Question 3)

Question 3) a)

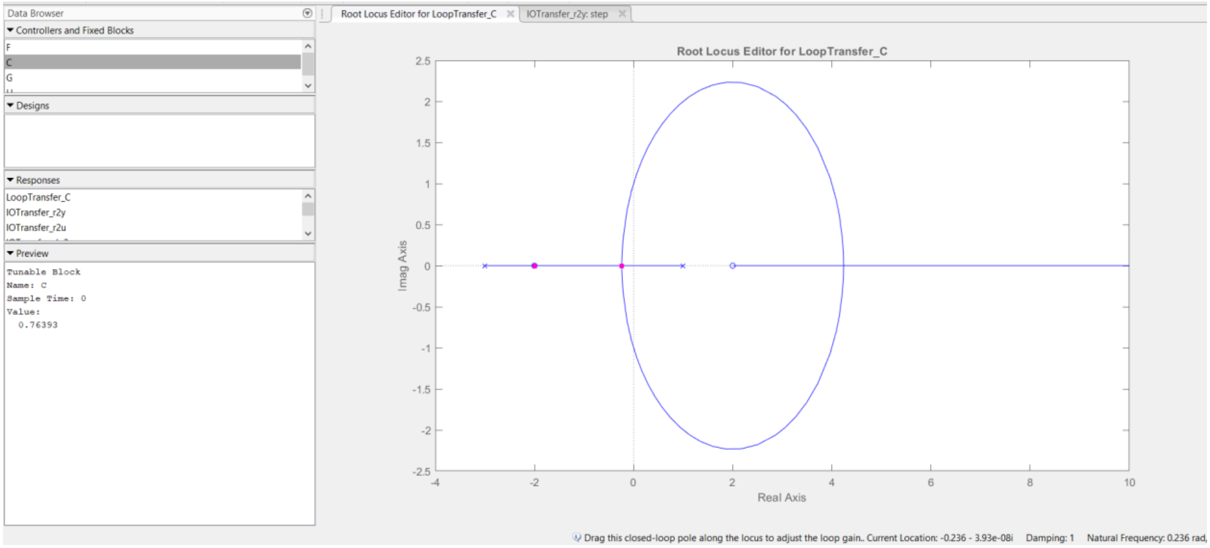


Figure 3.1: Root locus diagram with Pade's first order approximation

Question 3) b)

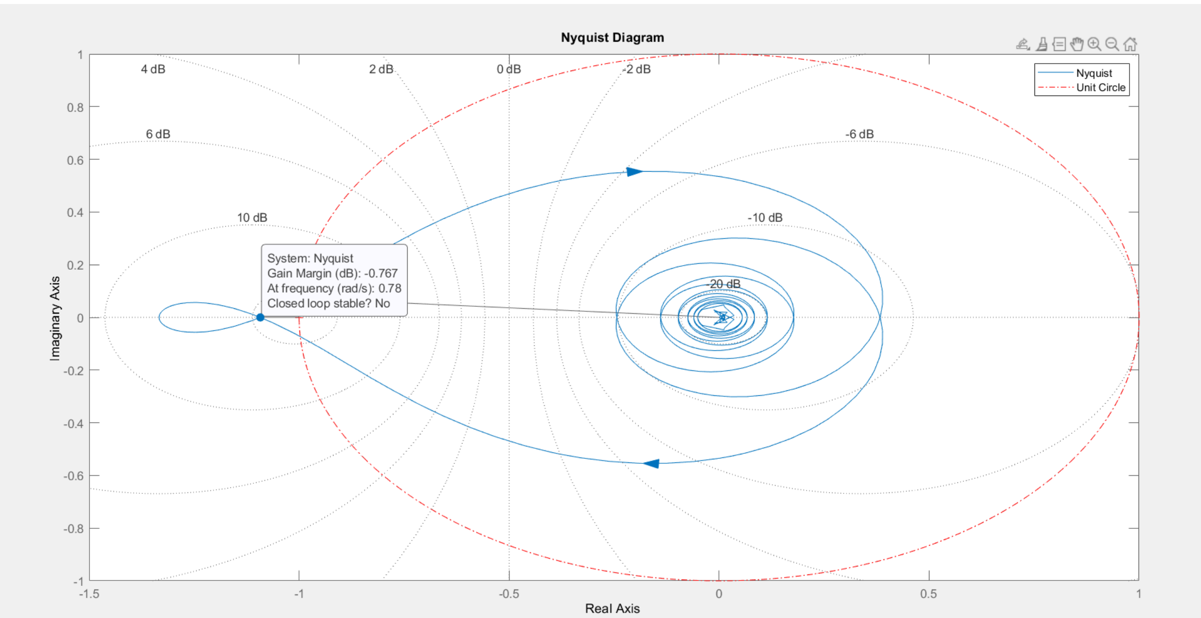


Figure 3.2: Nyquist diagram. $K_C = 1$. The red dotted curve is the unit circle.

Using the above diagram, we see that GM = -0.767 dB for $K_C = 1$.

So the required K_C for which gain margin is 10.5 dB is given by,

$$K_C = 10^{\frac{-0.767-10.5}{20}}$$

This gives $K_C = 0.2733$. However this system is unstable as depicted below in the Nyquist plot.

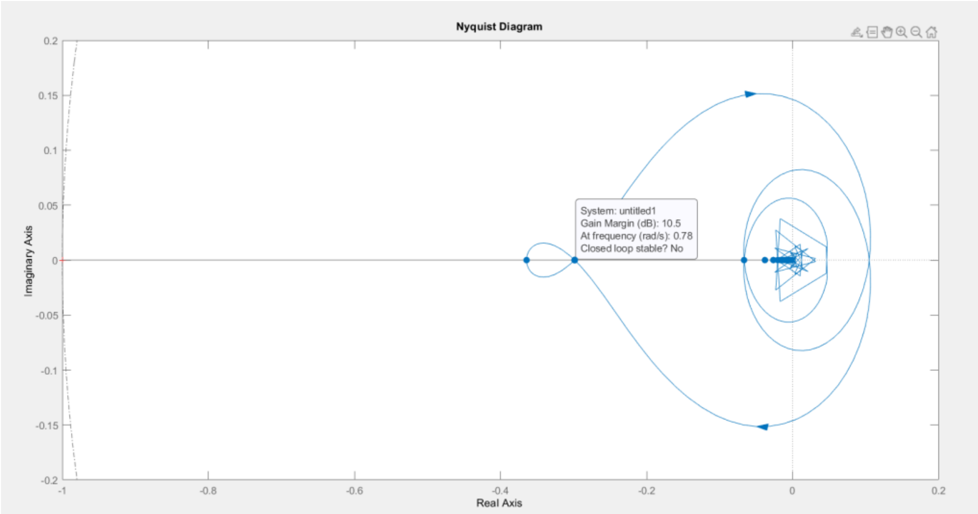


Figure 3.3: Nyquist Diagram for the system with gain margin 10.5 dB (K_C = 0.2733.)

As shown in handwritten part we can also get GM = 10.5 dB at w = 0 rad/s if K_C = 0.224.

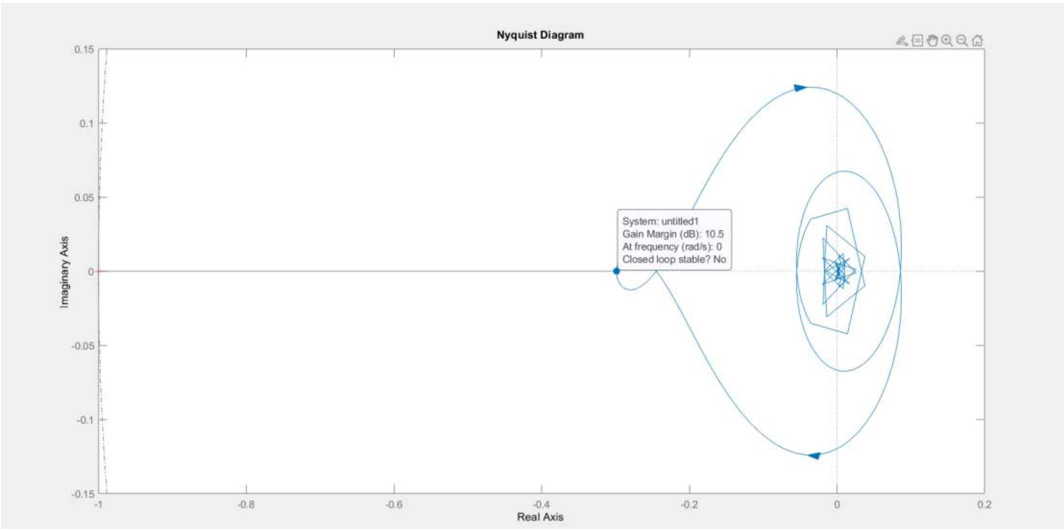


Figure 3.4: Nyquist Diagram for the system with gain margin 10.5 dB (K_C = 0.224)

Since the system is closed loop unstable, the step response of the system blows up. (So we can't define offset for this situation; offset -> infinity)

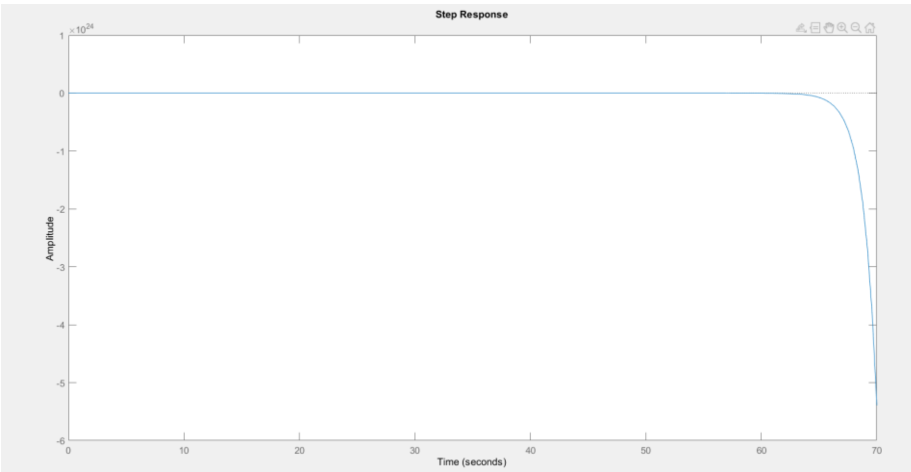


Figure 3.3: Step response of the system.

Question 3) c)

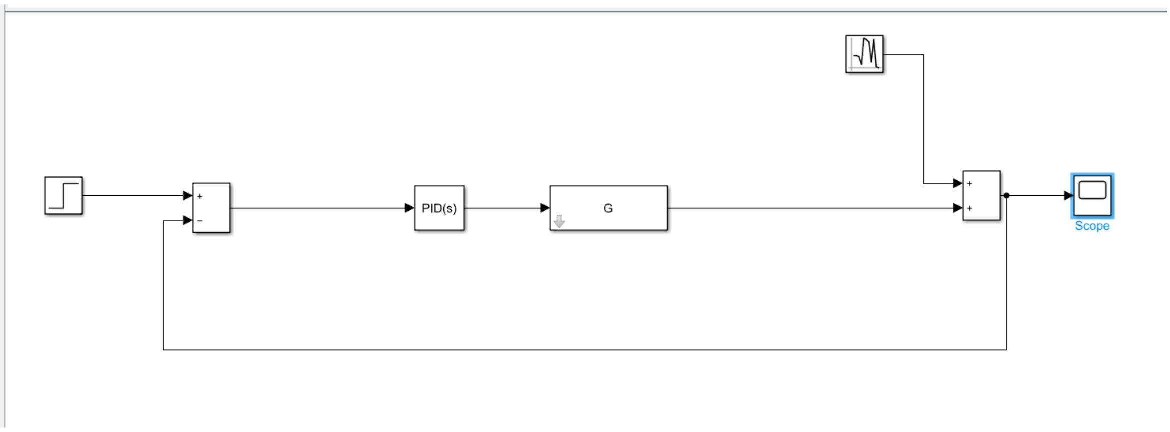
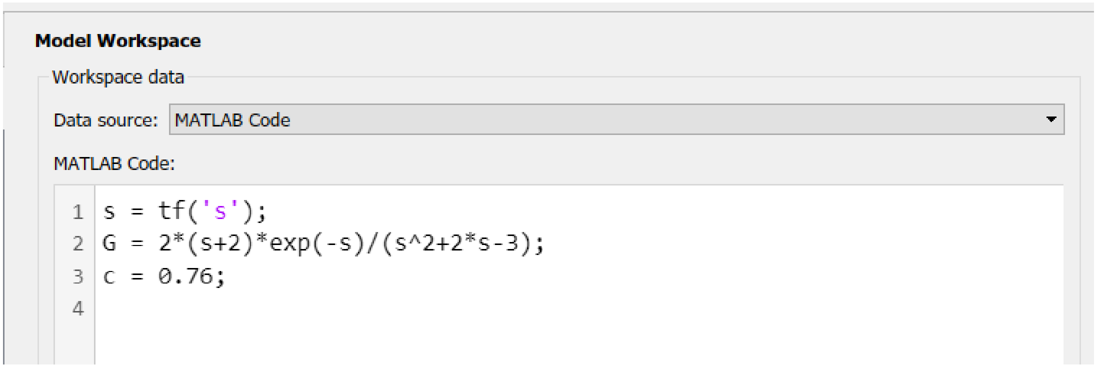


Figure 3.5 Simulink diagram

Variance of the disturbance is set to be 0.1.



Using Pade’s second order approximation

Since we can’t impose p = -2 is dominant condition I simply just used a K_C which gives CL stability when approximating the function using Pade’s.

L for the case of **second order Pade’s** approximation:

$$L = 2 * \frac{s+2}{s^2 + 2 * s - 3} * \frac{1 - \frac{s}{2} + \frac{s^2}{8}}{1 + \frac{s}{2} + \frac{s^2}{8}}$$

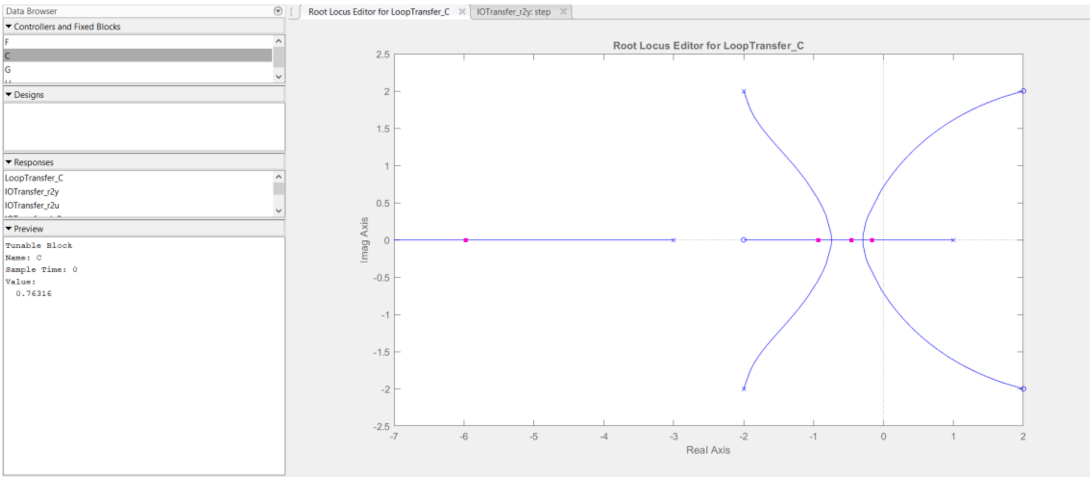


Figure 3.6: Rlocus plot for second order Pade's approximation.

A stable value of K_C came out to be **0.76** as shown in the above figure.

The same was used in the SIMULINK file and the following response was obtained.

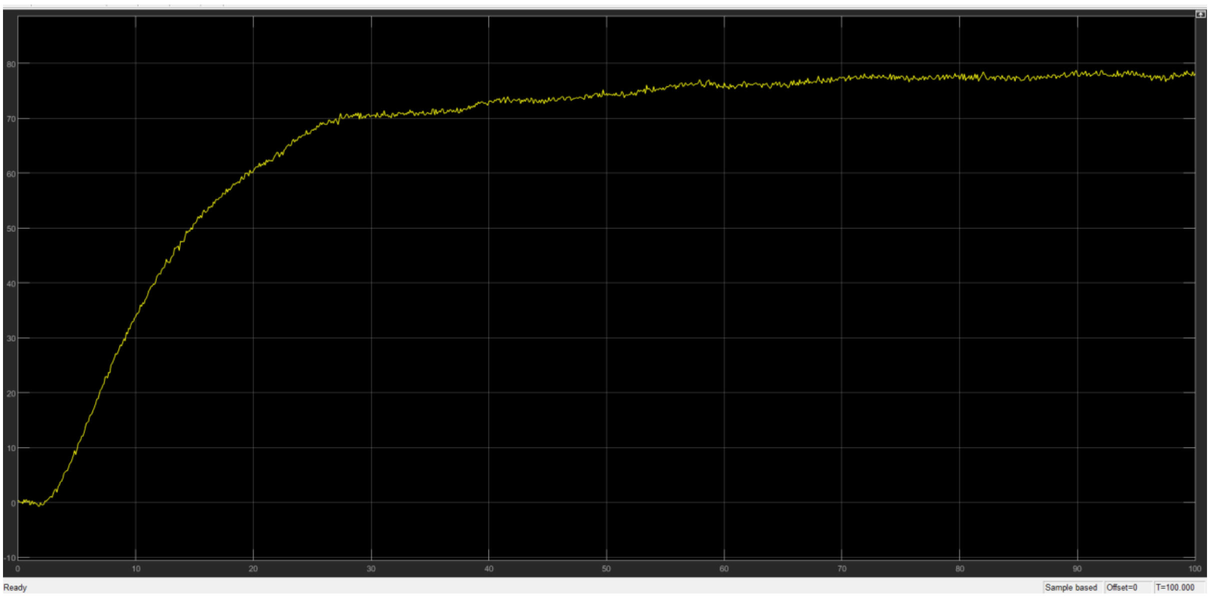


Figure 3.7: Step response controller obtained using second order Pade's approximation.

As we can see, the system is stable but there is a huge offset (~78 units)

Using the controller designed from Nyquist diagram

As shown in the figure below, step response is unbounded for K_C obtained using the Nyquist criterion. (K_C = 0.2733)

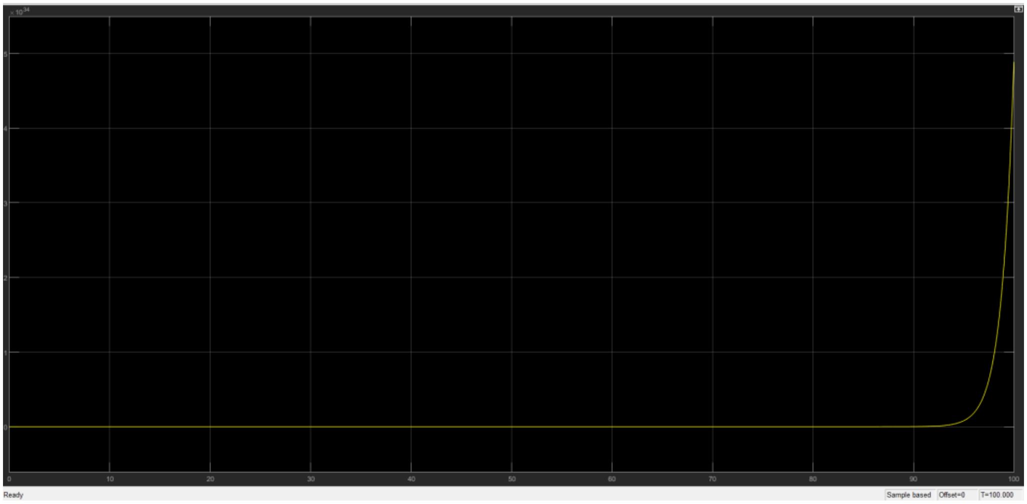


Figure 3.8: Step response for Kc = 0.2733

<u>Summary of question 3:</u>

1. With first order Pade's approximation, we were not able to find any controller that satisfies the given condition of p = -2 being dominant.
2. Using the given condition on Gain Margin and Nyquist plot, we obtained a K_C but that again gave unbounded response as shown by the above SIMULINK simulation.
3. Also, Nyquist plot has multiple cuts on real axis, so it is not easy to find out a gain margin. (Need the notion of a 'minimum stability' gain margin)
4. Controller gain obtained from checking RL plot of a second order Pade's approximated G_P seems to be stable, but the final value has a huge offset of about 79 units.
5. Pade's second order approximation improved the prediction but again, the choice of c was random, we weren't able to impose all the required conditions. So nothing special yielded from this.(could just be luck that I somehow got a stable K_C)
6. **Conclusion**: Delay seems to be badly affect the design of a P-Controller. It imposes severe restrictions on K_C and the responses are also not easily predictable.

Codes

Q1

```
clear; close all;
%% Setup the system
s = tf('s');
Gp = (s^2-4*s+8)/(s*(s+1)*(s+3));
G_sens = 1/(s+10);
L = Gp*G_sens;
%% Rootlocus plot
%rltool(L);
rlocus(L);
%% Solve for break in point
p = conv([4 42 86 30],[1 -4 8]) - conv([2 -4],[1 14 43 30 0]);
r = roots(p);
%% Part d)
Kcu = 4.69;
k = 0.01:0.01:Kcu;
r1 = zeros(length(k),1);
for i = 1:length(k)
   G = Gp*G_sens;
   sys = k(i)*G/(1+k(i)*G);
   S = stepinfo(sys);
   r1(i) = S.SettlingTime;
end
[val,loc] = min(r1);
Kc = k(loc);
%% Part e)
Lnew = tf([1 -4 8],([1 14 43 30 0 0]+Kc*[0 0 1 -4 8 0]));
figure;
rlocusplot(Lnew);
```

Q2

```
clear; close all;
%% Setup a P controller
Gp = tf([2 8],[10 7 1],'iodelay',2);
L = Gp;
[Gm,Pm,Wcg,Wcp] = margin(L);
margin(Gp);
Gm= 20*log10(Gm);
Gm_reqd = 8.2;
K_cu = 10^(Gm/20);
Kc = K_cu*10^((-Gm_reqd)/20);
[Gm2,Pm2,Wcg2,Wcp2] = margin(L*Kc);
%% Delay Uncertainity
w = Wcp2; % Here wcp and wcg correspond to Wgc, and Wpc respectively.
pm_verify = 180 + (atan(w/4)-atan(7*w/(1-10*w^2)) - 2*w)*180/pi;
```

```
figure;
margin(L*Kc);
DM = pm_verify/(w*180)*pi;
%% Designing a PI controller
s = tf('s');
[tauI,fval,exitflag]= fsolve(@(tauI) func(tauI,L,Kc),0.5);
figure;
margin(L*Kc*(1+1/tauI/s));
%% Evaluating the sensitivity integral
% P Controller
logmod2 = @(Kc,w) (log(abs(Q2_So2(Kc,w))));
int_val2 = integral(@(w)logmod2(Kc,w), 0, 10^5);
% PI controller
logmod = @(tauI,Kc,w) (log(abs(Q2_So(tauI,Kc,w))));
int_val = integral(@(w)logmod(tauI,Kc,w), 0, 10^4);
%% function that gives 60-PM for a given tau
function P  = func(tauI,L,Kc)
   s = tf('s');
   [~,PM,~,~] = margin(L*Kc*(1+1/tauI/s));
   P = 60 - PM;
end

%% Sensitivity function PI Controller
function So  = Q2_So(tauI,Kc,w)
   Gp = 2*(1j*w + 4)./(-10*w.^2+1+7*1j*w).*exp(-2j*w);
   Gc = Kc*(1+1/tauI./(1j*w));
   So = 1./(1+Gp.*Gc);
end


%% Sensitivity function P controller
function So  = Q2_So2(Kc,w)
   Gp = 2*(1j*w + 4)./(-10*w.^2+1+7*1j*w).*exp(-2j*w);
   Gc = Kc;
   So = 1./(1+Gp.*Gc);
end
```

Q3

```
clear; close all;
%% Setup the system
s = tf('s');
Gp = 2*(s+2)/(s^2+2*s-3)*exp(-s);
Gp_pade = 2*(2-s)*(s+2)/(s^2+2*s-3)/(s+2);
%% rl plot for fun
%rlocusplot(sys)
% Kc = 0.76393 gives stable roots (from rltool
%% Plot the nyquist diagram
figure();
nyquist(Gp);
grid on;
```

```
hold on;
% Plot the unit circle
n = 500;
theta = linspace(0,2*pi,n);
x = cos(theta); y = sin(theta);
plot(x,y,'r-.')
hold off;
legend('Nyquist','Unit Circle');
w = 0.78; % From nyquist plot
GM = -0.767; % From Nyquist Plot
Kc1 = 10^(-10.5/20)/4*3; % Derived by hand
figure;nyquist(Kc1*Gp)
Kc2 = 10^((-0.767-10.5)/20);
figure;nyquist(Kc2*Gp);
L = 2*(s+2)*(1-s/2+s^2/8)/(s^2+2*s-3)/(1+s/2+s^2/8);
c = 0.76566;
```