Hacettepe University - Computer Engineering

Advanced Robust Control MMÜ 749

Homework-1

Student Name Ayça KULA
Student ID: 202237285

Spring 2022

1 Derive G_d

From our Simulink model 1, it can be seen that the final transfer functions for plant and G_d can be written as:

$$G(s) = \frac{200}{10s+1}\frac{1}{(0.05s+1)^2}, \quad G_d(s) = \frac{100}{10s+1} \tag{1}$$

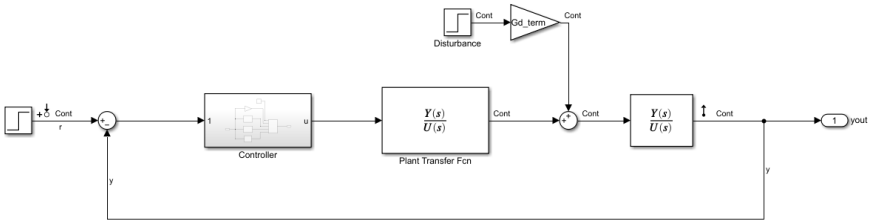


Figure 1: Simulink model for our configuration

2 Design PID Controller

Our control objectives are,

- Command tracking: $t_r < 0.3\text{sec}, M_p < 5\%$
- Disturbance rejection: response to unit step disturbance should stay within the range $[-1, 1]$ and should return to 0 as quickly as possible ($|y(t)| < 0.1$ after 3sec)
- Input constraints: $u(t)$ should remain within $[-1, 1]$

2.1 PID Design Requirements

Using the maximum overshoot M_p (percent), we can estimate the damping ratio:

$$\zeta = \frac{-\ln(\%\text{OS}/100)}{\sqrt{\pi^2 + \ln^2(\%\text{OS}/100)}} \tag{2}$$

Then, using the figure 7 and the damping ratio that we have obtained, we find beta as:

$$\beta = \tan^{-1}\frac{\sqrt{1-\zeta^2}}{\zeta} \tag{3}$$

Phase margin is defined as:

$$\Phi_M = \tan^{-1} \frac{2\zeta}{\sqrt{-2\zeta^2 + \sqrt{1+4\zeta^4}}} \tag{4}$$

These values can be found from the matlab script given in algorithm 2.

```
%% PID Controller

% Transient-Response Specifications

% Overshoot less than 5%
percent_Mp = 5;
damping_ratio = -log(percent_Mp/100)/(sqrt(pi^2+(log(percent_Mp/100))^2));
% For a desirable transient response,damping ratio must be between 0.4 and 0.8.

% Find beta using damping ratio
beta = atan(sqrt(1-damping_ratio^2)/damping_ratio);

% Rise time -- Should be less than 0.3
t_rise = 0.2;
w_d = (pi - beta)/t_rise;
w_n = w_d/(sqrt(1-damping_ratio^2));

% Settling time (usually 2% or 5%)
t_s_2per = 4/(damping_ratio*w_n); % --> For 2% criterion
t_s_5per = 3/(damping_ratio*w_n); % --> For 5% criterion

% We shall see that the maximum overshoot and the rise time conflict with
% each other!!

% Bandwidth is related to natural frequency by
bandwith = w_n*sqrt((1-2*damping_ratio^2)+...
sqrt(4*damping_ratio^4-4*damping_ratio^2+2));

% Desired Phase margin
PM = atan(2*damping_ratio/(sqrt(-2*damping_ratio^2+sqrt(1+4*damping_ratio^4))));
PM_approx = 100*damping_ratio; % Approximation for phase margin
PM_deg = rad2deg(PM);

% % GM, PM , Phase and gain crossover frequency
% [Gm,pm,wcp,wcg] = margin(sys);
% GmdB = 20*log10(Gm);
% results = [GmdB pm wcp wcg];
```

Algorithm 1: Design requirements for PID Controller

Table 1: Design reqirements values

Parameters	Values
Percent Overshoot	5
Damping Ratio	0.6901
beta	0.8092
Rise time	0.2
Bandwith	16.5017
Phase Margin (deg)	64.6253

And the final desired values are given in the table 1.

2.2 Design PID with sisotool

We have defined our system transfer functions in Matlab. Now, we can use the sisotool in order to design our PID controller.

1. Call *sisotool(Plant)* from command window.
2. Edit the control architecture [1] given in figure 7.In this figure, G2 is defined as the static gain zero.

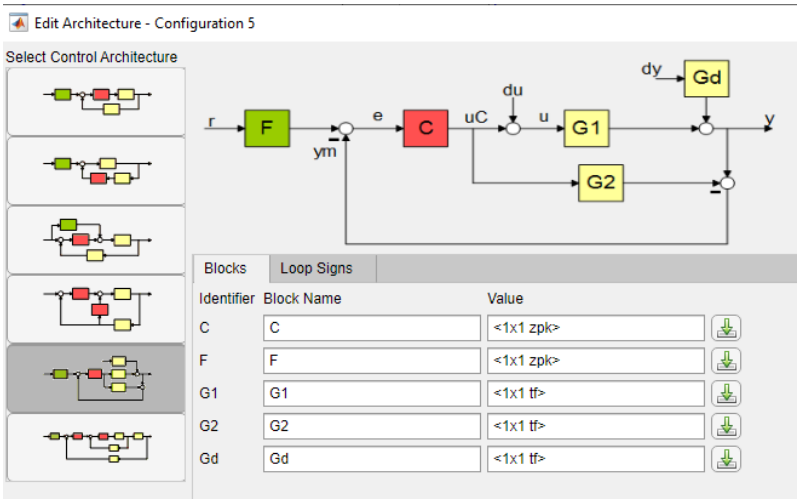


Figure 2: General control configuration

3. By using compansator editor zeros,poles etc. are added (For integrator add pole). All the K1,K2,K3 values are implemented and looked for their open-loop bode, nyquist, and closed loop step responses. Moreover, in order to design our pid controller we go into "pid tunning" in "Tunning methods" section.Then, we choose the controller method as "PID" and the chose the design mode as "frequency". We chose the bandwith as 17 and the phase margin as 64 as in 3.

   As seen, in figure 4 the design requirements are not satisfied.Due to this reason, a prefilter has been added in order to meet the design requirements.

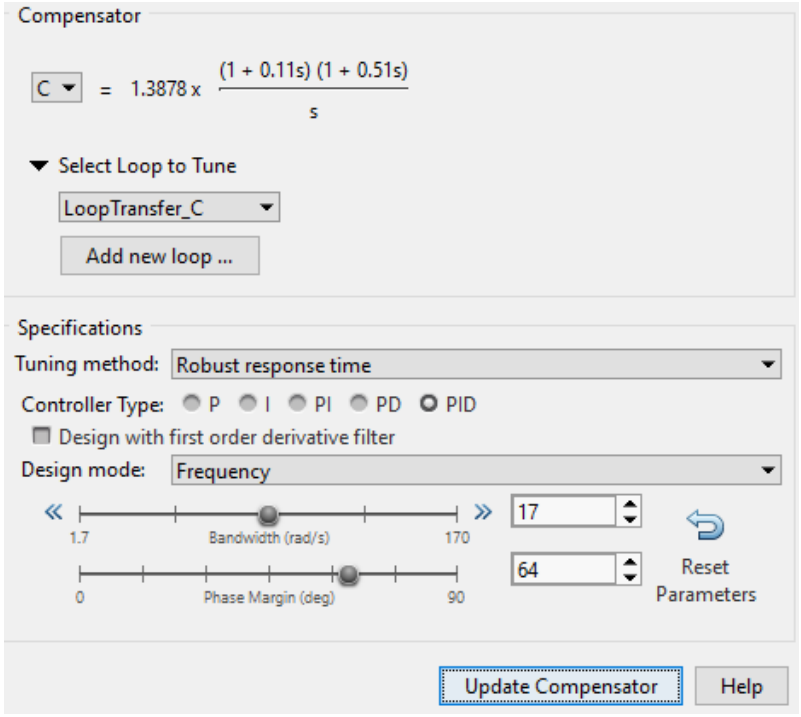


Figure 3: PID design requirements

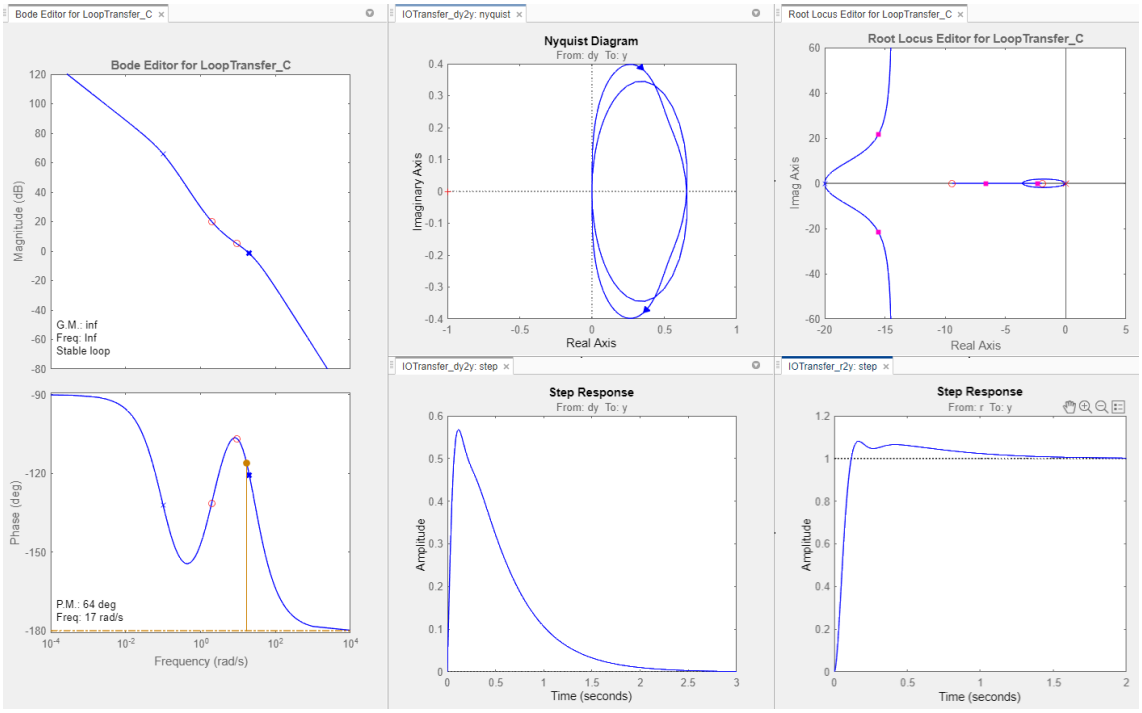


Figure 4: PID without Prefilter

2.3 PID Design with prefilter

In order to decrease overshoot we have added a pre-filter. In order to add a prefilter we include the "F" block in Sisotool in the compensator editor. And

we find the optimum value by adding a real pole -7 as in 5. Moreover, by adding this pole the design requirements are satisfied as seen in 6.

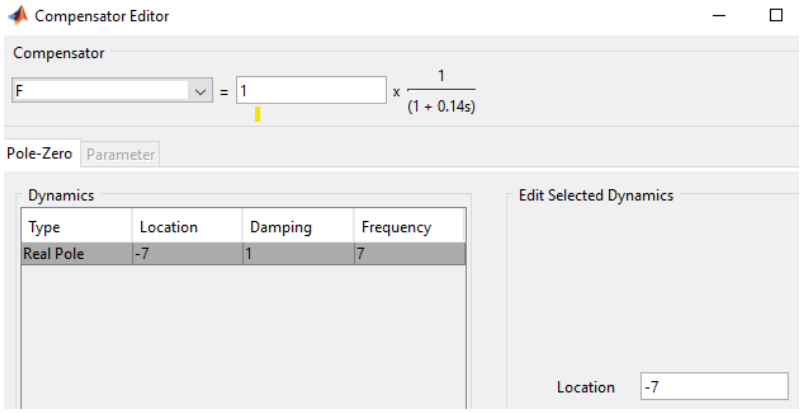


Figure 5: Add prefilter to the system

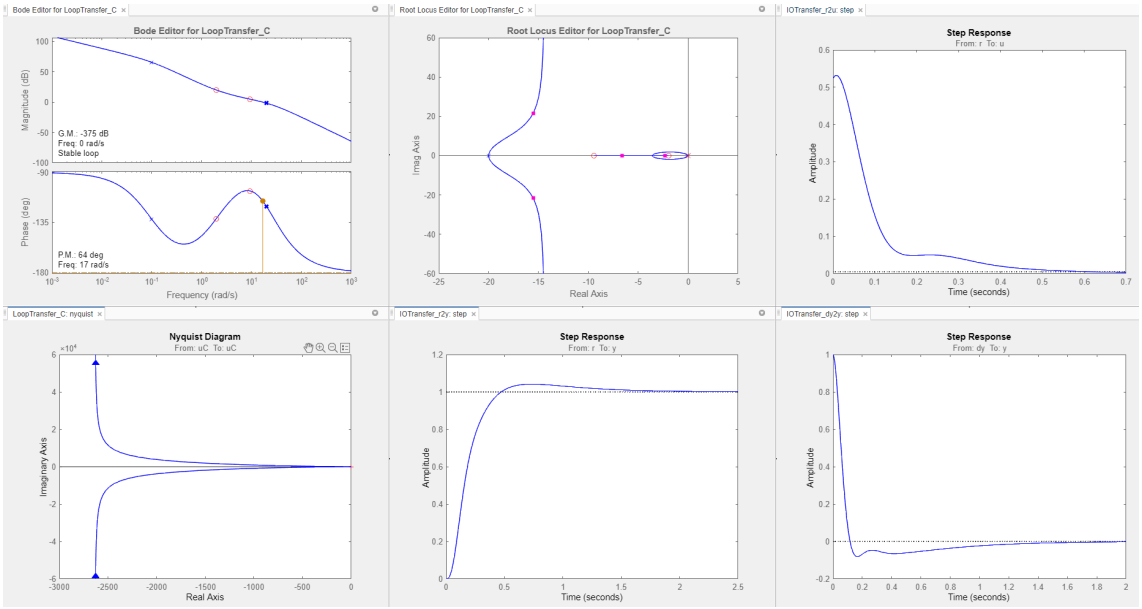


Figure 6: PID with prefilter

3 Solution by "Skogestad"

Step 1 Initial design
[2] states that a reasonable initial loop shape $L_{min}(s)$ is then one that just satisfies the condition

$$|L_{\min}| \approx |G_d| \tag{5}$$

Therefore,

$$|L_{\min}| = |G_d| = \left|\frac{100}{10s+1}\right| \tag{6}$$

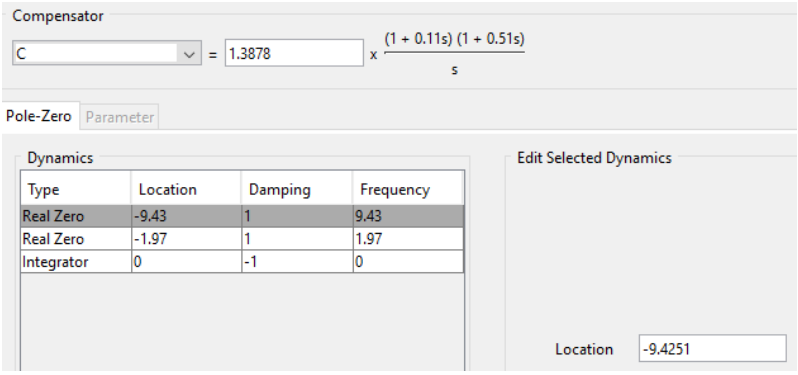


Figure 7: Tuned PID parameter values

And the controller is,

$$|K_{\min}| \approx \left|G^{-1}G_d\right| \tag{7}$$

$$K(s) = G^{-1}L_{\min} = (0.05s+1)^2 0.5 \tag{8}$$

Since this transfer function has more zeros than poles, this controller is not proper. The term $(0.05s+1)^2$ will effect at $1/0.05 = 20rad/s$.

$$\omega_c = 10\text{rad/s} < 20\text{rad/s} \tag{9}$$

The ω_c which is the desired gain crossover frequency is smaller than our frequency. Therefore, we may replace it by a constant gain of 1 resulting in a proportional controller.

$$K_1(s) = 0.5 \tag{10}$$

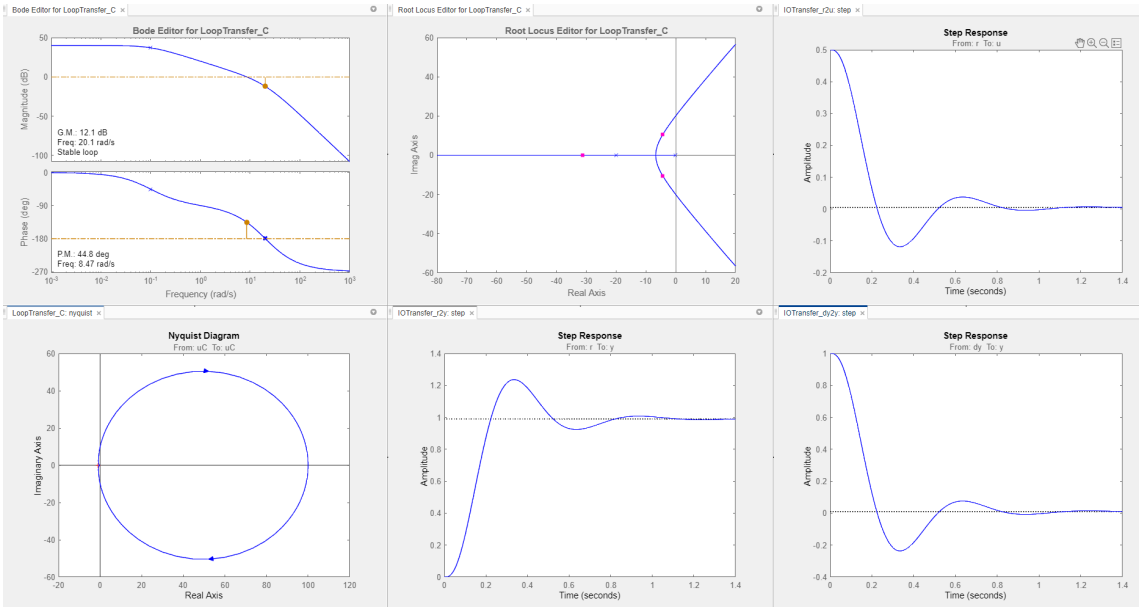


Figure 8: K1 Controller

Step 2 More gain at low frequency
In order to improve low-frequency performance (e.g. to get zero steady-state offset), an integral action is added at low frequencies

$$|K| = \left| \frac{s + \omega_I}{s} \| G^{-1} G_d \right| \tag{11}$$

where ω_I is the frequency up to which the term is effective. "For performance we want large gains at low frequencies, so we want ω_I to be large, but in order to maintain an acceptable phase margin(which is 44.7 degree for controller K1) the term should not add too much negative phase at frequency ω_c, so ω_I should not be too large. A reasonable value is $\omega_I = 0.2\omega_c$ for which the phase contribution from $(s + \omega_I)/s$ is $arctan(1/0.2) - 90° = -11\circ$ at ω_c."

$$K_2(s) = 0.5\frac{s+2}{s} \tag{12}$$

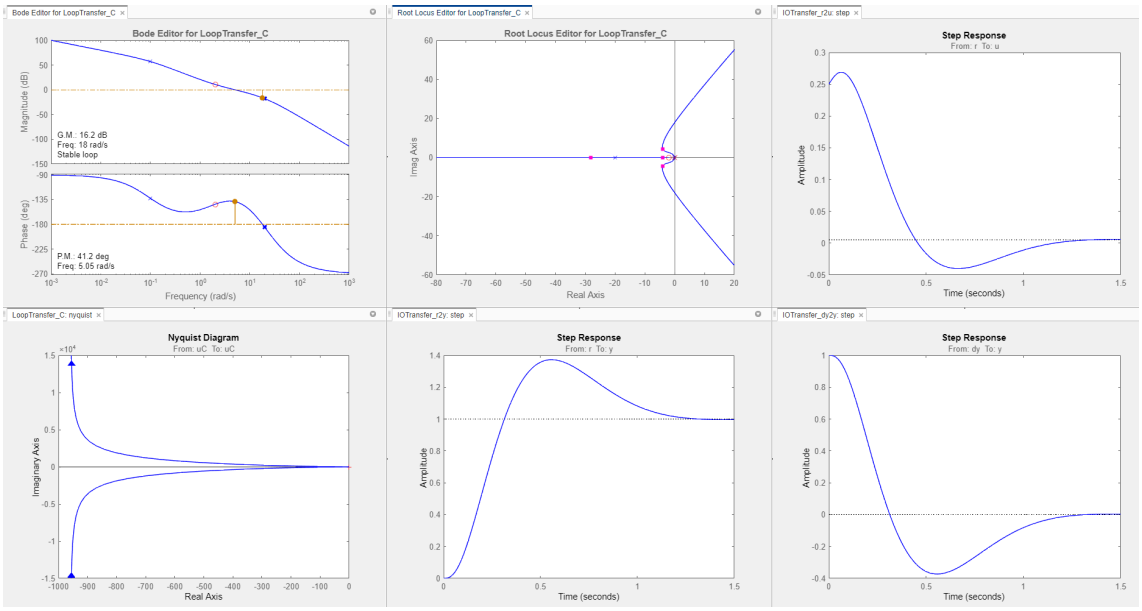


Figure 9: K2 Controller

Step 3 High-frequency correction
Controller is supported with "derivative action" in order to increase the phase margin and improve the transient response. Therefore, we multiply $K_3(s)$ by a a lead-lag term.

$$K_3(s) = 0.5\frac{s+2}{s}\frac{0.05s+1}{0.005s+1} \tag{13}$$

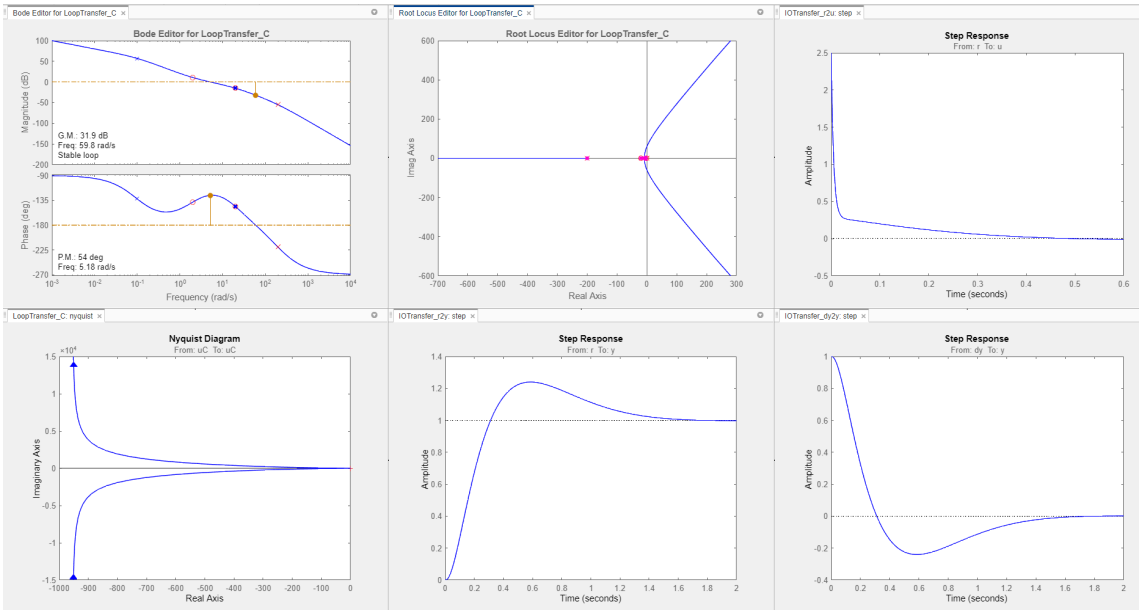


Figure 10: K3 Controller

4 Compare the Performance of Controllers

Now, lets examine all responses of both controller and system under step and/or disturbance.

4.1 Disturbance Responses

Firstly, the disturbance responses are examined. And it can be said that the PID controller is the best for rejecting the disturbance as seen in 11. For disturbance rejection a good choice for the controller is one which contains the dynamics of the disturbance and inverts the dynamics of the inputs (at least at frequencies just before crossover).Adding an integrator yields zero steady-state offset to a step disturbance. In order to improve the settling time and to reduce the steady-state offset, increasing the loop gain at low frequencies is a solution.

4.2 Step Responses in Sisotool

Here, we can talk about the damping factor by looking at the oscillations. We know that if damping ratio is smaller than 1, then the system will oscillate when "kicked" by a transient such as a step function.It can be clearly seen that K1, K2, K3 controllers are underdamped. And, PID is closed to being critically damped as can be seen from 12.Also, by looking at this figure we will have an insight about when does the system settles; how much is the overshoot,steady-state error; how fast does it rise etc.

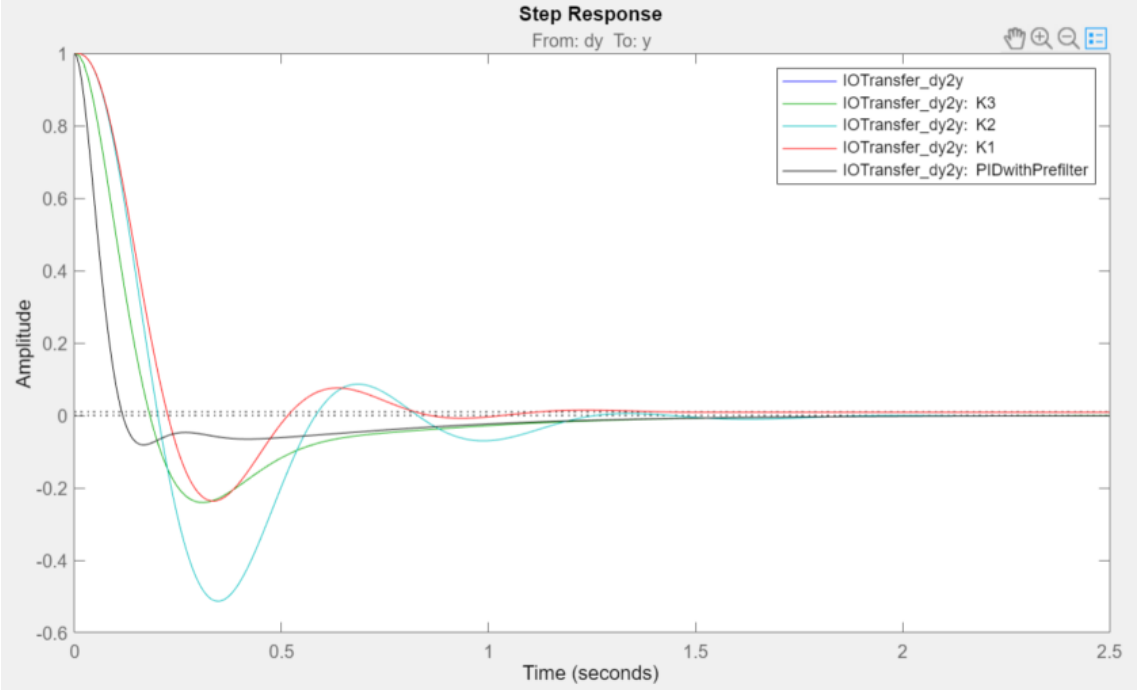


Figure 11: Disturbance Responses for different Controller Configurations

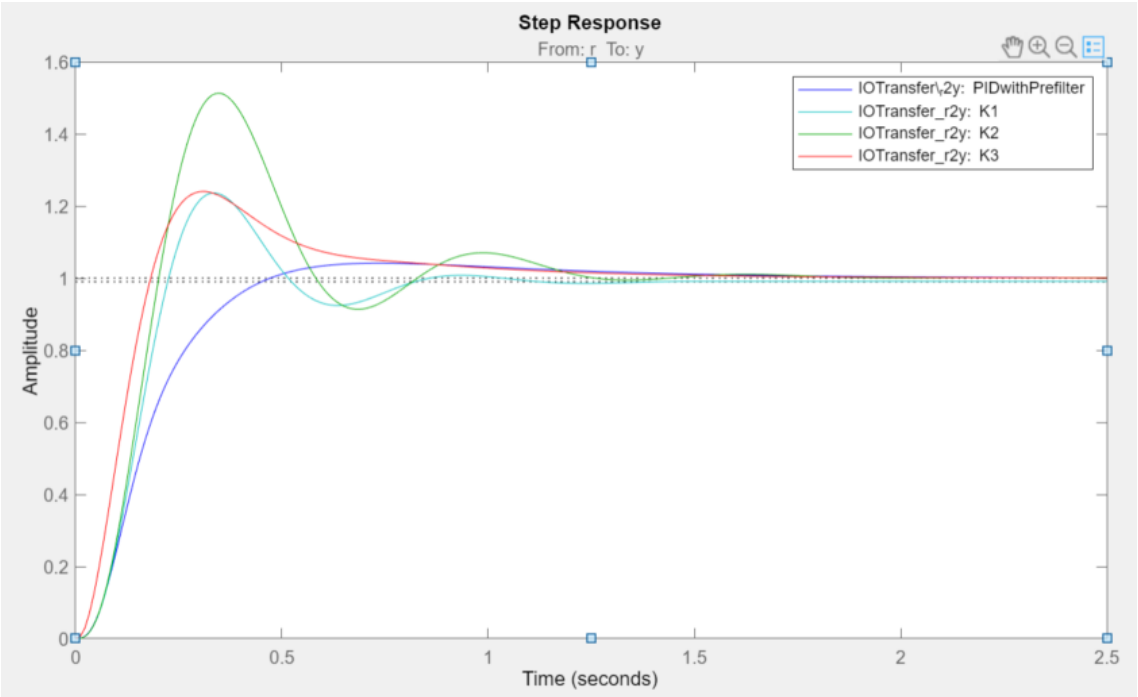


Figure 12: Step Responses For different Controller Configurations

4.3 Step Responses for both Controller Output and System Output

My plots are taken from Simulink.And for figure 13 same comments can be done as in previous figure.

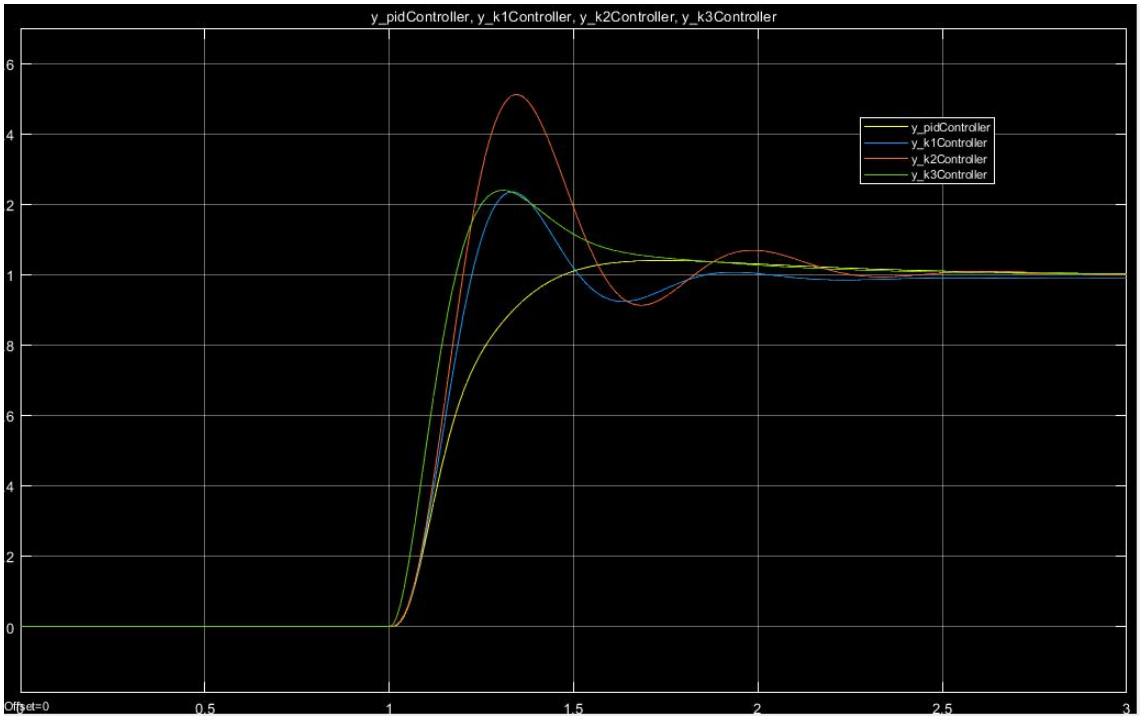


Figure 13: Output Step Responses For different Controller Configurations

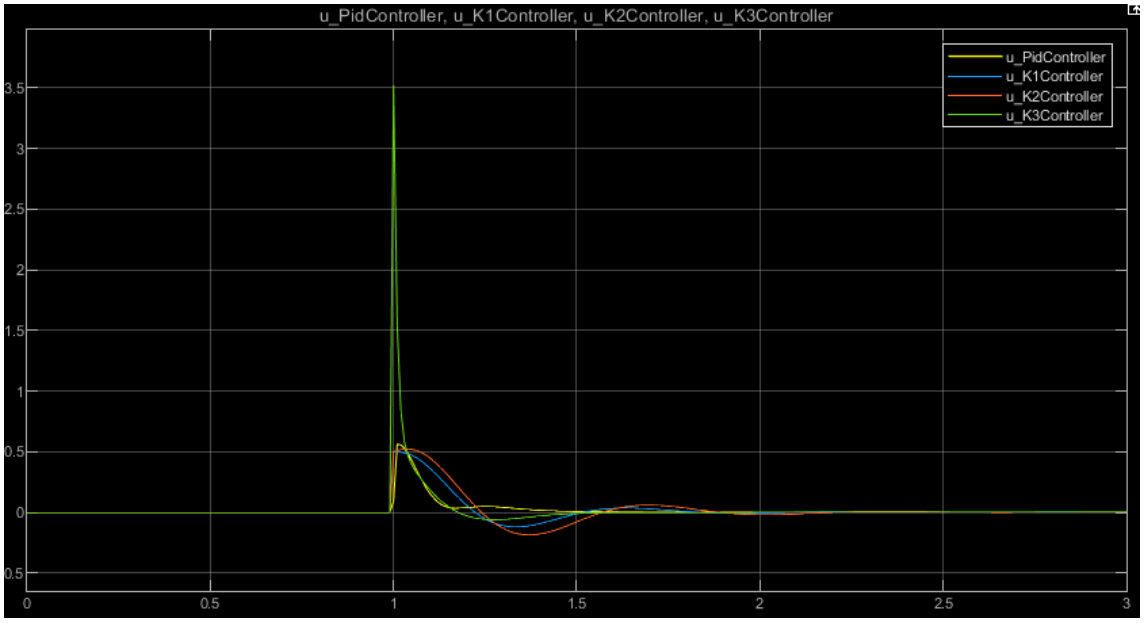


Figure 14: Controller Output Step Responses For different Controller Configurations

4.4 Disturbance Input Responses for both Controller Output and System Output

In this part, step input to reference is eliminated and only step input to disturbance is considered.

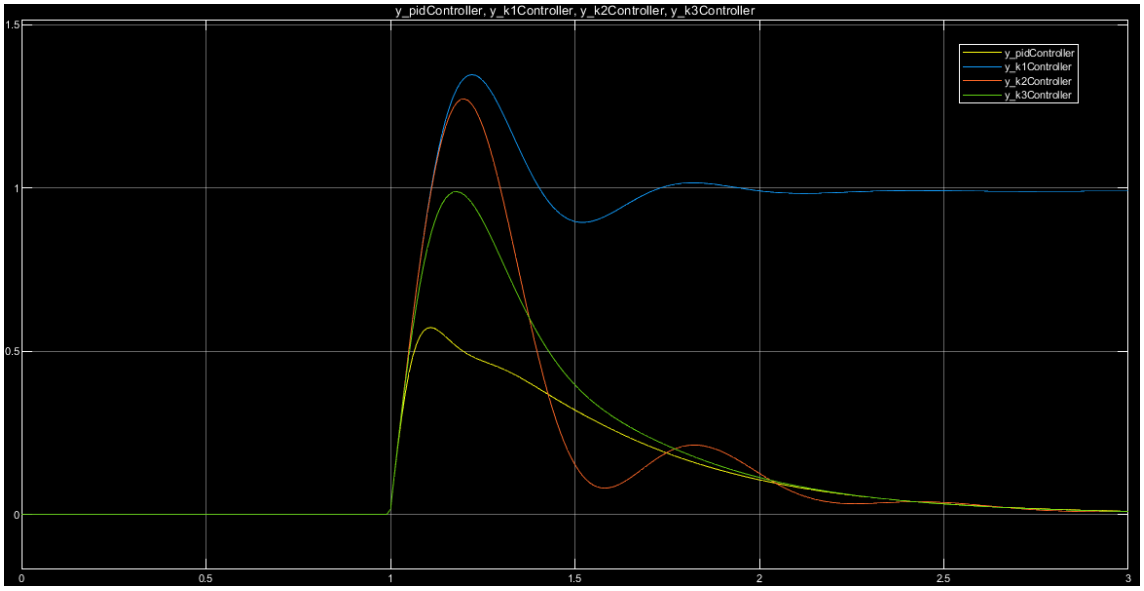


Figure 15: Output Disturbance Responses For different Controller Configurations

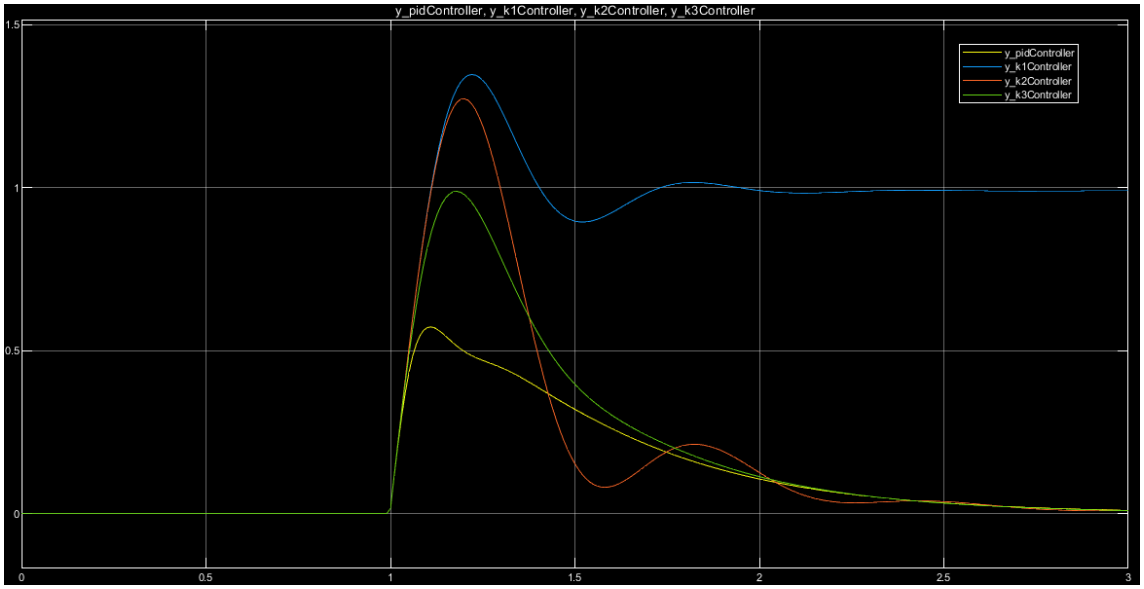


Figure 16: Controller Output Disturbance Responses For different Controller Configurations

4.5 Both Input Responses for both Controller Output and System Output

Response of y when r and d act simultaneously on the system is drawn in 17.Moreover, response of u when r and d act simultaneously on the system is drawn in 18

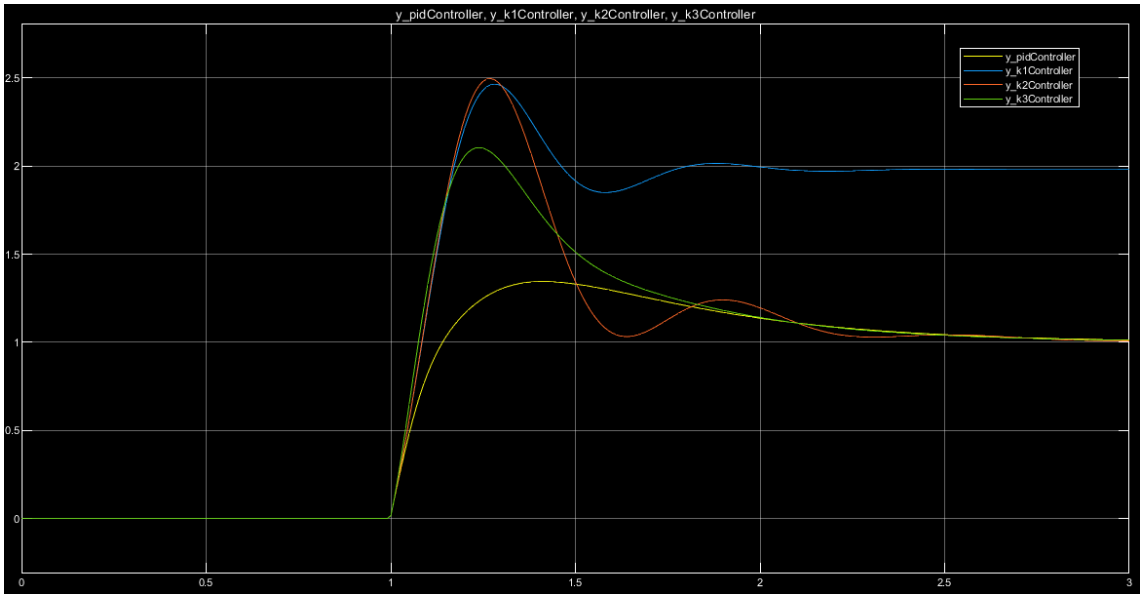


Figure 17: Output Step Responses For different Controller Configurations

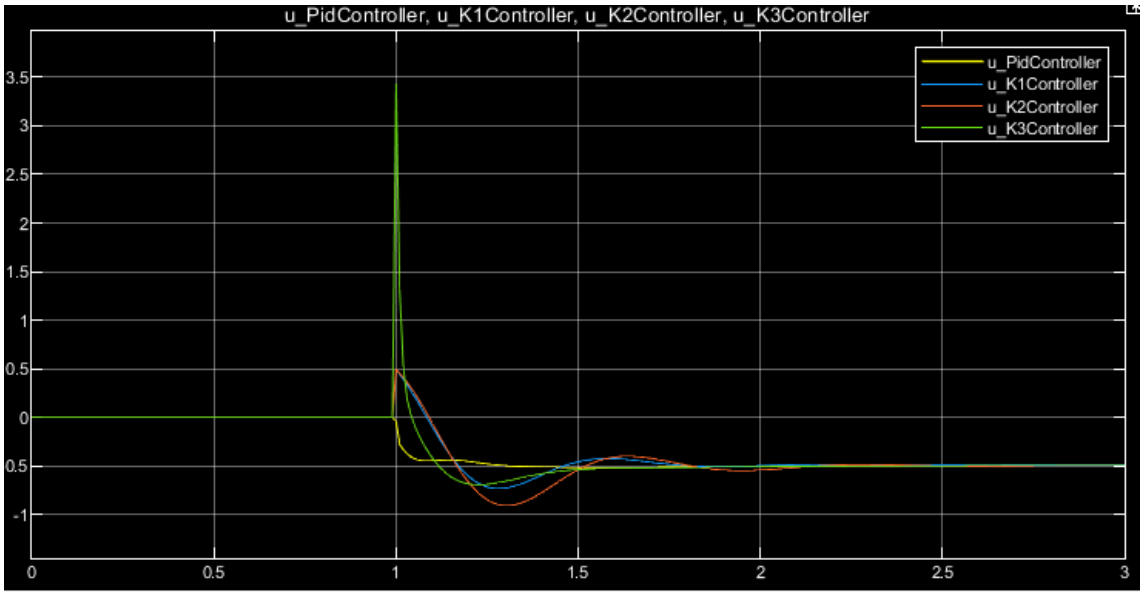
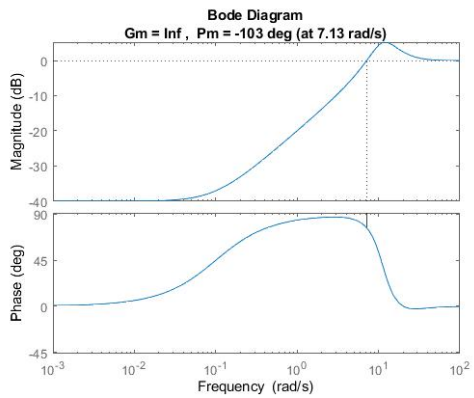


Figure 18: Controller Output Step Responses For different Controller Configurations

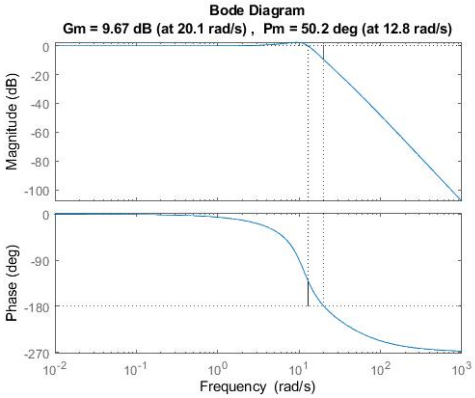
5 Draw S, T, KS, and SGd

Sensitivity is important because we ideally want S small.It is sufficient to consider just its magnitude. Therefore, for these bode plots it is alright not to look at the phase diagrams. If you look at the plots given in 22, you will see that sum of sensitivity and complementary sensitivity is equal to 1. For low frequencies, I want sensitivity to be small in order to reduce the error. Moreover, in high frequencies I want small complementary sensitivity function for noise attenuation. There is also a resonance peak which is caused by the poorly damped poles of the loop transfer function. Output can track reference only frequencies lesser than the crossover

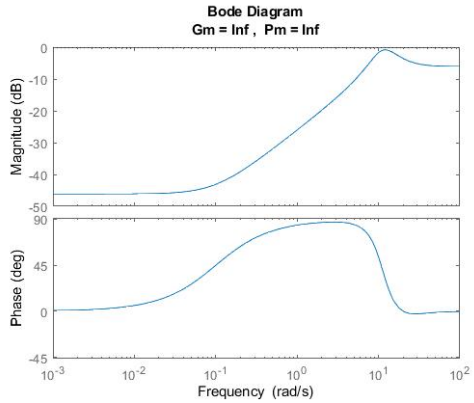
frequency and it can rise fast as those frequencies allow.



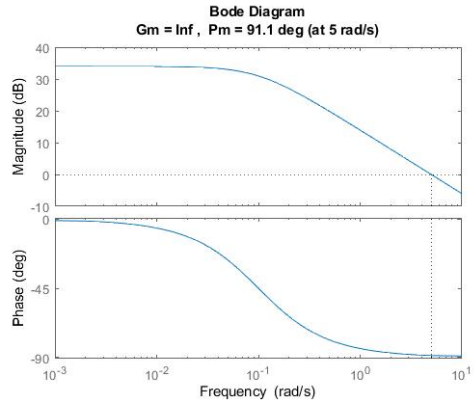
(a) Sensitivity



(b) Complementary Sensitivity

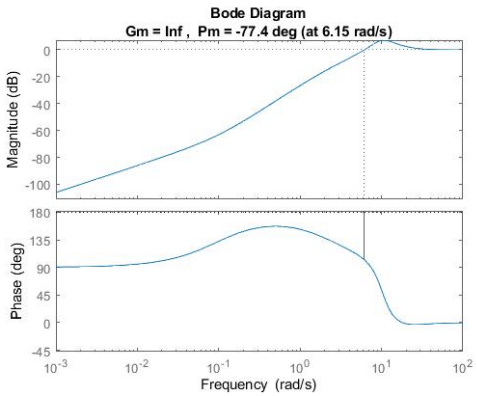


(c) KS

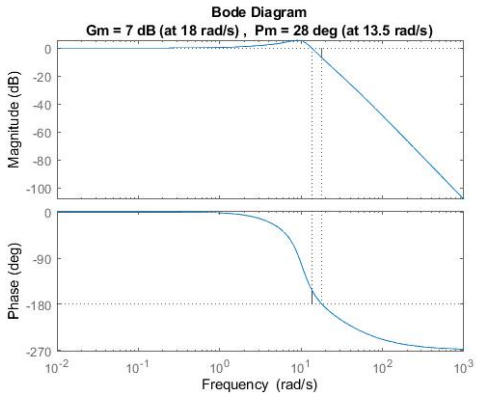


(d) SGd

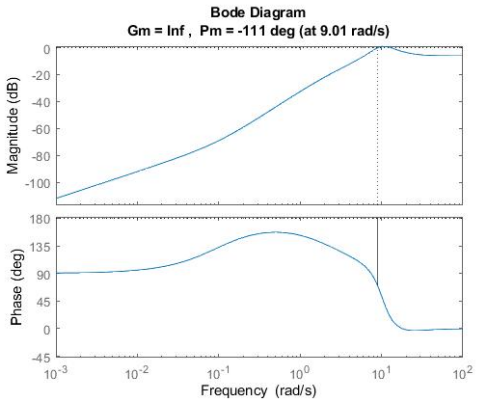
Figure 19: Sensitivity, Complementary Sensitivity, KS, SGd for K1



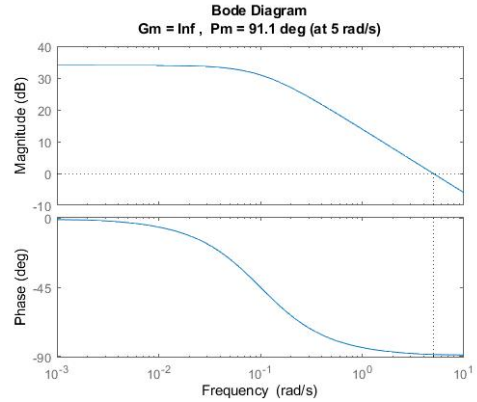
(a) Sensitivity



(b) Complementary Sensitivity

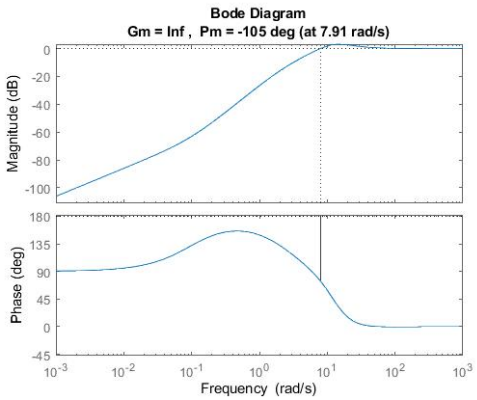


(c) KS

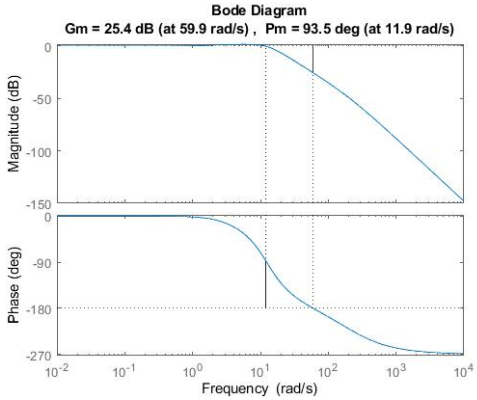


(d) SGd

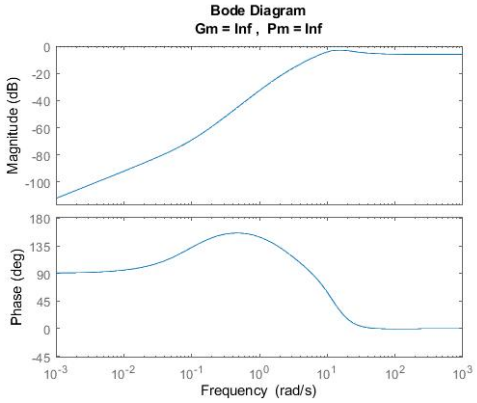
Figure 20: Sensitivity, Complementary Sensitivity, KS, SGd for K2



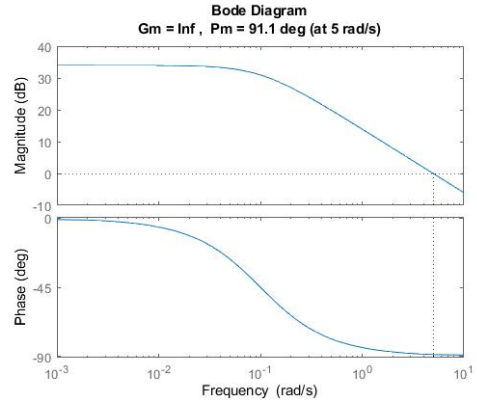
(a) Sensitivity



(b) Complementary Sensitivity

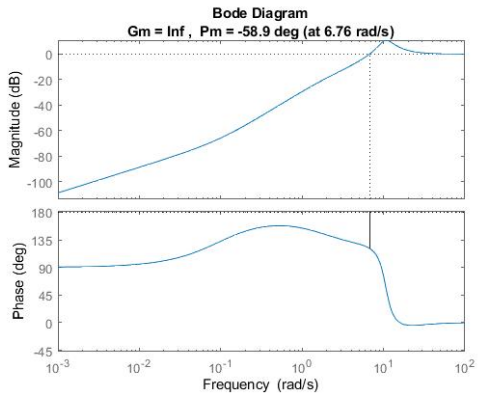


(c) KS

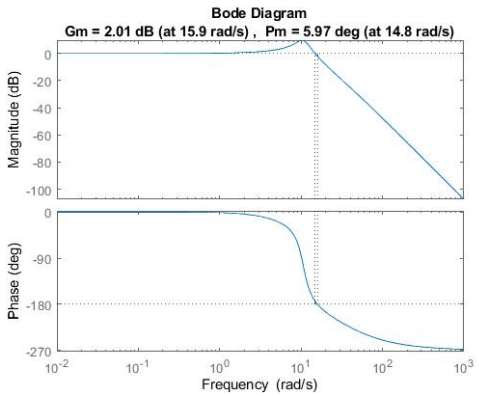


(d) SGd

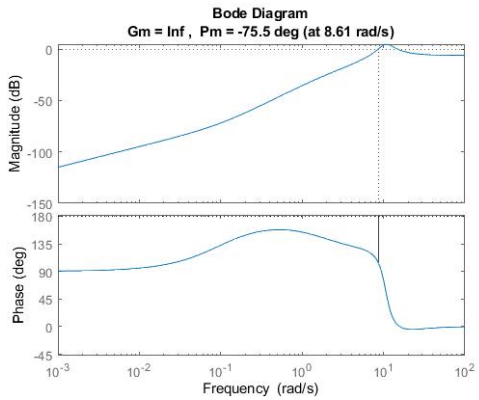
Figure 21: Sensitivity, Complementary Sensitivity, KS, SGd for K3



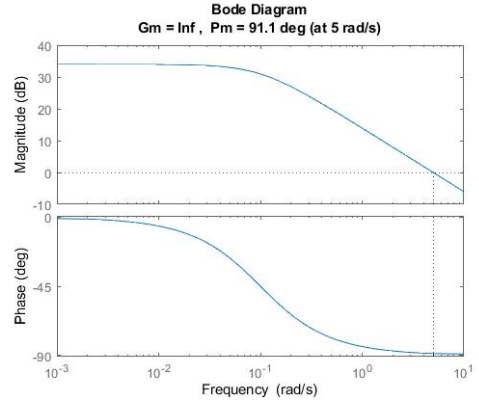
(a) Sensitivity



(b) Complementary Sensitivity



(c) KS

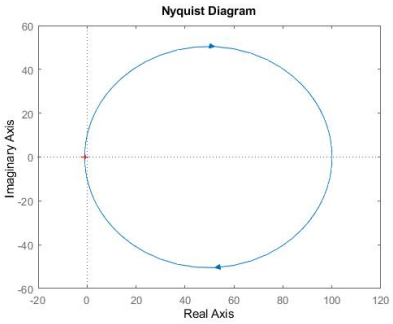


(d) SGd

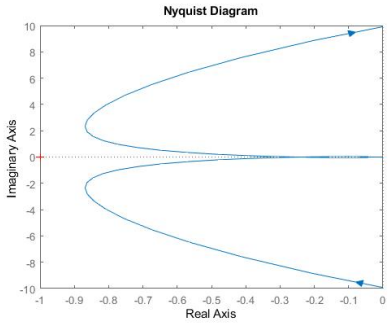
Figure 22: Sensitivity, Complementary Sensitivity, KS, SGd for PID

6 Nyquist plots

The Nyquist stability criterion relates the open loop frequency response to the number of closed loop poles of the system in the right hand side plane. Closed-loop instability occurs if $L(j\omega)$ encircles the critical point -1. From all plots given below, we can see that all the systems are stable since non of them encircles -1. In order to see the non-encirclement, zoom nyquist plots are more suitble to interpret.

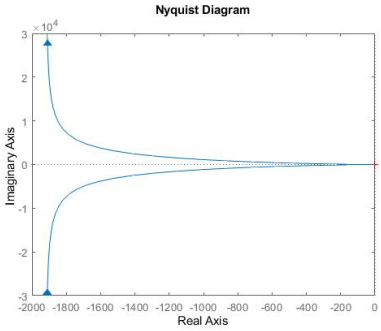


(a) Nyquist Diagram for K1

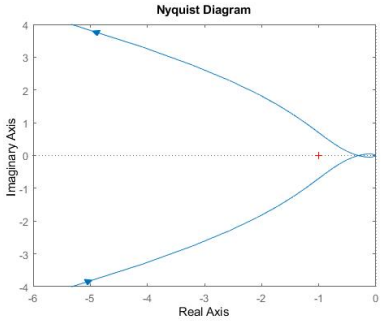


(b) Zoomed nyquist Diagram for K1

Figure 23: Nyquist Diagram for K1.

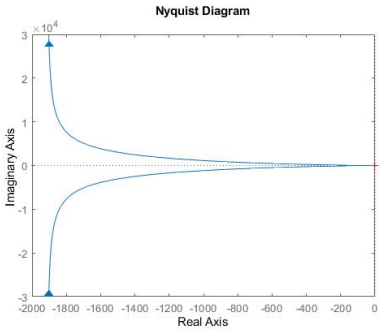


(a) Nyquist Diagram for K2

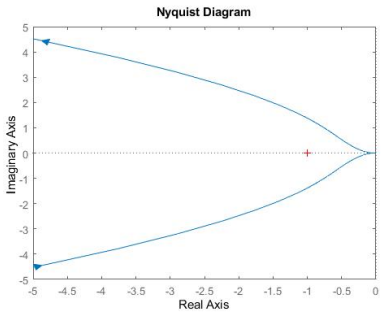


(b) Zoomed nyquist Diagram for K2

Figure 24: Nyquist Diagram for K2.



(a) Nyquist Diagram for K3

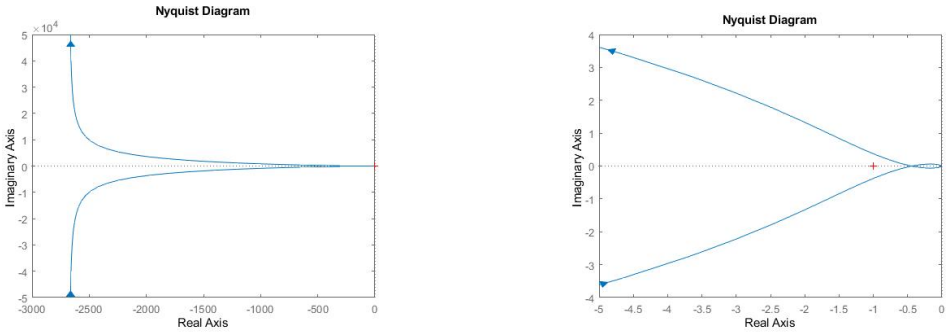


(b) Zoomed nyquist Diagram for K3

Figure 25: Nyquist Diagram for K3.

7 Obtain the generalized plant manually

The generalized plant model is given as 27.



(a) Nyquist Diagram for PID

(b) Zoomed nyquist Diagram for PID

Figure 26: Nyquist Diagram for PID.

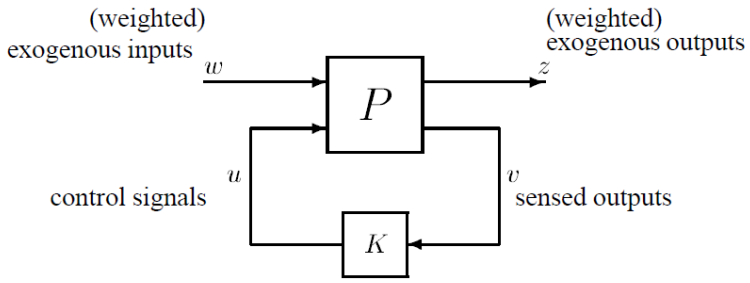


Figure 27: General control configuration for the case with no model uncertainty

If we want to find the generalized plant. Our first step is to identify the signals for the generalized plant:

- w is called generalized disturbance

$$w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} d \\ r \end{bmatrix} \tag{14}$$

- z is called controlled variable. (In our example, z = e.)

$$z = e = y - r = Gu + G_d d - r = Gu + IG_d w_1 - Iw_2 \tag{15}$$

- u denote the output signal of the controller,the so-called control input.
- v denote the signal that enters the controller, the so-called measurement output.

$$v = r - y = r - Gu - G_d d = Iw_2 - IG_d w_1 - Gu \tag{16}$$

- P which represents the transfer function matrix from

$$\begin{bmatrix} w & u \end{bmatrix}^T \text{ to } \begin{bmatrix} z & v \end{bmatrix}^T \tag{17}$$

which is:

$$P = \begin{bmatrix} IG_d & -I & G \\ -IG_d & I & -G \end{bmatrix} \tag{18}$$

Note that P does not contain the controller

```
clc;clear;close all;
s = tf('s');
% Uses the Mu-toolbox
G=nd2sys(1,conv([10 1],conv([0.05 1],[0.05 1])),200); % Plant is G.
Gd = 100/(10*s + 1);
Gd = nd2sys(1, cell2mat(Gd.Denominator),100); % Disturbance plant is Gdp.
% M=1.5; wb=10; A=1.e-4; Wp = nd2sys([1/M wb], [1 wb*A]); Wu = 1; % Weights.
M=1.5; wb=10; A=1.e-4; Ws = nd2sys([1/M wb], [1 wb*A]); Wks = 1; % Weights.

%Creating the generalized plant P
systemnames = 'G Gd Ws Wks';
inputvar = '[d(1); r(1); u(1)]'; %all inputs are scalar
                            %r(2) would be a 2dim signal
outputvar = '[Ws; Wks; r-G-Gd]';
input_to_G = '[u]';
input_to_Gd = '[d]';
input_to_Ws = '[r-G-Gd]';
input_to_Wks = '[u]';
sysoutname = 'P';
cleanupsysic = 'yes';
sysic
```

Algorithm 2: Design generalized plant in matlab

8 Read example 2.11 of Skogestad and design a H inf mixed sensitivity controller

H infinity controller has been designed in Matlab using 3. And the resulting frequecy-magnitude figure for inverse performance weight and sensitivty function is given in 28.

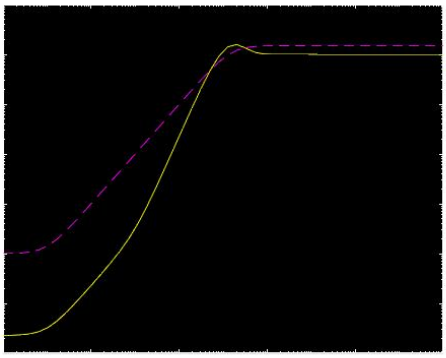


Figure 28: Inverse of performance weight and resulting sensitivity function

```
% Find H-infinity optimal controller:
nmeas=1; nu=1; gmn=0.5; gmx=20; tol=0.001;
[K,CL,gopt] = hinfsyn(P,nmeas,nu,gmn,gmx,tol);
w = logspace(-4,6,50);
CLw = vsvd(frsp(CL,w));
figure(1); vplot('liv,m',CLw);
title('singular values of weighted closed loop system');

[type,out,in,n] = minfo(G);
I = eye(out);
S = minv(madd(I,mmult(G,K))); %sensitivity
T = msub(I,S); %complementary sensitivity
KS = mmult(K,S); %input to G
GK = mmult(G,K); %loop transfer function

%singular values as a function of frequency
Sw = vsvd(frsp(S,w));
Tw = vsvd(frsp(T,w));
Kw = vsvd(frsp(K,w));
KSw = vsvd(frsp(KS,w));
GKw = vsvd(frsp(GK,w));

%Plot singular value plots
figure(2); vplot('liv,lm',Sw,'-',Tw,'--',GKw,'-.');
title('\sigma(S(jw)) (solid), \sigma(T(jw)) (dashed) and \sigma(GK(jw)) (dashdot)')
xlabel('Frequency [rad/sec]'); ylabel('Amplitude')
figure(3); vplot('liv,lm',Kw);
title('\sigma(K(jw))');
xlabel('Frequency [rad/sec]'); ylabel('Amplitude')

Sd = minv(Ws); Sdw = vsvd(frsp(Sd,w)); %"desired" sensitivity
KSd = minv(Wks); KSdw = vsvd(frsp(KSd,w)); %"desired" output
figure(4); vplot('liv,lm',Sw,'-',Sdw,'--');
title('\sigma(S(jw)) (solid) and \sigma(Ws^{-1}(jw)) (dashed)');
xlabel('Frequency [rad/sec]'); ylabel('Amplitude')
figure(5); vplot('liv,lm',KSw,'-',KSdw,'--')
title('\sigma(KS(jw)) (solid) and \sigma(Wks^{-1}(jw)) (dashed)');
xlabel('Frequency [rad/sec]'); ylabel('Amplitude')
```

9 Perturb all plant parameters by 10 percent

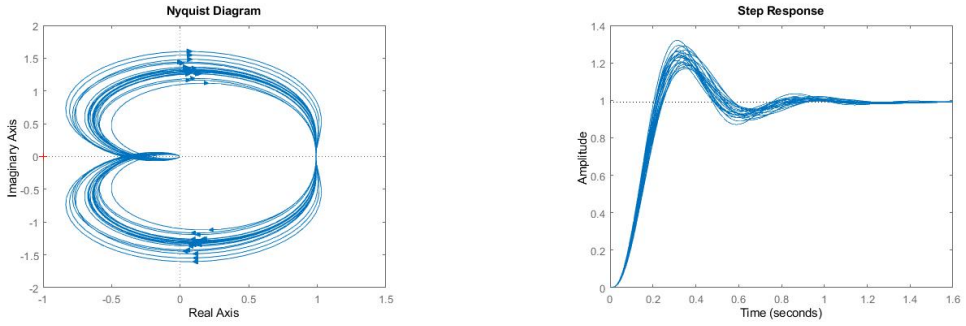
Plant parameters are perturbed using the "ureal" function as given in algorithm 4. Here, it can be said that if we perturb the plant parameters we are still stable for all the controllers. In order to obtain the generalized plant in matlab see algorithm 4.

```
%% Perturb all plant parameters by 10%

p1 = ureal('p1',200,'Percentage',10);
p2 = ureal('p2',0.025,'Percentage',10);
p3 = ureal('p3',1.002,'Percentage',10);
p4 = ureal('p4',10.1,'Percentage',10);
p5 = ureal('p5',1,'Percentage',10);

uncertain_plant = tf([p1],[p2 p3 p4 p5]);
```

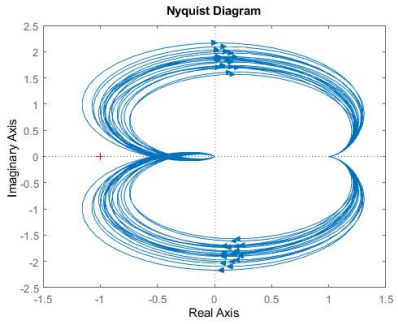
Algorithm 4: Perturb all plant parameters by 10 percent



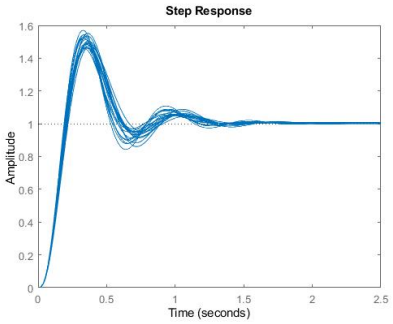
(a) Nyquist Diagram for K1

(b) Step for K1

Figure 29: Uncertain Plant for K1.

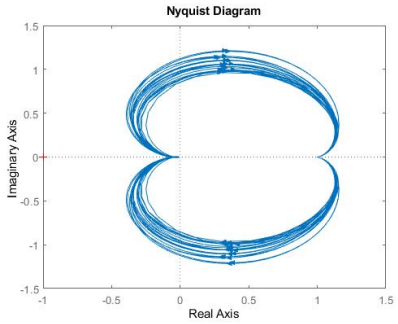


(a) Nyquist Diagram for K1

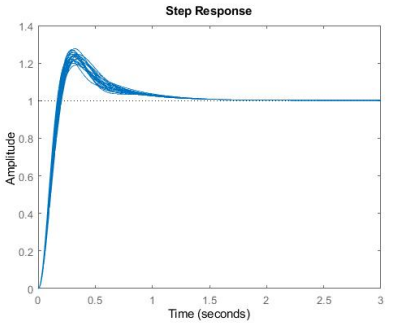


(b) Step for K1

Figure 30: Uncertain Plant for K2.

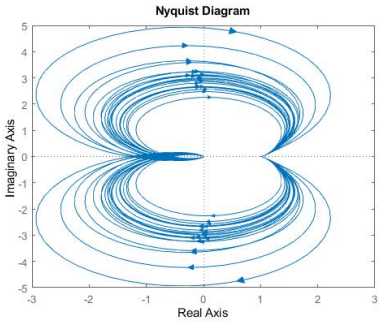


(a) Nyquist Diagram for K3

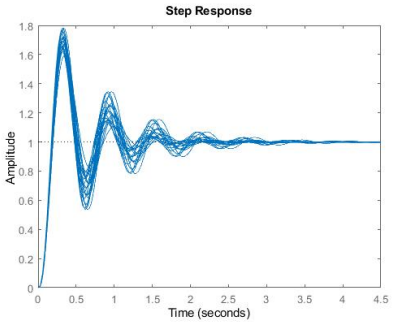


(b) Step for K3

Figure 31: Uncertain Plant for K3



(a) Nyquist Diagram for PID



(b) Step for PID

Figure 32: Uncertain Plant for PID.

References

[1] emehiel1. Pid design and sisotool, Jul 2012. URL https://www.youtube.com/watch?v=bvch8farB8U.

[2] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. John Wiley Sons, Inc., Hoboken, NJ, USA, 2005. ISBN 0470011688.