ASEN 5114
AUTOMATIC CONTROL SYSTEMS

PROJECT - 1

University of Colorado Boulder

Done by:
Anshul Jain, Surya Shashank Sekhar Hari

Under the guidance of:
Prof. Dale Lawrence

TABLE OF CONTENTS

Purpose of Project:.......... 3

Individual Contribution to the Project:.......... 3

Q1.
Curve Fitting.......... 4

Q2.
Stability Analysis for various Proportional Controller Gains using Bode & Nyquist Plots.......... 8

Q3:
Designing a Suitable Compensator.......... 20

Q4:
Applying Compensator Design to Empirical Data.......... 29

Q5:
Simulating the obtained System on Simulink.......... 35

Q6:
Spacecraft Model with First Resonance Occurring Before Anti-Resonance.......... 42

APPENDIX.......... 46

Purpose of Project:

The objective of this project is to utilize frequency domain modeling and design techniques to control the attitude of a spacecraft mockup. A hybrid approach, combining both analytic and empirical modeling, will be employed to derive a transfer function that represents the spacecraft's dynamics. Following this, a controller will be designed using frequency response methods and implemented in a simulation of the hardware mockup. The predicted system responses will be compared to the simulated outcomes. Additionally, the project will explore the challenges associated with collocated and non-collocated control configurations.

Individual Contribution to the Project:

The present project is presented by Anshul Jain and Surya Shashank Sekhar Hari both of whom made equal contributions to this project. Their contributions are as:

Task	Execution of the Code	Report Writing
Question 1	Surya Shashank Sekhar Hari	Anshul Jain
Question 2	Surya Shashank Sekhar Hari	Anshul Jain
Question 3	Anshul Jain	Surya Shashank Sekhar Hari
Question 4	Anshul Jain	Surya Shashank Sekhar Hari
Question 5	Anshul Jain	Surya Shashank Sekhar Hari
Question 6	Surya Shashank Sekhar Hari	Anshul Jain

Q1. Curve Fitting

Develop an analytic transfer function model to match the empirical frequency response obtained from the spacecraft mockup hardware, for frequencies up through the first resonance/anti-resonance.

In this task, the objective is to generate a transfer function that accurately fits the empirical data provided, which includes magnitude and phase data at various frequencies for the spacecraft mockup. Initially, an Empirical Bode plot was constructed to analyze the frequency response and identify key characteristics such as corner frequencies, which will be instrumental in deriving an analytical transfer function.

Upon inspection of the empirical Bode magnitude plot, it was observed that the plot exhibited a characteristic initial slope, gradually decreasing at lower frequencies, followed by two distinct pairs of anti-resonance (notches) and resonance (peaks) features. The gradual decline in slope at low frequencies suggests the presence of a first-order pole situated near the origin, though not exactly at the origin, as indicated by the phase plot, which deviates from a flat line. The occurrence of two resonance and anti-resonance pairs strongly points to the presence of two second-order poles and two second-order zeros. These observations collectively suggest that the system is characterized by a transfer function with two zeros and three poles. Therefore, the transfer function (H) can be represented in the general form:

$$H = \frac{K_x * (s^2 + 2\zeta_1 . w_{n1} + w_{n1}^2)(s^2 + 2\zeta_3 . w_{n3} + w_{n3}^2)}{(s-a)((s^2 + 2\zeta_2 . w_{n2} + w_{n2}^2)(s^2 + 2\zeta_4 . w_{n4} + w_{n4}^2)}$$

where,

a	=	First Order Pole present close to origin
$\zeta_1, \zeta_2, \zeta_3, \zeta_4$	=	Damping factors of each peak or notch
$w_{n1}, w_{n2}, w_{n3}, w_{n4}$	=	Natural Frequency of each peak or notch
K_x	=	Multiplication Factor to adjust the curve

Upon further examination of the Bode plot, the exact values of the natural frequencies corresponding to the resonance (peaks) and anti-resonance (notches) locations were identified. These frequencies are critical in accurately characterizing the system's behavior and are directly related to the natural frequencies of the second-order poles and zeros. The values are as follows:

w_{n1} = 0.7323 rad/s w_{n2} = 4.88315 rad/s w_{n3} = 1.32844 rad/s w_{n4} = 6.13774 rad/s

The damping factors can be derived from the amplitude (magnitude) of the peaks (resonances) or notches (anti-resonances) observed in the empirical magnitude data. This can be achieved using the following equation,

$$Magnitude = -20 * log_{10}(2\zeta)$$

$\zeta_1 = 0.0163$ $\zeta_2 = 0.0306$ $\zeta_3 = 0.0312$ $\zeta_4 = 0.0312$

By substituting all the relevant values and adjusting the multiplication factor along with the first-order pole, the resulting transfer function is derived as follows:

$$\boxed{H = \frac{0.0397*(s^4 + 0.3284s^3 + 24.39s^2 + 0.7331s + 12.79)}{s^5 + 0.5899s^4 + 39.53s^3 + 8.703s^2 + 66.95s + 8.37}}$$

The Final Bode plot, comparing the Analytical Transfer Function with the Empirical Data, is presented below:

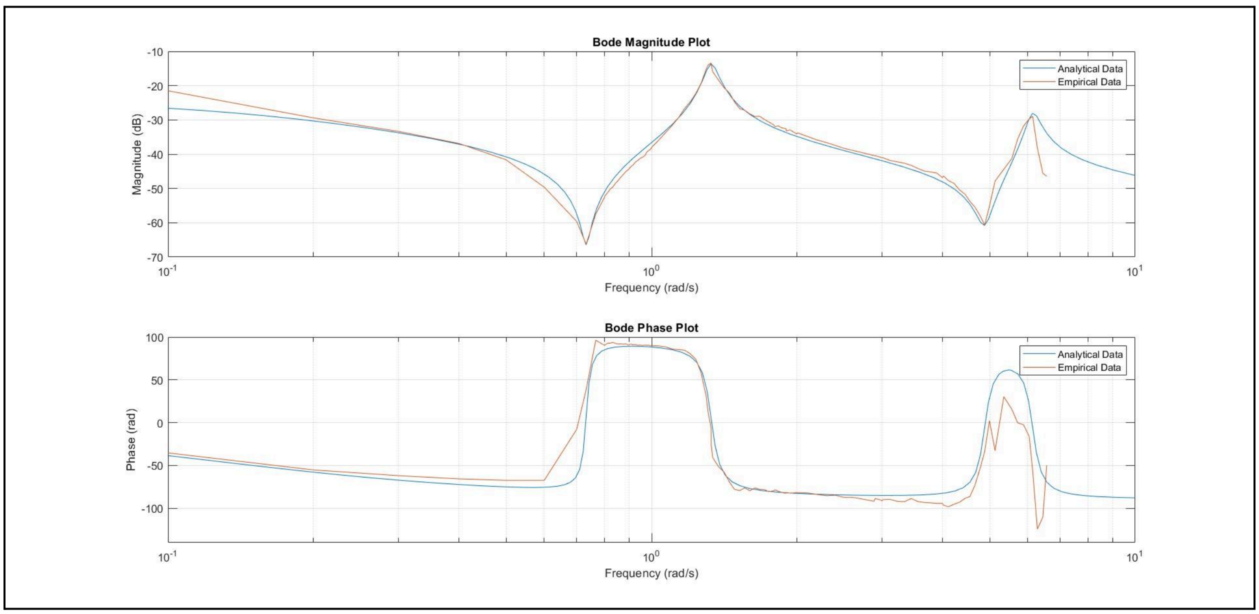


Figure 1.1 Analytic Transfer Function Bode Plot vs Empirical Data Bode Plot

MATLAB Code for Curve Fitting:

```
% Read Data from Excel
rawTable = readtable('Spacecraft_spin_module_frequency_response_data.xlsx','Sheet','Sheet1');

freq_data = rawTable.Frequency_Hz_;
mag_data = rawTable.Magnitude_rad_sec_V_;
mag_data = 20*log10(mag_data);
phase_data = rawTable.Phase_rad_;
phase_data = phase_data*360/(2*pi);

% Curve Fitting Transfer function
% Defining Variables
wn1 = 0.7323;
mag1 = 29.7268;
zeta1 = (10^(-mag1/20))/2;
wn2 = 1.32844;
mag2 = 24.2747;
zeta2 = (10^(-mag2/20))/2;
wn3 = 4.88315;
mag3 = 24.1018;
zeta3 = (10^(-mag3/20))/2;
wn4 = 6.13774;
mag4 = 24.1018;
zeta4 = (10^(-mag4/20))/2;
Kx = 10^(-28.034/20);

a = 10^-0.9;
b1 = 2*zeta1*wn1;
c1 = wn1^2;
d1 = 2*zeta3*wn3;
e1 = wn3^2;
b = 2*zeta2*wn2;
c = wn2^2;
d = 2*zeta4*wn4;
e = wn4^2;

% Substituting Values into the Transfer Function
H = tf(Kx*[1     b1+d1    e1+(b1*d1)+c1    (b1*e1)+(c1*d1)  c1*e1]  ,
[1     (d+b+a)     (e+((b+a)*d)+c+ (a*b))     (((b+a)*e)+((c+(a*b))*d)+(a*c))     (((c+(a*b))*e)+(a*c*d))     a*c*e])

[mag, phase, freq] = bode(H);
```

```
% Generating Bode Plot and comparing Empirical & Analytical Data

Figure;
subplot(2,1,1);
semilogx(freq, 20*log10(squeeze(mag)));
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');
title('Bode Magnitude Plot'); grid on;
hold on;
plot(freq_data,mag_data);
legend('Analytical Data', 'Empirical Data');
xlim([0.1 10]);
hold off

subplot(2,1,2);
semilogx(freq, squeeze(phase));
xlabel('Frequency (rad/s)');
ylabel('Phase (rad)');
title('Bode Phase Plot');
axis([0 12 -140 96]);
grid on;
hold on;
plot(freq_data,phase_data);
legend('Analytical Data', 'Empirical Data');
axis([0.1 10 -140 100]);
```

```
H =

          0.03966 s^4 + 0.01302 s^3 + 0.9671 s^2 + 0.02907 s + 0.5071
  ---------------------------------------------------------------------------------
          s^5 + 0.5899 s^4 + 39.53 s^3 + 8.703 s^2 + 66.95 s + 8.37
```

Q2. Stability Analysis for various Proportional Controller Gains

Examine the two loop transfer functions (analytic and empirical) via Bode plots and Nyquist plots using a proportional controller C(s)=K. Would the control system be stable for some values of K? For all values of K?

For this task, a proportional controller is assumed in series with the transfer function, and the goal is to analyze the stability of the system for various values of the proportional gain K. To assess the system's stability, Bode plots and Nyquist plots are used varying K logarithmically over the range

$$K = \{-10, -1, -0.1, -0.01, 0.01, 0.1, 1, 10, 100\}$$

The system's stability will be determined by whether the phase margin is positive and whether the Nyquist plot encircles the $(-1,0)$ point. For this purpose, the Nyquist plots are generated after multiplying the transfer function with an integrator. The results can be summarised as follows:

Value of "K"	Description of the Nyquist Plot	Stability
K = -10	The plot clearly encircles the -1+jw in the clockwise direction	Unstable
K = -1	There's smaller but considerable loopings around -1+jw in clockwise direction	Unstable
K = -0.1	Presence of a very small loop close to origin and wrapping around -1+jw in clockwise direction	Unstable
K = -0.01	A divergent curve almost near origin exploding in 90° presumably wrapping around -1+jw	Unstable
K = 0.01	A tiny loop far right from -1+jw	Stable
K = 0.1	A considerable loop that does not wrap around -1+jw	Stable
K = 1	Loop is still away from -1+jw but gets closer to region of instability	Marginally Stable
K = 10	The plot clearly encircles the -1+jw in the clockwise direction	Unstable
K = 100	The plot clearly encircles the -1+jw in the clockwise direction	Unstable

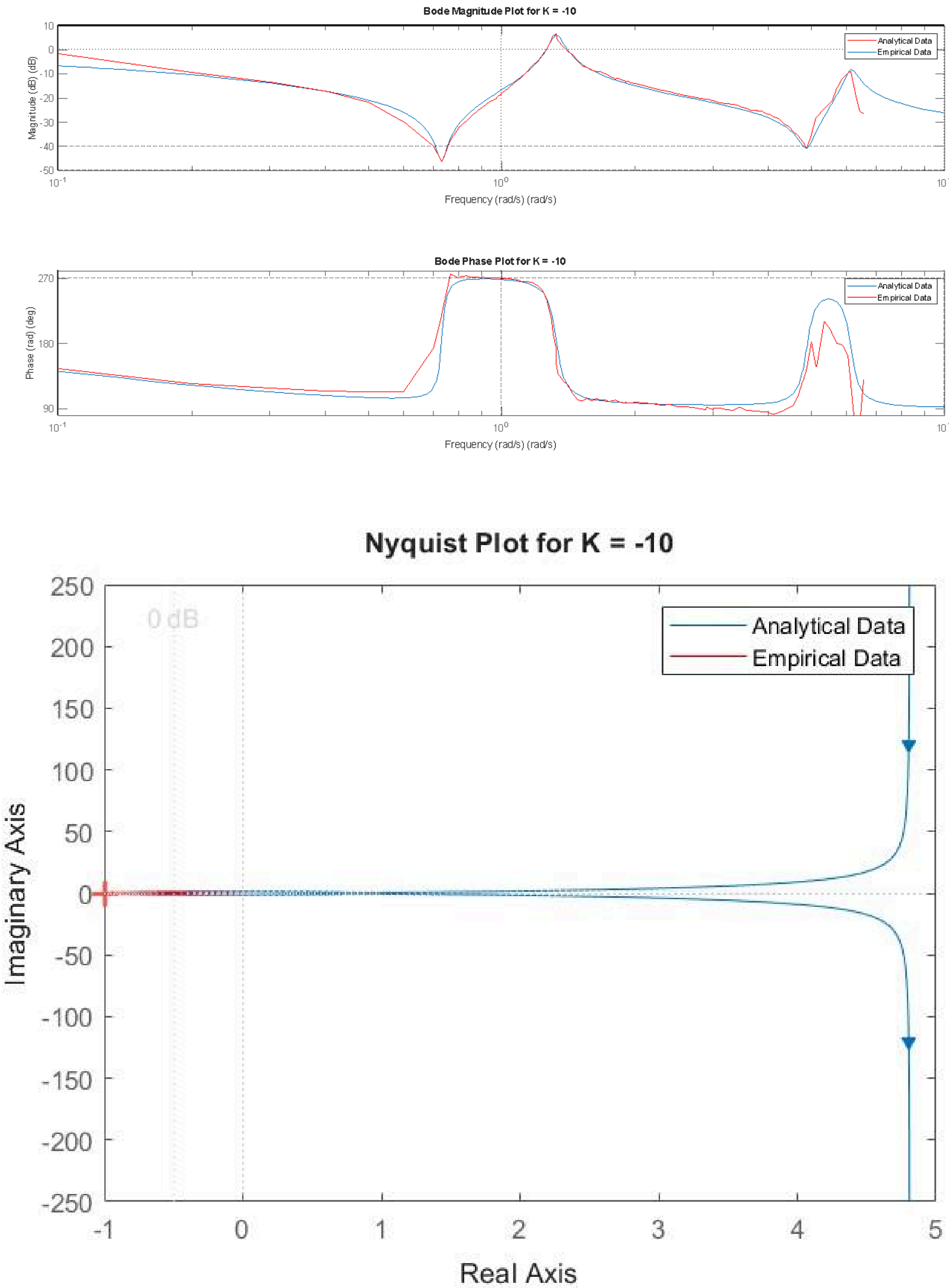
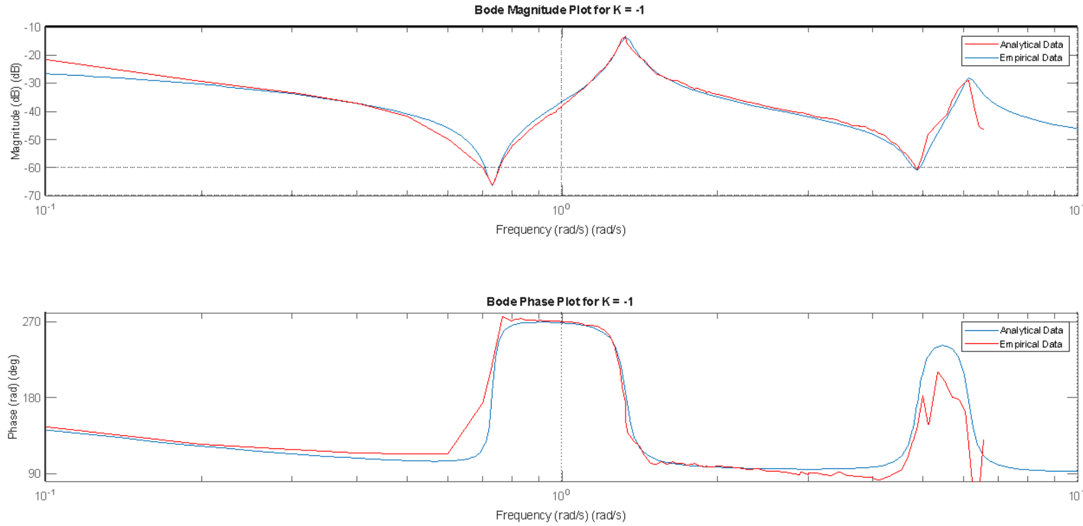


Figure 2.1 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = -10



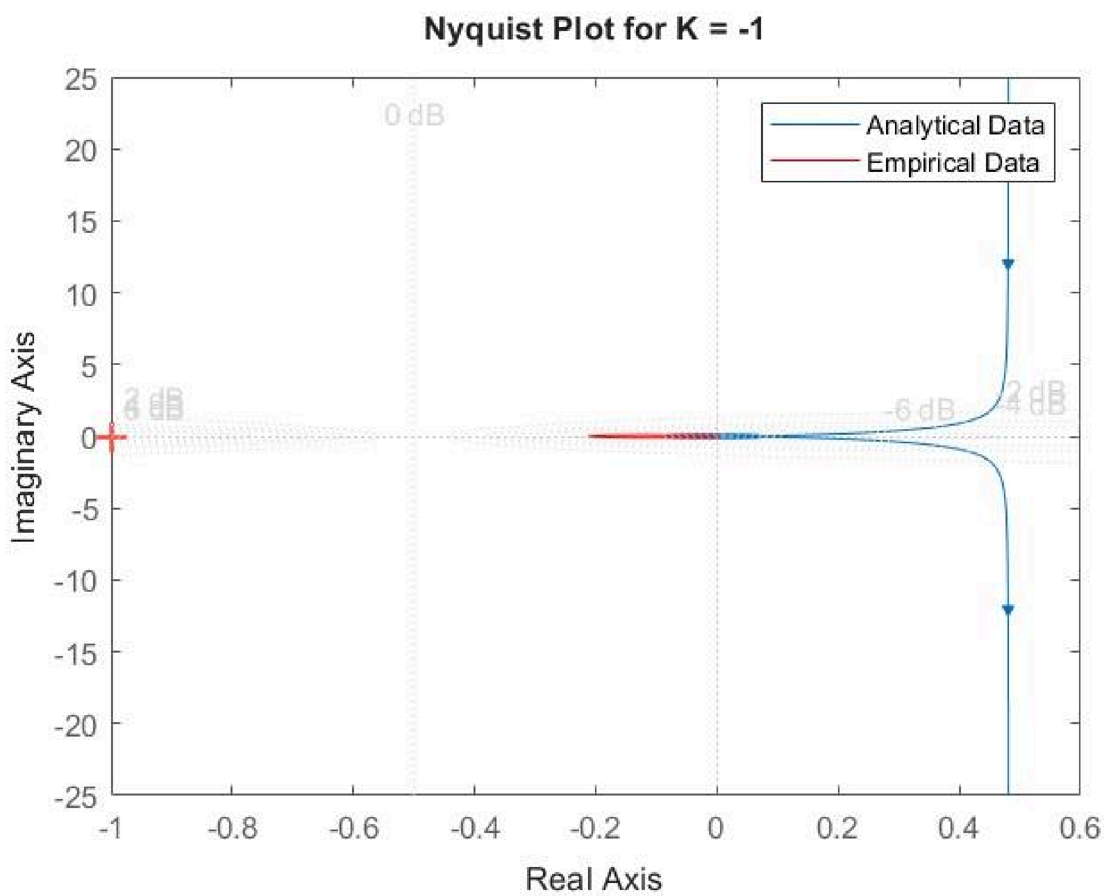


Figure 2.2 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = -1

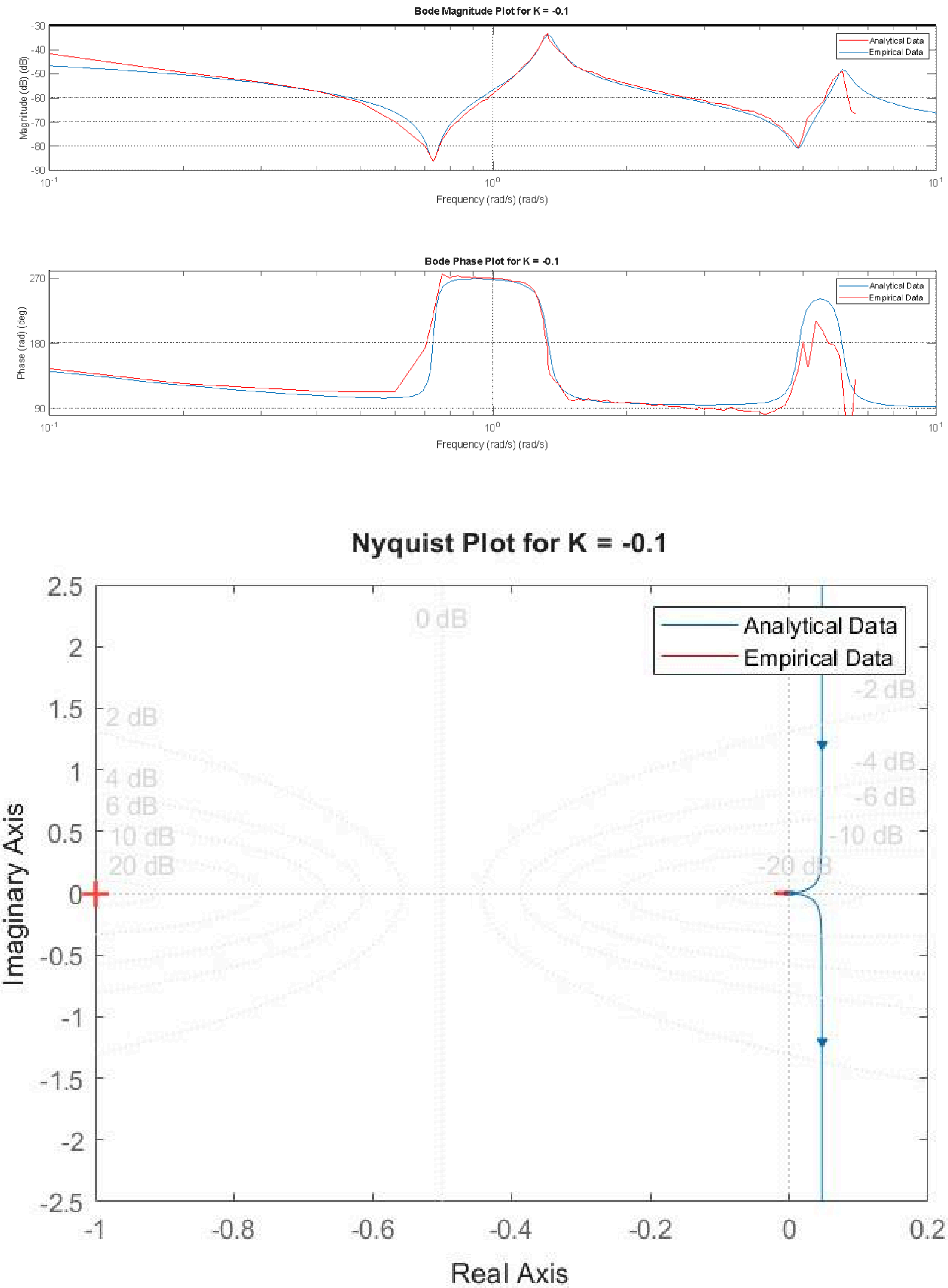


Figure 2.3 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = -0.1

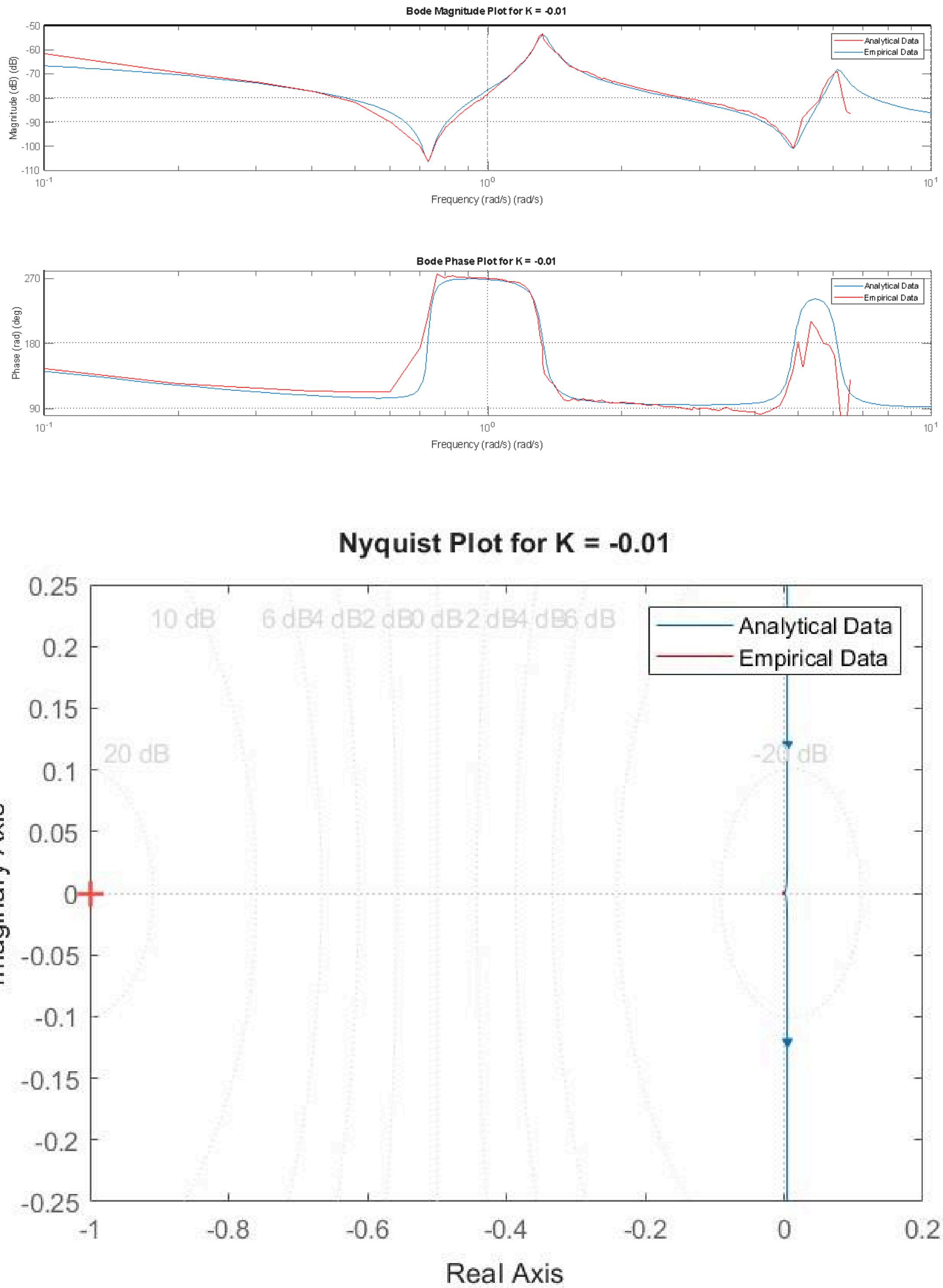


Figure 2.4 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = -0.01

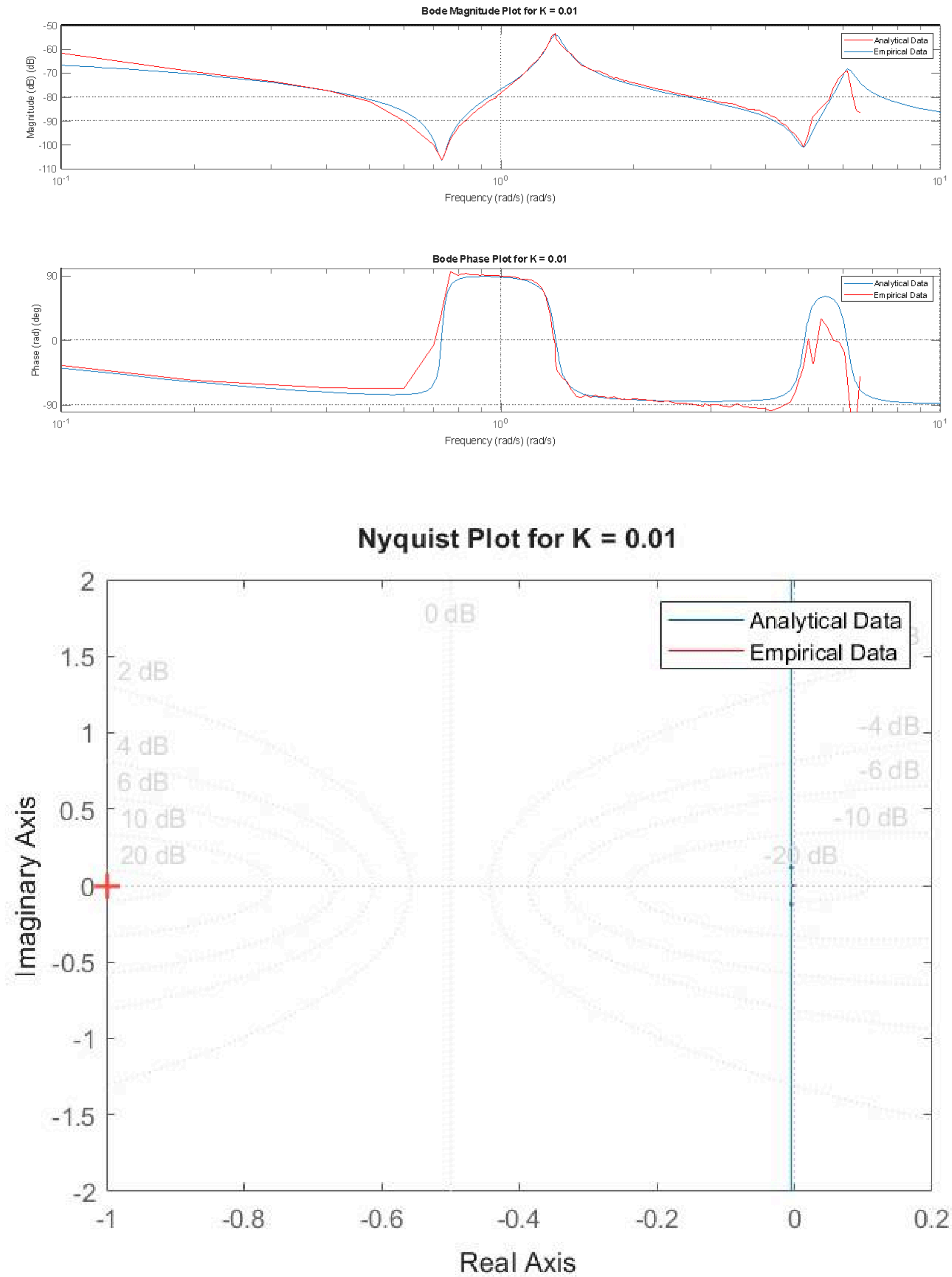


Figure 2.5 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = 0.01

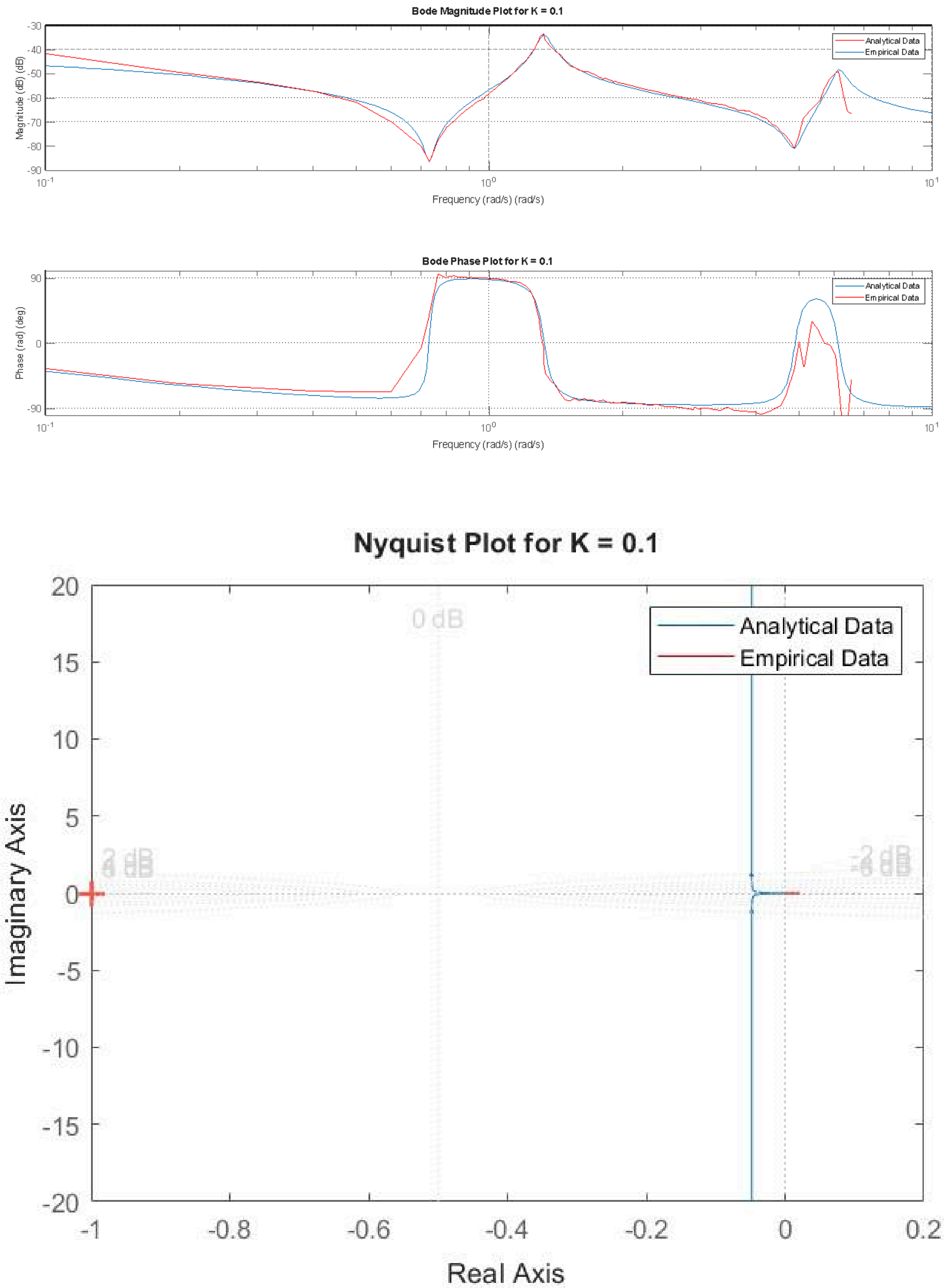


Figure 2.6 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = 0.1

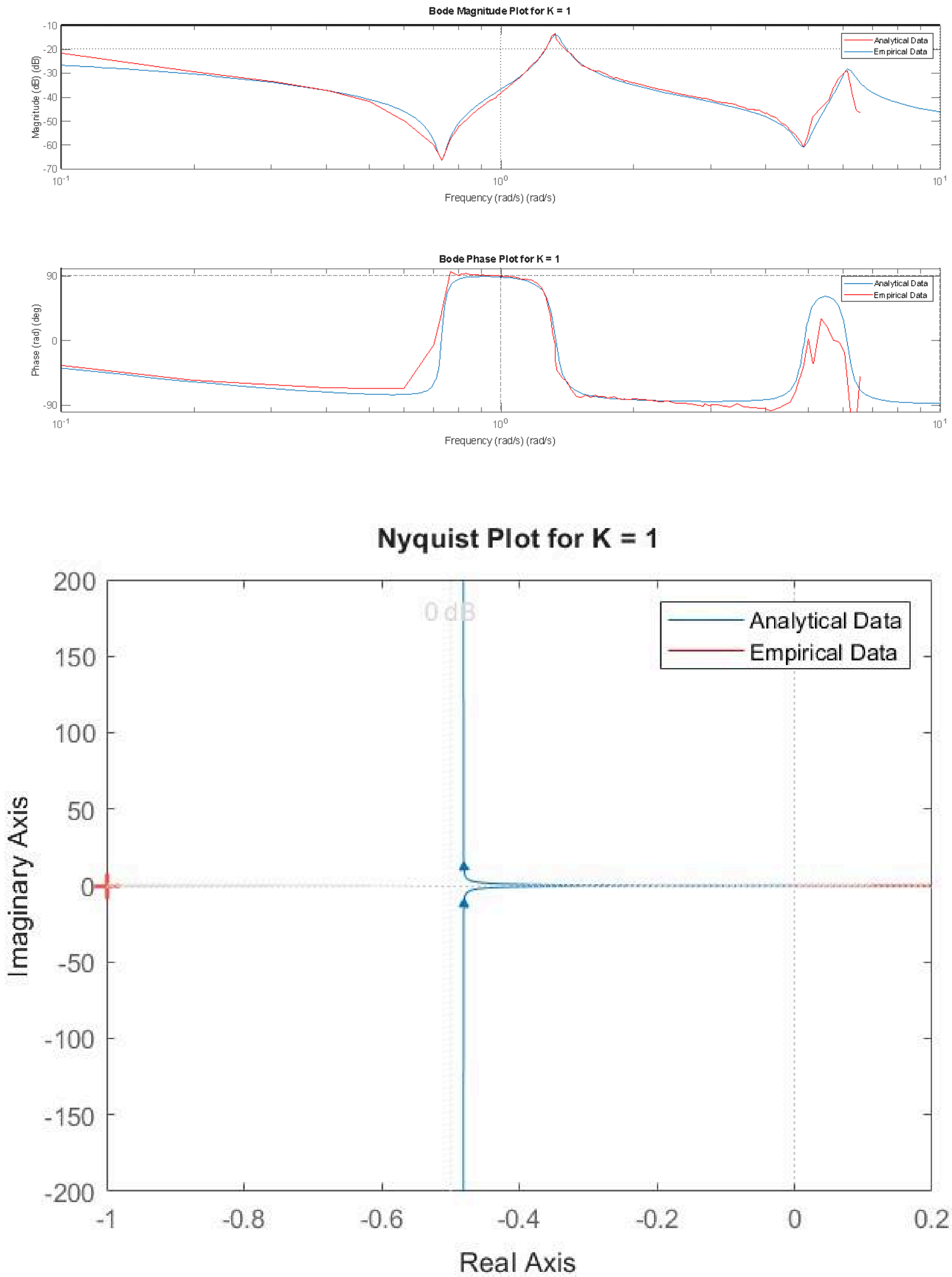
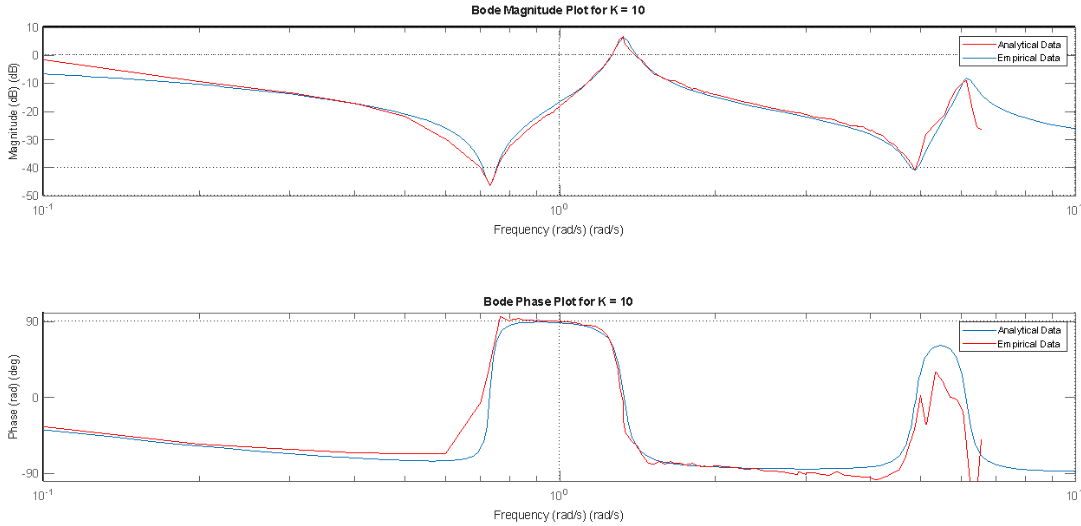


Figure 2.7 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = 1



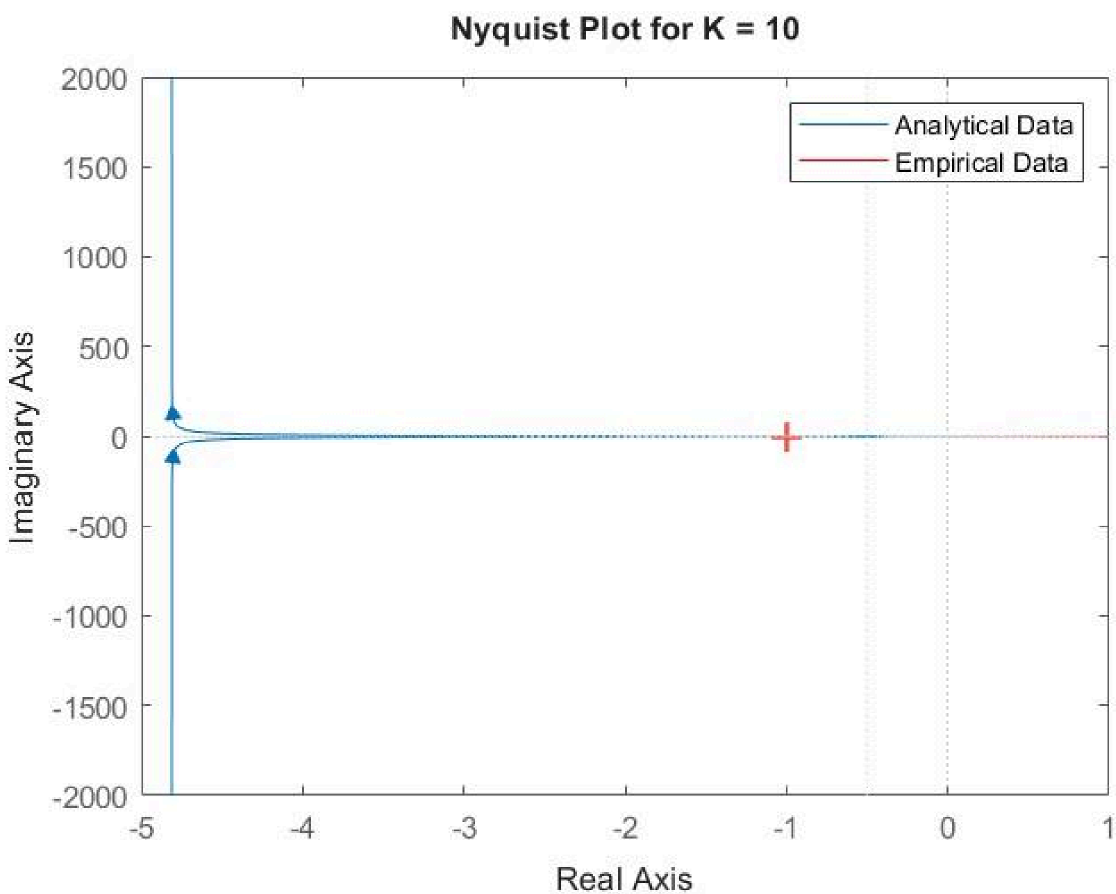


Figure 2.8 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = 10

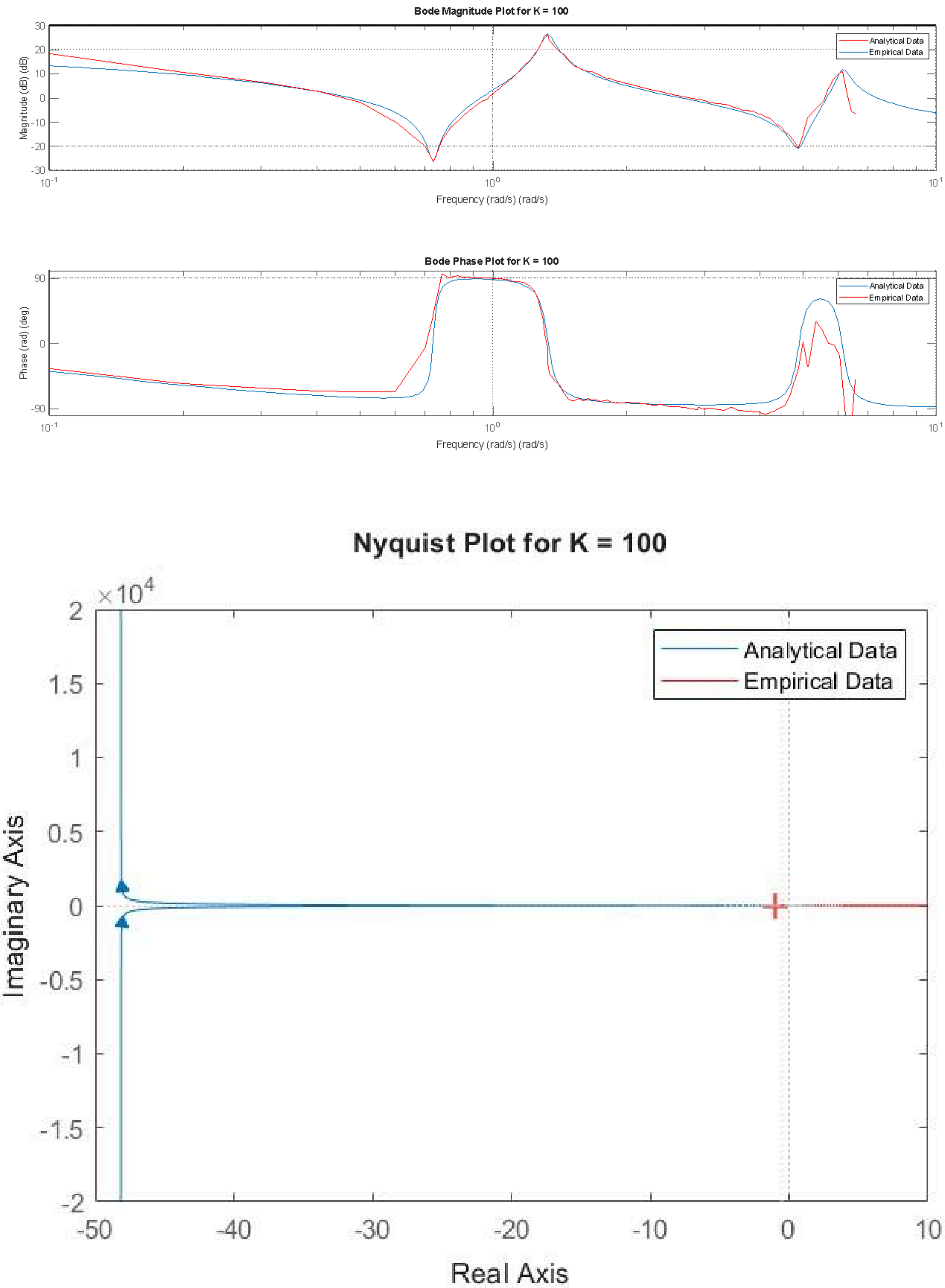


Figure 2.9 Bode & Nyquist Plots of Analytical & Empirical System for Proportional gain K = 100

- It is clear from the graphs that the plot is not stable for all values of K
- For negative values of K, even small values cause the system to become unstable
- When K is very small and positive, the plot stays well away from -1, so the system is stable. However, the curve slowly starts to stretch towards -1+jw
- For larger values of K, the systems becomes unstable again.

MATLAB Code for Stability Analysis:

```
% Define the range of K values, including both small and large values
% Specific values of K
K_values = [-10, -1, -0.1, -0.01, 0.01, 0.1, 1, 10, 100, 1000];

phases_deg = deg2rad(phase_data);
mag_nyq = 10.^(mag_data/20);
nyq_real = mag_nyq .* cos(phases_deg);
nyq_imag = mag_nyq .* sin(phases_deg);


% Loop over the K values
for i = 1:length(K_values)
K = K_values(i);
 % Open-loop transfer function L(s) = K * G(s)
L = K * H;

% Bode plot
figure;
subplot(2,1,1);
plot(freq_data,mag_data+(20*log10(K)), 'r');
hold on;
x = bodeplot(L);
setoptions(x, 'PhaseVisible', 'off');
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');
title(['Bode Magnitude Plot for K = ', num2str(K)]);
grid on;
plot(freq_data,K*mag_data,'r');
legend('Analytical Data', 'Empirical Data');
xlim([0.1 10]);
hold off

subplot(2,1,2);
x = bodeplot(L);
setoptions(x, 'MagVisible', 'off');
xlabel('Frequency (rad/s)');
ylabel('Phase (rad)');
axis([0 12 -140 96]);
title(['Bode Phase Plot for K = ', num2str(K)]);
grid on;
```

```
hold on;
plot(freq_data,phase_data-(((abs(K)-K)/(2*K))*180), 'r');
legend('Analytical Data', 'Empirical Data')
axis([0.1 10 -140 100]);

% Nyquist plot
figure
nyquist(L);
hold on
plot(K*nyq_real,K*nyq_imag,"r");
grid on;
title(['Nyquist Plot for K = ', num2str(K)]);
legend('Analytical Data', 'Empirical Data');

pause(0.5);


end
```

Q3: Designing a Suitable Compensator

Design a compensator $C_1(s)$ so that the unity feedback control system with $C(s) = K * C_1(s)$ has at least 40° phase margin, 10 dB of gain margin, and a closed-loop (-3 dB) bandwidth as close to 1 Hz as possible. Use the analytic plant model in this design. Compute and plot the closed-loop tracking frequency response.

The goal was to design a compensator, $C_1(s)$, such that the unity feedback system with:
$C(s) = K * C_1(s)$
meets the following performance requirements:

- Phase Margin (PM) at least 40°
- Gain Margin (GM) at least 10 dB
- Closed-Loop Bandwidth ~ 1 Hz

The analytic plant transfer function was used to make this compensator.

As shown in Figure 3.1, the open loop Bode Plot of the plant indicates no gain crossover point (the magnitude plot never crosses 0 dB), PM and GM are reported as Infinite. Although the plant is stable in an open loop, it cannot be used directly in the closed loop as it does not achieve a meaningful crossover.

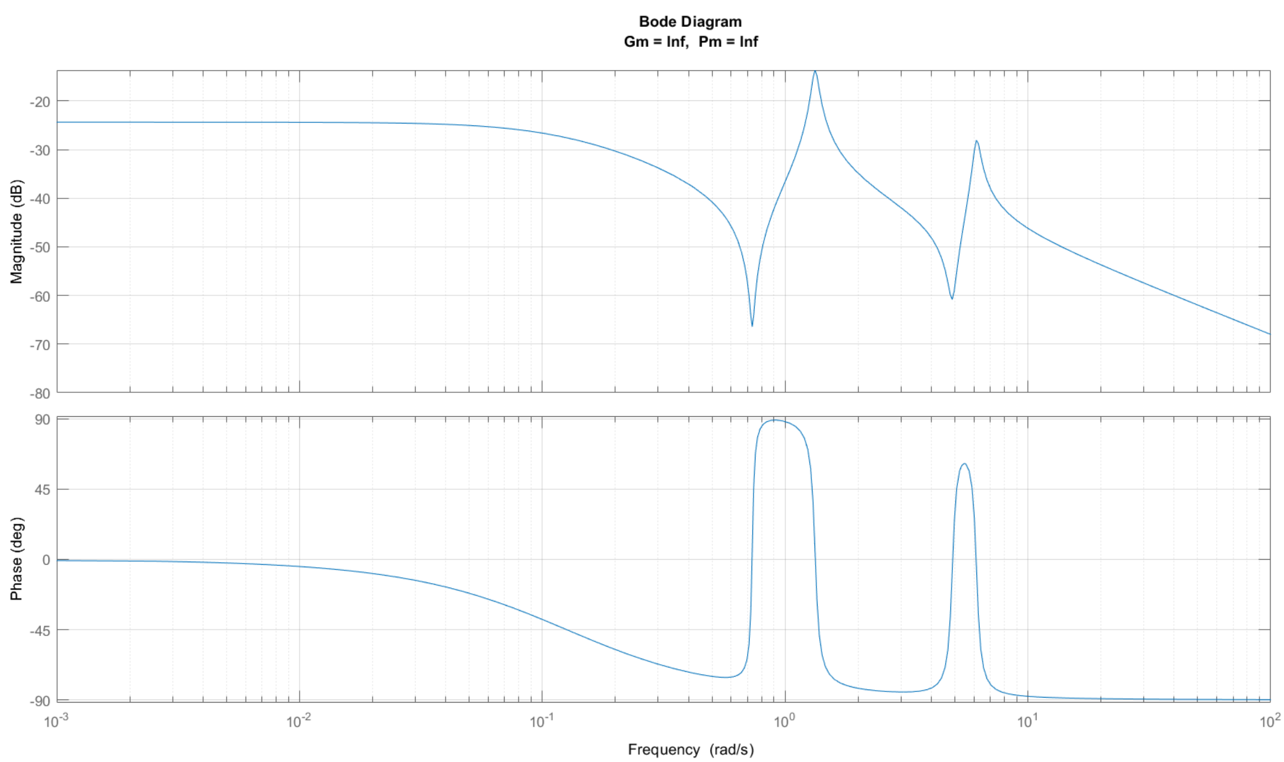


Figure 3.1 Analytic Transfer Function Bode Plot

This indicates a highly-passive or low-gain system, and a controller is needed to shift the crossover to around 1 Hz while ensuring stability margins.

MATLAB CODE:

```
% Analytic Transfer function from Part 1
H = tf(K*[1 b1+d1 e1+(b1*d1)+c1 (b1*e1)+(c1*d1) c1*e1],[1 (d+b+a)
(e+((b+a)*d)+c+(a*b)) (((b+a)*e)+((c+(a*b))*d)+(a*c))
(((c+(a*b))*e)+(a*c*d)) a*c*e])

% Check the margins of H
margin(H)
grid on;

% The GM and PM are Inf.
% We have the target gain crossover frequency at 6.28 rad/sec.
% The Phase at 6.28 rad/sec is -40 deg. Phase margin calculation
% with this phase is 140 deg.
% We need to add positive gain to the compensator such that
% the magnitude plot crosses 0 dB at 6.28 rad/sec.
```

A proportional gain was first introduced to bring the crossover frequency to 1 Hz.

- Required gain:
  - $K_c = \frac{1}{|H(jw_c)|}$, $where\ w_c = 2\pi * 1$,
  - While the crossover was achieved, the phase margin at this point was **-96.7°**, implying a highly unstable system, as depicted in Figure 3.2.

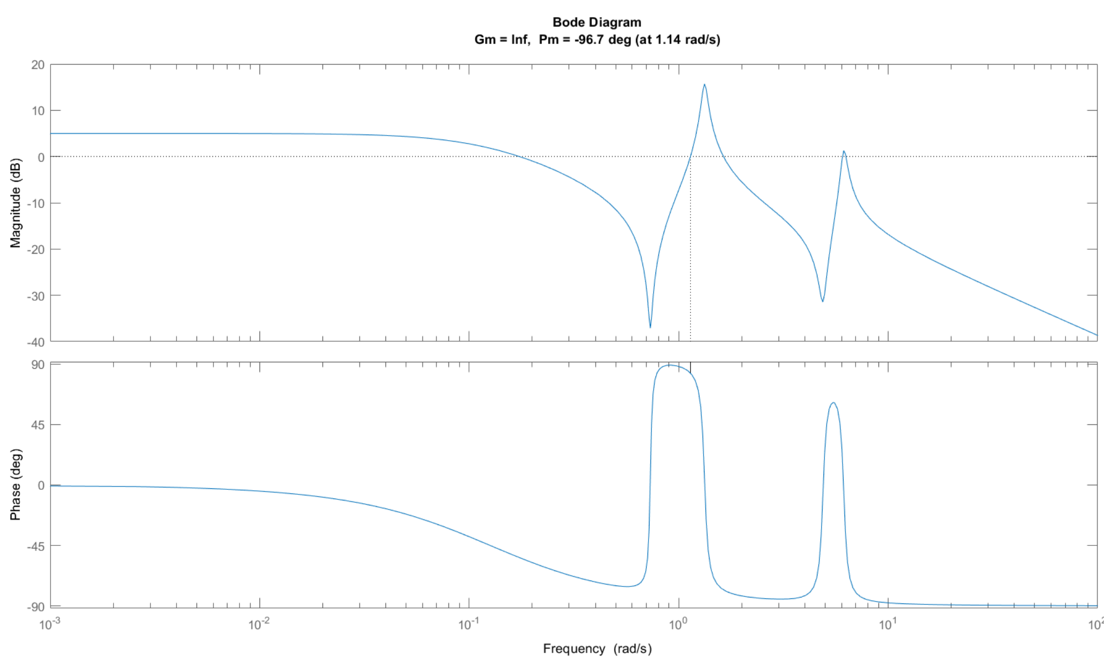


Figure 3.2 Open Loop Bode Plot with Proportional Gain

This confirmed the need for phase correction.

MATLAB CODE:

```
% As we need to shift the magnitude plot using some gain
% to have the gain crossover frequency at 1 Hz and not change
% the phase as the PM is already at least 40 deg, we check
% with just a proportional controller.

w_c = 2*pi*1; % crossover frequency at 1 Hz
mag_K_c = abs(squeeze(freqresp(H,w_c)));
K_c = 1/mag_K_c % gain required for target crossover frequency
K_c = 29.3132

% Check margins without the compensator
margin(K_c * H)

[Gm_H, Pm_H, Wcg_H, Wcp_H] = margin(K_c * H)
Gm_H = Inf
Pm_H = -96.6922
Wcg_H = NaN
Wcp_H = 1.1383

% GM is still Inf. Negative PM shows that the system is unstable.
% We need a compensator that stabilizes the system.
```

A lag compensator was selected in this design to meet the specific requirement of increasing the open-loop gain at around 1 Hz without significantly affecting the phase margin. The phase response near the desired crossover frequency (6.28 rad/s) was near –40°, suggesting that sufficient phase margin could be preserved. The lag compensator effectively boosts the low-frequency gain to bring the magnitude up to 0 dB near the target frequency, enabling a crossover at 1 Hz. Compared to a lead compensator, which adds phase lead and tends to push the crossover to higher frequencies, the lag compensator was more appropriate in this context without introducing unwanted phase shift.

A single lag compensator was introduced:

$$C_1(s) = \frac{s + 62}{s + 0.636}$$

As shown in Figures 3.3 and 3.4, this improved the PM to 23.5°, and the Bandwidth to 0.656 Hz, satisfying both the requirements. However, the gain margin remained undefined because the phase never crossed -180°. To make the GM measurable and ensure robustness to gain variation, a second lag compensator was introduced.

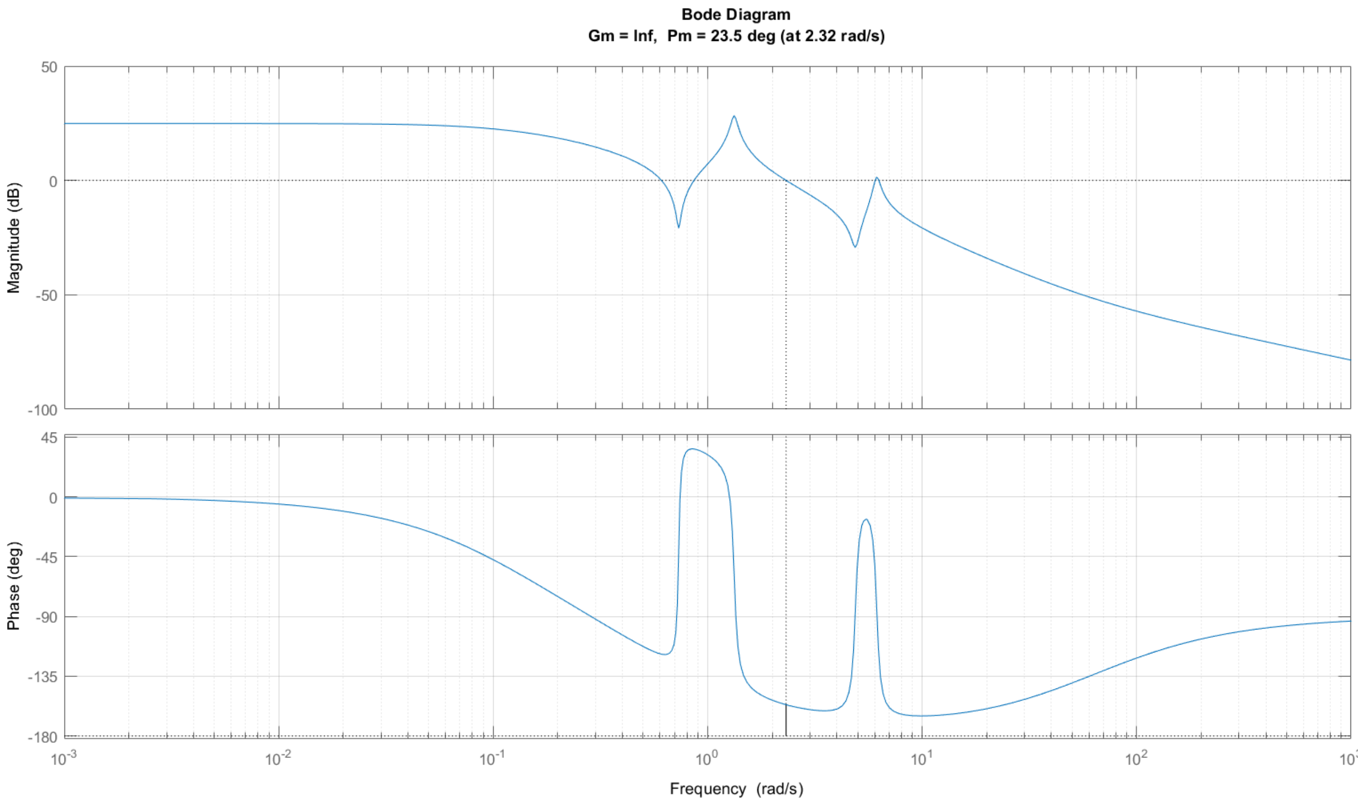


Figure 3.3 Open Loop Bode Plot with One Lag Compensator

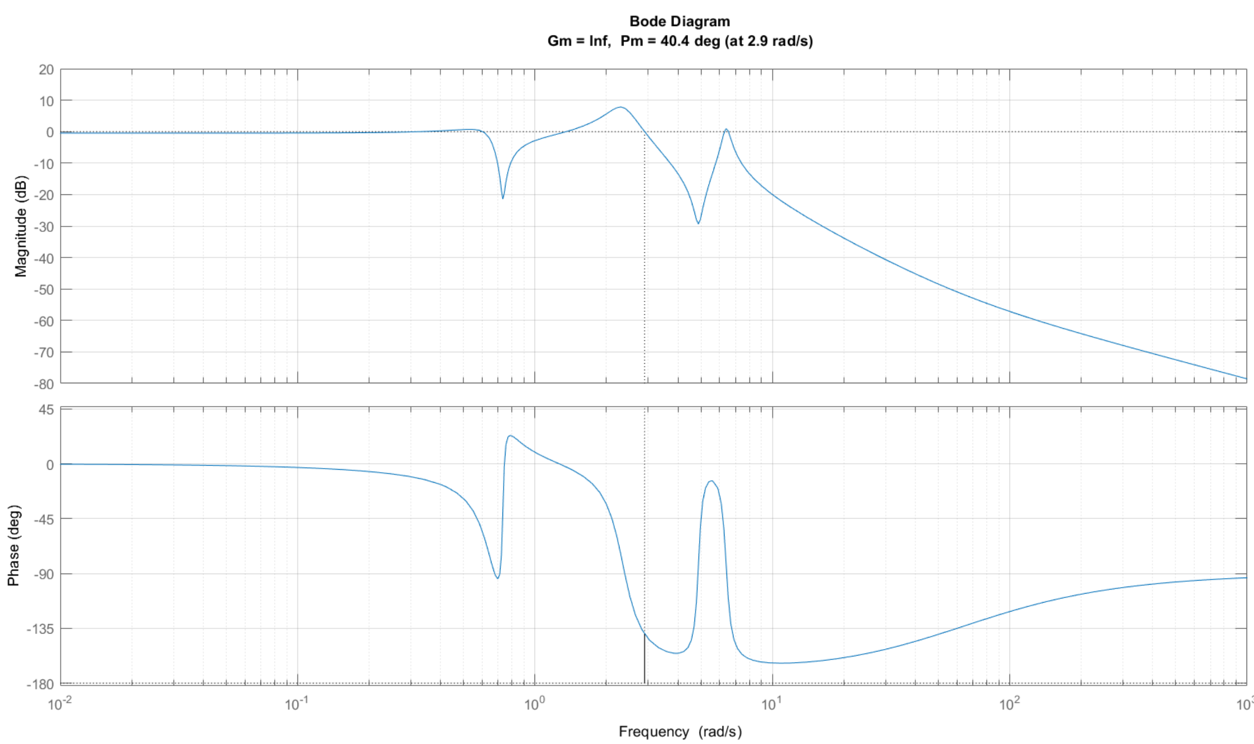


Figure 3.4 Closed Loop Bode Plot with One Lag Compensator

MATLAB CODE:

```
% As we need to shift the magnitude plot up to set the gain
% crossover frequency at 1 Hz, we use a positive gain.
% We do not need to alter the phase right now, although
% we can reduce the phase by some amount. Main purpose
% of designing a compensator is to give positive gain for
% the magnitude plot to cross 0 dB.
% Hence, we choose a LAG COMPENSATOR for the same, as
% a Lead Compensator will add phase, which is the opposite
% of what we want to achieve.

s = tf('s');
w_c = 2*pi*1; % crossover frequency at 1 Hz

% Compensator Design
% Lag compensator (z > p)
% Lag compensator reduces the phase in this frequency range:
% w_start = 0.1*p to w_end = 10*z.
% Also, max phase lag occurs at: w = sqrt(z*p).

C1_lag = (s+62)/(s+0.636);
C_H_lag = C1_lag * H;
% Gain
mag_K_lag = abs(squeeze(freqresp(C_H_lag, w_c)));
K_lag = 1/mag_K_lag
K_lag = 2.9706

% Open Loop transfer function
L_lag = K_lag * C_H_lag;

% LEAD COMPENSATOR:
%phi = 25; % phase change required
%alpha = (1-sind(phi))/(1+sind(phi));
%tau = 1/(w_c*sqrt(alpha));
%C_lead = tf([tau 1], [alpha*tau 1]); % compensator transfer function
%C = K_c * C_lead; % Compensator Design
%L = C * H; % Open Loop transfer function

% Analyzing the margins
margin(L_lag)
grid on;

[Gm_lag, Pm_lag, Wcg_lag, Wcp_lag] = margin(L_lag)
Gm_lag = Inf
Pm_lag = 23.5168
Wcg_lag = NaN
Wcp_lag = 2.3205
```

```
% Closed-loop system with compensator
T = feedback(L_lag, 1);
% Check the new margins and bandwidth with the compensator
margin(T)
grid on;
[bw] = bandwidth(T);
disp(['Closed-Loop Bandwidth: ', num2str(bw), 'Hz']);
Closed-Loop Bandwidth: 0.65684Hz

% We see from the above closed loop margin plot that
% PM for the closed loop system with the tuned lag
% compensator is 40.4 deg, perfect according to the
% requirements, and the Bandwidth is 0.656 Hz, which
% is closed enough to 1 Hz. But, the GM is still undefined.
```

The final compensator structure became:

$$C_1(s) = \frac{s+62}{s+0.636} \cdot \frac{s+20}{s+7.4}$$

The gains, *K_lag_lag and K_new*, were tuned to bring the gain crossover frequency to 1 Hz, and tune the GM, PM, and Closed-loop bandwidth to the desired values.

Results:

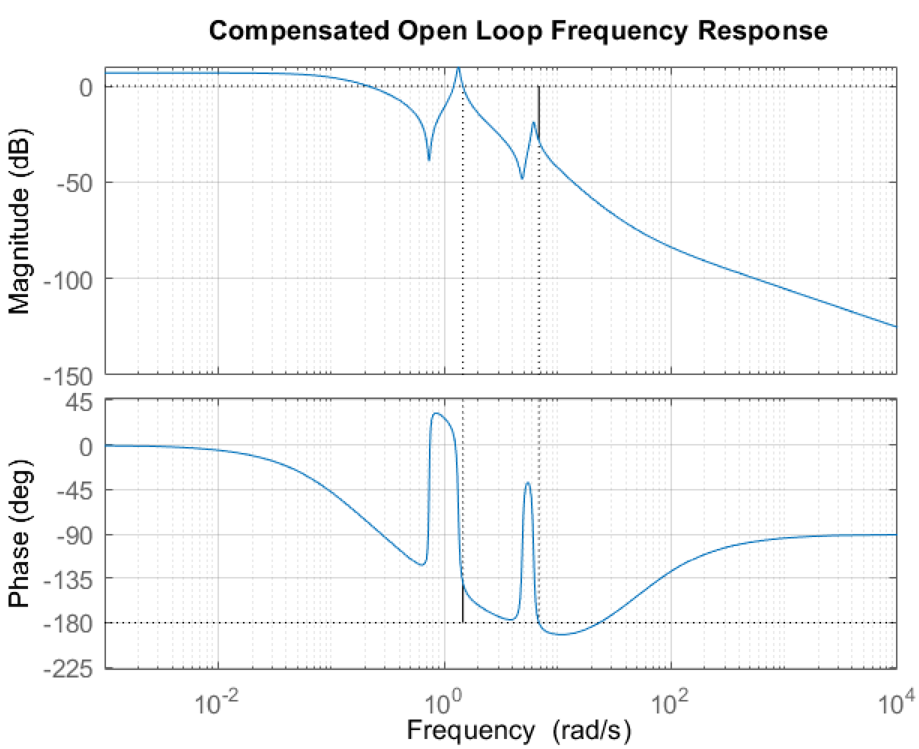


Figure 3.5 Open Loop Bode Plot after Compensation

From Figure 3.5, we see that:

- Crossover Frequency at ~ 6.28 rad/sec,
- Phase Margin = 40.3021°,
- Gain Margin = 26.6109 dB, as shown in the output of the MATLAB code below.

```
[Gm_total, Pm_total, Wcg_total, Wcp_total] = margin(L_total)
Gm_total = 26.6109
Pm_total = 40.3021
Wcg_total = 6.8473
Wcp_total = 1.4563
```

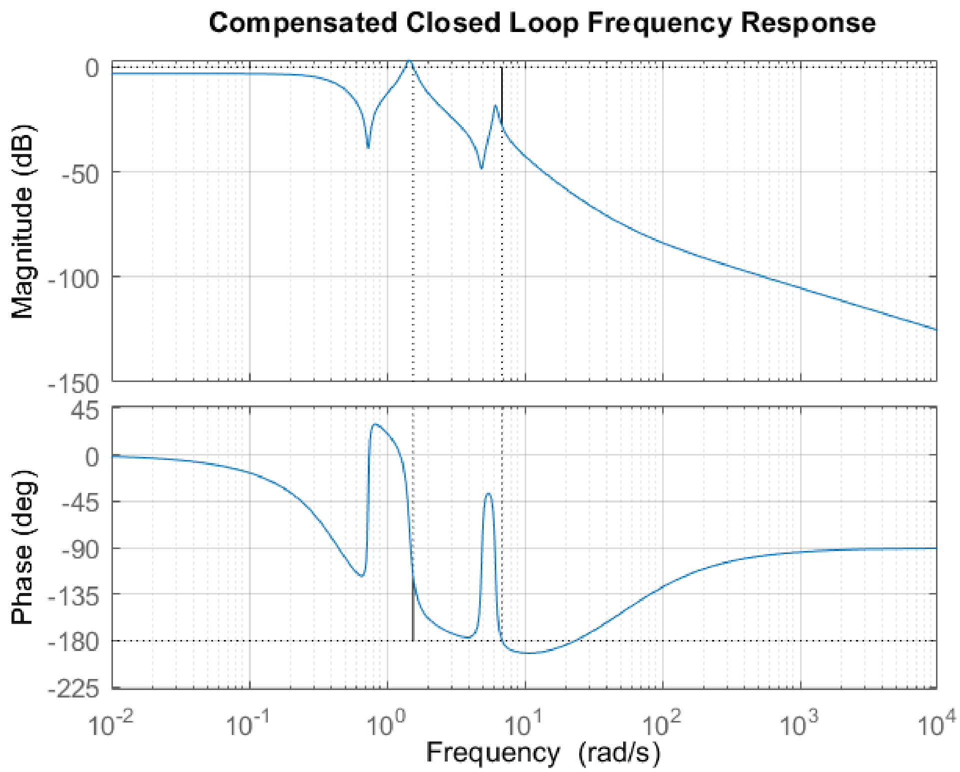


Figure 3.6 Closed Loop Bode Plot after Compensation

From Figure 3.6, we see that:

- Closed-loop (-3 dB) bandwidth = 0.37 Hz, as shown in the output of the MATLAB code below.
- Stable system indicating good phase/gain tradeoff.

```
[bw_total] = bandwidth(T_total);
disp(['Closed-Loop Bandwidth: ', num2str(bw_total), 'Hz']);
Closed-Loop Bandwidth: 0.3693Hz
```

For the designed compensation system with the tuned gains for the closed loop system, a closed loop bandwidth of 0.37 Hz could be achieved which is about ~ 1Hz. The controller successfully shifted the gain crossover to 1 Hz. The phase lag introduced by the plant was compensated by designing and tuning the two lag compensators. This validates the analytic plant model's control feasibility and prepares the foundations for comparisons with empirical plant data in later parts.

MATLAB CODE:

```
% We choose to add another lag compensator that makes
% the phase plot go down at higher frequencies and potentially
% cross the -180 deg line and then tune that pole to achieve
% the required GM. We can see that there are 3 regions in the
% magnitude plot where the magnitude is below -10 dB:
% 1. w --> 0.7 - 0.8 rad/sec
% 2. w --> 3.7 - 5.7 rad/sec
% 3. w --> 7.4 rad/sec and beyond.
% If we observe these three regions in the phase plot, we can see
% that the first two regions have high phase values and opting
% them for reducing the phase below -180 deg is not a wise choice.
% Hence, the new lag compensator will focus on frequencies:
% w --> 7.4 - 20 rad/sec. We are choosing 20 rad/sec as the upper
% limit as the phase starts to increase beyond that frequency.

% Another Lag Compensators
s = tf('s');
C2_lag = (s+20)/(s+7.4);
% Maximum reduction in phase occurs at 12.165 rad/sec (sqrt(20*7.4)).
% Minimum required reduction in phase:
% 180 - phase at 12.165 rad/sec = 180 - 162 = 18 deg
% Expected reduction in phase:
% asind((z-p)/(z+p)) = 27.32 deg
% Hence, we can hope that this works!

C_total = C2_lag * C_H_lag;
% Gain
mag_K_lag_lag = abs(squeeze(freqresp(C_total, w_c)));
K_lag_lag = 1/mag_K_lag_lag
K_lag_lag = 1.3756

K_new = 0.1;
L_total = K_new * K_lag_lag * C_total;

% Analyzing the margins
margin(L_total)
title('Compensated Open Loop Frequency Response')
grid on;

[Gm_total, Pm_total, Wcg_total, Wcp_total] = margin(L_total)
Gm_total = 26.6109
Pm_total = 40.3021
Wcg_total = 6.8473
Wcp_total = 1.4563
```

```
% Closed-loop system with compensator
T_total = feedback(L_total, 1);
% Check the new margins and bandwidth with the compensator
margin(T_total)
grid on;
title('Compensated Closed Loop Frequency Response')

[bw_total] = bandwidth(T_total);
disp(['Closed-Loop Bandwidth: ', num2str(bw_total), 'Hz']);
```

```
Closed-Loop Bandwidth: 0.3693Hz
```

Q4: Applying Compensator Design to Empirical Data

***Repeat part 3 using the empirical plant frequency response. Adjust the control design as necessary to try to maintain the stability and performance objectives. Comment on the effects of the unmodeled dynamics at high frequency*.**

This part repeats the compensator design process from the analytic plant, but uses empirical frequency response data instead of a mathematical transfer function. This goal is to maintain similar closed-loop specifications.

The bode plot for the measured frequency response data is shown in Figure 4.1. From Figure 4.1 we observed some non-ideal behavior at higher frequencies which suggest unmodelled high frequency dynamics.

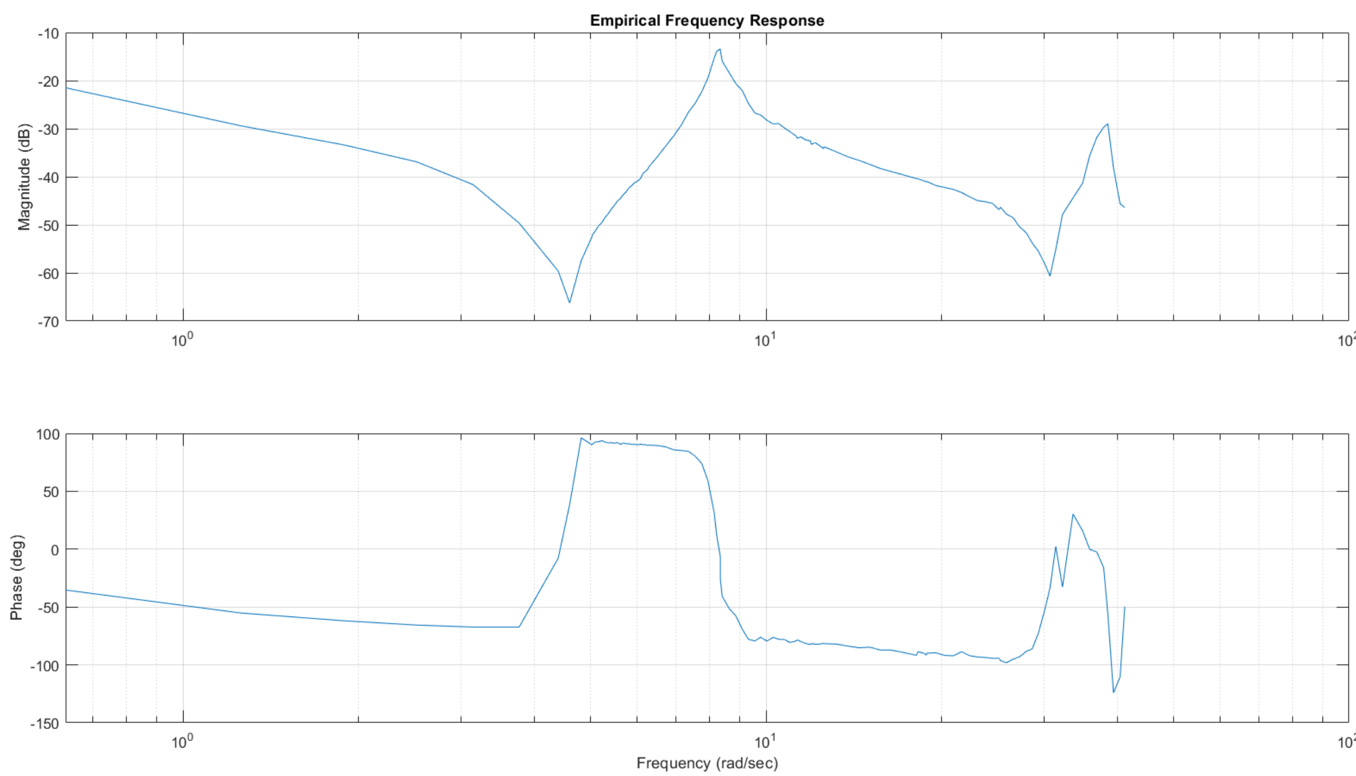


Figure 4.1 Empirical Frequency Response

MATLAB CODE:

```
% Designing the compensator for the empirical data

% Read Data from Excel
rawTable =
readtable('Spacecraft_spin_module_frequency_response_data.xlsx','Sheet','S
heet1');
```

```
% Freq in Hz
freq_data_Hz = rawTable.Frequency_Hz_;
% Freq in rad/sec
freq_data_rad_sec = 2*pi*freq_data_Hz;

% Mag in (rad/sec)/V
mag_data_linear = rawTable.Magnitude__rad_sec__V_;
% Mag in dB
mag_data_dB = 20*log10(mag_data_linear);

% Phase in rad
phase_data_rad = rawTable.Phase_rad_;
% Phase in deg
phase_data_deg = phase_data_rad*360/(2*pi);

% Converting the empirical data to complex frequency response
H_empirical = mag_data_linear .* exp(1j * phase_data_rad);

% Empirical Bode Plot
figure;
subplot(2,1,1)
semilogx(freq_data_rad_sec, mag_data_dB)
ylabel('Magnitude (dB)')
title('Empirical Frequency Response')
grid on

subplot(2,1,2)
semilogx(freq_data_rad_sec, phase_data_deg)
xlabel('Frequency (rad/sec)')
ylabel('Phase (deg)')
grid on
```

Rather than designing from scratch, the compensator used in Question 3 was used as a starting point. However, since the empirical plant showed sufficient gain characteristics in the low-frequency range, the number of lag compensators was reduced to one, simplifying the design while still achieving the desired margins.

Final compensator used:

$$C_1(s) = \frac{s+70}{s+1}$$

Figures 4.2 and 4.3 depict the open loop and closed loop frequency response with the compensator and the measured frequency response data.

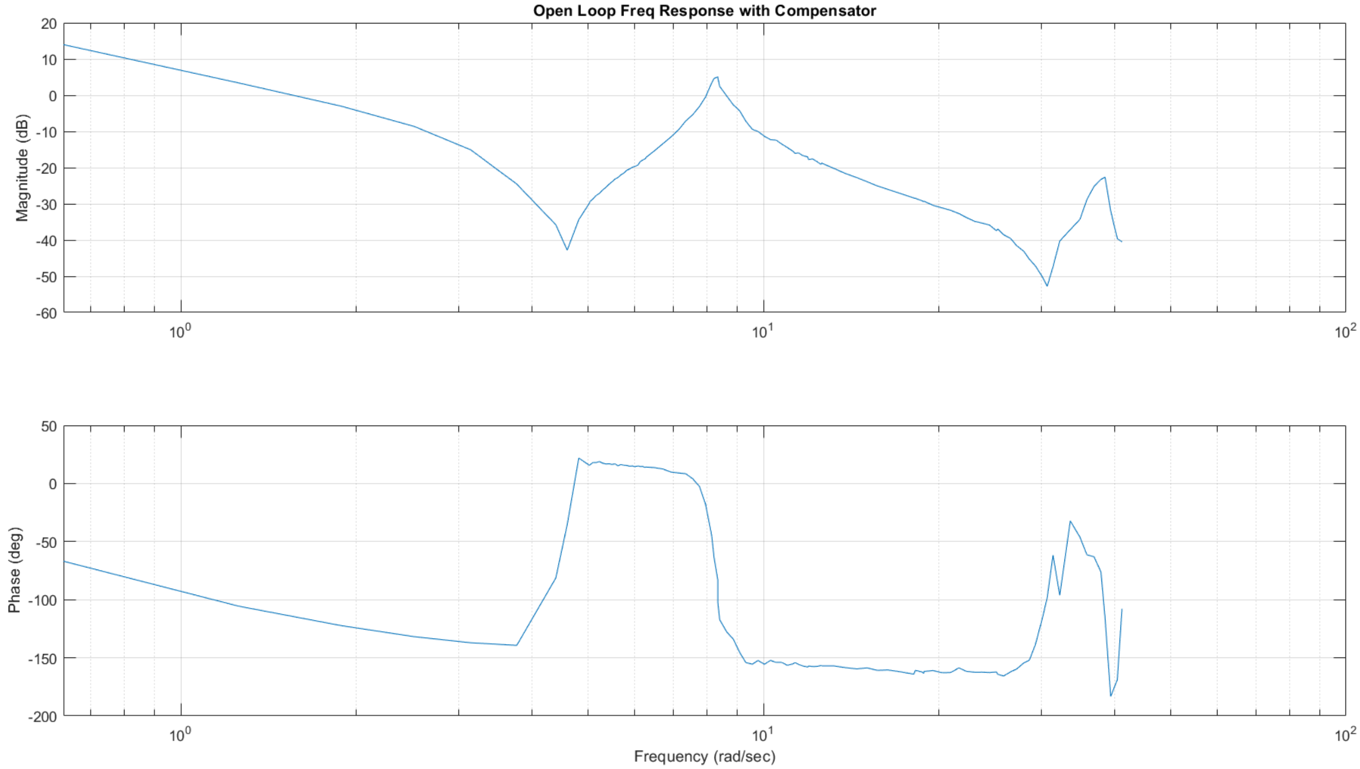


Figure 4.2 Open Loop Frequency Response after Compensation

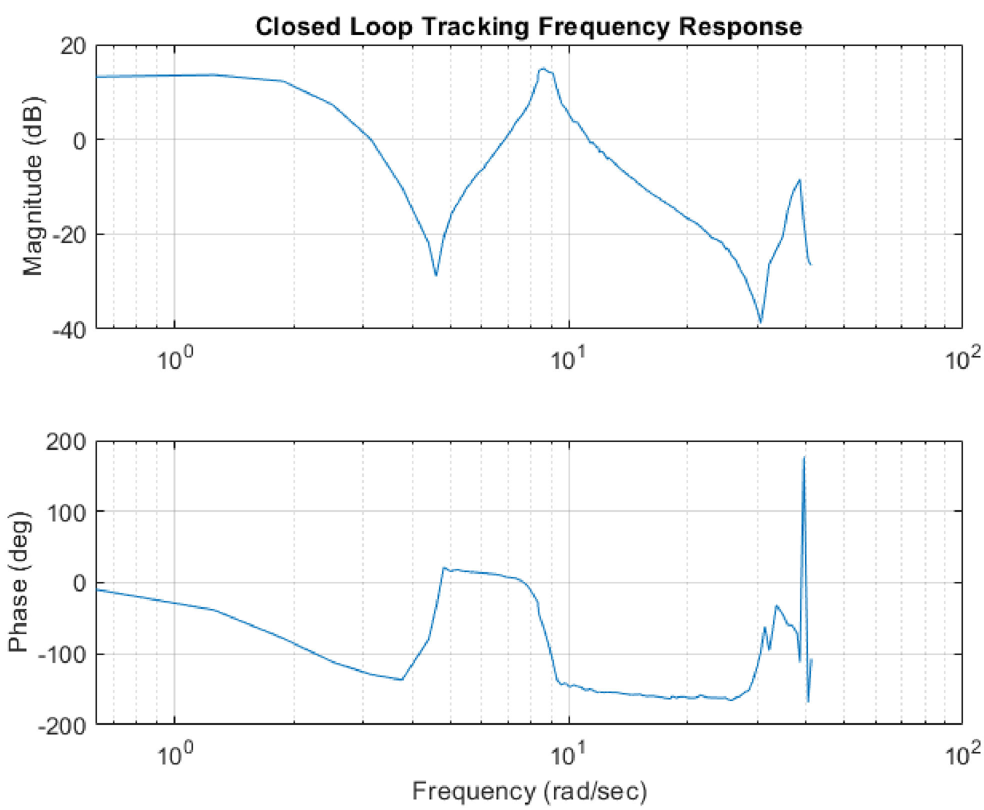


Figure 4.3 Closed Loop Frequency Response after Compensation

We observe from the above plots that there are distortions in the high frequency range possibly due to unmodelled dynamics. Also, the PM and GM as seen from the output of MATLAB for the open loop frequency response are:

- PM = 52.27°, and
- GM = 31.98 dB

The closed loop bandwidth as seen from the output of MATLAB is:

- Closed loop bandwidth = 0.60 Hz.

These results meet the original required target values for the margins and closed-loop bandwidth. The system behaves slightly more conservatively than the analytically modelled one, which is acceptable.

Effects of Unmodelled High Frequency Dynamics

The empirical frequency response shows significant irregularities beyond 20 rad/sec which suggests unmodelled behavior such as structural vibrations (also seen in the class while playing with the satellite), high-frequency sensor or actuator noise, or even non-linearities. While the designed compensator remains robust against these effects, they highlight the importance of bandwidth control, and the final model is designed to keep the bandwidth below the frequency at which the unmodelled dynamics become dominant.

So, the lag compensator first designed for the previous question was adopted and tuned for the empirical data keeping in mind the desired margin and bandwidth values. The closed loop system now meets all the stability and performance objectives and is robust against physical deviations.

MATLAB CODE:

```
% Designing the compensator for empirical data
% We will choose the same compensator as designed for
% the analytic transfer function as the initial compensator
% for the empirical data and then tune as required.

% Lag Compensator
s = tf('s');
w_c = 2*pi*1; % crossover frequency at 1 Hz

% Compensator Design
% Lag compensator (z > p)
% Lag compensator reduces the phase in this frequency range:
% w_start = 0.1*p to w_end = 10*z.
% Also, max phase lag occurs at: w = sqrt(z*p).

C1_lag_empirical = (s+70)/(s+1);
[mag_lag_empirical, phase_lag_empirical, ~] = bode(C1_lag_empirical,
freq_data_rad_sec);
mag_lag_empirical = squeeze(mag_lag_empirical);
phase_lag_empirical = squeeze(phase_lag_empirical);
```

```
% Open Loop Frequency Response with Compensator
mag_L_empirical = mag_lag_empirical .* mag_data_linear;
mag_L_empirical_dB = 20*log10(mag_L_empirical);
phase_L_empirical_deg = phase_lag_empirical + phase_data_deg;
L_empirical = mag_L_empirical .* exp(1j * deg2rad(phase_L_empirical_deg));

% Plot the bode plot for the open loop frequency response
subplot(2,1,1)
semilogx(freq_data_rad_sec, mag_L_empirical_dB);
%hold on;
%yline(0, 'r--', '0 dB line');
ylabel('Magnitude (dB)')
title('Open Loop Freq Response with Compensator')
grid on

subplot(2,1,2)
semilogx(freq_data_rad_sec, phase_L_empirical_deg);
%hold on;
%yline(-180, 'r--', '-180 deg line')
xlabel('Frequency (rad/sec)')
ylabel('Phase (deg)')
grid on

% Unity Feedback Closed Loop system
T_empirical = L_empirical ./ (1 + L_empirical);
K_new_empirical = 5;
T_empirical_total = K_new_empirical .* T_empirical;
mag_T_empirical_total = 20*log10(abs(T_empirical_total));
phase_T_empirical_total = rad2deg(angle(T_empirical_total));

% Plot the closed loop empirical frequency response
subplot(2,1,1)
semilogx(freq_data_rad_sec, 20*log10(abs(T_empirical_total)))
ylabel('Magnitude (dB)')
title('Closed Loop Tracking Frequency Response')
grid on
subplot(2,1,2)
semilogx(freq_data_rad_sec, rad2deg(angle(T_empirical_total)))
xlabel('Frequency (rad/sec)')
ylabel('Phase (deg)')
grid on

% Find the Gain Margin and Phase Margin for the closed loop response
% from the above plots

% Find index of point closest to 0 dB
```

```
[~, idx_gc] = min(abs(mag_L_empirical_dB));  % smallest difference from 0
dB
% Estimate phase margin
phase_at_gc = phase_L_empirical_deg(idx_gc);
PM = phase_at_gc + 180;
w_gc = freq_data_rad_sec(idx_gc);
fprintf('Approximate Gain Crossover at %.2f rad/s → PM ≈ %.2f deg\n',
w_gc, PM);
```

```
Approximate Gain Crossover at 8.64 rad/s → PM ≈ 52.27 deg
```

```
% Find index of point closest to -180 deg
[~, idx_pc] = min(abs(phase_L_empirical_deg + 180));  % phase closest to
-180 deg
% Estimate gain margin
mag_at_pc = mag_L_empirical_dB(idx_pc);
GM = -mag_at_pc;
w_pc = freq_data_rad_sec(idx_pc);
fprintf('Approx Phase Crossover at %.2f rad/s → GM ≈ %.2f dB\n', w_pc,
GM);
```

```
Approx Phase Crossover at 39.43 rad/s → GM ≈ 31.98 dB
```

```
% Find the bandwidth of the open loop
idx_bw = find(mag_T_empirical_total < -3, 1);
bw_closed_loop = freq_data_rad_sec(idx_bw);
fprintf('Closed-Loop Bandwidth ≈ %.2f rad/s (%.2f Hz)\n', bw_closed_loop,
bw_closed_loop/(2*pi));
```

```
Closed-Loop Bandwidth ≈ 3.77 rad/s (0.60 Hz)
```

```
% We can see from the output that the PM for the closed-loop system
% is 52.27 deg (> 40 deg), the GM is 31.98 dB (> 10 dB) and the
% Bandwidth is 0.60 Hz (~ 1 Hz). All these values are close to what
% was achieved using the analytic transfer function, although we could
% achieve the same performance with one less lag compensator.
```

Q5: Simulating the obtained System on Simulink

Simulate the analytic model in a unity feedback control loop in Simulink using the control design from part 4. Plot the responses to a 0.5 rad step input, and to single-sinusoid inputs at 0.5 Hz and 2.0 Hz, each with amplitude 0.5 rad. Also plot the torque input signal, along with its limiting values. Compare these responses with expectations from loop gain analysis for closed loop frequency response, DC tracking accuracy, and stability margins.

The control system uses the analytic plant transfer function, H(s), and the compensator structure from Question 3 which includes two lag compensators and gains tuned according to the desired stability and performance objectives. The simulink setup incorporates this loop configuration and probes the output angle and input torque signals.

Step Input Simulation and Results

Simulink Model:

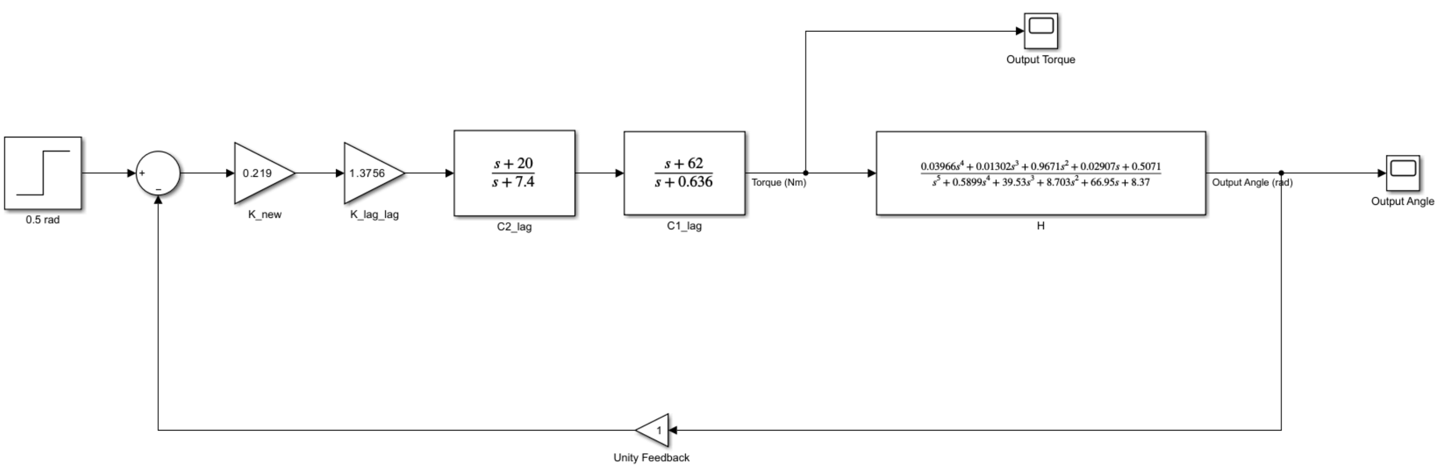


Figure 5.1 Simulink Model for 0.5 rad step input

Figure 5.1 shows the snapshot of the Simulink model where we designed the unity feedback control system for the analytic plant transfer function along with the compensators designed in Question 3. The output angle is taken as the output of the system, while the torque signal is taken right after the compensators as that is the input entering in the plant transfer function.

Figures 5.2 and 5.3 show the output angle and torque signals as simulated in Simulink. We see from Figure 5.2 that the output angle does not track the input angle very closely but shows a smooth oscillatory convergence towards 0.5 rad with some steady state error. The system overshoots slightly consistent with the frequency response margins. Figure 5.3 shows the torque input for the plant transfer function, where we can see that the torque rises sharply to 20 Nm initially as there is sudden change in the input angle and the satellite wishes to get to the desired angle fast. The torque then settles based on the instantaneous angular displacement of the satellite.

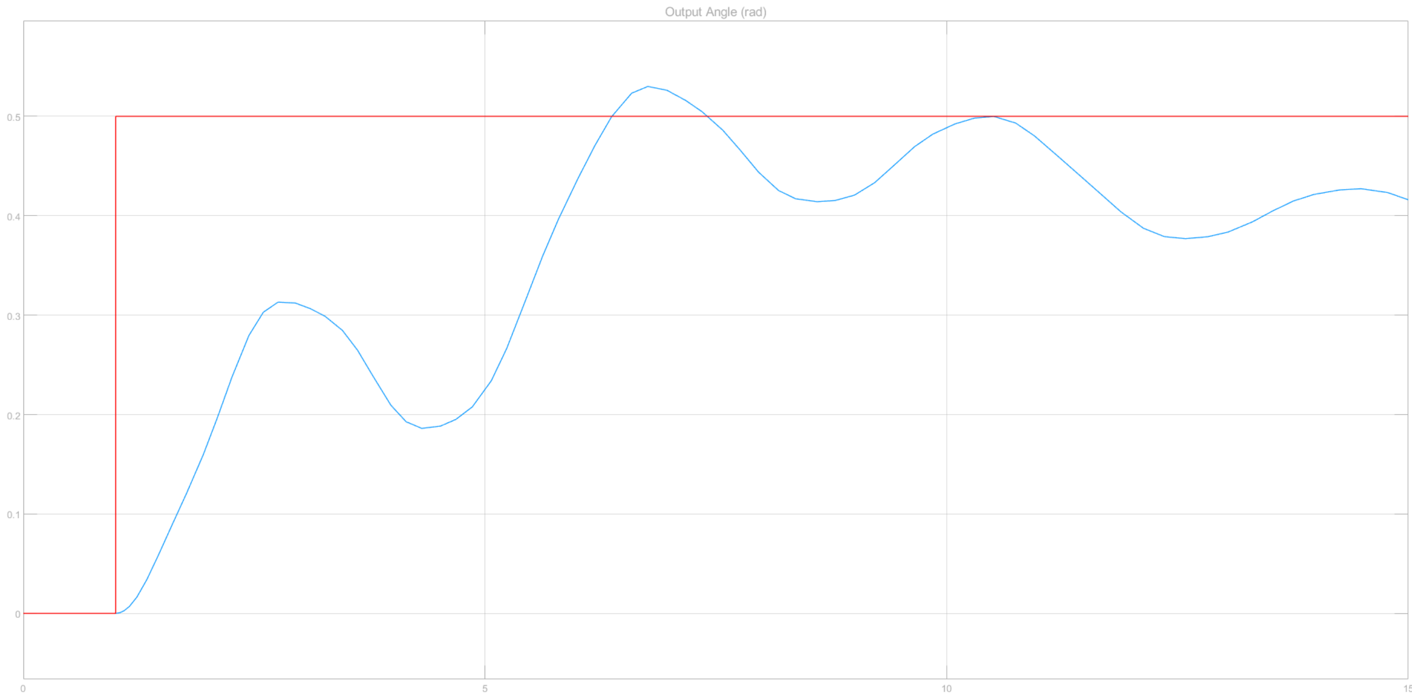


Figure 5.2 Output Angle against Input Angle

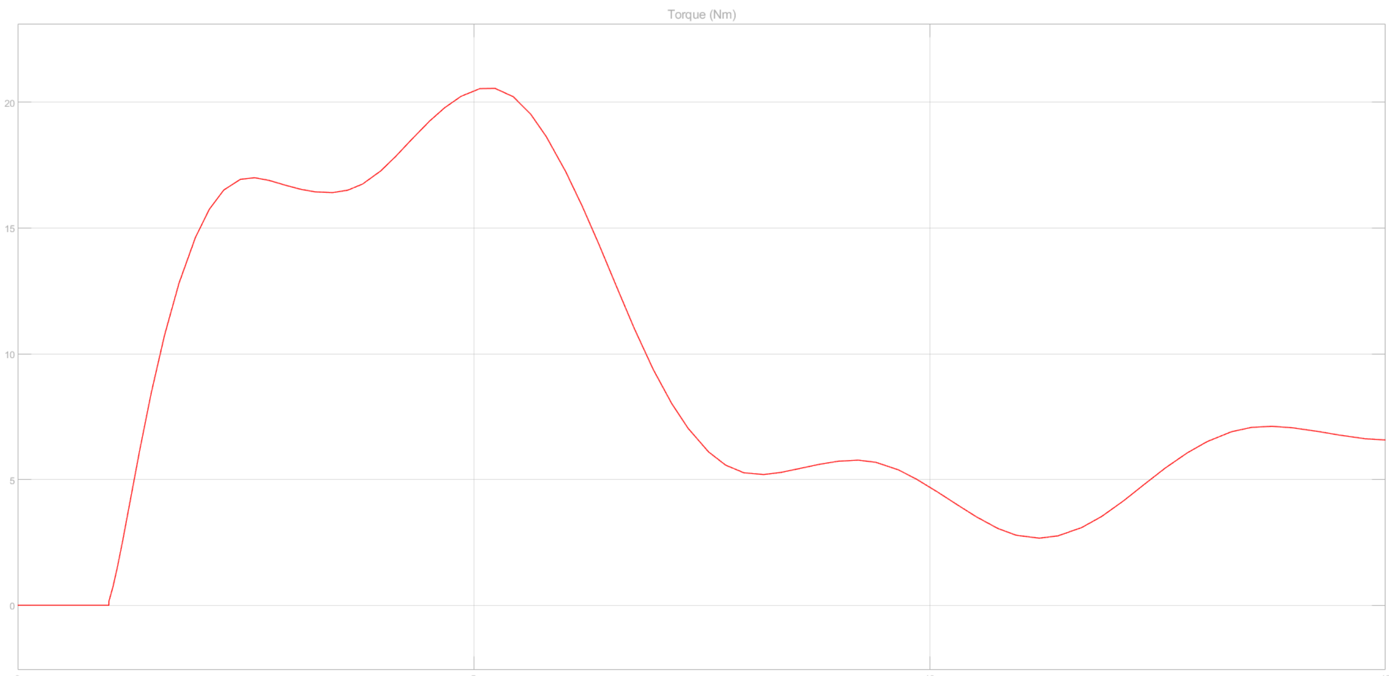


Figure 5.3 Torque input signal to Plant Transfer Function

The system tracks the input with good DC accuracy and moderate settling time, indicating that the closed-loop bandwidth is appropriate for this level of performance.

Sinusoidal Input Simulations and Results

Sine Input with 0.5Hz Frequency:

Simulink Model:

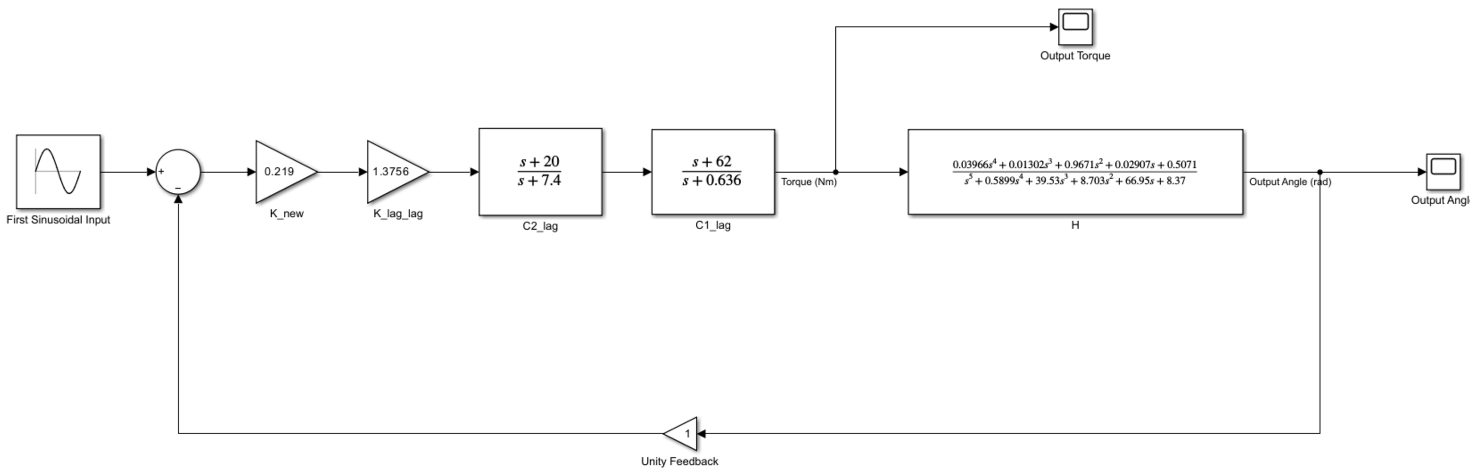


Figure 5.4 Simulink Model for 0.5 Hz Sine Input

Figure 5.4 shows the snapshot of the Simulink model with 0.5 rad amplitude and 0.5 Hz frequency sinusoidal input signal to the control system.

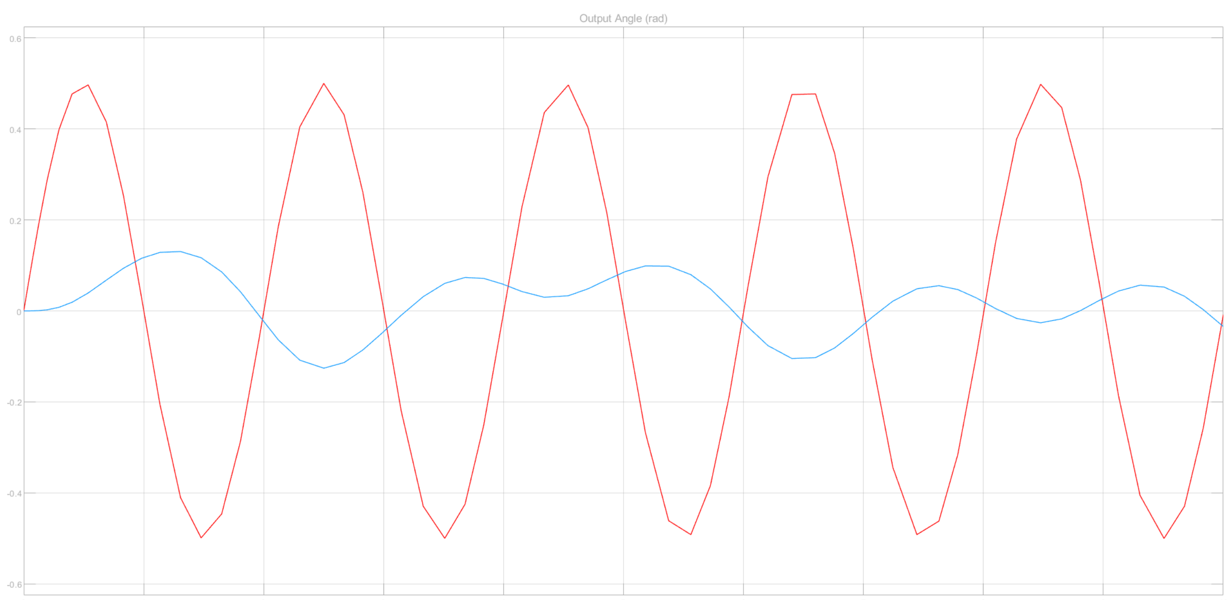


Figure 5.5 Output Angle against Input Angle for Sine Input with 0.5 Hz Frequency

Figures 5.5 and 5.6 show the output angle and torque signals as simulated in Simulink. We see from Figure 5.5 that the output angle approximately tracks the input signal but with noticeable difference in the amplitude. We reason that this change in the amplitude is because of the signal being sinusoidal which changes the input angle from positive to negative values at a frequency where the satellite is not able to reach the amplitude of the input angle before the input signal itself changes its direction. We can see this more evidently in the second sine input signal which has higher frequency. Figure 5.6 shows the torque generated in the satellite as required by the satellite to track the input signal. We can see that the torque oscillates in sync with the input signal and peaks at roughly 10 Nm.

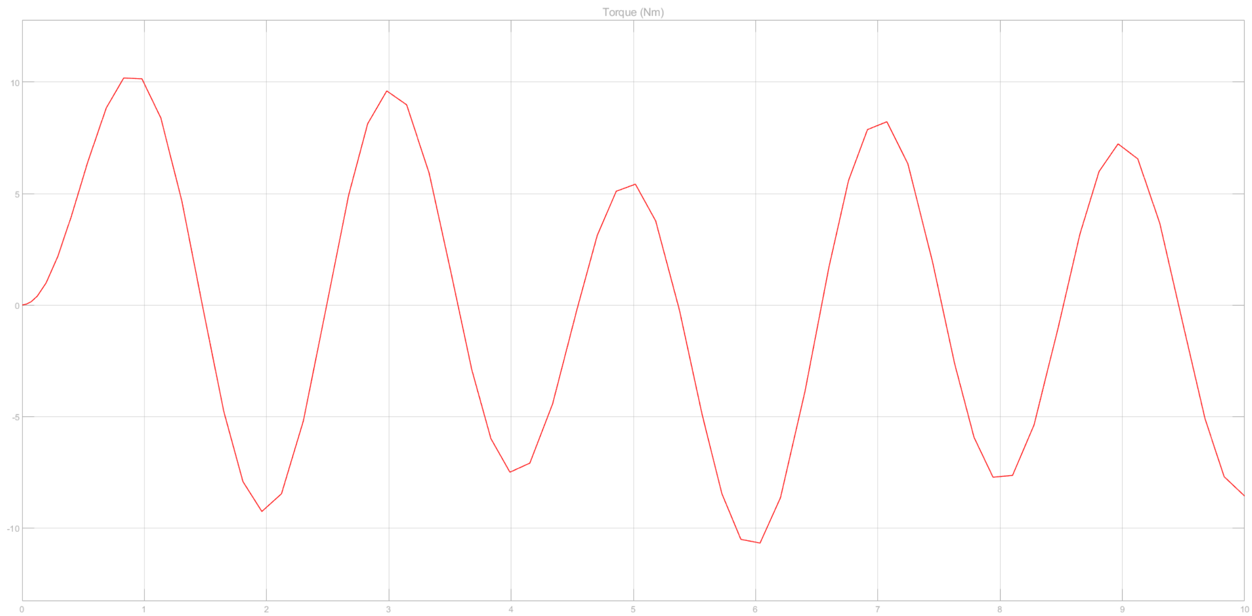


Figure 5.6 Torque Input signal to Plant Transfer Function at 0.5 Hz Frequency

Sine Input with 2.0Hz Frequency:
Simulink Model:

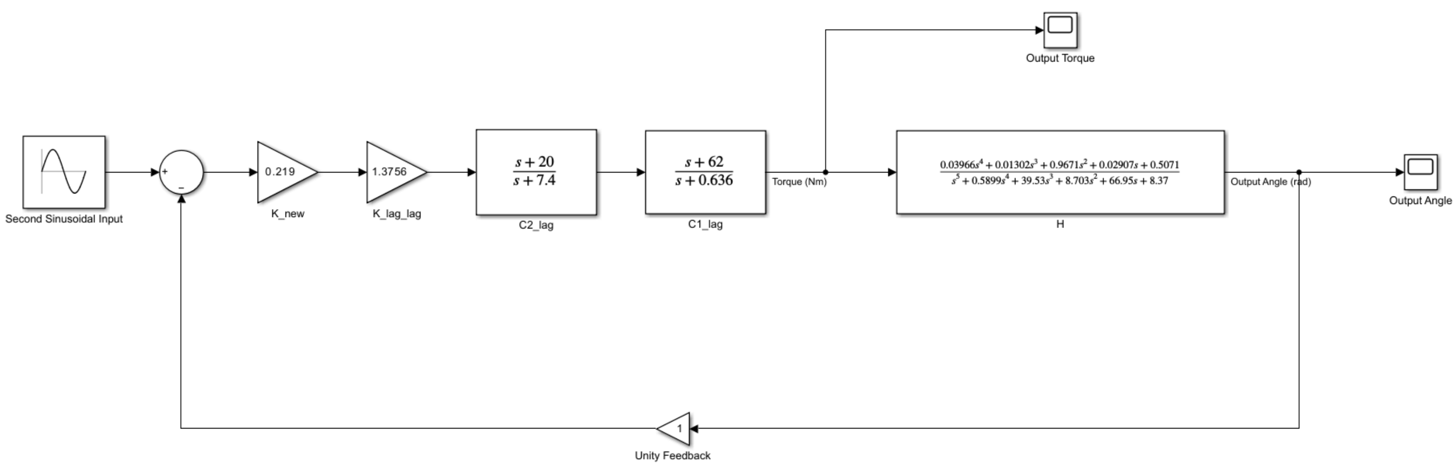


Figure 5.7 Simulink Model for 2.0 Hz Sine Input

Figure 5.7 shows the snapshot of the Simulink model with 0.5 rad amplitude and 2.0 Hz frequency sinusoidal input signal to the control system.

Figures 5.8 and 5.9 show the output angle and torque signals as simulated in Simulink. We see from Figure 5.8 that the output angle does not track the input signal. We reason that this change in the amplitude is because of the signal being sinusoidal which changes the input angle from positive to negative values at a frequency where the satellite is not able to reach the amplitude of the input angle before the input signal itself changes its direction. This is more evident in the second sine input as the frequency is 4 times of that in the first sine input. Figure 5.9 shows the torque generated in the satellite as required by the satellite to track the input signal. We can see that the torque oscillates in sync with the input signal and peaks at roughly 2.5 Nm.

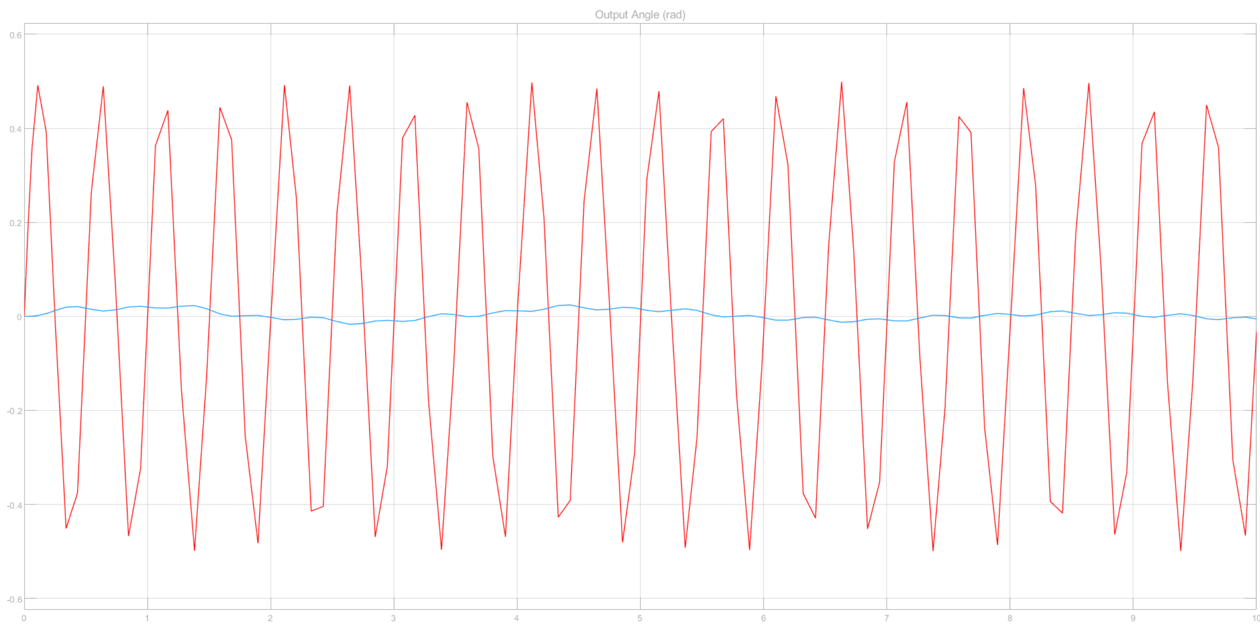


Figure 5.8 Output Angle against Input Angle for Sine Input with 2.0 Hz Frequency

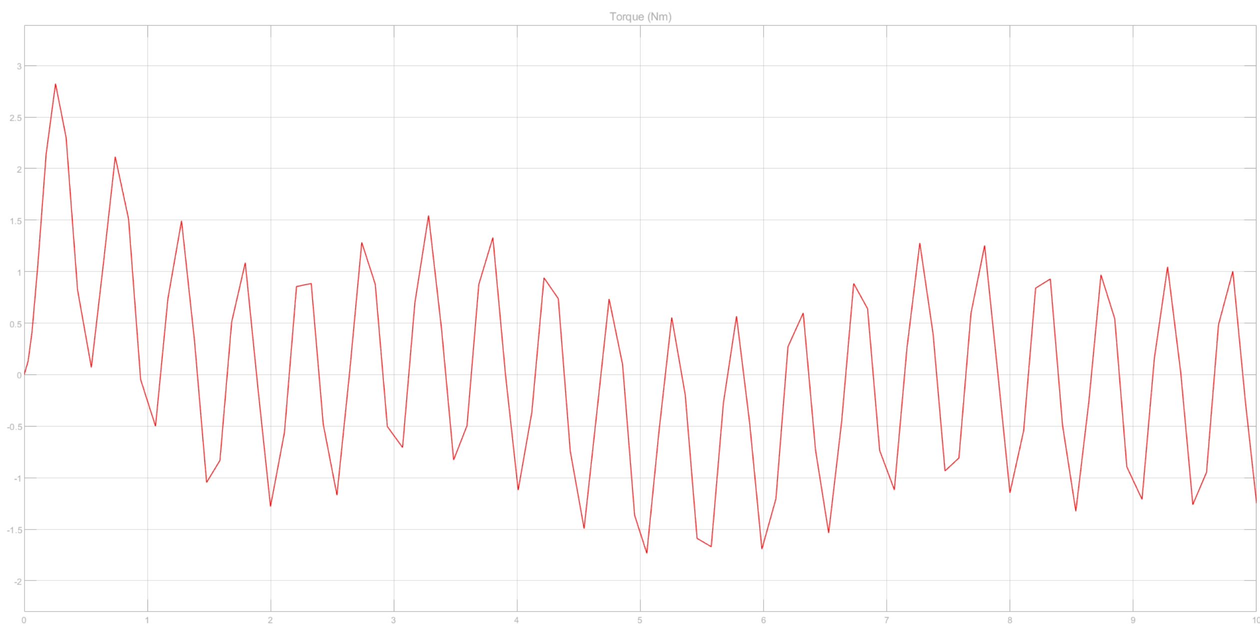


Figure 5.9 Torque Input signal to Plant Transfer Function at 2.0 Hz Frequency

MATLAB CODE:

```
% Simulating the analytic model in a unity feedback control loop
% in Simulink using the designed compensator for the analytic transfer
% function.

% Inputs:
% 1. 0.5 rad step input
% 2. Sine input at 0.5 rad, 0.5 Hz
% 3. Sine input at 0.5 rad, 2.0 Hz

% Call the analytic transfer function
H
H =

  0.03966 s^4 + 0.01302 s^3 + 0.9671 s^2 + 0.02907 s + 0.5071
  -----------------------------------------------------------
   s^5 + 0.5899 s^4 + 39.53 s^3 + 8.703 s^2 + 66.95 s + 8.37

Continuous-time transfer function.
Model Properties

% Designed compensator for the analytic transfer function
C_total
C_total =

    0.03966 s^6 + 3.265 s^5 + 51.21 s^4 + 95.48 s^3 + 1202 s^2 + 77.63 s + 628.8
  --------------------------------------------------------------------------------
  s^7 + 8.626 s^6 + 48.97 s^5 + 329.1 s^4 + 322.9 s^3 + 587.4 s^2 + 382.4 s + 39.39

Continuous-time transfer function.
Model Properties

% Open Loop Transfer function with the compensator and gains
L_total
L_total =

   0.005455 s^6 + 0.4491 s^5 + 7.044 s^4 + 13.13 s^3 + 165.4 s^2 + 10.68 s + 86.49
  --------------------------------------------------------------------------------
  s^7 + 8.626 s^6 + 48.97 s^5 + 329.1 s^4 + 322.9 s^3 + 587.4 s^2 + 382.4 s + 39.39

Continuous-time transfer function.
Model Properties

% Unity Feedback Closed Loop Transfer Function with the
% analytic transfer function
T_total
T_total =

  0.005455 s^6 + 0.4491 s^5 + 7.044 s^4 + 13.13 s^3 + 165.4 s^2 + 10.68 s + 86.49
  ------------------------------------------------------------------------------
  s^7 + 8.631 s^6 + 49.42 s^5 + 336.2 s^4 + 336.1 s^3 + 752.7 s^2 + 393 s + 125.9

Continuous-time transfer function.
Model Properties
```

```
% Step Input Simulation
open_system('stepinput_closedloop_simulation.slx')

% First Sinusoidal Input (0.5 rad, 3.14 rad/sec)
open_system('sineinput1_closedloop_simulation')

% Second Sinusoidal Input (0.5 rad, 12.56 rad/sec)
open_system('sineinput2_closedloop_simulation')
```

Q6: Spacecraft Model with First Resonance Occurring Before Anti-Resonance

Suppose the spacecraft had a model where the first resonance occurred before the anti-resonance in frequency. Sketch this Bode plot, and determine if your controller designed in Part 4 would produce a stable system. Comment on the difficulty of achieving the above closed loop stability and tracking bandwidth performance with this "non-collocated" plant.

In this task, the objective was to swap the locations of the resonance and anti-resonance frequencies and to determine the stability condition while using the same compensator as in the previous task. This was accomplished effortlessly by simply reversing the natural frequency values from Task 1. The resulting Bode plot is shown below:

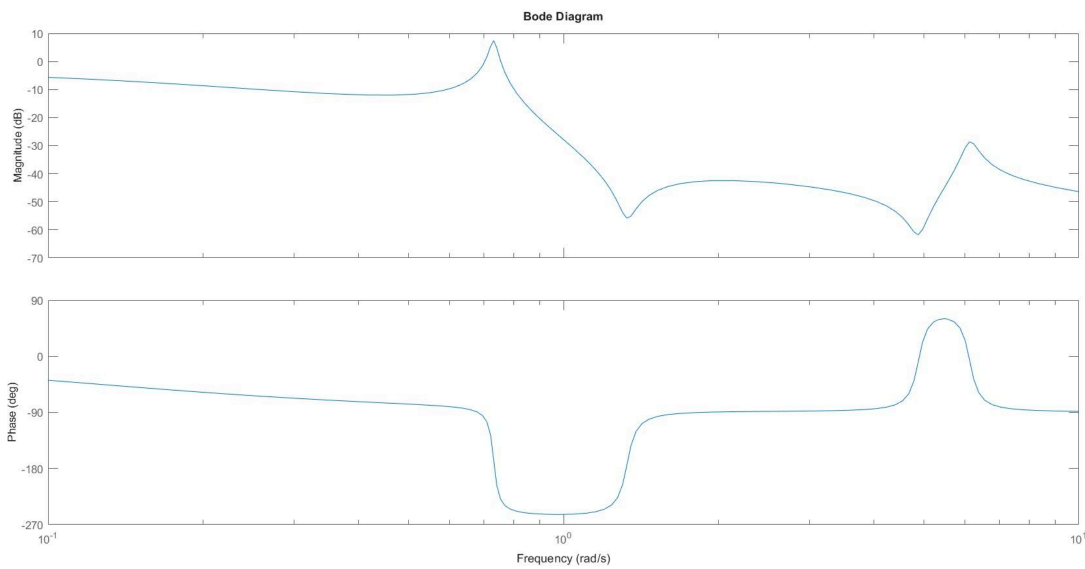


Figure 6.1 Revised Analytic Transfer Function Bode Plot

If this system is connected with the compensator designed in task 3, the following margins are obtained:

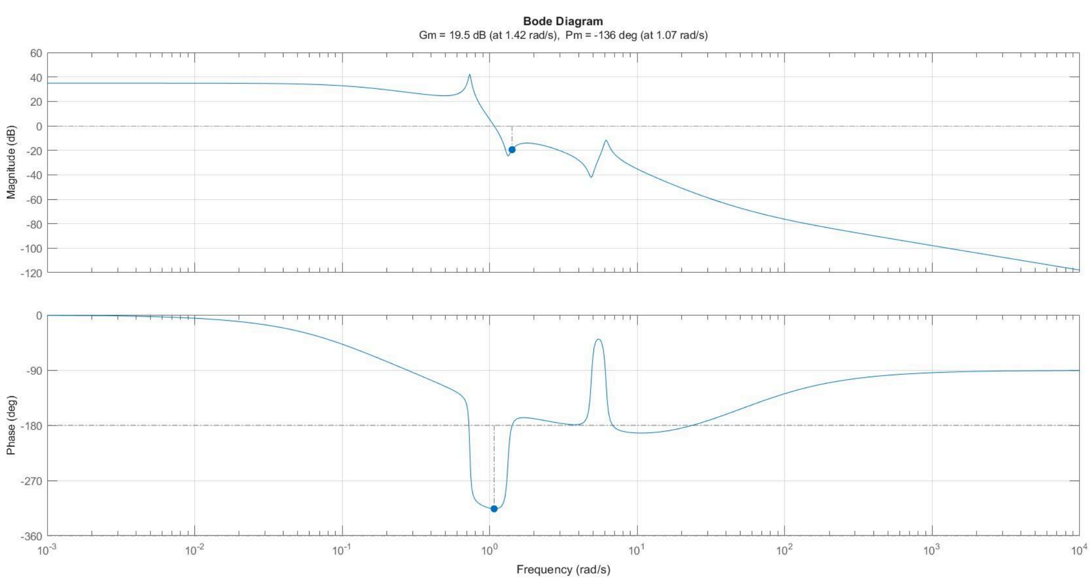


Figure 6.2 Gain & Phase Margins of the Analytic Function

As observed, the system exhibits a Gain Margin of 18.5 dB and a phase margin of -136 degrees. Additionally, the closed-loop bandwidth is 0.99954. While the Gain Margin and Closed-Loop Bandwidth appear to be reasonably good, these values are not meaningful in this context due to the significantly negative phase margin. This negative phase margin indicates that the system is far from stable.

The main challenge in assessing the stability of this system arises from the dip in phase values at lower frequencies, which leads to an early phase lag. Since a lag-lag compensator is used in the system, this further worsens the stability issues, causing a deterioration in overall system stability.

MATLAB Code:

```
% Analytic Transfer function from Part 1
wn2 = 0.7323;
mag2 = 29.7268;
zeta2 = (10^(-mag2/20))/2;
wn1 = 1.32844;
mag1 = 24.2747;
zeta1 = (10^(-mag1/20))/2;
wn3 = 4.88315;
mag3 = 24.1018;
zeta3 = (10^(-mag3/20))/2;
wn4 = 6.13774;
mag4 = 24.1018;
zeta4 = (10^(-mag4/20))/2;

b1 = 2*zeta1*wn1;
c1 = wn1^2;
d1 = 2*zeta3*wn3;
e1 = wn3^2;
a = 10^-0.9;
b = 2*zeta2*wn2;
c = wn2^2;
d = 2*zeta4*wn4;
e = wn4^2;

Kx = 10^(-28.034/20);

H = tf(Kx*[1     b1+d1    e1+(b1*d1)+c1    (b1*e1)+(c1*d1)  c1*e1]   ,
[1      (d+b+a)      (e+((b+a)*d)+c+ (a*b))     (((b+a)*e)+((c+(a*b))*d)+(a*c))     (((c+(a*b))*e)+(a*c*d))     a*c*e])
```

```
figure;
bode(H);
xlim([0.1 10]);

 w_c = 2*pi*1;
% crossover frequency at 1 Hz
mag_K_c = abs(squeeze(freqresp(H,w_c)));

K_c = 1/mag_K_c
% gain required for target crossover frequency

s = tf('s');

w_c = 2*pi*1;
% crossover frequency at 1 Hz

% Compensator Design
 Lag compensator (z > p)

C1_lag = (s+62)/(s+0.636);
C_H_lag = C1_lag * H;
mag_K_lag = abs(squeeze(freqresp(C_H_lag, w_c)));
K_lag = 1/mag_K_lag
L_lag = K_lag * C_H_lag;
C2_lag = (s+20)/(s+7.4);
C_total = C2_lag * C_H_lag;

% Gain

mag_K_lag_lag = abs(squeeze(freqresp(C_total, w_c)));
K_lag_lag = 1/mag_K_lag_lag
K_new = 0.219;
L_total = K_new*K_lag_lag * C_total;

% Closed-loop system with compensator
T_total = feedback(L_total, 1);

figure;
margin(L_total)
[Gm_total, Pm_total, Wcg_total, Wcp_total] = margin(L_total)
grid on;


[bw_total] = bandwidth(T_total);
disp(['Closed-Loop Bandwidth: ', num2str(bw_total), 'Hz']);
```

```
>>
Gm_total =  9.3918

Pm_total = -136.1776

Wcg_total = 1.4171

Wcp_total = 1.0708

Closed-Loop Bandwidth: 0.99954Hz
```

APPENDIX

Question 1:

```
% Read Data from Excel
rawTable = readtable('Spacecraft_spin_module_frequency_response_data.xlsx','Sheet','Sheet1');
freq_data = rawTable.Frequency_Hz_;
mag_data = rawTable.Magnitude__rad_sec__V_;
mag_data = 20*log10(mag_data);
phase_data = rawTable.Phase_rad_;
phase_data = phase_data*360/(2*pi);

% Curve Fitting Transfer function
wn1 = 0.7323;
mag1 = 29.7268;
zeta1 = (10^(-mag1/20))/2;
wn2 = 1.32844;
mag2 = 24.2747;
zeta2 = (10^(-mag2/20))/2;
wn3 = 4.88315;
mag3 = 24.1018;
zeta3 = (10^(-mag3/20))/2;
wn4 = 6.13774;
mag4 = 24.1018;
zeta4 = (10^(-mag4/20))/2;

b1 = 2*zeta1*wn1;
c1 = wn1^2;
d1 = 2*zeta3*wn3;
e1 = wn3^2;
a = 10^-0.9;
b = 2*zeta2*wn2;
c = wn2^2;
d = 2*zeta4*wn4;
e = wn4^2;


Kx = 10^(-28.034/20);
H = tf(Kx*[1 b1+d1 e1+(b1*d1)+c1 (b1*e1)+(c1*d1) c1*e1],[1 (d+b+a) (e+((b+a)*d)+c+(a*b))
(((b+a)*e)+((c+(a*b))*d)+(a*c)) (((c+(a*b))*e)+(a*c*d)) a*c*e]);
[mag, phase, freq] = bode(H);

figure;
```

```
subplot(2,1,1);
semilogx(freq, 20*log10(squeeze(mag)));
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');
title('Bode Magnitude Plot');
grid on;
hold on;
plot(freq_data,mag_data);
legend('Analytical Data', 'Empirical Data');
xlim([0.1 10]);
hold off

subplot(2,1,2);
semilogx(freq, squeeze(phase));
xlabel('Frequency (rad/s)');
ylabel('Phase (rad)');
title('Bode Phase Plot');
axis([0 12 -140 96]);
grid on;
hold on;
plot(freq_data,phase_data);
legend('Analytical Data', 'Empirical Data');
axis([0.1 10 -140 96]);
```

Question 2:

```
% Define the range of K values, including both small and large values
K_values = [-10, -1, -0.1, -0.01, 0.01, 0.1, 1, 10, 100];  % Specific values of K


phases_deg = deg2rad(phase_data);
mag_nyq = 10.^(mag_data/20);
nyq_real = mag_nyq .* cos(phases_deg);
nyq_imag = mag_nyq .* sin(phases_deg);

% Loop over the K values
for i = 1:length(K_values)
   % Get the current value of K
   K = K_values(i);

   % Open-loop transfer function L(s) = K * G(s)
   L = K * H;
```

```
    % Bode plot
    figure;
    subplot(2,1,1);
    plot(freq_data,mag_data+(20*log10(K)),'r');
    hold on;
    x = bodeplot(L);
    setoptions(x, 'PhaseVisible', 'off');
    xlabel('Frequency (rad/s)');
    ylabel('Magnitude (dB)');
    title(['Bode Magnitude Plot for K = ', num2str(K)]);
    grid on;
    plot(freq_data,K*mag_data,'r');
    legend('Analytical Data', 'Empirical Data');
    xlim([0.1 10]);
    hold off

    subplot(2,1,2);
    x = bodeplot(L);
    setoptions(x, 'MagVisible', 'off');
    xlabel('Frequency (rad/s)');
    ylabel('Phase (rad)');
    axis([0 12 -140 96]);
    title(['Bode Phase Plot for K = ', num2str(K)]);
    grid on;
    hold on;
    plot(freq_data,phase_data-(((abs(K)-K)/(2*K))*180),'r');
    legend('Analytical Data', 'Empirical Data');
    axis([0.1 10 -140 100]);

    % Nyquist plot
    figure
    nyquist(L);
    hold on
    plot(K*nyq_real,K*nyq_imag,"r");
    grid on;
    title(['Nyquist Plot for K = ', num2str(K)]);
    legend('Analytical Data', 'Empirical Data');

    % Pause for a moment to allow the plots to update
    pause(0.5);
end
```

Question 3:

```
% Analytic Transfer function from Part 1
H = tf(K*[1 b1+d1 e1+(b1*d1)+c1 (b1*e1)+(c1*d1) c1*e1],[1 (d+b+a) (e+((b+a)*d)+c+(a*b))
(((b+a)*e)+((c+(a*b))*d)+(a*c)) (((c+(a*b))*e)+(a*c*d)) a*c*e])

% Check the margins of H
margin(H)
grid on;

% The GM and PM are Inf.
% We have the target gain crossover frequency at 6.28 rad/sec.
% The Phase at 6.28 rad/sec is -40 deg. Phase margin calculation
% with this phase is 140 deg.
% We need to add positive gain to the compensator such that
% the magnitude plot crosses 0 dB at 6.28 rad/sec.

% As we need to shift the magnitude plot using some gain
% to have the gain crossover frequency at 1 Hz and not change
% the phase as the PM is already at least 40 deg, we check
% with just a proportional controller.

w_c = 2*pi*1; % crossover frequency at 1 Hz
mag_K_c = abs(squeeze(freqresp(H,w_c)));
K_c = 1/mag_K_c % gain required for target crossover frequency

% Check margins without the compensator
margin(K_c * H)
[Gm_H, Pm_H, Wcg_H, Wcp_H] = margin(K_c * H)
% GM is still Inf. Negative PM shows that the system is unstable.
% We need a compensator that stabilizes the system.

% As we need to shift the magnitude plot up to set the gain
% crossover frequency at 1 Hz, we use a positive gain.
% We do not need to alter the phase right now, although
% we can reduce the phase by some amount. Main purpose
% of designing a compensator is to give positive gain for
% the magnitude plot to cross 0 dB.
% Hence, we choose a LAG COMPENSATOR for the same, as
% a Lead Compensator will add phase, which is the opposite
% of what we want to achieve.

s = tf('s');
```

```
w_c = 2*pi*1; % crossover frequency at 1 Hz

% Compensator Design
% Lag compensator (z > p)
% Lag compensator reduces the phase in this frequency range:
% w_start = 0.1*p to w_end = 10*z.
% Also, max phase lag occurs at: w = sqrt(z*p).

C1_lag = (s+62)/(s+0.636);
C_H_lag = C1_lag * H;
% Gain
mag_K_lag = abs(squeeze(freqresp(C_H_lag, w_c)));
K_lag = 1/mag_K_lag
% Open Loop transfer function
L_lag = K_lag * C_H_lag;

% LEAD COMPENSATOR:
%phi = 25; % phase change required
%alpha = (1-sind(phi))/(1+sind(phi));
%tau = 1/(w_c*sqrt(alpha));
%C_lead = tf([tau 1], [alpha*tau 1]); % compensator transfer function
%C = K_c * C_lead; % Compensator Design
%L = C * H; % Open Loop transfer function

% Analyzing the margins
margin(L_lag)
grid on;
[Gm_lag, Pm_lag, Wcg_lag, Wcp_lag] = margin(L_lag)

% Closed-loop system with compensator
T = feedback(L_lag, 1);
% Check the new margins and bandwidth with the compensator
margin(T)
grid on;
[bw] = bandwidth(T);
disp(['Closed-Loop Bandwidth: ', num2str(bw), 'Hz']);

% We see from the above closed loop margin plot that
% PM for the closed loop system with the tuned lag
% compensator is 40.4 deg, perfect according to the
% requirements, and the Bandwidth is 0.656 Hz, which
% is closed enough to 1 Hz. But, the GM is still undefined.
```

```
% We choose to add another lag compensator that makes
% the phase plot go down at higher frequencies and potentially
% cross the -180 deg line and then tune that pole to achieve
% the required GM. We can see that there are 3 regions in the
% magnitude plot where the magnitude is below -10 dB:
% 1. w --> 0.7 - 0.8 rad/sec
% 2. w --> 3.7 - 5.7 rad/sec
% 3. w --> 7.4 rad/sec and beyond.
% If we observe these three regions in the phase plot, we can see
% that the first two regions have high phase values and opting
% them for reducing the phase below -180 deg is not a wise choice.
% Hence, the new lag compensator will focus on frequencies:
% w --> 7.4 - 20 rad/sec. We are choosing 20 rad/sec as the upper
% limit as the phase starts to increase beyond that frequency.

% Another Lag Compensators
s = tf('s');
C2_lag = (s+20)/(s+7.4);
% Maximum reduction in phase occurs at 12.165 rad/sec (sqrt(20*7.4)).
% Minimum required reduction in phase:
% 180 - phase at 12.165 rad/sec = 180 - 162 = 18 deg
% Expected reduction in phase:
% asind((z-p)/(z+p)) = 27.32 deg
% Hence, we can hope that this works!

C_total = C2_lag * C_H_lag;
% Gain
mag_K_lag_lag = abs(squeeze(freqresp(C_total, w_c)));
K_lag_lag = 1/mag_K_lag_lag
K_new = 0.1;
L_total = K_new * K_lag_lag * C_total;

% Analyzing the margins
margin(L_total)
title('Compensated Open Loop Frequency Response')
grid on;
[Gm_total, Pm_total, Wcg_total, Wcp_total] = margin(L_total)

% Closed-loop system with compensator
T_total = feedback(L_total, 1);
% Check the new margins and bandwidth with the compensator
margin(T_total)
grid on;
```

```
title('Compensated Closed Loop Frequency Response')
[bw_total] = bandwidth(T_total);
disp(['Closed-Loop Bandwidth: ', num2str(bw_total), 'Hz']);
```

Question 4:

```
% Designing the compensator for the empirical data

% Read Data from Excel
rawTable = readtable('Spacecraft_spin_module_frequency_response_data.xlsx','Sheet','Sheet1');

% Freq in Hz
freq_data_Hz = rawTable.Frequency_Hz_;
% Freq in rad/sec
freq_data_rad_sec = 2*pi*freq_data_Hz;

% Mag in (rad/sec)/V
mag_data_linear = rawTable.Magnitude__rad_sec__V_;
% Mag in dB
mag_data_dB = 20*log10(mag_data_linear);

% Phase in rad
phase_data_rad = rawTable.Phase_rad_;
% Phase in deg
phase_data_deg = phase_data_rad*360/(2*pi);

% Converting the empirical data to complex frequency response
H_empirical = mag_data_linear .* exp(1j * phase_data_rad);

% Empirical Bode Plot
figure;
subplot(2,1,1)
semilogx(freq_data_rad_sec, mag_data_dB)
ylabel('Magnitude (dB)')
title('Empirical Frequency Response')
grid on

subplot(2,1,2)
semilogx(freq_data_rad_sec, phase_data_deg)
xlabel('Frequency (rad/sec)')
ylabel('Phase (deg)')
grid on
```

```
% Desigining the compensator for empirical data
% We will choose the same compensator as designed for
% the analytic transfer function as the initial compensator
% for the empirical data and then tune as required.

% Lag Compensator
s = tf('s');
w_c = 2*pi*1; % crossover frequency at 1 Hz

% Compensator Design
% Lag compensator (z > p)
% Lag compensator reduces the phase in this frequency range:
% w_start = 0.1*p to w_end = 10*z.
% Also, max phase lag occurs at: w = sqrt(z*p).

C1_lag_empirical = (s+70)/(s+1);
[mag_lag_empirical, phase_lag_empirical, ~] = bode(C1_lag_empirical, freq_data_rad_sec);
mag_lag_empirical = squeeze(mag_lag_empirical);
phase_lag_empirical = squeeze(phase_lag_empirical);

% Open Loop Frequency Response with Compensator
mag_L_empirical = mag_lag_empirical .* mag_data_linear;
mag_L_empirical_dB = 20*log10(mag_L_empirical);
phase_L_empirical_deg = phase_lag_empirical + phase_data_deg;
L_empirical = mag_L_empirical .* exp(1j * deg2rad(phase_L_empirical_deg));

% Plot the bode plot for the open loop frequency response
subplot(2,1,1)
semilogx(freq_data_rad_sec, mag_L_empirical_dB);
%hold on;
%yline(0, 'r--', '0 dB line');
ylabel('Magnitude (dB)')
title('Open Loop Freq Response with Compensator')
grid on

subplot(2,1,2)
semilogx(freq_data_rad_sec, phase_L_empirical_deg);
%hold on;
%yline(-180, 'r--', '-180 deg line')
xlabel('Frequency (rad/sec)')
ylabel('Phase (deg)')
grid on
```

```
% Unity Feedback Closed Loop system
T_empirical = L_empirical ./ (1 + L_empirical);
K_new_empirical = 5;
T_empirical_total = K_new_empirical .* T_empirical;
mag_T_empirical_total = 20*log10(abs(T_empirical_total));
phase_T_empirical_total = rad2deg(angle(T_empirical_total));

% Plot the closed loop empirical frequency response
subplot(2,1,1)
semilogx(freq_data_rad_sec, 20*log10(abs(T_empirical_total)))
ylabel('Magnitude (dB)')
title('Closed Loop Tracking Frequency Response')
grid on
subplot(2,1,2)
semilogx(freq_data_rad_sec, rad2deg(angle(T_empirical_total)))
xlabel('Frequency (rad/sec)')
ylabel('Phase (deg)')
grid on

% Find the Gain Margin and Phase Margin for the closed loop response
% from the above plots

% Find index of point closest to 0 dB
[~, idx_gc] = min(abs(mag_L_empirical_dB));  % smallest difference from 0 dB
% Estimate phase margin
phase_at_gc = phase_L_empirical_deg(idx_gc);
PM = phase_at_gc + 180;
w_gc = freq_data_rad_sec(idx_gc);
fprintf('Approximate Gain Crossover at %.2f rad/s → PM ≈ %.2f deg\n', w_gc, PM);

% Find index of point closest to -180 deg
[~, idx_pc] = min(abs(phase_L_empirical_deg + 180));  % phase closest to -180 deg
% Estimate gain margin
mag_at_pc = mag_L_empirical_dB(idx_pc);
GM = -mag_at_pc;
w_pc = freq_data_rad_sec(idx_pc);
fprintf('Approx Phase Crossover at %.2f rad/s → GM ≈ %.2f dB\n', w_pc, GM);

% Find the bandwidth of the open loop
idx_bw = find(mag_T_empirical_total < -3, 1);
bw_closed_loop = freq_data_rad_sec(idx_bw);
fprintf('Closed-Loop Bandwidth ≈ %.2f rad/s (%.2f Hz)\n', bw_closed_loop,
bw_closed_loop/(2*pi));
```

```
% We can see from the output that the PM for the closed-loop system
% is 50.78 deg (> 40 deg), the GM is 25.53 dB (> 10 dB) and the
% Bandwidth is 0.60 Hz (~ 1 Hz). All these values are close to what
% was achieved using the analytic transfer function, although we could
% achieve the same performance with one less lag compensator.
```

Question 5:

```
% Simulating the analytic model in a unity feedback control loop
% in Simulink using the designed compensator for the analytic transfer
% function.

% Inputs:
% 1. 0.5 rad step input
% 2. Sine input at 0.5 rad, 0.5 Hz
% 3. Sine input at 0.5 rad, 2.0 Hz

% Call the analytic transfer function
H

% Designed compensator for the analytic transfer function
C_total

% Open Loop Transfer function with the compensator and gains
L_total

% Unity Feedback Closed Loop Transfer Function with the
% analytic transfer function
T_total

% Step Input Simulation
open_system('stepinput_closedloop_simulation.slx')

% First Sinusoidal Input (0.5 rad, 3.14 rad/sec)
open_system('sineinput1_closedloop_simulation')

% Second Sinusoidal Input (0.5 rad, 12.56 rad/sec)
open_system('sineinput2_closedloop_simulation')
```

Question 6:

```
% Analytic Transfer function from Part 1
wn2 = 0.7323;
mag2 = 29.7268;
zeta2 = (10^(-mag2/20))/2;
wn1 = 1.32844;
mag1 = 24.2747;
zeta1 = (10^(-mag1/20))/2;
wn3 = 4.88315;
mag3 = 24.1018;
zeta3 = (10^(-mag3/20))/2;
wn4 = 6.13774;
mag4 = 24.1018;
zeta4 = (10^(-mag4/20))/2;

b1 = 2*zeta1*wn1;
c1 = wn1^2;
d1 = 2*zeta3*wn3;
e1 = wn3^2;
a = 10^-0.9;
b = 2*zeta2*wn2;
c = wn2^2;
d = 2*zeta4*wn4;
e = wn4^2;


Kx = 10^(-28.034/20);
H = tf(Kx*[1 b1+d1 e1+(b1*d1)+c1 (b1*e1)+(c1*d1) c1*e1],[1 (d+b+a) (e+((b+a)*d)+c+(a*b))
(((b+a)*e)+((c+(a*b))*d)+(a*c)) (((c+(a*b))*e)+(a*c*d)) a*c*e])
figure;
bode(H);
xlim([0.1 10]);

w_c = 2*pi*1; % crossover frequency at 1 Hz
mag_K_c = abs(squeeze(freqresp(H,w_c)));
K_c = 1/mag_K_c % gain required for target crossover frequency

s = tf('s');
w_c = 2*pi*1; % crossover frequency at 1 Hz

% Compensator Design
% Lag compensator (z > p)
```

```
C1_lag = (s+62)/(s+0.636);
% This gives phi_min = -44.5 deg
C_H_lag = C1_lag * H;
% Gain
mag_K_lag = abs(squeeze(freqresp(C_H_lag, w_c)));
K_lag = 1/mag_K_lag
% Open Loop transfer function
L_lag = K_lag * C_H_lag;

% Another Lag Compensator
C2_lag = (s+20)/(s+7.4);

C_total = C2_lag * C_H_lag;

% Gain
mag_K_lag_lag = abs(squeeze(freqresp(C_total, w_c)));
K_lag_lag = 1/mag_K_lag_lag
K_new = 0.219;
L_total = K_new*K_lag_lag * C_total;

% Closed-loop system with compensator
T_total = feedback(L_total, 1);
% Check the new margins and bandwidth with the compensator
figure;
margin(L_total)
[Gm_total, Pm_total, Wcg_total, Wcp_total] = margin(L_total)
%step(T)
grid on;
[bw_total] = bandwidth(T_total);
disp(['Closed-Loop Bandwidth: ', num2str(bw_total), 'Hz']);
```