Hacettepe University - Computer Engineering

Advanced Robust Control MMÜ 749

Homework-2

Student Name	Ayça KULA
Student ID:	202237285

Spring 2022

1 Compute the 2 and infinity norm of the system with the indicated method for each case

1.1 Direct integration procedure for 2 norm

G must be strictly proper, otherwise the H2 norm is infinite. We say that $G(s)$ is stable if it is analytic in the closed right half-plane (Re s 0), proper if $G(j\omega)$ is finite (degree of denominator degree of numerator), strictly proper if $G(j\omega) = 0$ (degree of denominator $>$ degree of numerator), and biproper if G^ and G^1 are both proper (degree of denominator $=$ degree of numerator).

2-Norm

$$\|\hat{G}\|_2 := \left(\frac{1}{2\pi} \int_{-\infty}^{\infty} |\hat{G}(j\omega)|^2 d\omega \right)^{1/2} \tag{1}$$

- **First example**

$$\frac{s+2}{4s+1} \tag{2}$$

The system is stable and finite.

$$G_{j\omega} = \frac{j\omega+2}{4j\omega+1} = \frac{j\omega+2}{4j\omega+1} = \frac{(j\omega+2)(1-4j\omega)}{(4j\omega+1)(1-4j\omega)} = \frac{4\omega^2+2-7j\omega}{16\omega^2+1} \tag{3}$$

By calculating rest of the square and integral parts in Matlab as in algorithm 2. The result is obtained as infinity because the system is not strictly proper and the system has nonzero feedthrough.

```
clc;clear;
syms w
real_sys1 = (4*w^2+2)/(16*w^2+1);
comp_sys1 = (-7*w)/(16*w^2+1);
abs_g = simplify(real_sys1^2 + comp_sys1^2);
F = int(abs_g,w,[-inf inf]);
result = sqrt(F/(2*pi))

% Check the result
sys = tf([1 2],[4 1]);
norm(sys)
```

Algorithm 1: Calculating 2-Norm in Matlab

$$\|\hat{G}\|_2 := \infty \tag{4}$$

- **Second example** Suppose that G is strictly proper and has no poles on the imaginary axis (so its 2-norm is finite).

$$\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}, \quad \zeta, \omega_n > 0 \tag{5}$$

The system is stable and strictly proper. The poles for this system is,

$$p_{1,2} = -\zeta\omega_n \pm \omega_n\sqrt{(\zeta^2 - 1)} \tag{6}$$

Since $\zeta > 0$, the system could be overdamped($\zeta > 1$, two distinct real poles), critically damped($\zeta = 1$, two identical real poles), under damped ($1 > \zeta \geq 0$, two complex poles).

$$G_{j\omega} = \frac{\omega_n^2}{-\omega^2 + 2\zeta\omega_n\omega j + \omega_n^2} \tag{7}$$

$$\|G(s)\|_2 = (\frac{1}{2\pi}\frac{\omega_n \tan^{-1}\left(\frac{4\omega_n^2\zeta^2\omega - \omega_n^2 + \omega}{2\omega_n^3\zeta}\right)}{2\zeta})^{\frac{1}{2}} \tag{8}$$

According to Matlab, the result is $\|\hat{G}\|_2 := \infty$.

```
syms Wn W
syms zeta

% real part for the system
real = Wn^2*(Wn^2-W^2)/((Wn^2-W^2)+(2*zeta*Wn*W)^2);
% imaginary part for the system
imag = -Wn^2*(2*zeta*Wn*W)/((Wn^2-W^2)+(2*zeta*Wn*W)^2);
abs_g = simplify(real^2 + imag^2);
F = int(abs_g,W,[-inf inf]);
result = sqrt(F/(2*pi))
```

Algorithm 2: Calculating 2-Norm in Matlab

1.2 Calculating peak of bode plot for inf norm

The formula for inf norm is given as:

$$\|G(s)\|_\infty = \max_\omega |G(j\omega)| \tag{9}$$

Algorithm 4 can be used to calculate inf norm.

```
% bode plot
margin(sys)

% inf_norm of dyamic system
[ninf,fpeak] = hinfnorm(sys)

%  singular values of system
sigma(sys),grid
```

Algorithm 3: Calculating inf-Norm in Matlab

- **First example** Frequency of peak gain or largest singular value is 0 and the inf norm of dynamic system is 2. The bode plot can be seen in figure 2. The inf norm can be found as 2.

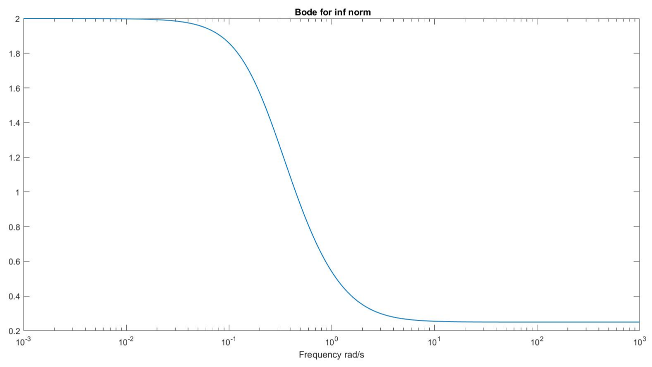


Figure 1: Bode plot for the first example

- **Second example** The inf norm plot can be done for different values of damping ratio. Taking the inverse Laplace transform of this equation yields the time solution for the response $c(t)$ as follows:
For $0 \leq \zeta < 1$,

$$c(t) = \frac{\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin \omega_n \sqrt{1-\zeta^2} t, \qquad \text{for } t \geq 0 \tag{10}$$

For $\zeta = 1$

$$c(t) = \omega_n^2 t e^{-\omega_n t}, \qquad \text{for } t \geq 0 \tag{11}$$

For $\zeta > 1$

$$c(t) = \frac{\omega_n}{2\sqrt{\zeta^2-1}} e^{-\left(\zeta-\sqrt{\zeta^2-1}\right)\omega_n t} - \frac{\omega_n}{2\sqrt{\zeta^2-1}} e^{-\left(\zeta+\sqrt{\zeta^2-1}\right)\omega_n t}, \qquad \text{for } t \geq 0 \tag{12}$$

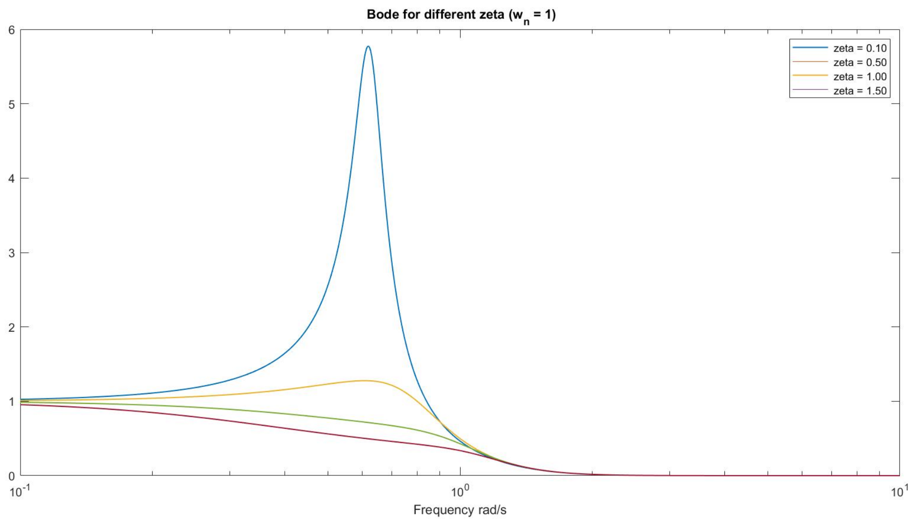


Figure 2: Bode plot for the second example

```
s = tf('s');
zeta = [0.1 0.5 1 1.5];
w_n = 1;
w = logspace ( -1 ,1 ,1000);

for i = 1: length(zeta)
    G = tf ([ w_n ^2] ,[1 2*zeta (i)*w_n w_n ^2]);
    % Bode response
    [mag,phase,wout ] = bode (G ,w);
    % Maximum value
    mag = squeeze(mag );
    [m,n] = max(mag );
    semilogx (wout,mag ,'Linewidth',1);
    hold on;
    plot (wout(n),m);

end
xlabel ('Frequency rad/s');
title ( sprintf (" Bode for different zeta (w_n = 1)"));
legend ( sprintf (" zeta = %.2f", zeta (1) ), sprintf (" zeta = %.2f", zeta (2) ),
```

Algorithm 4: Calculating inf-Norm in Matlab

1.3 Taking a derivative of freq response wrt freq for inf norm

- **First example** The differential result is:

$$\frac{d|\hat{H}|^2}{dw}(jw) = 0 \tag{13}$$

$$\frac{\partial}{\partial \omega}|H(j\omega)| = \frac{2\omega}{16\omega^2+1} - \frac{32\omega(\omega^2+4)}{(16\omega^2+1)^2} \tag{14}$$

The infinity norm is the smallest value of the bode frequency because ω has to be zero to make the derivative zero.

- **Second example** The amplitude-frequency response is given by when Stable complex-conjugate pole pairs occur in $0 < \zeta < 1$

$$|H(j\omega)| = \frac{\omega_n^2}{|\omega_n^2 - \omega^2 + j2\xi\omega_n\omega|} = \omega_n^2 \left[\left(\omega_n^2 - \omega^2\right)^2 + 4\xi^2\omega_n^2\omega^2\right]^{-1/2} \tag{15}$$

Also note that the peak value is not necessarily centered exactly at the corner frequency; to find the peak location we set the first derivative equal to zero, giving

$$\frac{\partial}{\partial \omega}|H(j\omega)| = 0 \quad \Rightarrow \omega = \omega_p = \omega_n\sqrt{1-2\xi^2} \tag{16}$$

This result tells us that there is a peak or maximum in the response only when $1 - \zeta^2 > 0$. In this range the peak amplitude is given by,(Low pass)

$$|H(j\omega_p)| = \frac{1}{2\xi\sqrt{1-\xi^2}} \Rightarrow -20\log\left(2\xi\sqrt{1-\xi^2}\right) \tag{17}$$

If we instead have a complex pair of zeroes in the transfer function,(High pass)

$$\frac{\partial}{\partial \omega}|H(j\omega)| = 0 \Rightarrow \omega = \omega_p = \frac{\omega_n}{\sqrt{1-2\xi^2}} \tag{18}$$

The infinity norm depends on ω_n and ζ.

1.4 Implementing a method based on Gramians for 2 norm

Definition: Let A be a stable matrix, then the matrix:

$$C_G = \int_0^{t_1} e^{At}BB^Te^{A^Tt}dt \tag{19}$$

is called the controllability Grammian. Definition: Let A be a stable matrix, then the matrix:

$$O_G = \int_0^{\infty} e^{A^Tt}C^TCe^{At}dt \tag{20}$$

is called the observability Grammian.

Let A be a stable matrix. The controllability Grammian C_G satisfies the Lyapunov equation

$$AC_G + C_GA^T = -BB^T \tag{21}$$

and is symmetric positive definite if and only if (A, B) is controllable. Let A be a stable matrix. The observability Grammian O_G satisfies the Lyapunov equation

$$O_G A + A^T O_G = -C^T C \tag{22}$$

and is symmetric positive definite if and only if (A, C) is observable.

The matrix A is stable and using MATLAB command lyap to solve Lyapunov equations in 21 and 22. Finally, in order to find the H_2 norm we will use:

$$\|G(s)\|_2 = \sqrt{\operatorname{tr}(B^T Q B)} \text{ or } \|G(s)\|_2 = \sqrt{\operatorname{tr}(C P C^T)} \tag{23}$$

where Q and P are the observability and controllability Gramians.In order to calculate these we use algorithm 5 in MATLAB.

```
%% Controllability and observability Gramians

% Take the transfer function of the system
sys = tf(num,den);

% Obtain state-space representation
[A,B,C,D] = tf2ss(num,den);
sys_ss = ss(A,B,C,D);

% The input is the state-space representations for the system
Wc = gram(sys_ss,'c'); % calculates the controllability Gramian of the state-space

Wo = gram(sys_ss,'o'); % calculates the observability Gramian of the ss model sys.

% Check the grammians controllability Gramian
Wc_check = lyap(A,B*transpose(B));

% Check the grammians controllability Gramian
Wo_check = lyap(A,C*transpose(C));

% H2 norm calculation --> For observability matrix
norm_Wo = sqrt(trace(B'*Wo*B));

% H2 norm calculation --> For controllability matrix
norm_Wc = sqrt(trace(C*Wc*C'));
```

Algorithm 5: Calculating Gramians in Matlab

First example And the the H_2 norm is found as $\|\hat{G}\|_2 = 0.6187$ from 5.

Second example It is hard to obtain a generic result. However, we can give some arbitrary values and calculate this matrices using 5.

1.5 Implementing a method based on Hamiltonian matrix for inf norm

The connection between the Hinf norm of a stable transfer function (i.e no poles in the right half plane) and its associated Hamiltonian matrix plays an important role in the algorithm for the state feedback case.

Consider the system:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

Definition: From the coefficients in the system above, we define M_r, the Hamiltonian matrix

$$M_r = \begin{bmatrix} A + BR^{-1}D^TC & BR^{-1}B^T \\ -C^T\left(I + DR^{-1}D^T\right)C & -\left(A + BR^{-1}D^TC\right)^T\Big) \end{bmatrix}$$

where $R = r^2I - D^TD$. This is an iterative procedure, where one may start with a large value of "r" and reduce it until imaginary eigenvalues for H appear.

Recall the transfer function G(s) defined from coefficient matrices of the system,

$$G(s) = C(si - A)^{-1}B + D \tag{24}$$

Let $G(s)$ be a stable transfer function and let $r > 0$. Then $\|G\|_\infty < r$ if and only if $\sigma_{\max}(D) < r$ and M_r has no purely imaginary eigenvalues.

$$H = \begin{bmatrix} 0.1094/*R - 0.25 & 1/R \\ -0.0120/R - 0.1914 & 0.25 - 0.1094/R \end{bmatrix} \tag{25}$$

$$\lambda_i = \pm\frac{\sqrt{16R - 63}}{16\sqrt{R}} \tag{26}$$

λ has chosen as bigger than 2 in order to eliminate the no imaginary eigenvalues. $\|G(s)\|_\infty \approx = 2$.

In order to run a generic Hamiltonian matrix, I have obtained an algorithm for the second system as in 6. For any, ω and ζ we can run this code. Inf norm is $\|G\|_\infty = \gamma = 1$ for these second order systems.

```
%% Find rlb & rub
n = 1;
tol = 10^(-6);
sigma = max(eig(A*A'));
rlb = max(sigma,sqrt(n*trace(Wc*Wo)/n));
rub = sigma + 2*sqrt(n*trace(Wc*Wo));
r = (rlb + rub)/2;
while 2*(rub-rlb) > tol
    R = r^2*eye(2)-D'*D;
    % Calculate the Hamiltonaian Matrix
    Mr = [A + B*inv(R)*D'*C B*inv(R)*B'; -C'*(eye(2)+D*inv(R)*D')*C ...
        -(A+B*inv(R)*D'*C)'];
    % Find the eigenvalues
    E = eig(Mr);
    imaginal = imag(E)
    % if there is complex eigenvalue
    if any(imaginal) == 1
        rlb = r;
    else
        rub = r;
    end
end
```

Algorithm 6: Calculating Hamiltonian matrix in Matlab

1.6 Implementing a method based on max singular value for inf norm over all freq

Given system with coefficient matrices A,B,C,D where A is stable one can find the Hinf norm using the following steps.

- Compute the transfer function $G(s)$ of the given system.
- Find the poles of $G(s)$. If the poles are strictly real valued, set $\omega_p = \max|\lambda_i| \cdot |$ Else set $\omega_p = \lambda_i$, where λ_i maximizes

$$\left| \frac{\operatorname{Im}(\lambda_i)}{\operatorname{Re}(\lambda_i)} \frac{1}{\lambda_i} \right| \tag{27}$$

- Compute r_{lb} using

$$r_{lb} = \max\left\{\sigma_{\max}G(0), \sigma_{\max}G(\omega_p i), \sigma_{\max}(D)\right\} \tag{28}$$

- Compute

$$r = (1 + 2\epsilon)r_{lb} \tag{29}$$

M_r Sort all of the purely imaginary eigenvalues of Mr, and label them in descending order $\omega_1, \ldots, \omega_k$. If M_r has no purely imaginary eigenvalues,set

- For $i = 1$ to $k{-}1$ (i) Compute $m_i = \frac{1}{2}\left(\omega_i + \omega_{i+1}\right)$. (ii) Compute $\sigma_{\max}\left(G\left(m_i i\right)\right) = svd_i$. Set$_{lb} = \max\left(svd_i\right)$.
- Compute $\|G\|_\infty = \frac{1}{2}\left(r_{lb} + r_{ub}\right)$.

  The singular values of a matrix A, j (A) are definted as

$$\sigma_j(A) = \lambda_j\left(AA^T\right) \tag{30}$$

  where $\lambda_j(E)$ denotes the j th eigenvalue of the matrix E. Note the Euclidean norm of a matrix is defined to be

$$\|A\| = \max_i \sigma_i(A) \tag{31}$$

The algorithm is really complex. Therefore I used the algorithm 7 taken from [1]. Here, the results are obtained as:

$$\begin{aligned} \|G_1\|_\infty &= 8.6652 \\ \|G_2\|_\infty &\approx 1 \end{aligned} \tag{32}$$

```
clc ; clear ;
zeta = 0.5;
wn = 1;
s = tf('s');
% sys = s + 2 / (4* s + 1);
sys = wn ^2 / (s^2 + 2* zeta *wn*s + wn ^2);
tol = 1e-5; % tolerance for inf - norm
[sv ,w] = sigma ( sys ); % compute singukar values and frequencies
% compress sv and w into 1-D array add 0 frequency
sv = squeeze (sv); sv = [ sigma(sys,0) sv ];
w = squeeze (w); w = [0; w];
[svmax ,ix] = max (sv); % compute maximum sigular value
wmax = w(ix); % frequency attaining max singular value
while true
% new frequency range
new_w = linspace (w( max (ix -1 ,1)),w( min (ix +1, length (2) )) ,100);
sv_next = sigma (sys , new_w ); % next singular values and frequencies
sv_next = squeeze ( sv_next ); % compress sv and w into 1-D array
[ svmax_next , ix_next ] = max ( sv_next ); % compute next maximum sigular value
wmax_next = new_w ( ix_next );
% if in the tolerance , break the loop
if ( abs ( svmax_next - svmax ) > tol)
svmax = svmax_next ;
w = new_w ;
ix = ix_next ;
else
svmax = svmax_next ;
wmax = wmax_next ;
break ;
end
end
fprintf (" The infinity norm of the system : %f at %f frequency \r\n",svmax , wmax
```

Algorithm 7: Calculating inf norm

2 Design Lead, Lag, PI, PD, PID compensators

The open-loop transfer function of a unity feedback system is

$$G(s) = \frac{K}{s\left(1+\frac{s}{5}\right)\left(1+\frac{s}{20}\right)} \tag{33}$$

2.1 Sketch the system block diagram

In figure 3 simple block diagram can be seen.

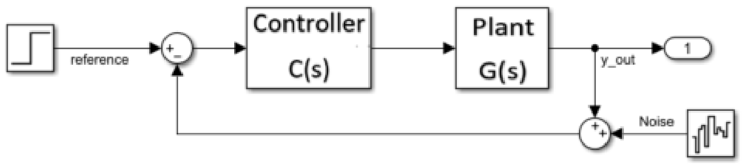


Figure 3: Control System block diagram

2.2 Design compensators

The design requirements are:

- The steady-state error to a unit ramp input is less than 0.01.

$$e_{\mathrm{ss}} = \frac{1}{K_v} < 0.01 \Rightarrow K_v > 100 \tag{34}$$

- $PM \geq 45^\circ$
- The steady-state error for sinusoidal inputs with $\omega < 0.2 rad/sec$ is less than $1/250$. $20log(250) = 47.9588dB$

$$y_{ss}(t) = a\cos(\omega t + \phi) \tag{35}$$

$$\text{where } a = |H(j\omega)|, \phi = \angle H(j\omega) \tag{36}$$

- Noise components introduced with the sensor signal at frequencies greater than 200 rad/sec are to be attenuated at the output by at least a factor of 100. $20log(0.01) = -40dB$.

  And the design requirements are defined as in 8.

```
%% Define the system
s = tf('s');
K = 1;
G = tf(100,[1 25 100 1]);
%% Define design requirements
% Kv should be bigger than 100
Kv = 100;
% Phase margin has to be bigger than 45
PM = 45;
% Gain should be bigger than 250 (w<0.2)
lowGainMargin_dB = 20*log10(250);
% Gain should be smaller than 0.01 (w>200)
HighGainMargin_dB = 20*log10(0.01);
```

Algorithm 8: Design Requirements are defined in Matlab

The designs are made using "Sisotool" and the open loop transfer function is plotted as 4.

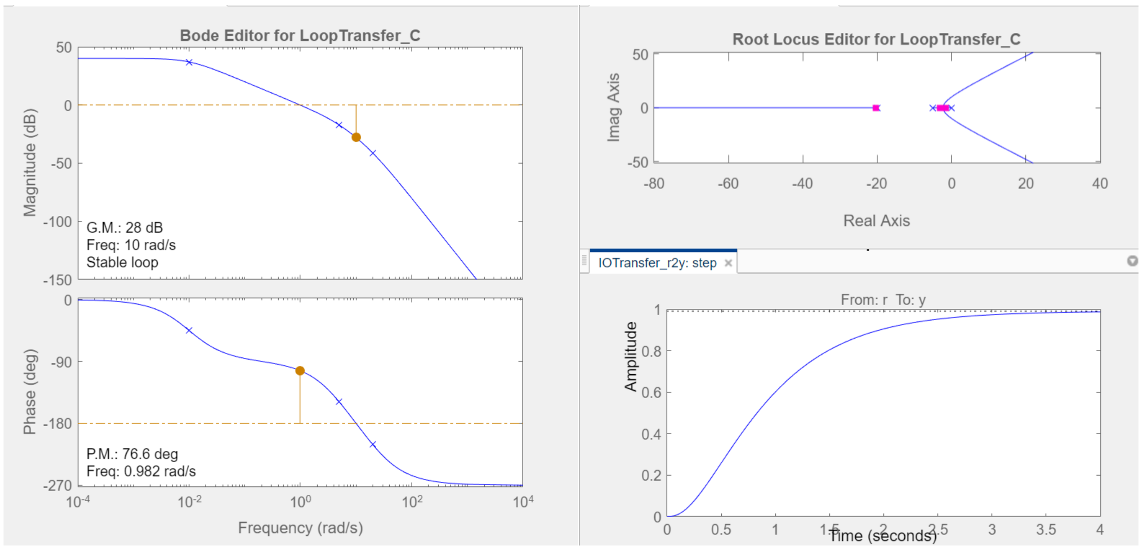
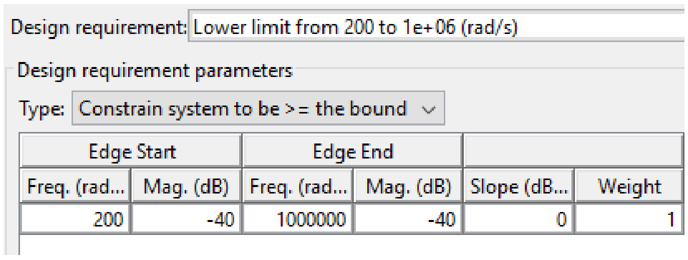


Figure 4: Open Loop response

And the design requirements are defined as in 5 and 6. From the open loop plot of $G(s)$ we can say that the open loop transfer function has desired phase margin and desired high frequency gain. And, we need to improve velocity constant and low frequency gain. The corresponding zero and poles are added considering this situation.



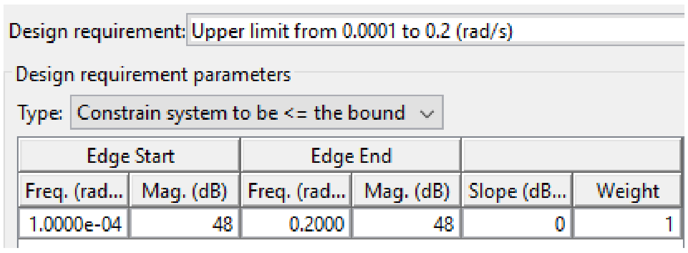
Design requirement: Lower limit from 200 to 1e+06 (rad/s)

Design requirement parameters

Type: Constrain system to be >= the bound

Edge Start		Edge End			
Freq. (rad...	Mag. (dB)	Freq. (rad...	Mag. (dB)	Slope (dB...	Weight
200	-40	1000000	-40	0	1

Figure 5: Define design requirements



Design requirement: Upper limit from 0.0001 to 0.2 (rad/s)

Design requirement parameters

Type: Constrain system to be <= the bound

Edge Start		Edge End			
Freq. (rad...	Mag. (dB)	Freq. (rad...	Mag. (dB)	Slope (dB...	Weight
1.0000e-04	48	0.2000	48	0	1

Figure 6: Define design requirements

2.2.1 Lead Compensator

The transfer function of the a compansator is defined as:

$$\text{Compensator } C(s) = K\left[\frac{T_d s + 1}{\alpha T_d s + 1}\right], \quad \alpha < 1 \tag{37}$$

As seen in figure 7, lead compensator has a good performance when we place the zero to -5 and pole to -50. From figure 7, the desired phase gain and high frequency are reached. However, no low frequency gain and no velocity constant is reached. There is a cancellation between the plant's pole and the compensator's zero. In order to meet the design requirements poles,zeros and gain has been changed. But, it is clear that at the same time lead compensator can not meet all the design requirements.In algorithm 9, there is another controller in 8, designed in order to satisfy the low and high frequency gain and make a comparison for the second part of the question. In this new transfer function phase margin requirement does not satisfy.

```
%% Designing Controllers
% Lead compensator where phase margin and desired high frequency gain satisfied
C (1) = 14.5*(1 + 0.2* s) /(1 + 0.02* s);
% Lead compensator where high and low frequency gains are satisfied
C (2) = 50*(1 + 0.2* s) /(1 + 0.02* s);
```

Algorithm 9: Designed Lead Compensators

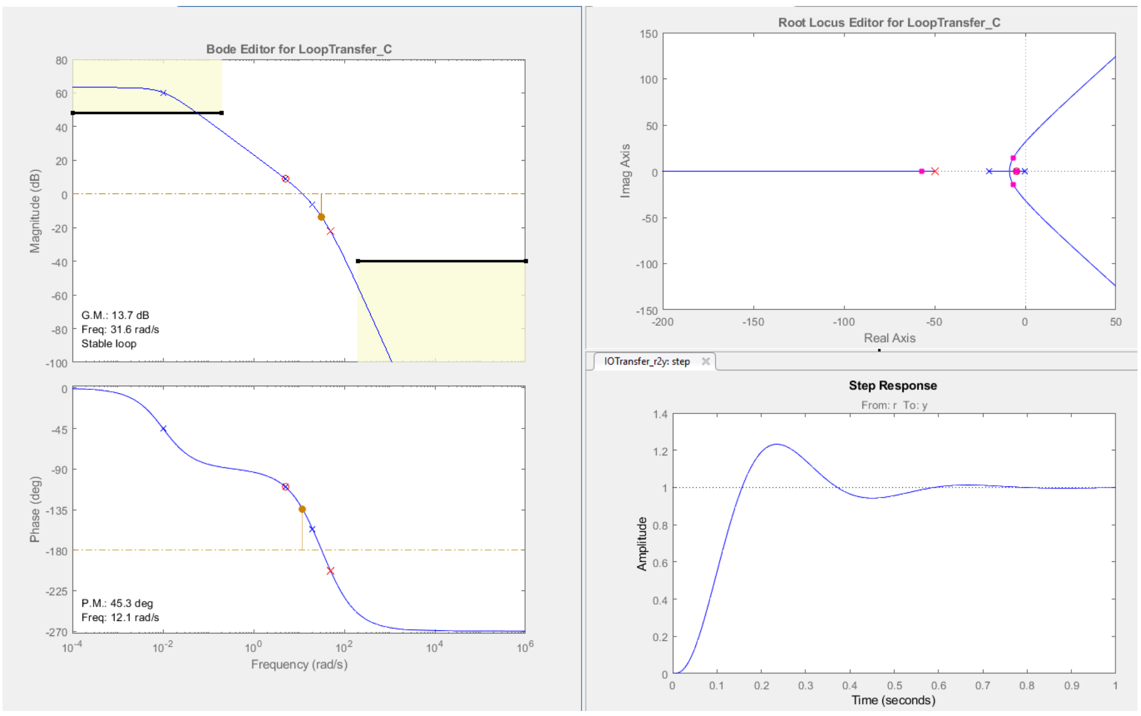


Figure 7: Lead Compensator

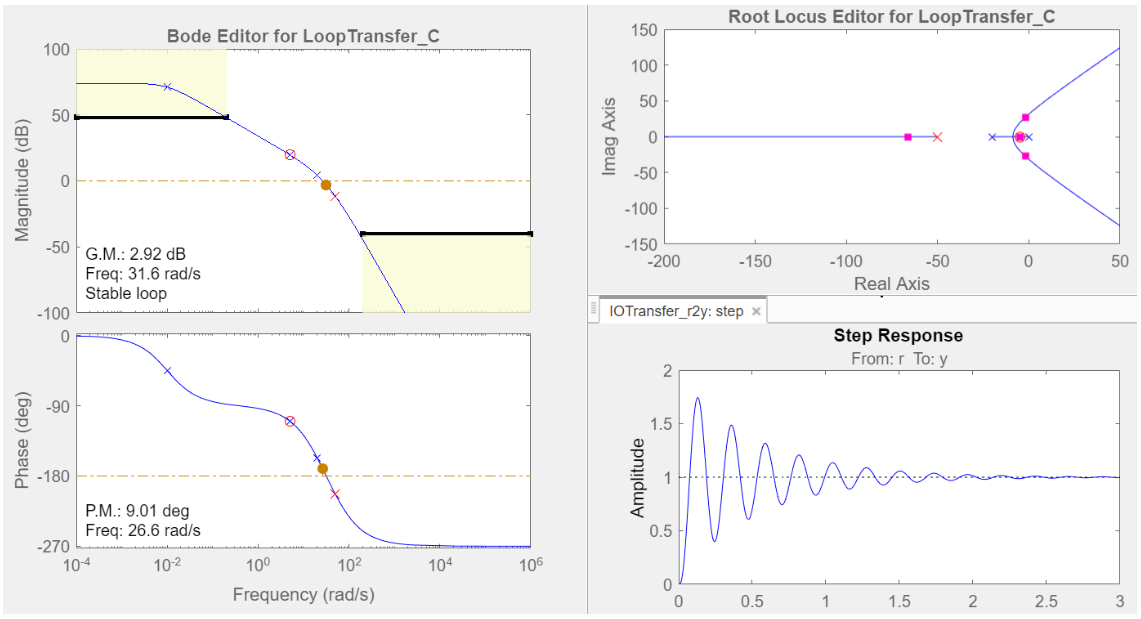


Figure 8: Second Lead Compensator

2.2.2 Lag Compensator

The pole is placed at -2 and the zero is placed at -0.2 as seen in figure 10. However, all of the design requirements are not satisfied at the same time. In figure 10, the low frequency gain and velocity constant is not satisfied. Many trials have been made. However, no combinations of poles,zeros and gains are obtained for all design requirements.In 10, there are two transfer functions that satisfies some of the requirements.

```
% Lag Compensator low frequency not satisfied
C (3) = 0.35*(1 + 0.5* s) /(1 + 5*s);
% Lag Compensator phase margin not satisfied
C (4) = 110*(1 + 0.5* s)/(1 + 5*s);
```

Algorithm 10: Designed Lag Compensators

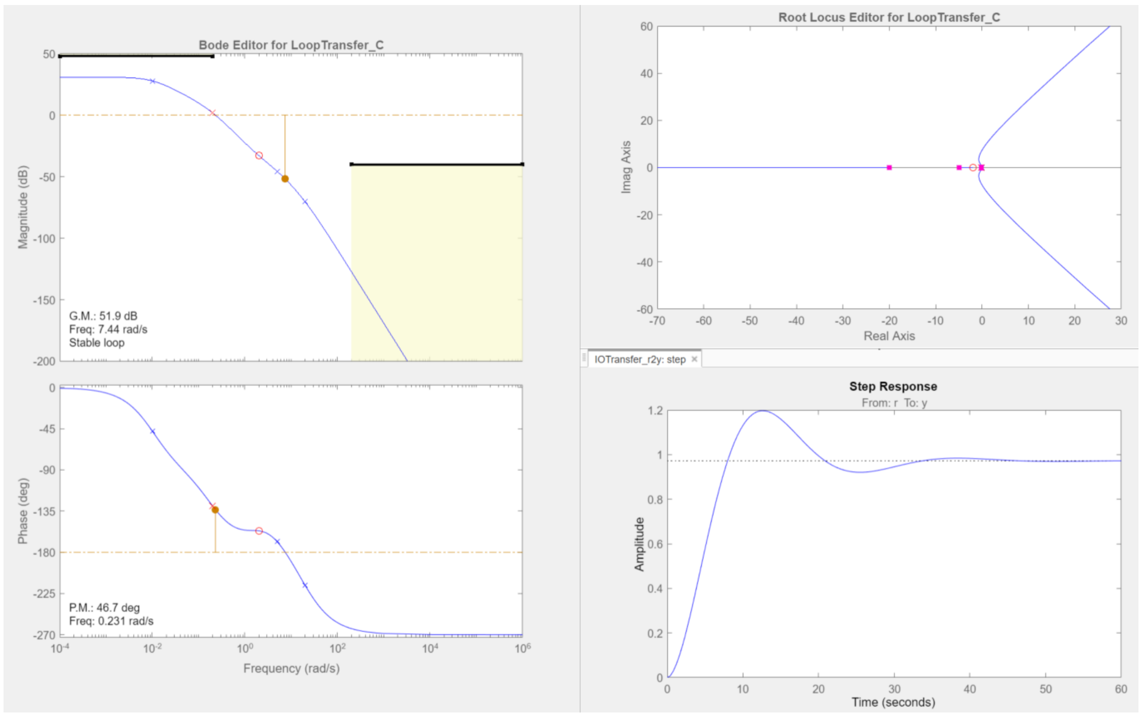


Figure 9: Lag Compensator

2.2.3 Lead Lag Compensator

As seen from figure 11, all the design requirements are satisfied. The appropriate transfer function can be seen from 11.

```
% Lead-Lag Compensator
C (5) = 150*((1 + 0.2* s) /(1 + 0.0091* s)) *((1 + 0.67* s)/(1 + 6.7* s));
```

Algorithm 11: Designed Lead-Lag Compensators

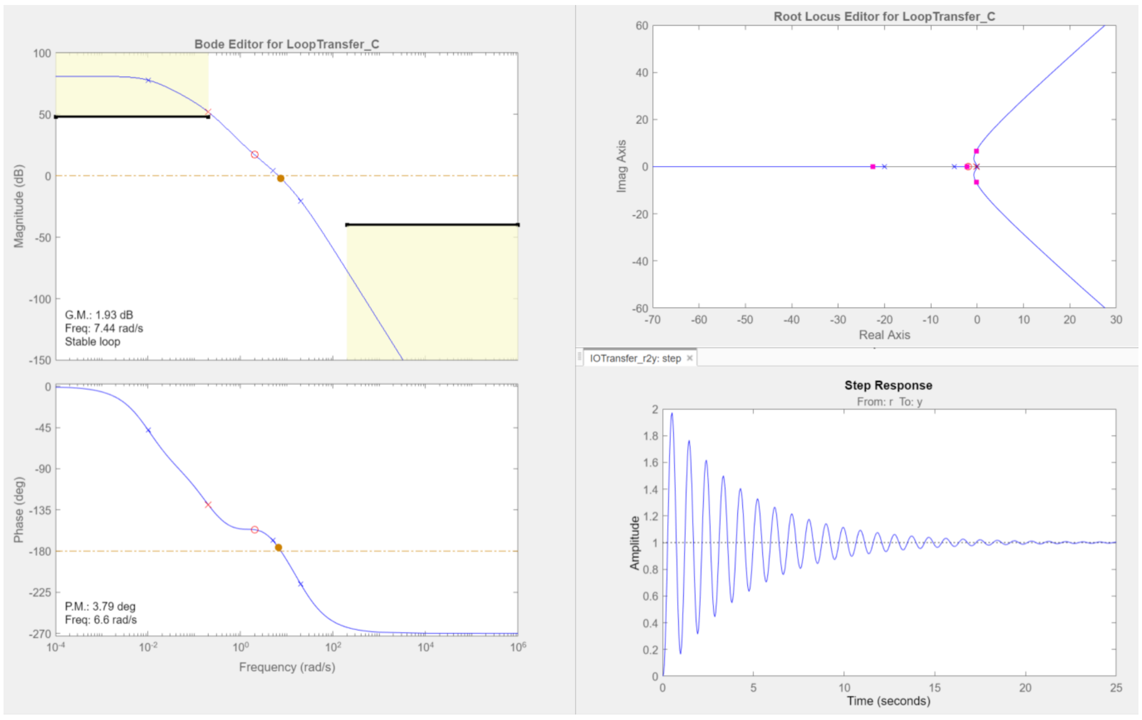


Figure 10: Second Lag Compensator

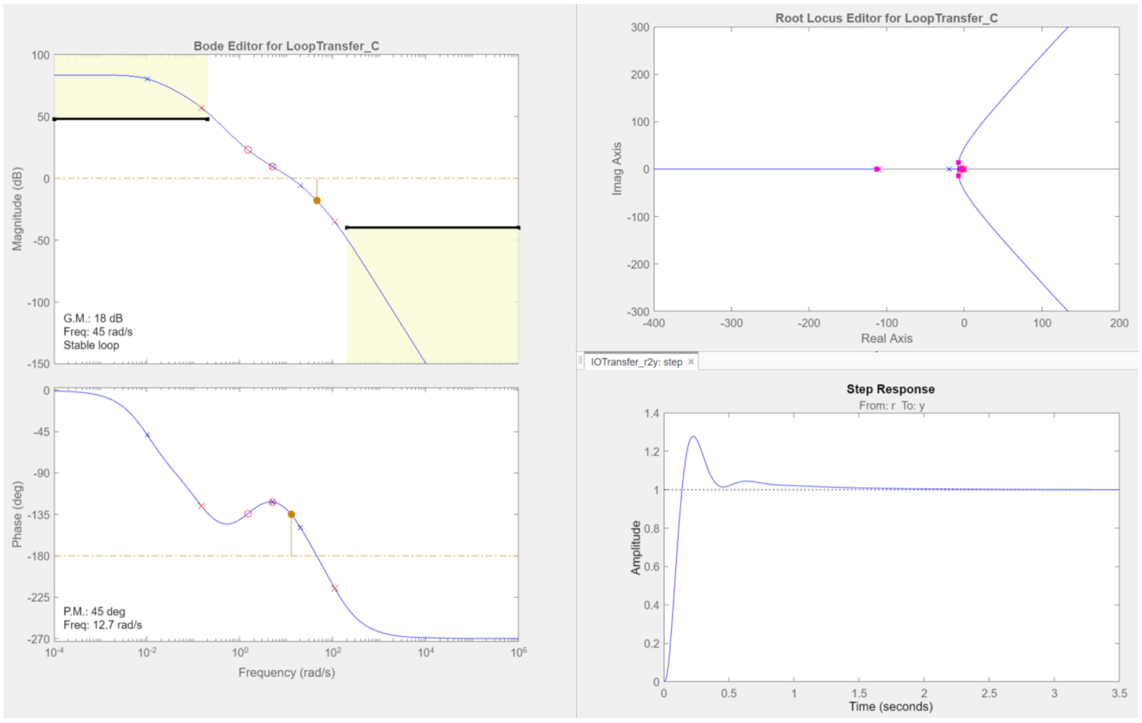


Figure 11: Lead Lag Compensator

2.2.4 PI

In order to improve the low frequency gain and the velocity constant we can change the zero locations. PI has high gain at low frequency. However, PI compensator decreases the phase at low frequencies. By putting a zero in -0.5, we meet the requirement for phase margin and high frequency gain (look at figure 12). On the other hand, the second function satisfies the low and high frequency gain criterion's in algorithm 12 and in figure 13. In conclusion,we can not satisfy all the requirements at the same time with a PI controller.

```
% PI Compensator where phase margin and desired high frequency gain satisfied
C (6) = 1.2*(1 + 2*s)/s;
% PI compensator where high and low frequency gains are satisfied
C (7) = 24*(1 + 0.56* s)/s;
```

Algorithm 12: Designed PI Compensators

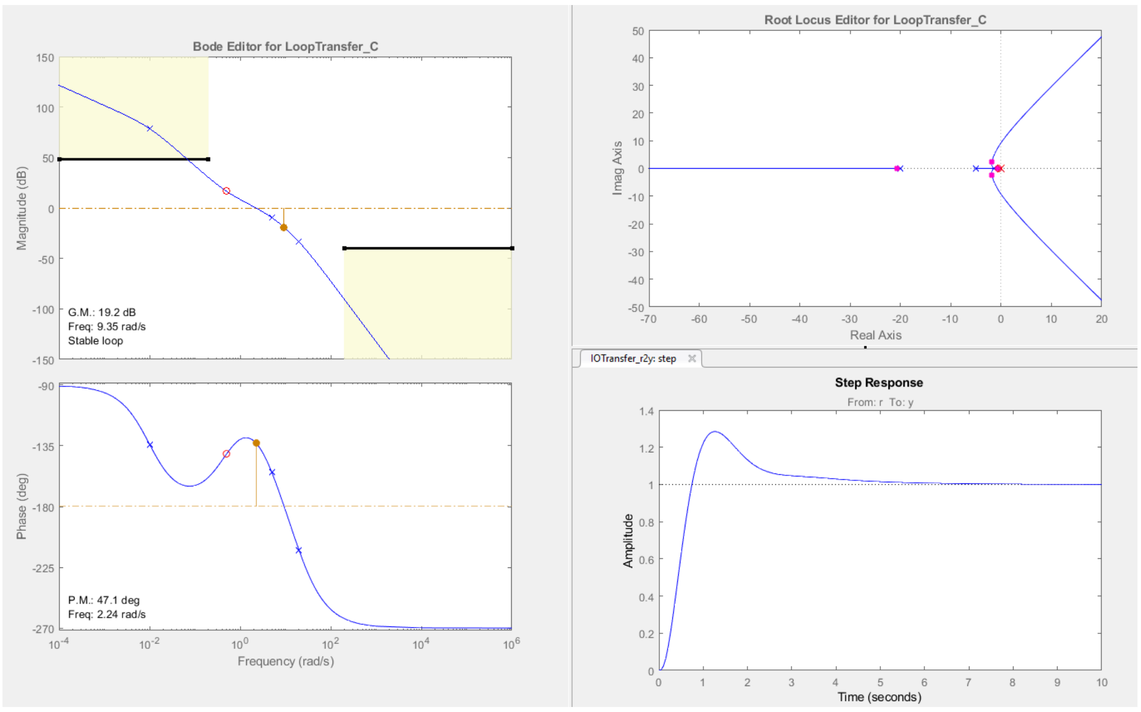


Figure 12: PI Controller

2.2.5 PD

By putting a zero in -8.3 as seen in 15, we meet the requirement for phase margin and high frequency gain. Moreover, in order to meet high and low frequency gains we obtain the second transfer functipn in 13. The appropriate transfer functions can be seen from 13.

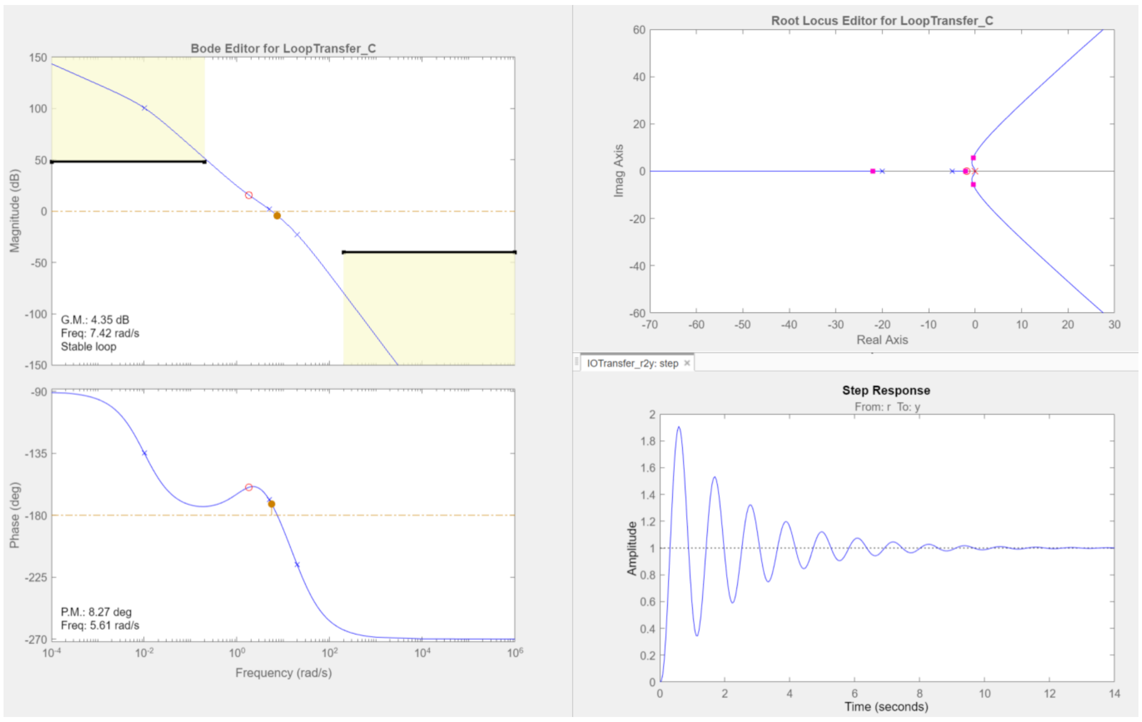


Figure 13: Second PI Controller

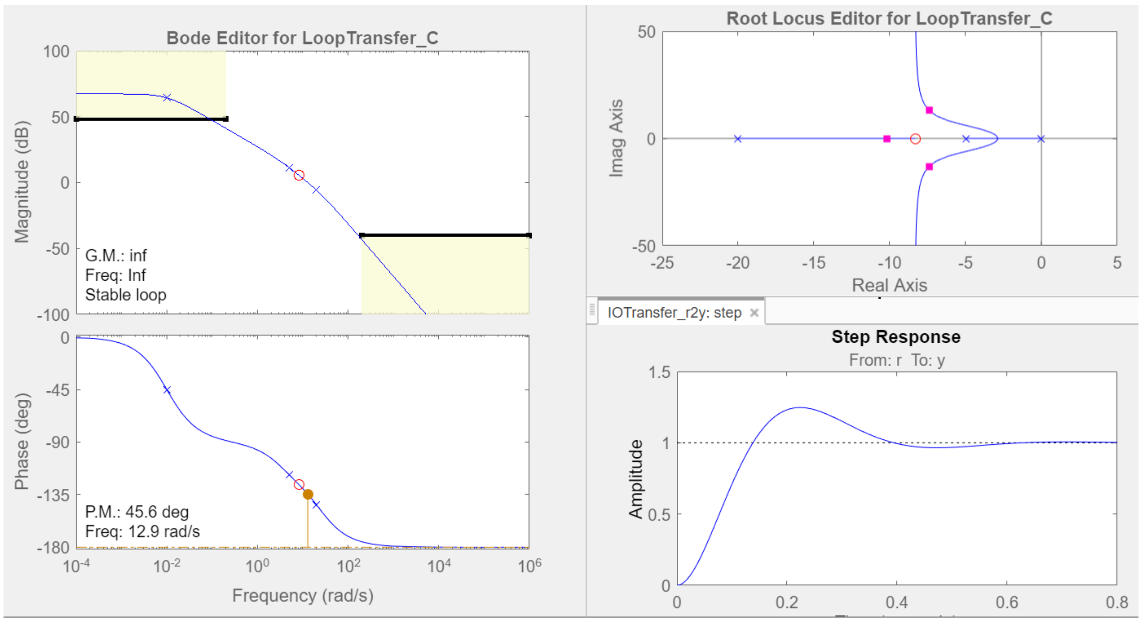


Figure 14: PD Controller

```
% PD Compensator where phase margin and desired high frequency gain satisfied
C (8) = 23*(1 + 0.12* s);
% PD Compensator high and low frequency gains are satisfied
C (9) = 110*(1 + 0.036* s);
```

Algorithm 13: Designed PD Compensators

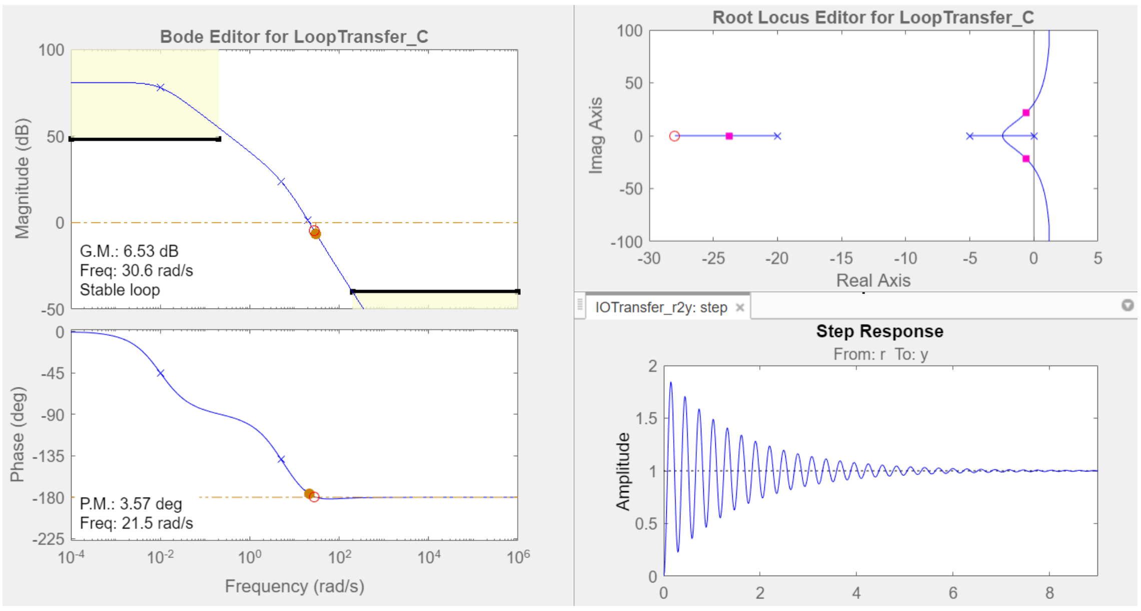


Figure 15: Second PD Controller

2.2.6 PID

The designed transfer functions can be seen in algorithm 14.The first PID controller in 16 satisfies the phase margin, low and high frequency gain criterion. However, the Kv value is not satisfied.Moreover, for the second figure in 17, the phase margin is not satisfied. Therefore, all conditions are again not met with the PID controller.

```
% PID Kv does not satisfy
C (10) = 12*(1 + 0.25* s)*(1 + 0.67* s)/s;
% PID phase margin does not satisfiy
C (11) = 100*(1 + 0.12* s) *(1 + 0.17* s)/s;
```

Algorithm 14: Designed PID Compensators

2.3 Verify and/or refine your design using MATLAB

You can check the following figures for each controller as in 18,19,20,21,22,23.

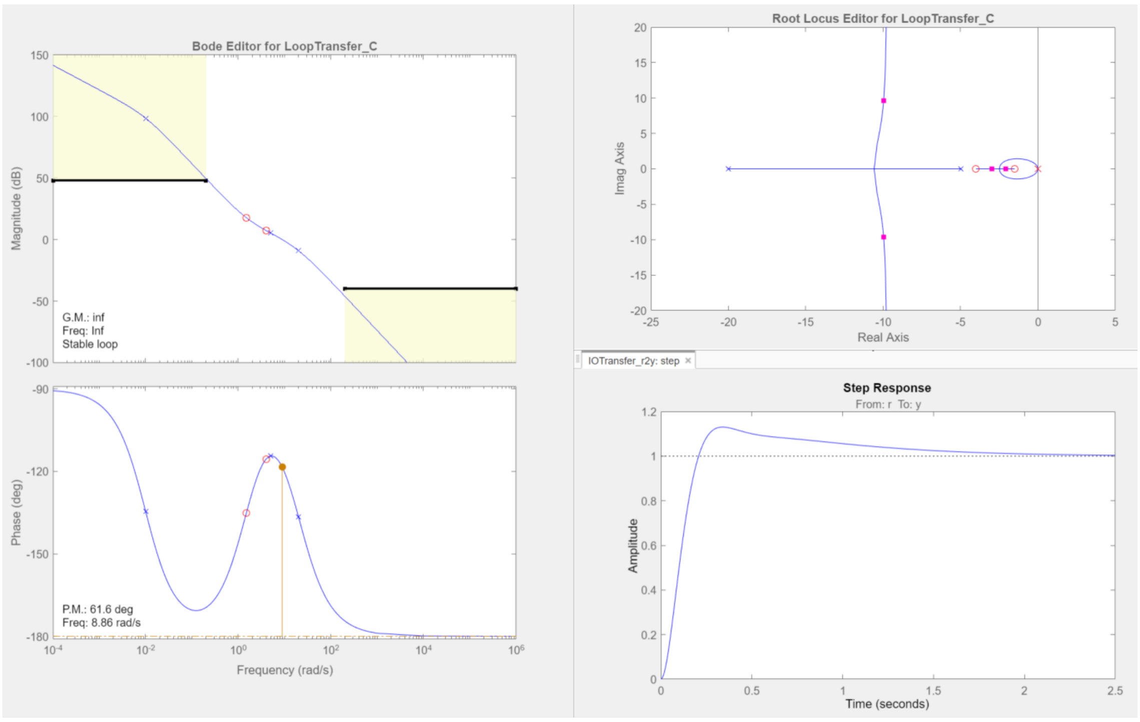


Figure 16: PID Controller

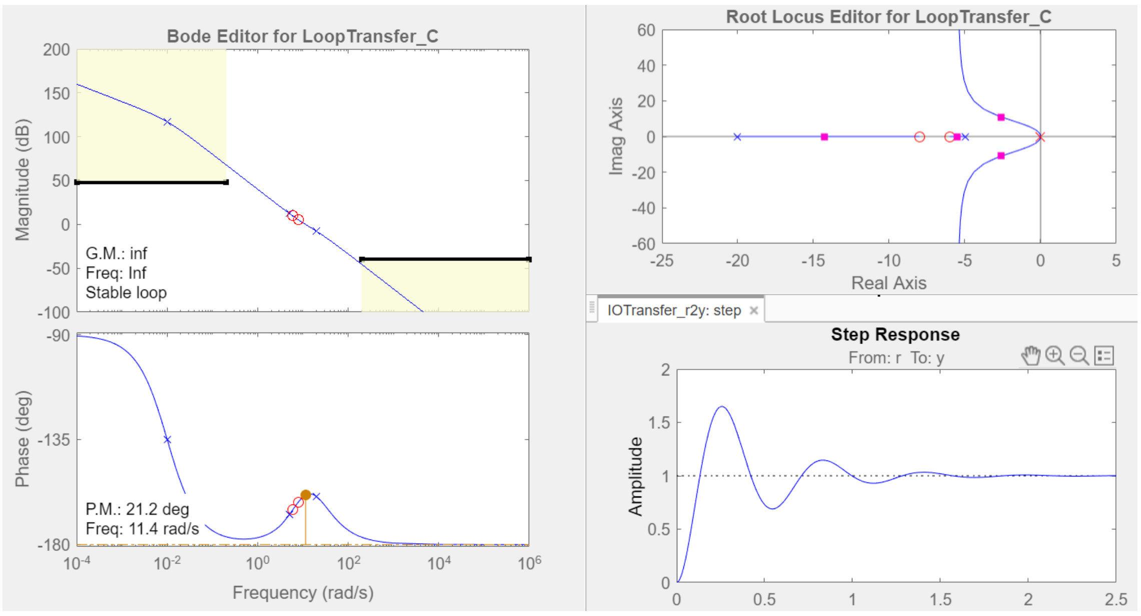


Figure 17: Second PID Controller

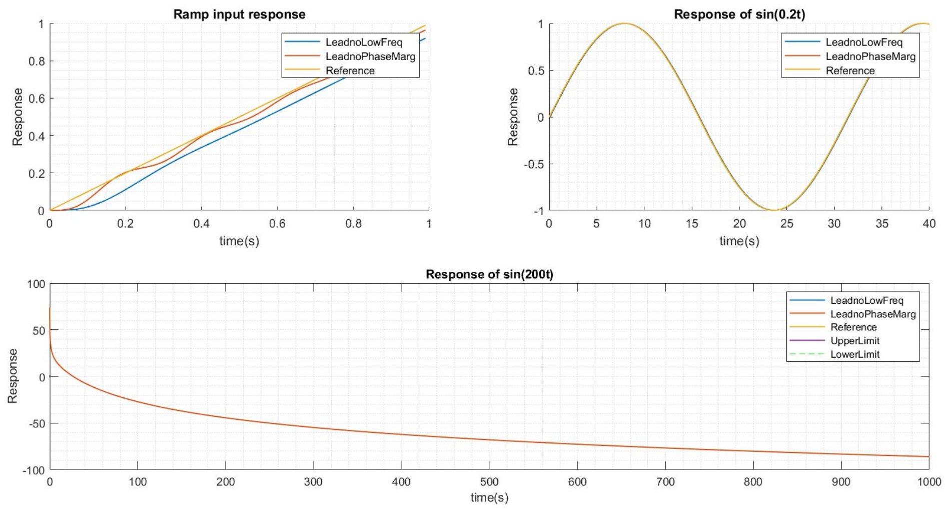


Figure 18: Check Lead Controller

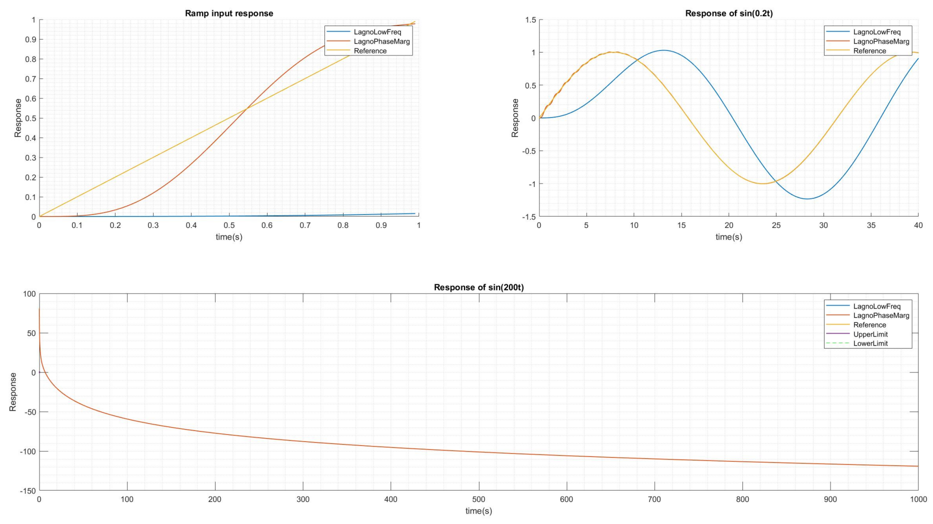


Figure 19: Check Lag Controller

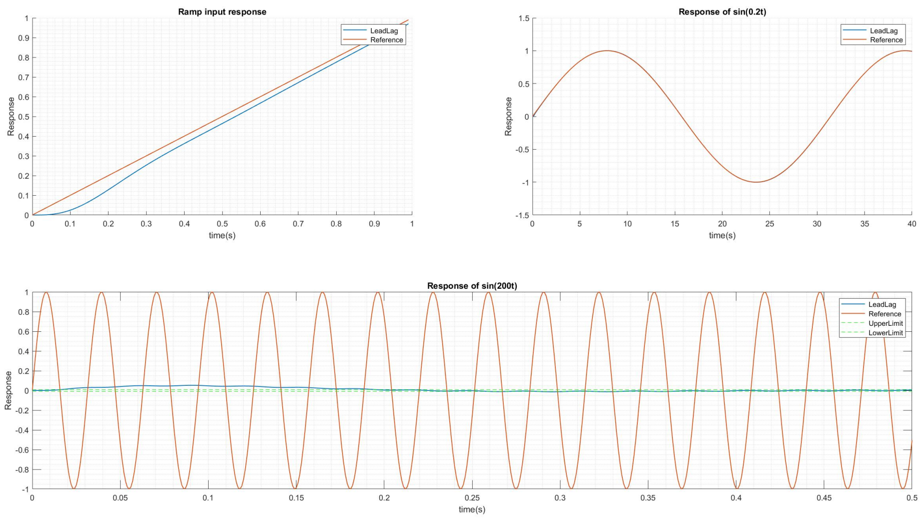


Figure 20: Check Lead-Lag Controller

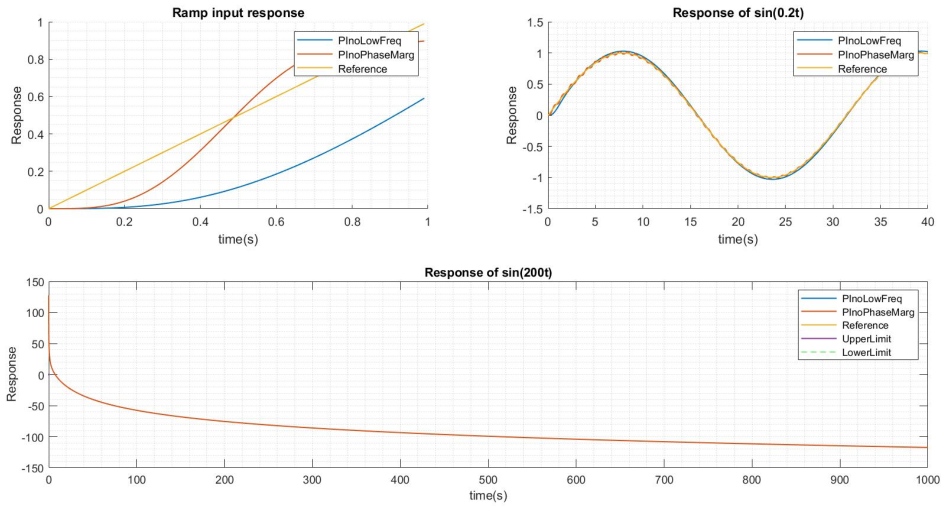


Figure 21: Check PI Controller

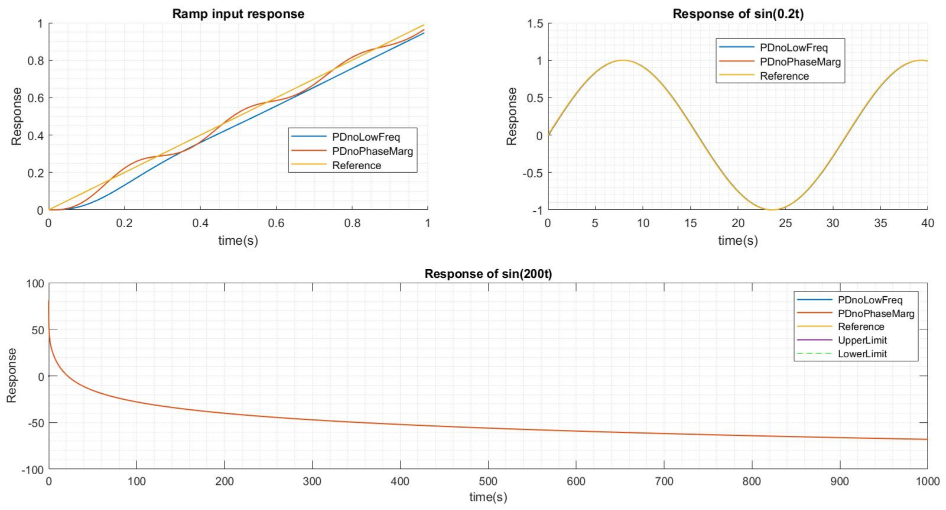


Figure 22: Check PD Controller

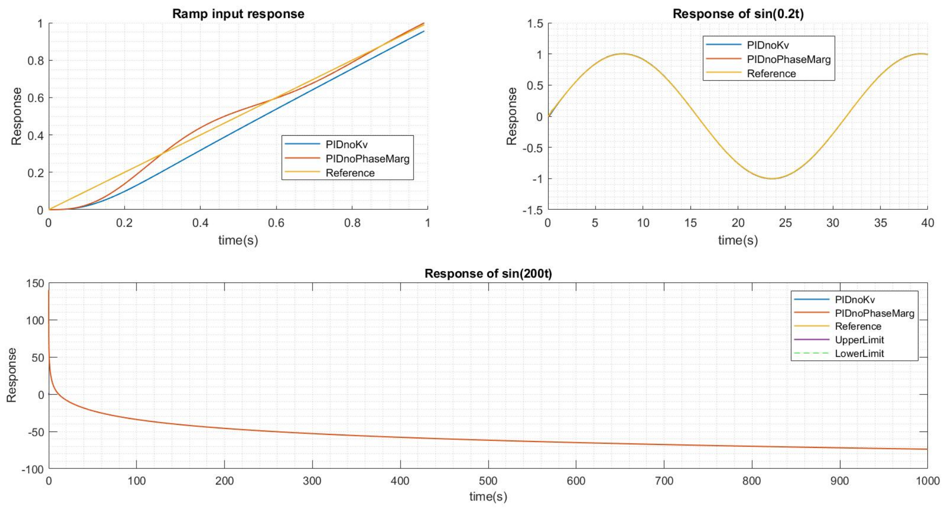


Figure 23: Check PID Controller

3 Find the least positive gain k

$$\hat{P}(s) = \frac{1}{10s+1}, \quad \hat{C}(s) = k, \quad \hat{F}(s) = 1 \tag{38}$$

- **Internally stable**: Theorem 1: The feedback system is internally stable iff there are no closed-loop poles in Res ≥ 0.

Firstly write the P,C,F as ratios of coprime polynomials(polynomials with no common factors):

$$P = \frac{N_P}{M_P}, \quad C = \frac{N_C}{M_C}, \quad F = \frac{N_F}{M_F} \tag{39}$$

The characteristic polynomial of the feedback system can be seen from equation 41

$$N_P N_C N_F + M_P M_C M_F \tag{40}$$

The closed-loop poles are the zeros of the characteristic polynomial. For our problem:

$$k + 10s + 1 \Rightarrow k + 1 \geq 0 \Rightarrow k > -1 \tag{41}$$

- $|e(\infty)| \leq 0.1$ **when** $r(t)$ **is the unit step and** $n = d = 0$:

This system is a type zero system. For a unit-step input, the steady-state error may be summarized as follows for a typical type 0 system:

$$e_{\text{ss}} = \frac{1}{1+K_p}, \quad \text{for type 0 systems} \tag{42}$$

and for our requirement:

$$-0.1 \leq e_{\text{ss}} = \frac{1}{1+K_p} \leq 0.1 \tag{43}$$

Then K_p needs to be bigger or equal to 9 ($K_p \geq 9$) since k needs to be bigger than -1. The static position error constant K_p is defined by:

$$K_p = \lim_{s\to 0} G(s) = \lim_{s \to 0} \frac{k}{10s+1} \tag{44}$$

where $G(s)$ is the transfer function. Therefore, Since $K_p \geq 9$ than $k \geq 9$.

- $\|y\|_\infty \leq 0.1$ **for all** $d(t)$ **such that** $\|d\|_2 \leq 1$ **when** $r = n = 0$: The closed loop transfer function $G(j\omega)$ is given as:

The H$_2$ norm is calculated using:

$$\|G(s)\|_2 = (\frac{1}{2\pi}\int_{-\infty}^{\infty} \underbrace{|G(j\omega)|^2}\, d\omega)^{\frac{1}{2}} \tag{45}$$

$$= \sqrt{\frac{1}{2\pi}\int_{-\infty}^{\infty}\left(\left(\frac{k(1+k)}{(1+k)^2+100w^2}\right)^2+\left(\frac{10kw}{(1+k)^2+100w^2}\right)^2\right)dw} \tag{46}$$

$$\|G(s)\|_2 = \sqrt{\frac{k^2}{20(k+1)}} \tag{47}$$

Then $\|G(s)\|_2 \geq 1$. Therefore, $k \geq 2001$

References

[1] Ozbay Hitay, Gumussoy Suat, Kenji Kashima, and Yutaka Yamamoto. *Frequency domain techniques for H control of distributed parameter systems*. SIAM, Society for Industrial and Applied Mathematics, 2018.