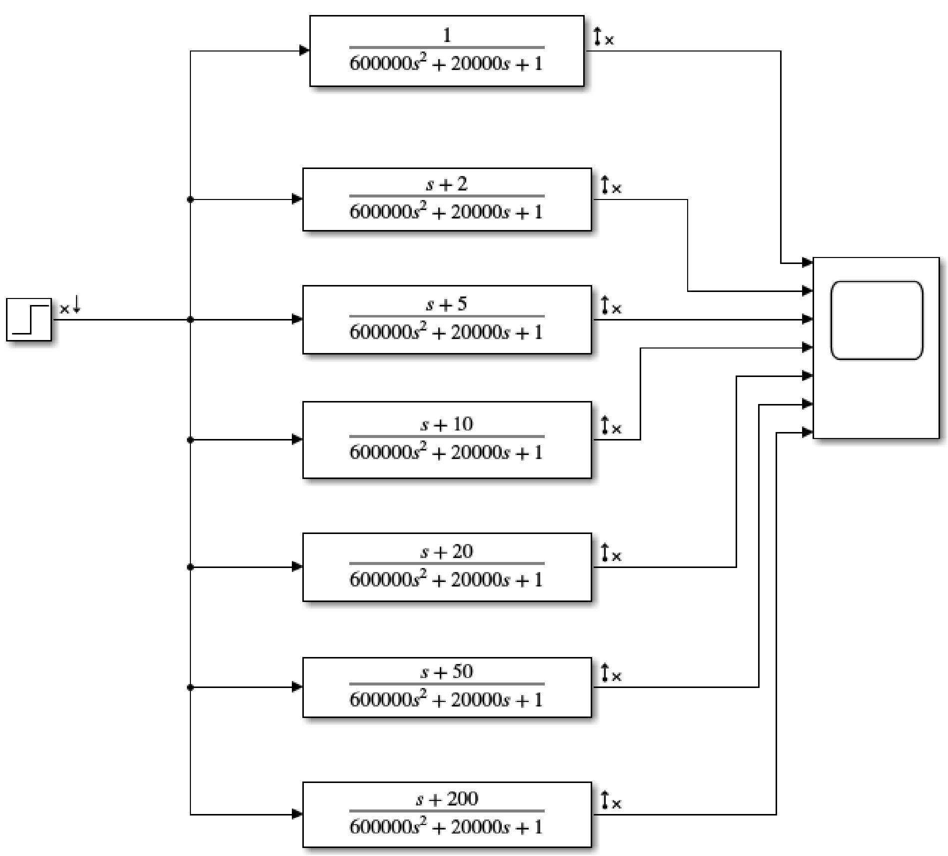
j. Discuss the effect upon the transient response of the proximity of a higher-order pole to the second-order system

When a higher-order pole is added to a second-order system, its proximity to the dominant poles significantly affects the transient response.
Some of these effects:

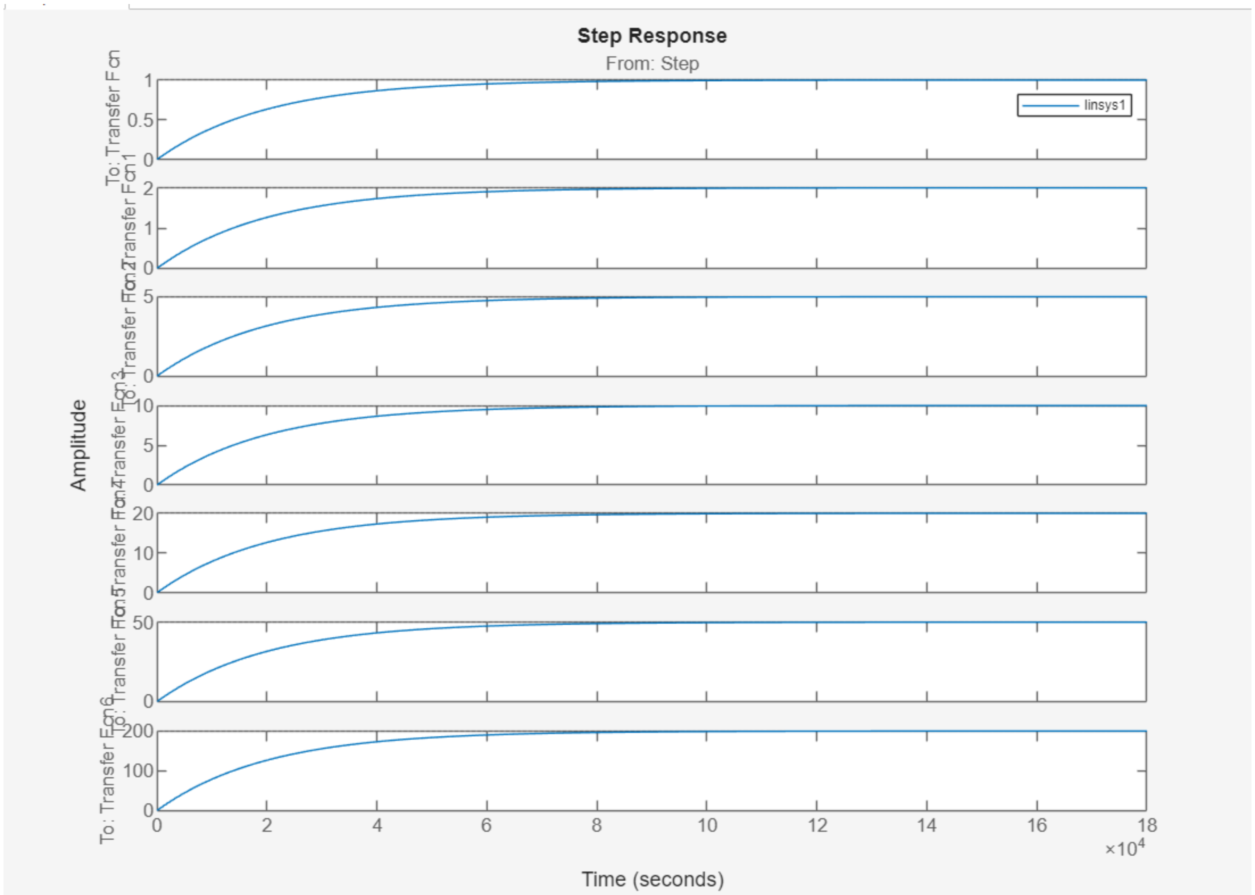
- **Dominant Poles:** If the higher-order pole is far from the dominant poles (much further left in the s-plane), its effect on the transient response is minimal because its effects decay much faster due to its larger real part (more negative). However, if the higher-order pole is close to the dominant poles, its influence becomes significant.
- **Rise Time and Settling Time:** The proximity of a higher-order pole to the second-order system may slow down the system's response, increasing rise and settling times, especially if the higher-order pole introduces a slower mode.
- **Overshoot and Damping:** If the higher-order pole is complex and close to the second-order poles, it may contribute additional oscillatory behavior, increasing overshoot and reducing overall damping.

k. Using Simulink, add a zero to the second-order system at -2, -5, -10, -20, -50, -200.

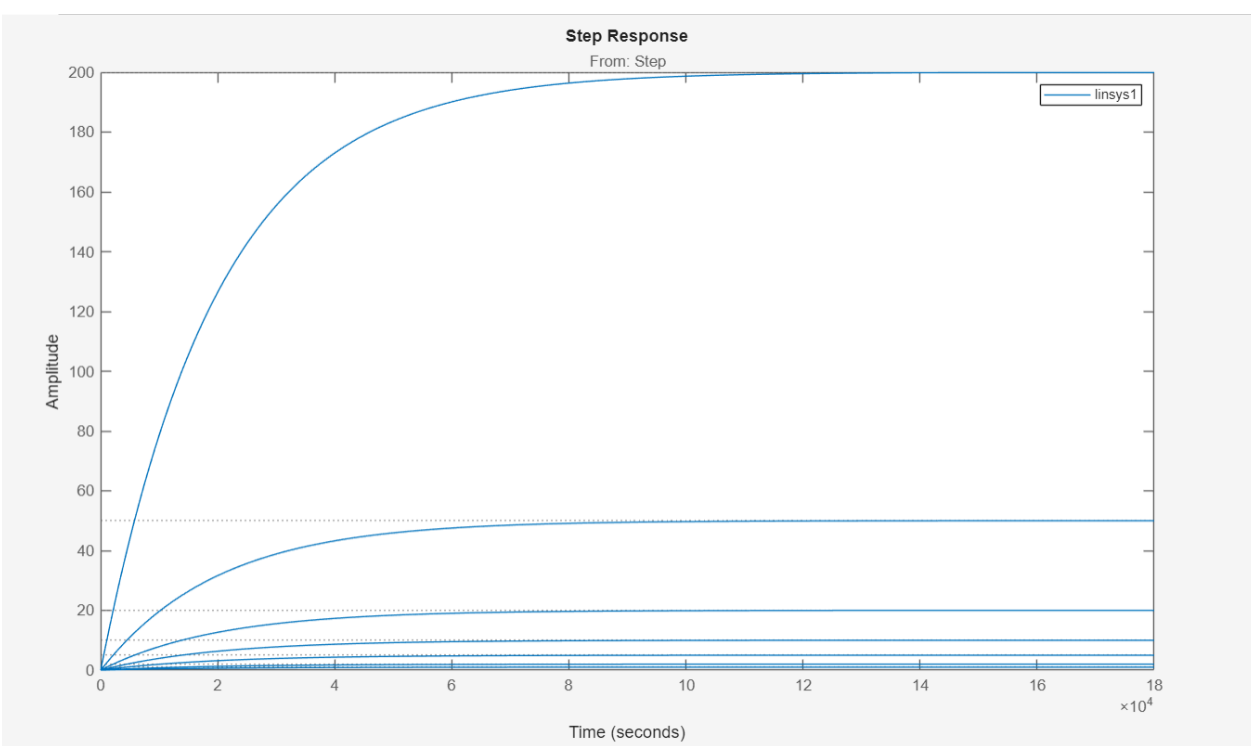
Block diagram:

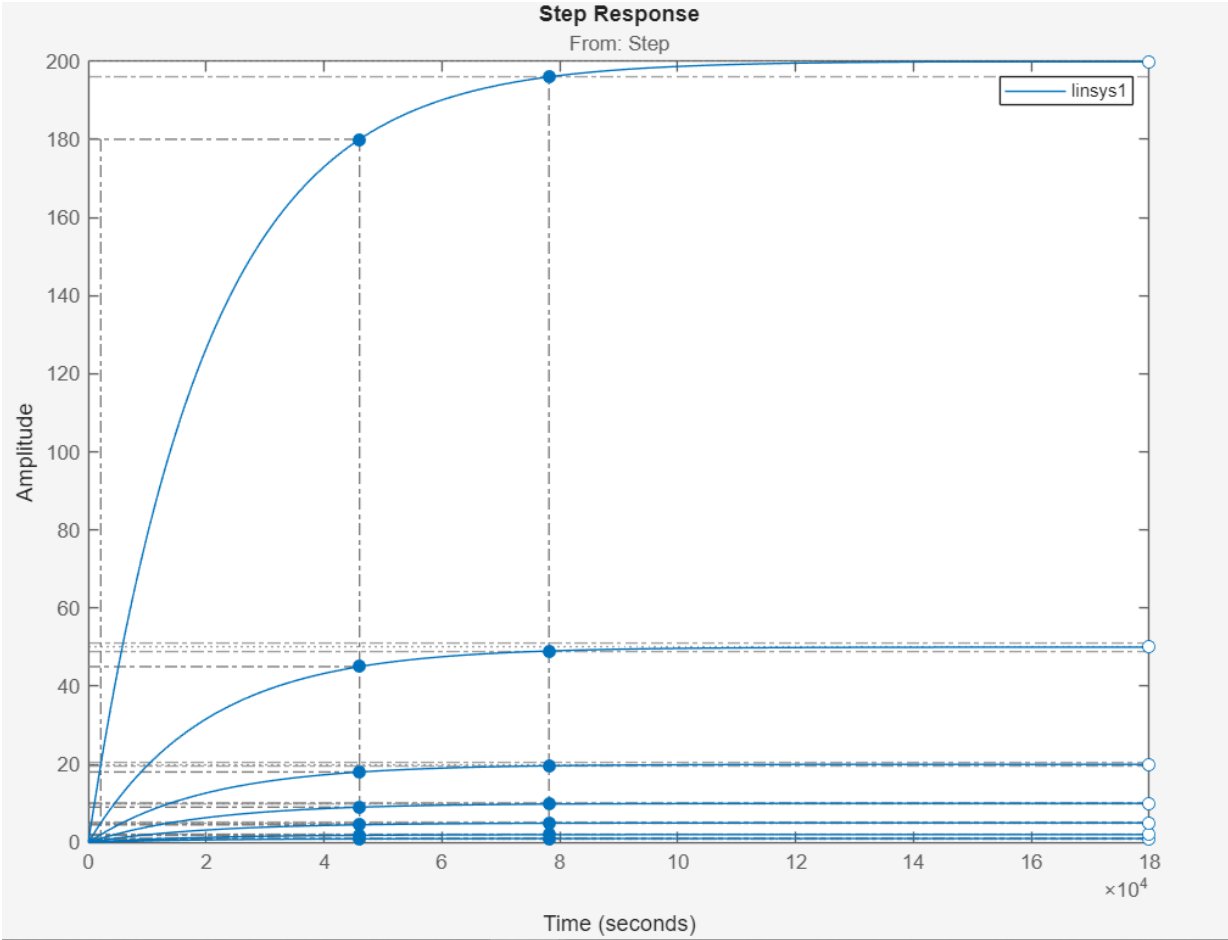


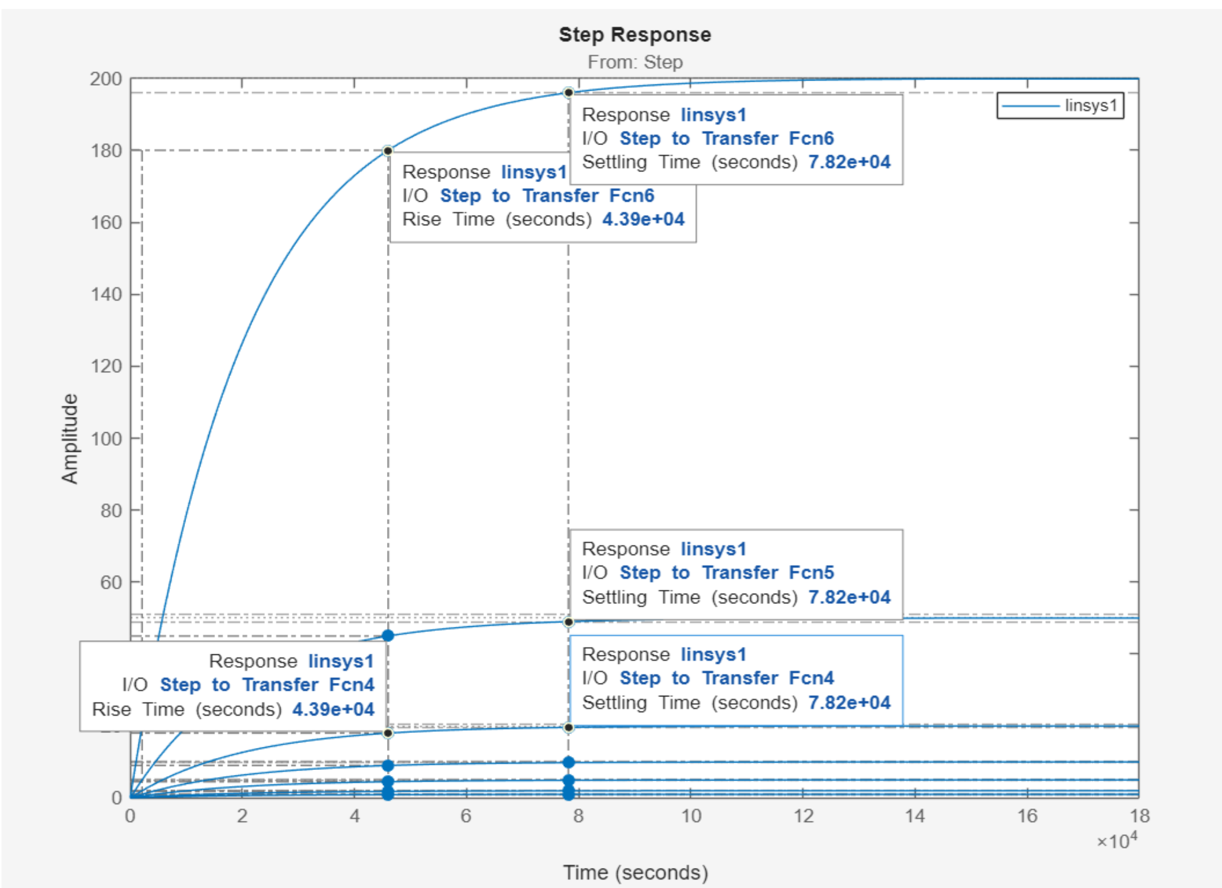
Plot the step responses of the system:



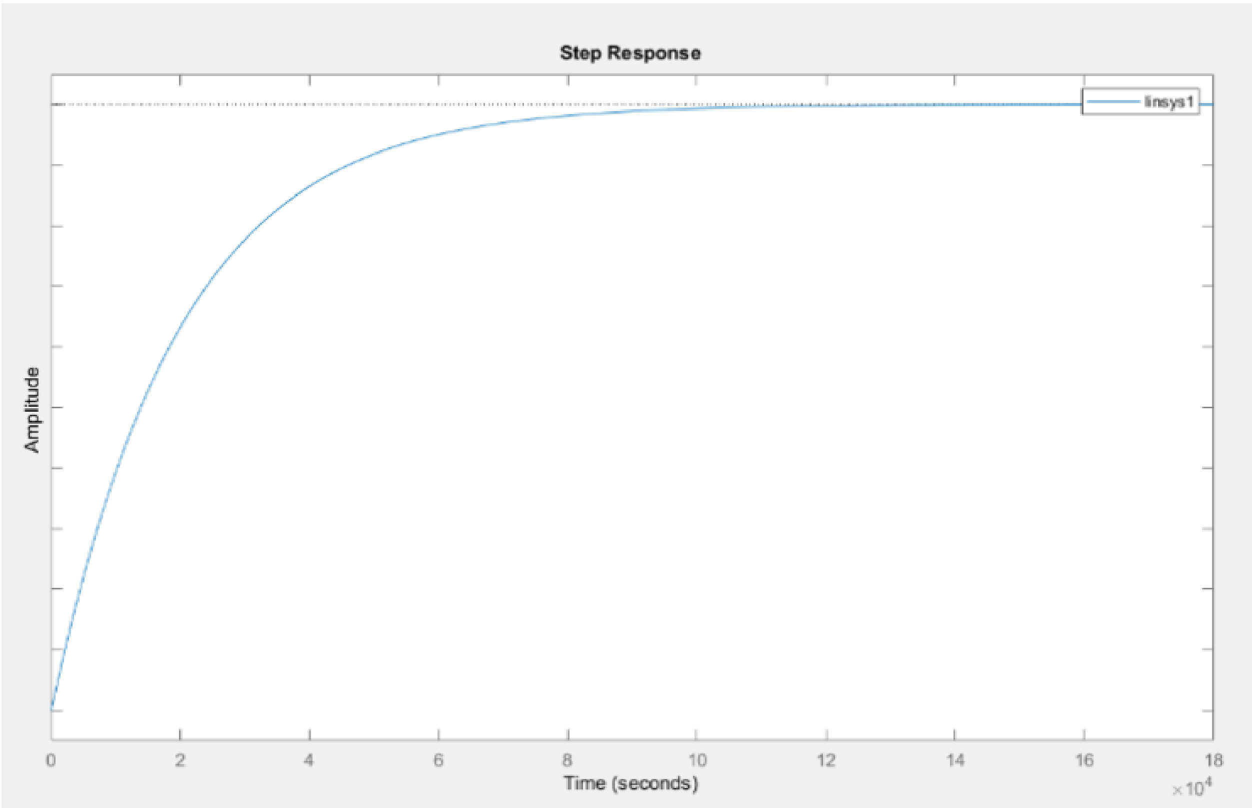
Make the plots on a single graph:

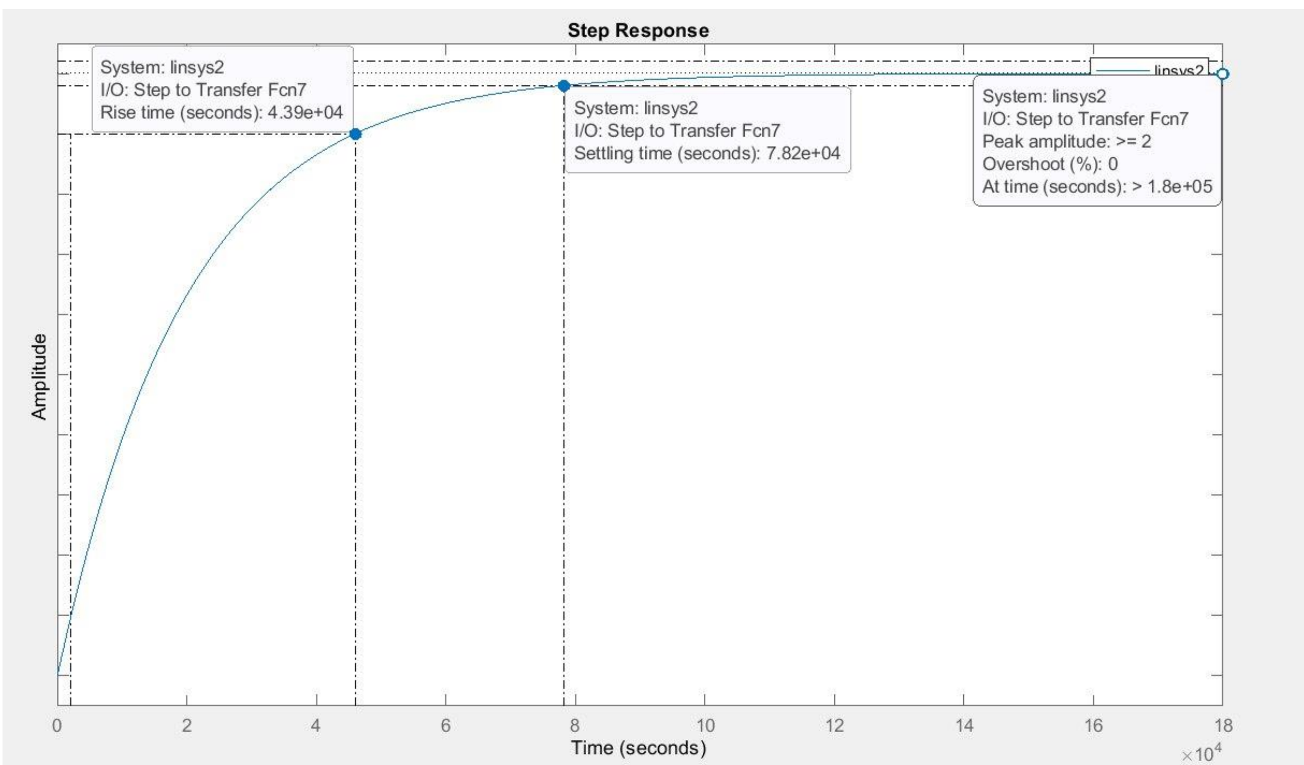



Step Response
From: Step
linsys1
Amplitude
Time (seconds)
×10^4


Step Response
From: Step
linsys1
Response linsys1
I/O Step to Transfer Fcn6
Settling Time (seconds) 7.82e+04
Response linsys1
I/O Step to Transfer Fcn6
Rise Time (seconds) 4.39e+04
Response linsys1
I/O Step to Transfer Fcn5
Settling Time (seconds) 7.82e+04
Response linsys1
I/O Step to Transfer Fcn4
Rise Time (seconds) 4.39e+04
Response linsys1
I/O Step to Transfer Fcn4
Settling Time (seconds) 7.82e+04
Amplitude
Time (seconds)
×10^4

The step response after normalization:





l. Discuss the effect upon the transient response of the proximity of a zero to the dominant second-order pole pair

When a zero is added to a second-order system, its proximity to the dominant poles affects the transient response in several ways:

- **Faster Response**: If the zero is close to the dominant poles, the system responds faster. This means the rise time and the peak time are reduced
- **Overshoot and Damping**: A properly placed zero (far from the imaginary axis) can effectively increase the damping of the system, reducing overshoot and making the response more controlled. However, if it is improperly positioned (close to the dominant poles), it can reduce the damping of the system, increasing the overshoot and makes the system more oscillatory.
- **Settling Time:** The zero's proximity can increase settling time if it destabilizes the dominant poles by reducing their damping ratio, the system can become more oscillatory. This means it takes longer time for the oscillations to die out. However, if the zero is well-placed, it can decrease settling time by pulling the response closer to critical damping. This reduces oscillations and helps the system settle faster.

So, the specific effects on the transient response depend on the zero's type, location, and proximity to the dominant second-order poles.

- If the zero is far from the dominant poles, its impact on the transient response is minimal. The system behaves almost like it did before the zero was added.