ECCE 323 Nov. 5, 2024

Final Assignment © R.K.

System Identification and Controller Design

Group 8

Shayma Mohammed Alteneiji | 100063072

Engy Emad Farouq | 100061659

Hind Ibrahim Kazim | 100060327

Due: 1/12/2024

Introduction

In this assignment we aim to identify the transfer function of a linear time-invariant system utilizing the frequency response, this is a key skill that we as engineers need in our system modelling and design journey, where we need to be able to Identify an unknow system using the input and the output behaviour of the system. We then design multiple types of controllers, in which the given specifications were satisfied using the different methods that were taught during this course, while constantly simulating our results to ensure the right track into modelling is taken. Where, by the end we would be completing a cycle of system identification, system modelling and system design.

Problem Statement

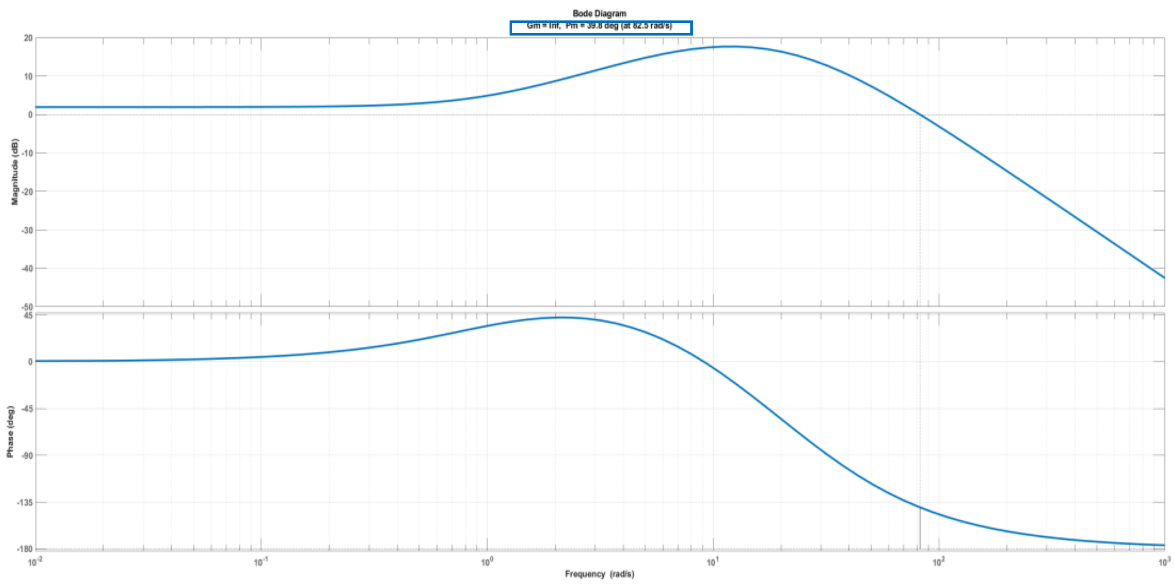
System Identification:

Consider the bode diagram of a linear time-invariant system shown in Figure 1. It is required to:

Identify the transfer function of the open-loop system.

Sketch the Bode diagram of the identified system and compare it with the one shown in Figure 1. You may need to tune the system parameters until you exactly meet the given information of the phase margin and the gain margin for the given range of frequencies.

Figure 1. Bode diagram of a transfer function of an open-loop system, $G_p(s)$.



Solution:

From the given Bode diagram (Figure 1), we infer the following about the system:

1) It's a type zero system, as the phase response is zero for lower frequencies.
2) The system has all it's zeros on the left half plane (i.e. minimum phase system), since the phase ranges only from $45°$ to $-180°$.
3) The system has one zero and three poles (single and a double pole). The phase response increases initially and then decrease to $-180°$ as $\omega \rightarrow \infty$, this indicates one zero contributing +90° and three poles contributing -270°, resulting in a net phase of -180°.
4) The system gain K >1 .The gain of the system for lower frequencies is greater than 0 dB.

(Solution to Part A Continued)

To identify the system parameters, we superimpose an asymptotic bode plot onto the given plot (Figure 2). The values obtained from the approximate asymptotic bode plot served as our initial estimates for the system parameters:

$$z_1 = 0.8,\ p_1 = 9,\ 2 \times p_2 = 30, K = 10^{\frac{2}{20}} = 1.2589$$

By carefully tuning these parameters, we matched the Bode plot with the crossover frequency and phase margin specifications (Figure 3). The resulting transfer function for the system is:

$$G_p'(s) = \frac{1.5551 \times \left(\frac{s}{0.9} + 1\right)}{\left(\frac{s}{6} + 1\right) \times \left(\frac{s}{27} + 1\right)^2}$$

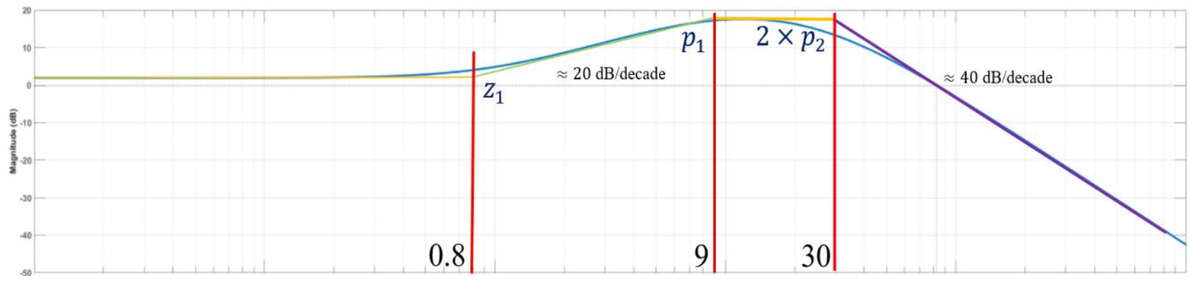


Figure 2. Superimposed approximate asymptotic bode plot on the bode diagram of Gp(s).

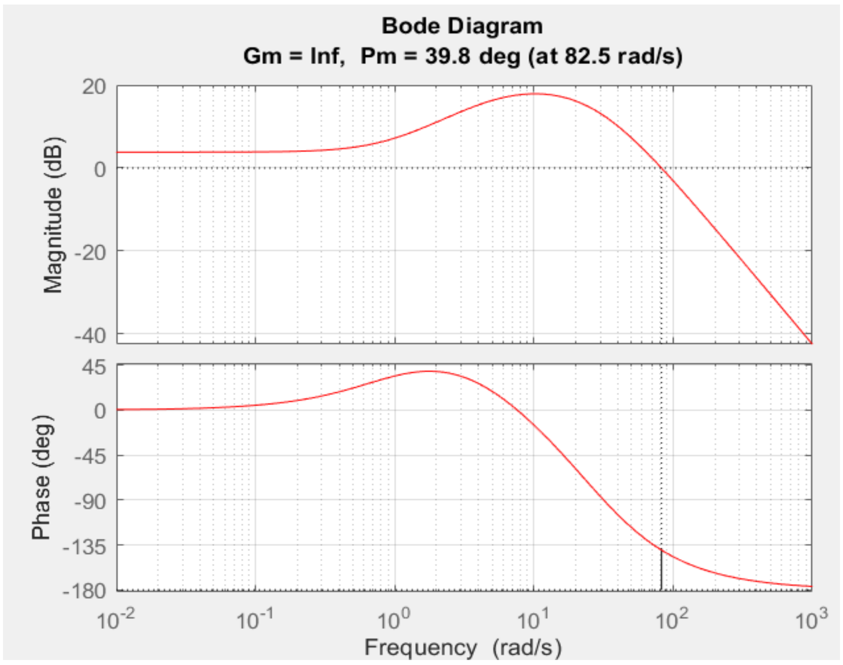


Figure 3. Bode diagram of the derived transfer function, $G_p'(s)$.

System Control Design:

The section consists of four questions on system control design. Modify the design (when necessary) to ensure zero tracking error for unit-ramp reference input signal. Give the transfer function of the controller.

Q1) Determine the order and the type of the system and find the gain K that could make the system critically stable.

B.1.1 Bode Plot- Based Solution

As explained above, the system is Type-0 and of order 3 (three poles). To determine the gain *K* required for critical stability, the following procedure is used:

1) Plot the Bode diagram of the open-loop transfer function *Gp*(*s*).
2) Identify the frequency ω_{GM} at which the phase response of the system is $-180°$. Record the corresponding magnitude response, $A_{dB}(\omega_{GM})$, at that frequency.
3) Calculate the adjusted gain *K'* that would shift the system's magnitude response to 0 dB at ω_{GM} :

$$K' = 10\text{^}(-A_{dB}(\omega_{GM})/20).$$

   - If $A_{dB}(\omega_{GM}) < 0$, the new plot shifts upward.
   - If $A_{dB}(\omega_{GM}) > 0$, the new plot shifts downward.

4) Multiply *K'* by the system's current gain *K* to calculate the critical gain *K*cri:

$$K_{cri} = K' \times K$$

(This assumes all terms of the T.F. are normalized to their corner frequencies).

In our case, as shown in Figure 3, the system's phase response never reaches -180° for any finite frequency. Instead, $\measuredangle G_p(j\omega) \to -180°$ as $\omega \to \infty$. The corresponding magnitude response is $A_{dB}(\infty) = -\infty$. Substituting this into the formula for K':

$$K' = 10\text{^}(-(-\infty)/20) = \infty$$

Thus, an infinite gain would be required to make the system critically stable.

B.1.2 Custom MATLAB Function- Based Solution

We developed a MATLAB function to determine the critical gain Kcri. The function takes the zeros, poles, and gain K of the open-loop system as inputs. The MATLAB code is provided in Appendix A.

The MATLAB function was tested on the open-loop system:

$$G(s) = \frac{40}{(s+2) \times (s+4) \times (s+5)}$$

(Solution to Part B.1.2 continued)

The MATLAB program returned the following results:

Kcri = 377.9439, and ω_{nc} *= 6.164 rad/s.*

This example corresponds to Lesson 10.7 (page 567) in *Control Systems Engineering* by Norman Nise. The textbook solution gives Kcri = 400 and ω_{nc} = 7 rad/s. The slight variation arises because the book's solution uses a graphical asymptotic Bode plot, whereas our MATLAB function provides a more precise numerical result.

Q2) (a) Design a proportional controller (if you can) to maintain a phase margin greater than 50° and a gain margin > 10 dB. (b) Simulate and plot the step response of the controlled system using MATLAB/SIMULINK.

(a)

B.2.1 Bode Plot- Based Solution

To design a proportional controller, the following steps were performed:

1) Determine Required Phase:

$$\phi_m = \phi_s + 180°$$

` For a phase margin of 50°:

$$\phi_s = \phi_m - 180° = 50° - 180° = -130°$$

2) Determine Frequency and Gain:

Using the Bode diagram (Figure 4), the cursor is placed at the frequency ω_s = 64.3 rad/s, where the phase is −130°. The magnitude response at this frequency is 3.78 dB.

3) Calculate Controller Gain:

The proportional controller gain Kp is calculated as:

$$K_p = 10^{-\frac{A_{dB}(\omega_s)}{20}} = 10^{-\frac{3.78}{20}} = 0.6471$$

As Kp < 1, the magnitude response is shifted downward. For a phase margin greater than 50°, the controller gain should be $K_p < 0.6471$.

The uncompensated and compensated systems for K_p = 0.6471 are plotted as shown in figure 5, with both phase and gain margins meeting the design specifications.

(Solution to Part (a) Q2 continued)

B.2.2 Custom MATLAB Function- Based Solution

A custom MATLAB function was developed to automate the gain controller design. The input to the function includes the zeros and poles of the open-loop transfer function, the gain, and the desired phase margin. The MATLAB program, provided in Appendix B, returns a controller gain Kp.

For the system tested, MATLAB results give:

Kp = 0.6483

The difference between manual and MATLAB results is minimal, as both gains satisfy the phase margin requirement and yield the same crossover frequency to one decimal point.

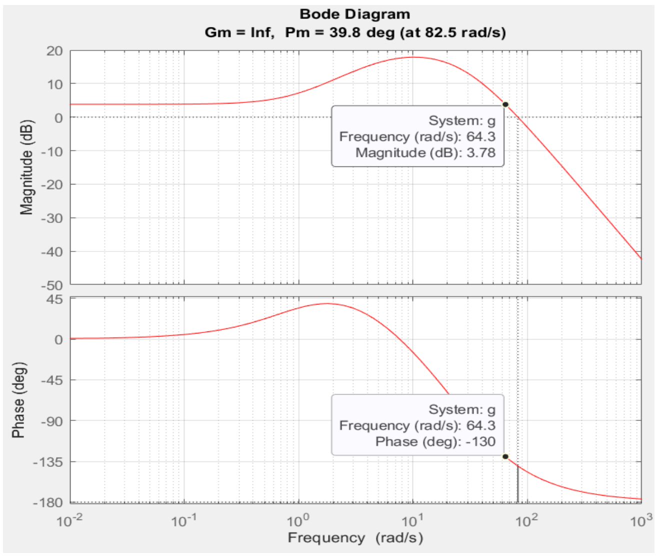


Figure 4. Bode diagram of $G(s)$ with the cursor placed at $\omega = 64.3$ rad/s.

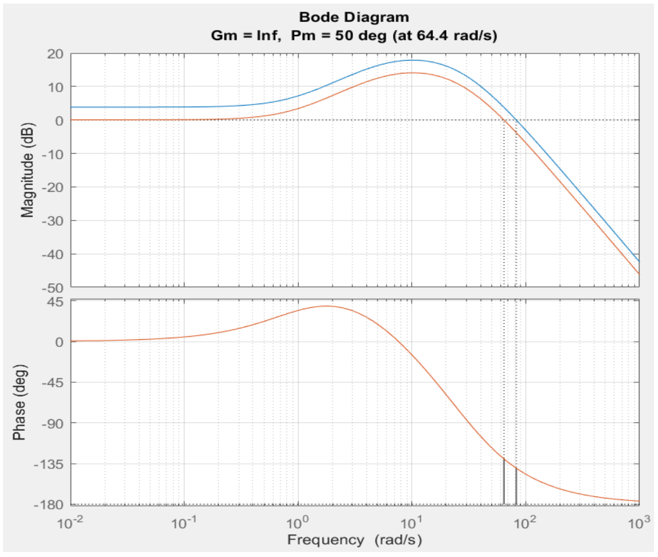


Figure 5. Bode diagrams of $G(s)$, the uncompensated and $K_pG(s)$,

the compensated system, the phase response is unchanged.

(b)

The SIMULINK block diagram of the controlled system is shown in Figure 6. The MATLAB code used to obtain the system's unit step response is provided below. The resulting step response is illustrated in Figure 7. It can be observed that step response dose not settle to unity. For a type 0 system, the steady-state error (SSE) due to a step input is given by:

$$SSE = \frac{1}{1 + K_p'}$$

Where K'$_p$ here is the static position error constant (denoted with a prime to differentiate it from the proportional controller gain). The static error constant is calculated as follows:

$$K_p' = \lim_{s\to 0} K_p G_p(s) = \lim_{s\to 0} K_p \times K \times \frac{\left(\frac{s}{0.9}+1\right)}{\left(\frac{s}{6}+1\right)\times\left(\frac{s}{27}+1\right)^2}$$

Simplifying:

$$= K_p \times K = 1.0063$$

Substituting into the SSE formula:

$$S.S.E = \frac{1}{1 + 1.0063} = 0.4984$$

From the system's unit step response (Figure 7), this result is verified as the system's output eventually settles at: 1 – 0.4984 = 0.5016.

```
clc; clear all
K = 1.5551; Kp = 0.6471;
Stop_time = 15;
out = sim("Proportional_Controller.slx",Stop_time )

plot(out.tout, out.System_Responce ,'r')
grid on
title("The Step Responce of the Controlled System Kp*Gp(s)")
```

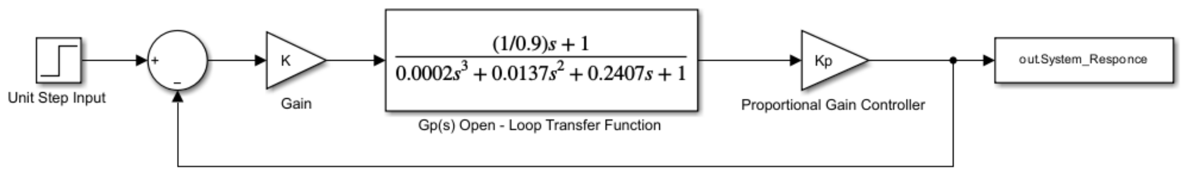

Unit Step Input
+
-
K
Gain
$\frac{(1/0.9)s+1}{0.0002s^3+0.0137s^2+0.2407s+1}$
Gp(s) Open - Loop Transfer Function
Kp
Proportional Gain Controller
out.System_Responce

Figure 6. Block diagram of the controlled closed-loop system using a P controller.

(Solution to Part (b) Q2 continued)

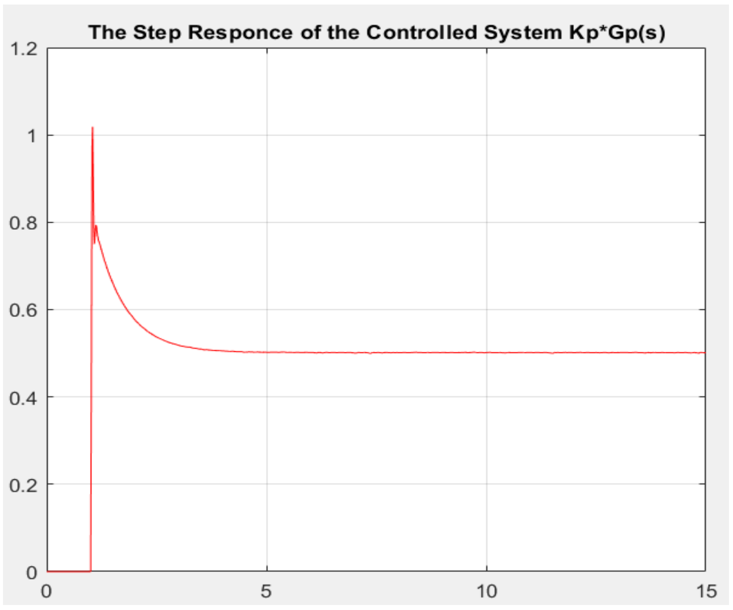


Figure 7. Step response of the controlled closed-loop system

using a P controller.

To ensure zero steady-state error for a ramp input, the block diagram was modified (Figure 8). This adjustment introduces integral action, enabling the system to track the ramp input without error. However, this modification affects the Bode plot of the system, as shown in Figure 9. While the gain margin requirement is satisfied, the phase margin decreases to 37.7°, and the new crossover frequency is 1.32 rad/s. The most effective approach to enhance the phase margin is to cascade the system with a phase-lead compensator. Using the El-Khazali method of design, we solve for the lead compensator parameters:

$$\alpha = \frac{50 - 37.7}{90} = 0.1367$$

$$\tau = \; tan\left(\frac{\pi\alpha}{2}\right) + sec\left(\frac{\pi\alpha}{2}\right) = 1.2416$$

The resulting compensator transfer function is given below, Kc = 1, as there is no specification on the crossover frequency:

$$G_c(s) = \frac{\left(\frac{1.2416s}{1.32}\right) + 1}{\left(\frac{s}{1.32}\right) + 1.2416}$$

With this compensator cascaded with the system, the new Bode plot is shown in figure 10, and the system response to a unit step and ramp input is shown in figures 11 and 12, where SSE=0.

(Solution to Part (b) Q2 continued)

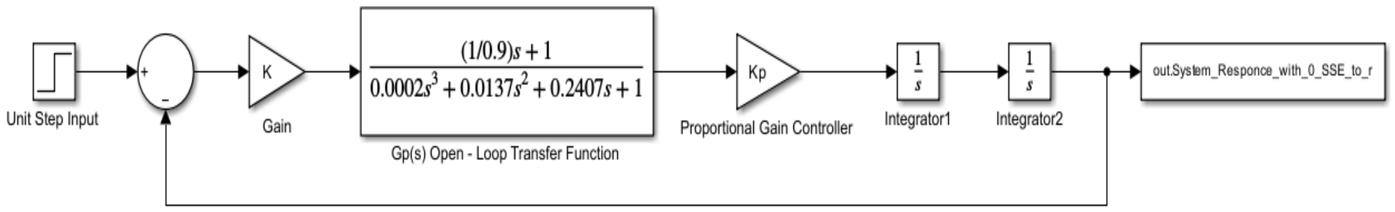

Unit Step Input
+
-
K
Gain
$\frac{(1/0.9)s+1}{0.0002s^3+0.0137s^2+0.2407s+1}$
Gp(s) Open - Loop Transfer Function
Kp
Proportional Gain Controller
$\frac{1}{s}$
Integrator1
$\frac{1}{s}$
Integrator2
out.System_Responce_with_0_SSE_to_r

Figure 8. Modified block diagram of the controlled system using a P controller. S.S.E for a ramp input is 0.

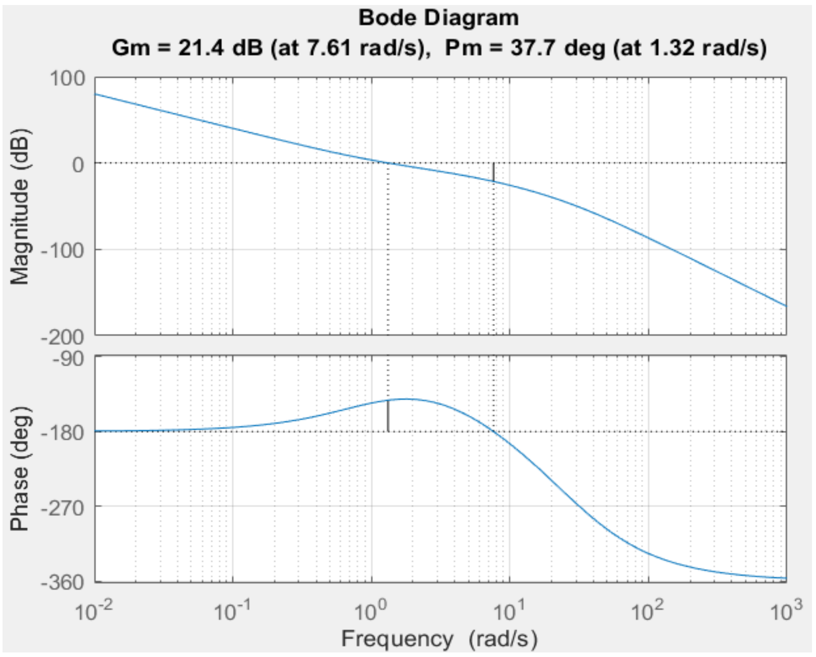


Figure 9. The Bode diagram of the controlled system with two integrators in the forward path.

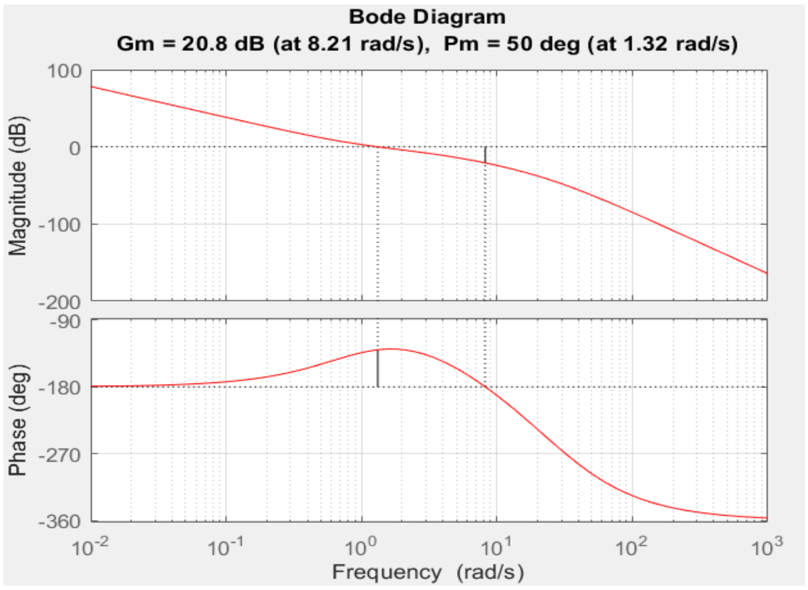


Figure 10. The Bode diagram of the controlled system with two integrators in the forward path cascaded with a phase- lead compensator.

(Solution to Part (b) Q2 continued)

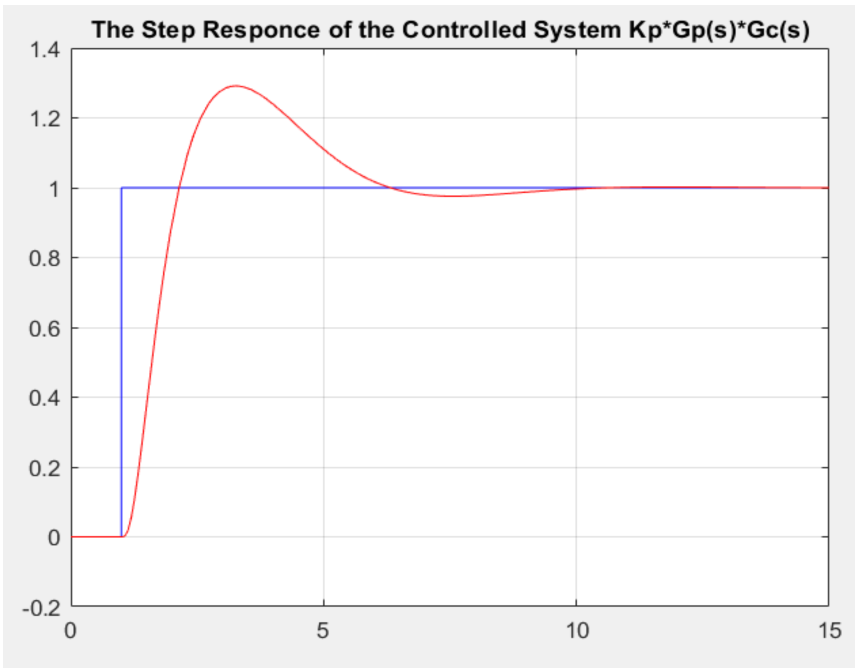


Figure 11. Unit step input response of the modified controlled system using

a P controller. S.S.E is 0.

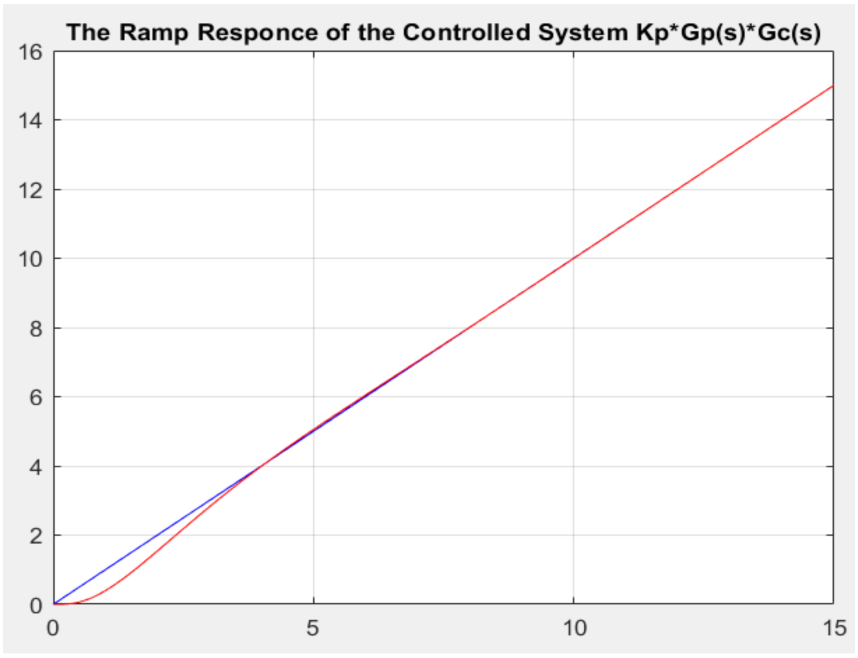


Figure 12. Unit ramp input response of the modified controlled system using

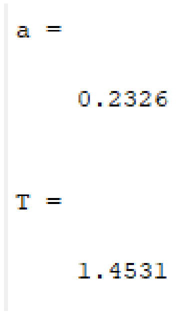
a P controller. S.S.E is 0.

Q2) (a) Design a lead controller to achieve an exact phase margin of 60.7° at the same crossover frequency of the uncontrolled system and a gain margin > 10 dB. (b) Simulate and plot the step response of the controlled closed-loop system using MATLAB/SIMULINK.

(a)

B.3.1 Manual Calculation Based - Design Solution

Since the required crossover frequency doesn't change, the required gain "Kc" will be equal to one. Moreover, we used El-Khazali method to design the controller to achieve the phase margin of 60.7° and gain margin>10dB. Using matlab, we calculated the variables α and T and got the variables seen in figure 13.



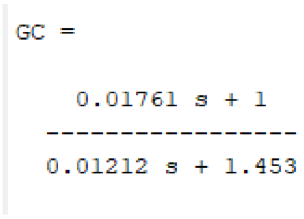
```
a =

    0.2326


T =

    1.4531
```

Figure 13. Calculated values of controller variables.

B.3.2 Custom MATLAB Function- Based Solution

A custom MATLAB function was developed to automate the lead controller design. The input to the function includes the open-loop transfer function, the crossover frequency, and the desired phase margin. The MATLAB program, provided in Appendix C, returns a controller transfer function Gc (Figure 14).

For the system tested, MATLAB results give:



```
GC =

    0.01761 s + 1
  -----------------
  0.01212 s + 1.453
```

Figure 14. Controller transfer function.

In order to test the functionality of our system we generated the bode plot (Figure 15) of the compensated open loop system and the results are like we desire.

(Solution to Part (a) B.3.2 continued)

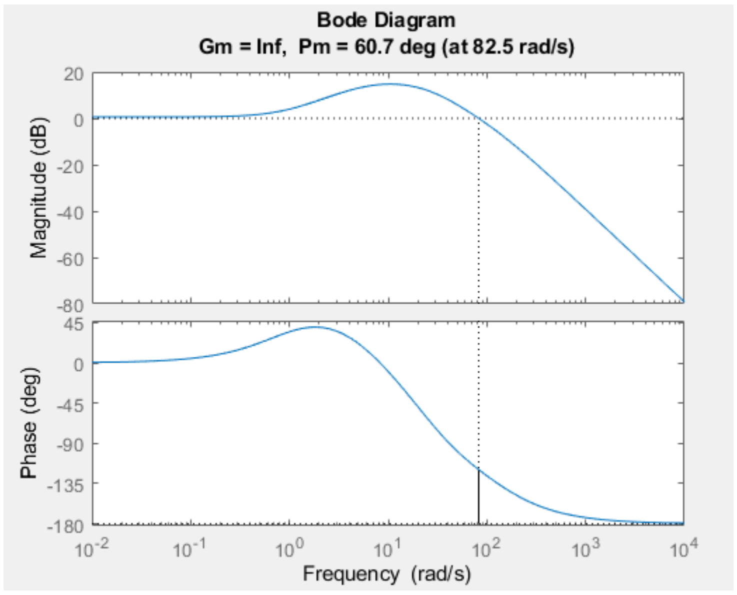


Figure 15. Bode plot of the controlled open-loop system using a lead controller.

(b)

From generating the step response using the function:

step(feedback(G*GC,1),3)

We are able to observe from the graph in figure 16 that for the compensated system, the steady state error is:

1-0.517=0.483

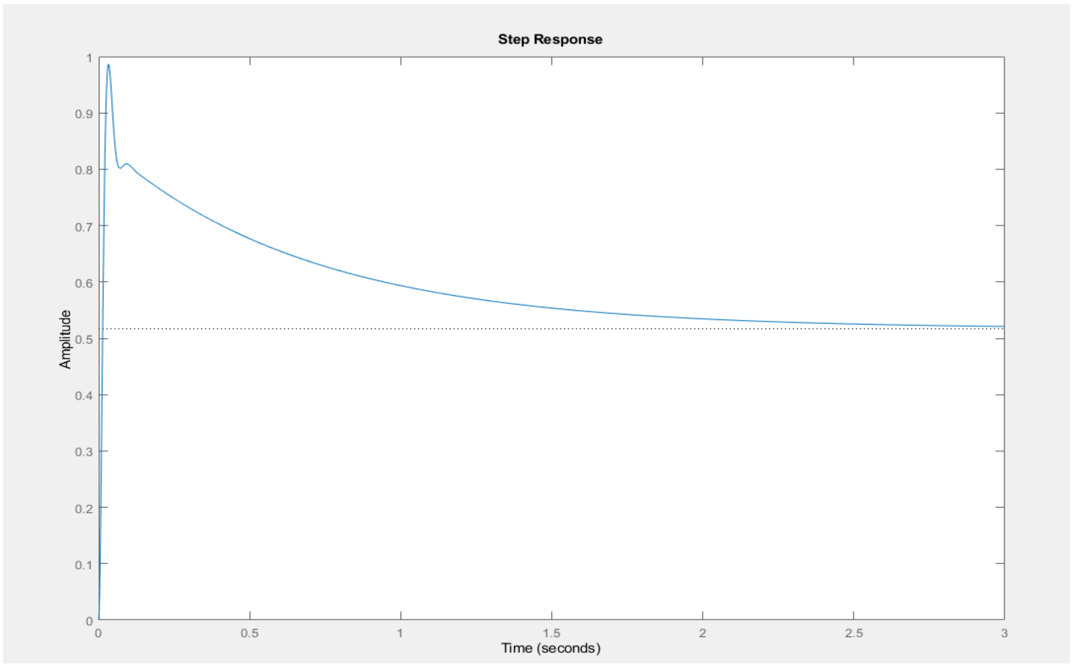
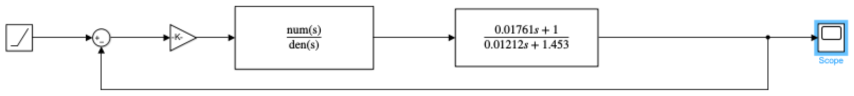


Figure 16. Step response of the controlled closed-loop system using a lead controller.

(Solution to Part (b) Q3 continued)

When we tested our system with a ramp input using SIMULINK we got the response shown in figure 17.



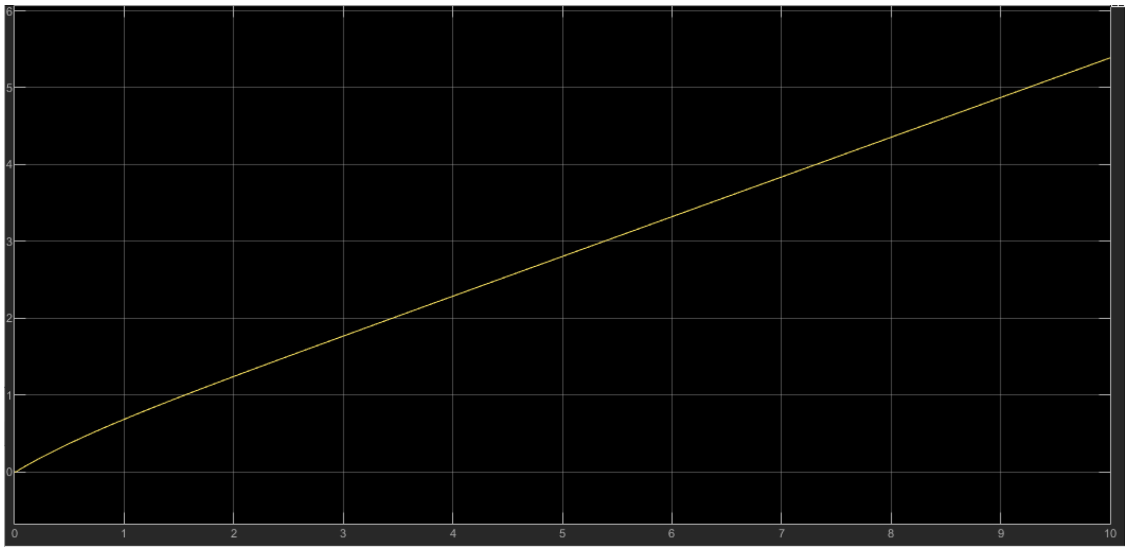
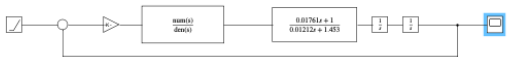


Figure 17. Response to a ramp input to the controlled closed-loop system using a lead controller.

Therefore, we added two integrators in order to remove the steady state error (Figure 18). This gave us the transfer function GGCs which gave us the desired steady state error zero with a ramp input.



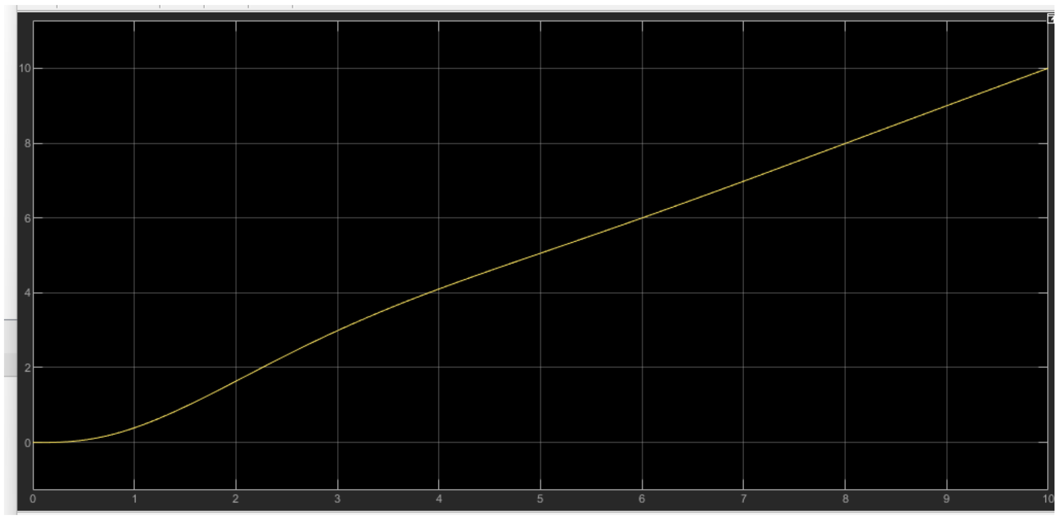
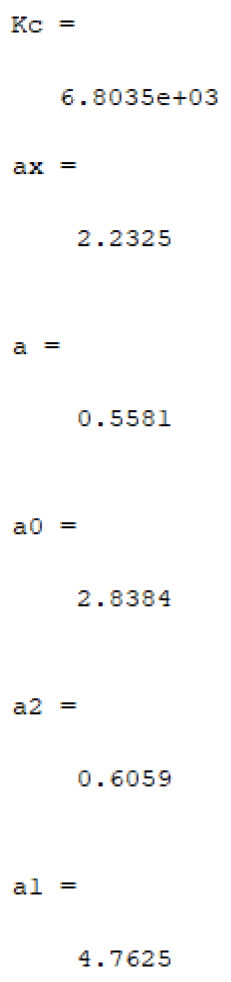


Figure 18. Response to a ramp input to the controlled closed-loop system with integrators using a lead controller.

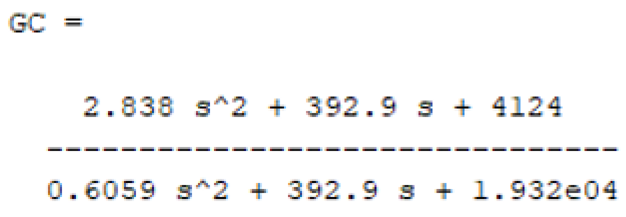
(Solution to Part (b) Q3 continued)

However, we observed that this made a change to the system's phase margin. Therefore, we had to make a few changes to the code of our function (since the bode plot needs to be shifted upwards). Furthermore, as shown in figure 19, the new value of "a" is more than 1 (2.2325), so we needed to divide it by 4 and design a second order lead controller with the new "a" and cascade the four similar controllers with our plant. The code can be found in Appendix C. We were able to find the new controller transfer function (Figure 20) along with the gain Kc (given in figure in Figure 19).



```
Kc =

   6.8035e+03

ax =

    2.2325


a =

    0.5581


a0 =

    2.8384


a2 =

    0.6059


a1 =

    4.7625
```

Figure 19. Calculated values of controller variables.



```
GC =

     2.838 s^2 + 392.9 s + 4124
  -------------------------------
  0.6059 s^2 + 392.9 s + 1.932e04
```

Figure 20. Controller transfer function.

(Solution to Part (b) Q3 continued)

After simulating the new compensated system we were able to get the bode plot in figure 21 with the phase margin of the system being 60.7 at 82.5rad/s. Our new gain margin is 15.8 dB which is still greater than 10dB.

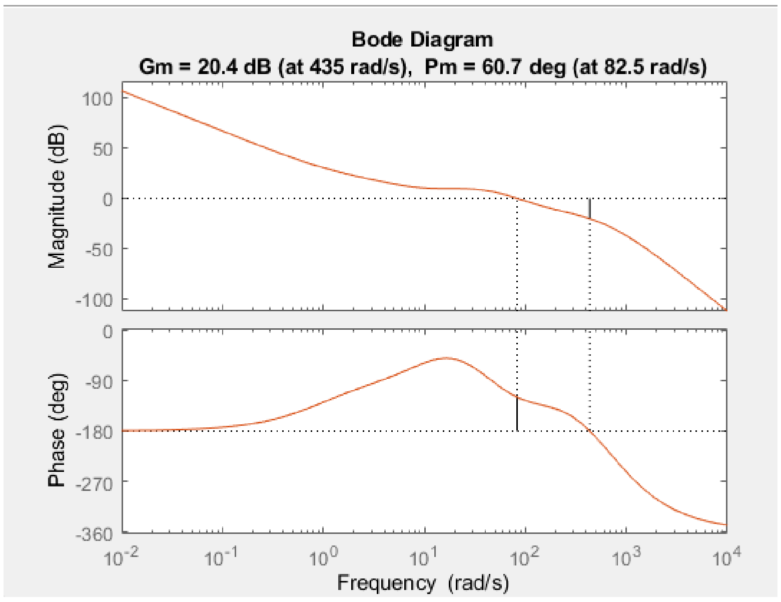


Figure 21. Bode plot of the controlled open-loop system using the new lead controller.

After simulating our response to both a step input (figure 22) and a ramp input (figure 23) we can conclude that with this new system our steady space error goes to zero.

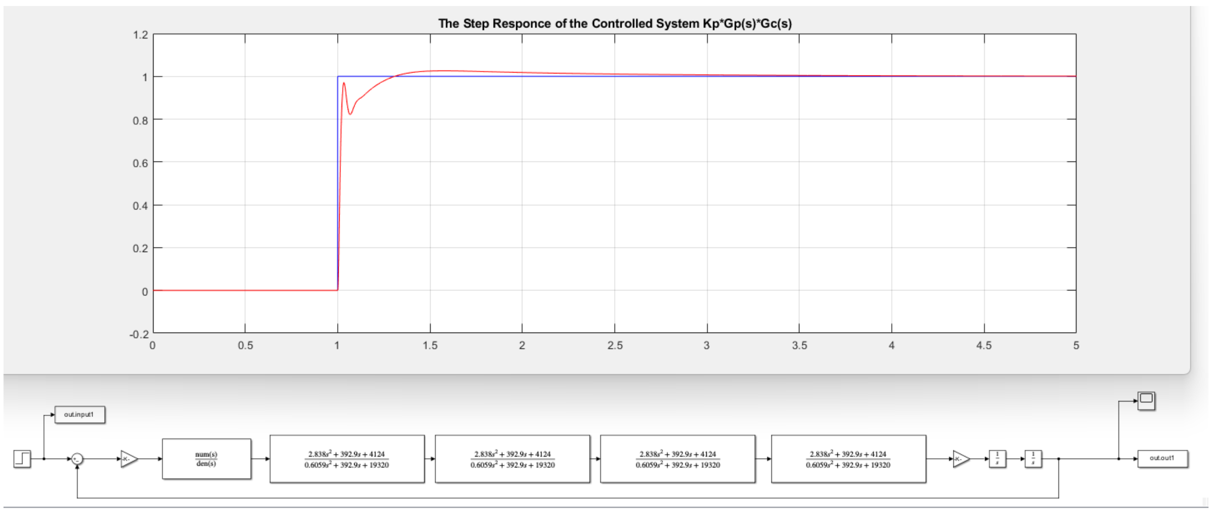


Figure 22. Response to a step input to the controlled closed-loop system with integrators using cascaded controllers.

(Solution to Part (b) Q3 continued)

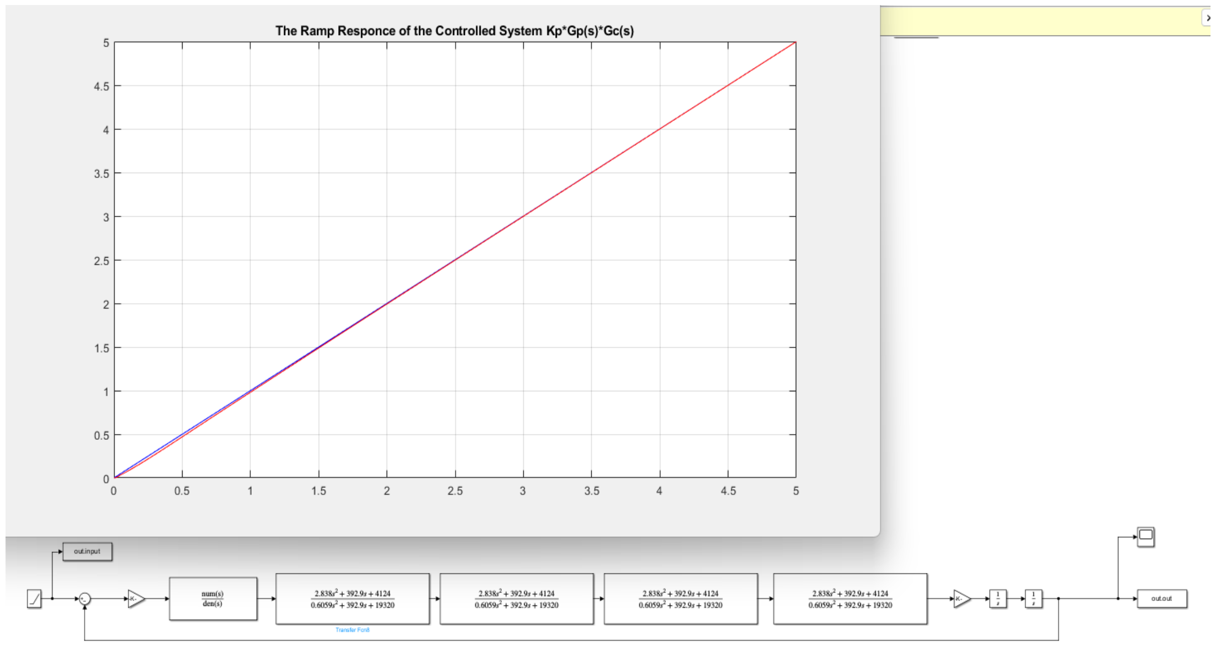


Figure 23. Response to a ramp input to the controlled closed-loop system with integrators using cascaded controllers.

Q4) (a) Design a PID controller to achieve (if possible) the same phase margin as in part (3). (b) Simulate the step response of the closed-loop system using MATLAB/SIMULINK.

(a)

B.4.1 Manual Calculation Based - Design Solution

The open-loop transfer function $G_p(s)$ is a type 0 system, and hence we follow the steps of Method one in designing the PID controller. Figure 24 shows the system unit step response, from which we obtain the T and L values. We determine that the the step input rises to 63.2% of it's final value at t = 1.0195 seconds, therefore, T = 0.0195, and L = 1.

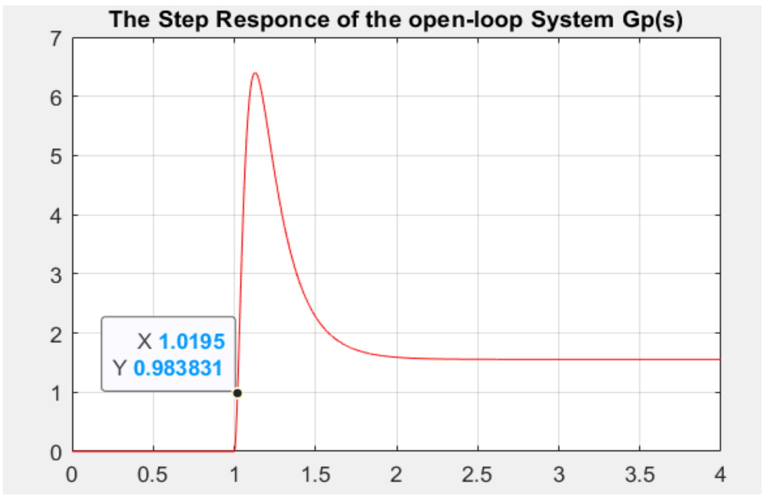


Figure 24. Open- loop system step response.

The Bode plot of the controlled system is shown in figure 26. Ideally, by carefully changing the values of T and L, we should get close to the required phase margin, however, we could not get close to satisfying the phase margin.

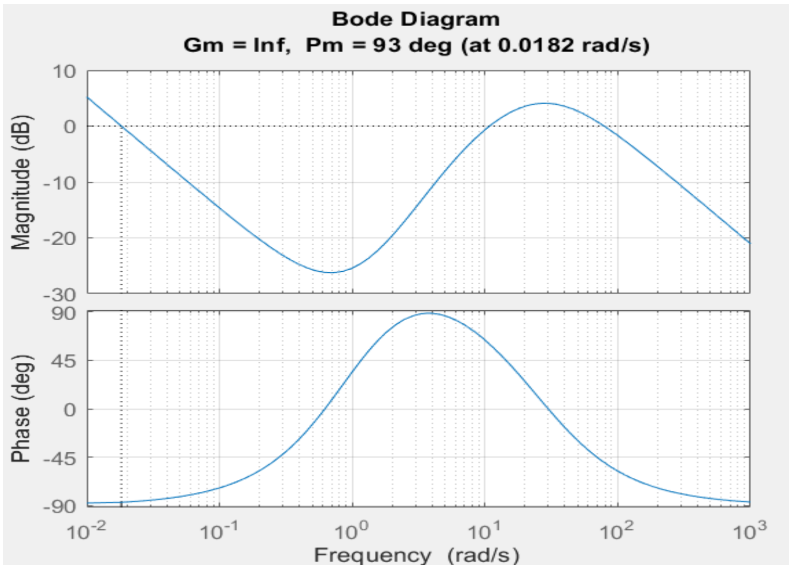
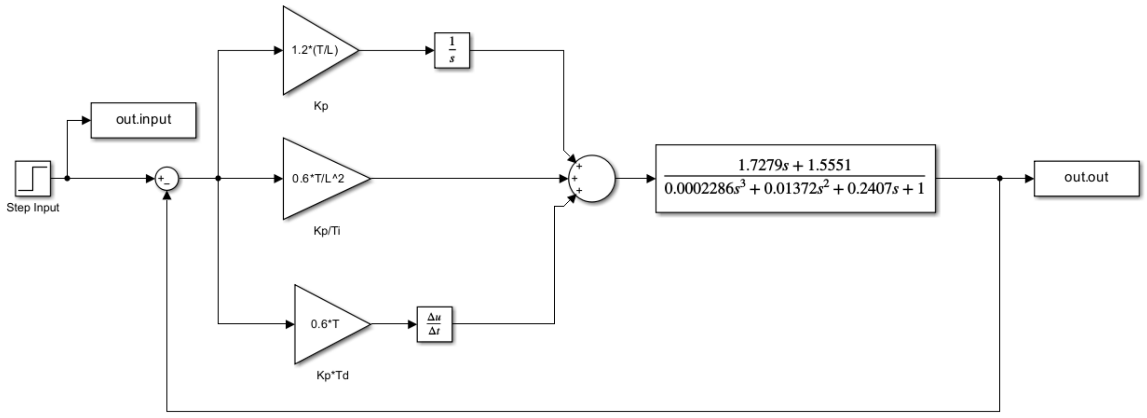


Figure 24. Bode diagram of the controlled system using a PID controller.

(b)

The block diagram of the controlled system is shown figure 25. The system unit step response is shown in figure 26.



1.2*(T/L)
Kp
$\frac{1}{s}$
out.input
0.6*T/L^2
Kp/Ti
$\frac{1.7279s+1.5551}{0.0002286s^3+0.01372s^2+0.2407s+1}$
out.out
Step Input
0.6*T
Kp*Td
$\frac{\Delta u}{\Delta t}$

Figure 25. Bode diagram of the controlled system using a PID controller.

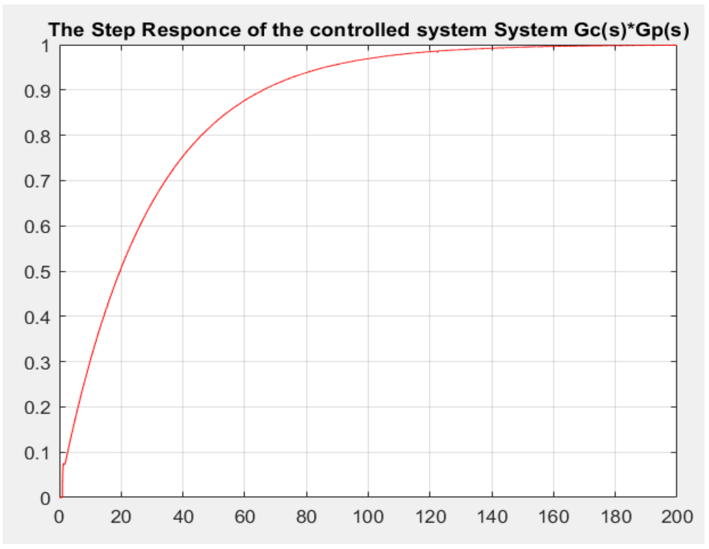


Figure 26. Unit step response of the controlled system using a PID controller.

To ensure zero tracking of the error due to a ramp input, we add one integrator in the forward path. The new Bode diagram of the system is shown in figure 27. To improve the phase margin to meet the required phase margin and to increase the speed of the system response we cascade the system with a phase-lead compensator. The compensator's parameters are calculated as shown below:

$$\alpha = \frac{60.7 - 22.4}{90} = 0.4256$$

$$\tau = \tan\left(\frac{\pi\alpha}{2}\right) + \sec\left(\frac{\pi\alpha}{2}\right) = 2.064$$

The resulting compensator transfer function is given below, Kc = 1, as there is no specification on the crossover frequency:

(Solution to Part (b) Q4 continued)

$$G_c(s) = \frac{\left(\frac{2.064s}{0.137}\right) + 1}{\left(\frac{s}{0.137}\right) + 2.064}$$

The Bode diagram of the compensated system is shown in figure 27, where we see that the phase margin requirement is satisfied. The block diagram of the modified system is shown in figure 28. The system step and ramp responses are shown in figures 29 and 30, respectively. The S.S.E is zero. Note that the system is much faster due to the addition of the phase lead compensator.

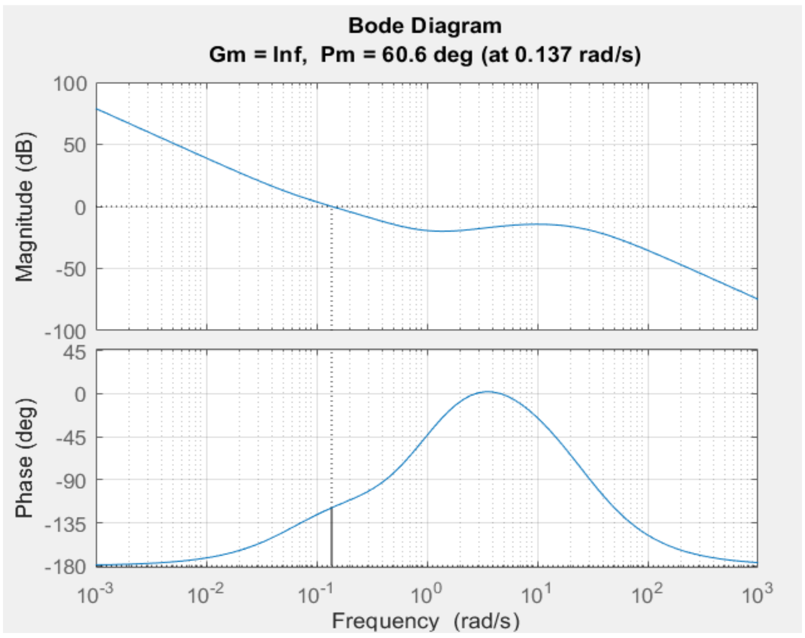
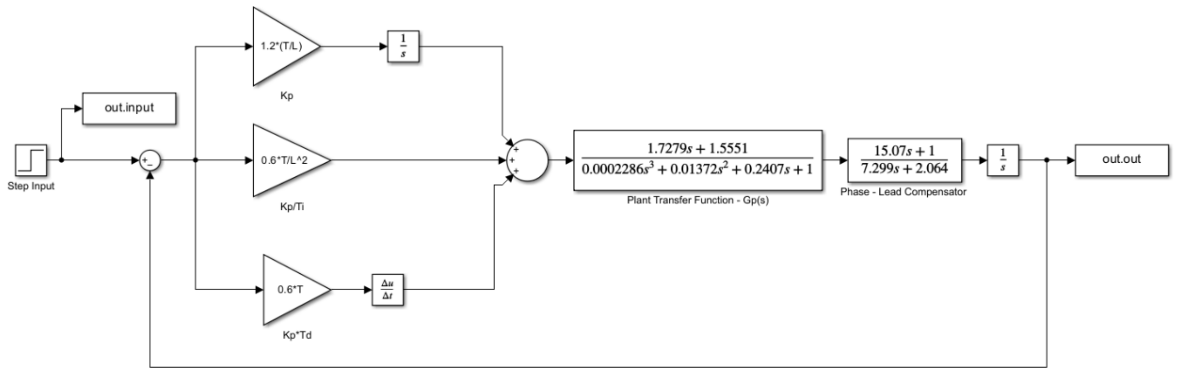


Figure 27. Bode diagram of the controlled system using a PID controller cascaded with an integrator and a phase-lead compensator.



out.input
Step Input
1.2*(T/L)
Kp
$\frac{1}{s}$
0.6*T/L^2
Kp/Ti
0.6*T
Kp*Td
$\frac{\Delta u}{\Delta t}$
$\frac{1.7279s + 1.5551}{0.0002286s^3 + 0.01372s^2 + 0.2407s + 1}$
Plant Transfer Function - Gp(s)
$\frac{15.07s + 1}{7.299s + 2.064}$
Phase - Lead Compensator
$\frac{1}{s}$
out.out

Figure 28. Bode diagram of the modified controlled system to achieve the phase margin requirement and a zero SSE for a ramp input.

(Solution to Part (b) Q4 continued)

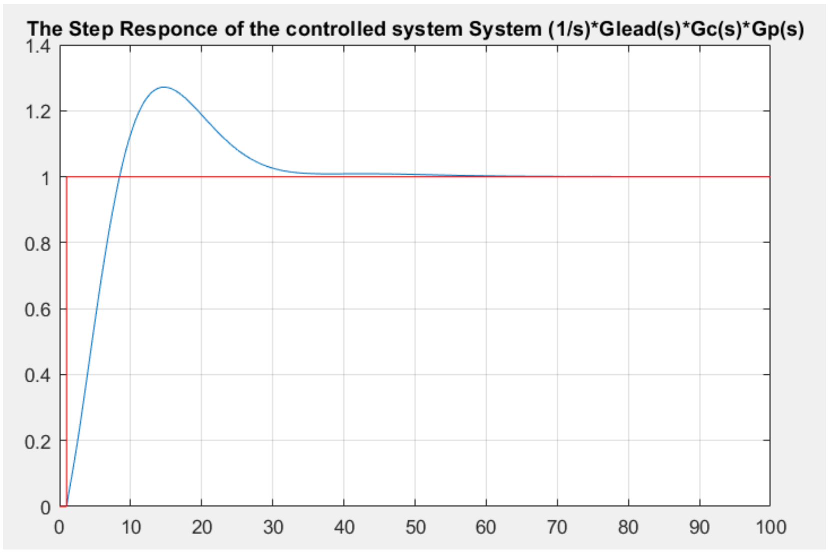


Figure 29. Step response of the PID controlled system cascaded with an integrator and a phase-lead compensator.

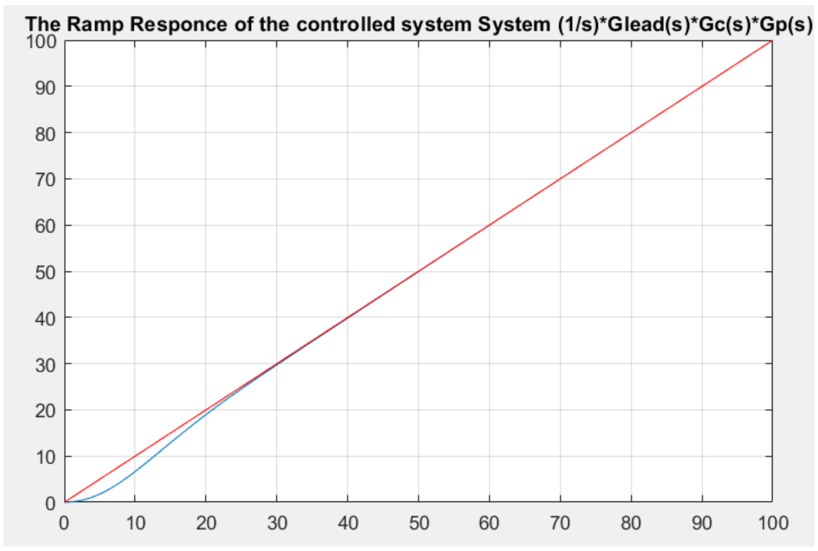


Figure 30. Ramp response of the PID controlled system cascaded with an integrator and a phase-lead compensator.

Q5) Discuss the effect of each controller of parts (2), (3), and (4) on the performance of your system (i.e., static error coefficients, overshoot, settling time, steady-state errors, … etc). Which one would you recommend to your system? Why?

Solution:

The frequency response characteristics of an open-loop system can provide important insights into the time-domain performance specifications of the system. These characteristics include bandwidth and phase margin. The phase margin is primarily a function of the system's damping ratio and is directly proportional to it, while the bandwidth is inversely related to the settling time and peak time. The static error constants are directly linked to the low-frequency gain of the system (i.e., the DC gain).

Proportional Controller:

For a proportional controller, the controlled system had a lower DC gain, lower crossover frequency, and higher phase margin. This means that:

- The controller increases the steady-state error (SSE) for a step input.
- The system response slows down, and the overshoot percentage is reduced. However, achieving higher phase margins with a proportional controller results in an even slower system response, and the SSE becomes higher.

Phase-Lead Controller:

For a phase-lead controller, when designed to maintain the same crossover frequency as the uncontrolled system, the controlled system exhibited:

- Higher DC gain and higher phase margin. This means that the compensated system:
- Decreases the steady-state error (SSE), especially for step inputs.
- Increases the system response speed by improving the transient behaviour.

PID Controller:

For a PID-controlled system, the controlled system had: Lower crossover frequency, higher phase margin, and higher DC gain. This configuration leads to:

- A significantly slower system response due to the reduction in crossover frequency.
- Increased phase margin, which improves stability but can further slow down the response.
- The system exhibits a higher DC gain, improving steady-state accuracy but not compensating for the slow response.

(Solution to Q5 continued)

As the phase-lead compensator is the easiest to design and can improve both phase margin and response speed we recommend using it to our system. Note however that it may not always be the best choice in systems that require significant improvement in steady-state error or precise disturbance rejection. For systems requiring a balance between speed, stability, and accuracy, a more robust design of a PID controller may be more appropriate.

Appendices

Appendix A – Program code for a MATLAB function that determines the gain K that makes the system critically stable.

```
function [kcr] = K_cri(z,p,k)

w = 10e-3:10e-4:10e4;
n = length(z); % determines the number of zeros and poles
m = length(p);
% determines the number of zeros and poles at the origin
n_z_index = find(z == 0);z
m_z_index = find(p == 0);
n_z = length(find(z == 0));
m_z = length(find(p == 0));

% Calculates the phase due the poles and zeros at the origin
phase = 90*n_z -90*m_z;

% Calculates the phase due the zeros not at the origin
      for q = 1:n
           if (q == n_z_index)
               continue;
           else
           phase = atand(w/z(q))+ phase;
           end
       end

% Calculates the phase due the poles not at the origin
    for l = 1:m
        if (l == m_z_index)
            continue;
        else
            phase = phase - atand(w/p(l));
        end
    end
% Accounts for the 180 degree for K<0
    if k<0

    phase = phase +180;
    end

% Determines the frequency at which the phase is -180 degree
% Takes the absolute of the subtraction of the phase by -180 degrees
% The minimum of this difference is where w = w GM
% Defines a variable w_new_crossover = w GM
diff = abs(phase + 180);
[min_valu index] = min(diff);
w_new_crossover = w(index);
```

(Continued code, Appendix A)

```
% Calculates the gain at the new_crossover frequency

G = k;
  num_abs = sqrt((w_new_crossover./z).^2 + 1);

    den_abs = sqrt((w_new_crossover./p).^2+ 1);

    for q = 1:n
    G = num_abs(q)*G;
    end

    for l = 1:m
    G = G*(1/den_abs(l));
     end

% Calculates Kcri

kn = 1./G; % As G was calculated in linear scale K_ci
kcr = kn*k;
    for q = 1:n
    kcr = kcr*z(q);
    end

    for l = 1:m
    kcr = kcr*p(l);
     end
```

Appendix B – Program code for a MATLAB function that designs a proportional controller to satisfy a given phase margin requirement.

```
function [kp] = p_con(z,p,k, phase_margin)

w = 10e-3:10e-4:10e4;
n = length(z); % determines the number of zeros and poles
m = length(p);
% determines the number of zeros and poles at the origin
n_z_index = find(z == 0);z
m_z_index = find(p == 0);
n_z = length(find(z == 0));
m_z = length(find(p == 0));

% Calculates the phase due the poles and zeros at the origin
phase = 90*n_z -90*m_z;

% Calculates the phase due the zeros not at the origin
       for q = 1:n
            if (q == n_z_index)
                continue;
            else
            phase = atand(w/z(q))+ phase;
            end
        end

% Calculates the phase due the poles not at the origin
    for l = 1:m
        if (l == m_z_index)
            continue;
        else
            phase = phase - atand(w/p(l));
        end
    end
% Accounts for the 180 degree for K<0
    if k<0

    phase = phase +180;
    end

% Determines the required phase at the new crossover frequency
    req_phase = phase_margin - 180;
% Determines the new crossover frequency
diff = abs(phase - req_phase);
[min_valu index] = min(diff);
w_new_crossover = w(index);

% Step 4: Calculate the gain at the new_crossover frequency
G = k;
```

(Continued code, Appendix B)

```
 num_abs = sqrt((w_new_crossover./z).^2 + 1);
  den_abs = sqrt((w_new_crossover./p).^2+ 1);

    for q = 1:n
    G = num_abs(q)*G;
    end


    for l = 1:m
    G = G*(1/den_abs(l));
     end

% Step 6: Calculate the controller gain
kp = 1./G;

% to plot the compensated and uncompensated system
% margin(g)
% hold
% margin(kp*g), where g is the tf of the uncompensated system
```

Appendix C – Program code for a MATLAB function that designs a PD controller to satisfy a given phase margin requirement.

```
G=tf([1/0.9, 1],[(1/(27^2*6)), (1/(27*3)+1/(27^2)),
(2/27+1/6),1])*(10^(3.83515/20))
GC=PD_controller(G,82.5,60.7)

function [GC]=PD_controller(G,Wc,w)
Gain=abs(evalfr(G,Wc*i))
Kc=1/Gain
G1=Kc*G
[Gm,Pm,WCG,WCP]=margin(G1)
a=(w-Pm)/90
T=tan(a*pi/2)+sec(a*pi/2)
GC=tf([(T/Wc), 1],[(1/Wc), T])
margin(G1*GC)
end
```

Updated code for system with integrator:

```
G=tf([1/0.9, 1],[(1/(27^2*6)), (1/(27*3)+1/(27^2)), (2/27+1/6),1,
0])*(10^(3.83515/20))
GC=PD_controller(G,82.5,60.7)
%margin(G*GC)

function [GC]=PD_controller(G,Wc,w)
Gain=abs(evalfr(G,Wc*i))
k=log10(Gain)
Kc=10^-k
G1=Kc*G
[Gm,Pm,WCG,WCP]=margin(G1)
a2=(w-Pm)/90
a=a2/2
T=tan(a*pi/2)+sec(a*pi/2)
GC=tf([(T/Wc), 1],[(1/Wc), T])
margin(G1*GC*GC)
end
```