EE324 Control Systems Lab

Problem sheet 7

Tarun Sai Goddu | 190070024

Question 1

Given

$$G(s) = 1 / s(s^2+4s+8)$$

1.1 Part A

Scilab Code:

```
s = poly(0,'s');

G = 1/(s*(s^2 + 4*s + 8));
sys = syslin('c', G);
K = 0.01:0.01:100;

phm = zeros(length(K), 1);
gm = zeros(length(K), 1);
frp = zeros(length(K), 1);
frg = zeros(length(K), 1);

i = 1;
for  k = K
   sys1 = (k*sys);
   [phm(i),frp(i)] = p_margin(sys1);
   [gm(i), frg(i)] = g_margin(sys1);
   i = i + 1;
end

scf();
plot(K,phm,'b');
plot(K,gm,'r');

K = 32;
sys = (K*sys);
char = 1 + sys;
poles = roots(char.num);
disp(poles);
show_margins(sys);
```

By the use of scilab defined functions, we obtain the following plot for the open loop transfer function G(s):

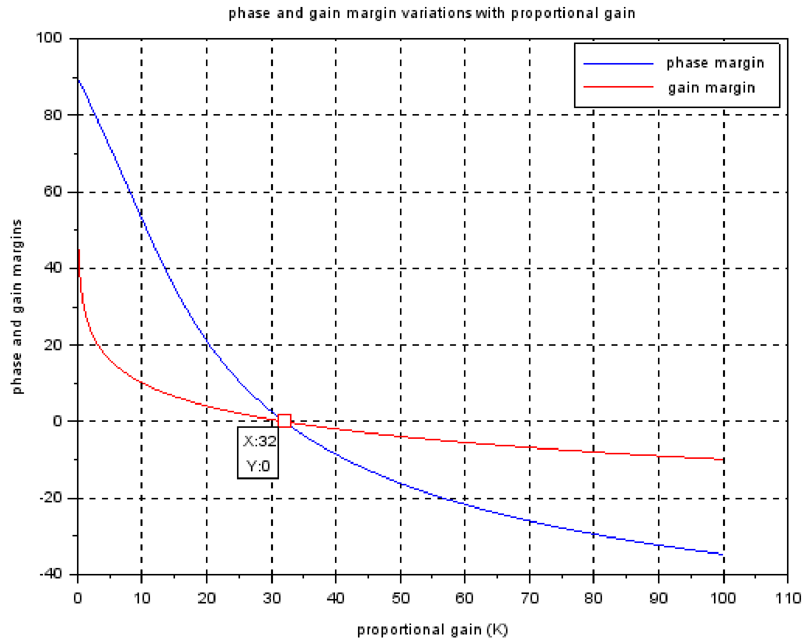


Figure 1: Phase Margin and Gain Margin vs Proportional Gain (K)

As is visible from the above plot, the value of K (gain) for which the gain and phase margin are both equal to zero is 32.

1. Part (b)
No, as shown in Figure 1, there is therefore no gain (K), with a gain of 0, but a margin of the phase is not null. The two only cross the x-axis once and again.

1. Part (c)
Poles of closed loop system when K = 32 are -4, ±2.8284i
Since we have poles of the closed loop system on the imaginary axis, the system is marginally stable.
The bode plot for the above system is shown below:

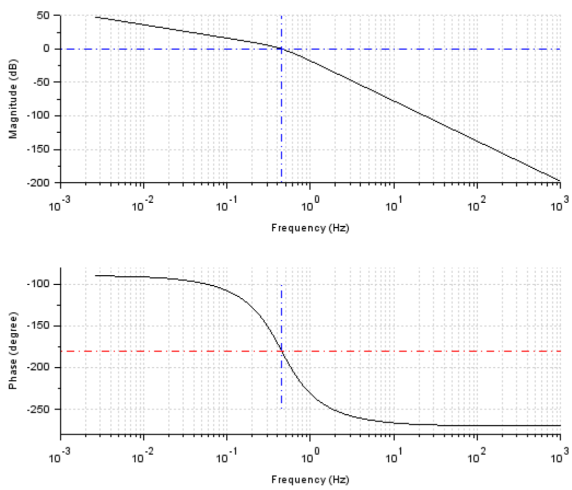


Figure 2: Bode plot of system for K = 32

Question 2

We are given a closed loop system with negative unity feedback with open loop transfer function as follows

Gopen(s) = 1/(s+2)(s+1)

For obtaining the desired specifications, a lag compensator is used with a transfer function:

C(s) = K(s+z)/(s+p) with z/p = 20

2a

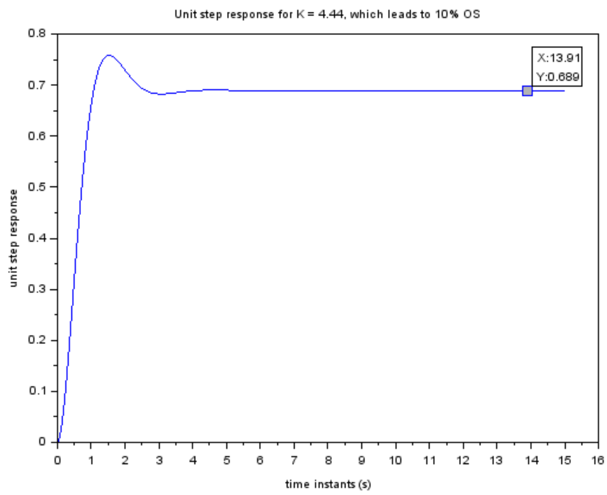
Constant gain K to achieve 10% OS in the closed-loop. We obtain the locus of 10% OS as follows:
Therefore, PI controller to be used is:
It is a straight line with angle of tan^-1((1-p^2)^1/2/p)

((1-p^2)^1/2/p = Pi/ln(10)

Therefore, we obtain the open loop contant gain K = 4.44

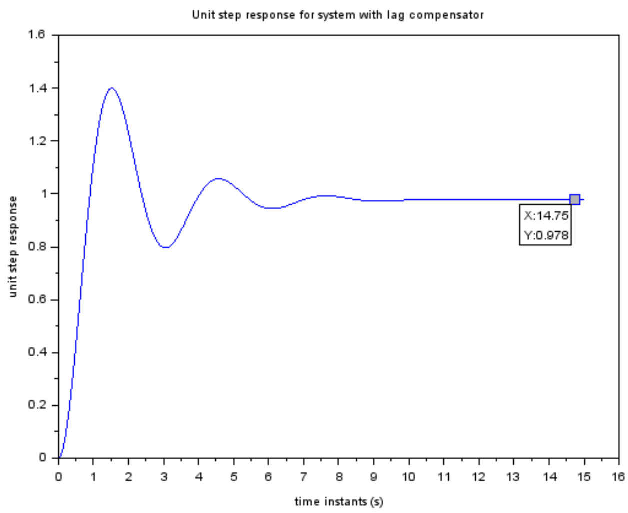
2b

The steady state error of the above system upon step input comes out be = 1-0.689 = 0.311 The unit step response is shown below:



Step response for 2b without Lag compensator

After placing the lag compensator, we obtain the following step response, which gives steady state error = 1- 0.978 = 0.022



Step response for 2b without Lag compensator

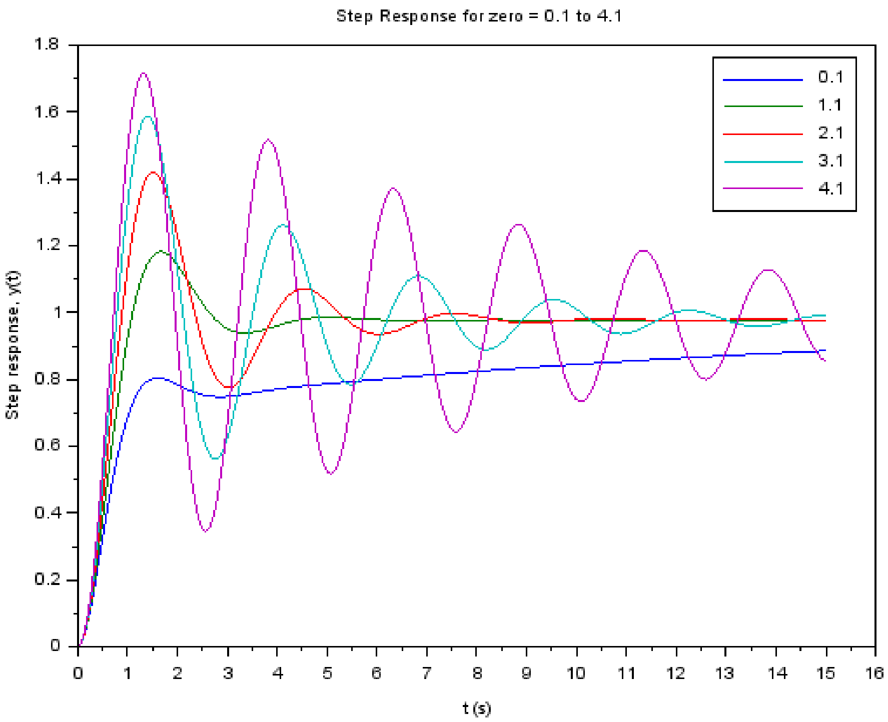
Lag compensator used had the following trransfer function:

$$C(s) = 4.44(s + 2)/ (s + 0.1)$$

```
s=poly(0,'s');
z = 0.01;
G = 1/((s+1)*(s+2));
sysG = syslin('c',G);
pi = 3.1415;
//from  10 % OS we get k = 4.44
k = 4.44;
t=0:0.01:15;
sysT = syslin('c',k*G/(1+k*G));
step_r = csim('step',t,sysT);
fig=scf();
```

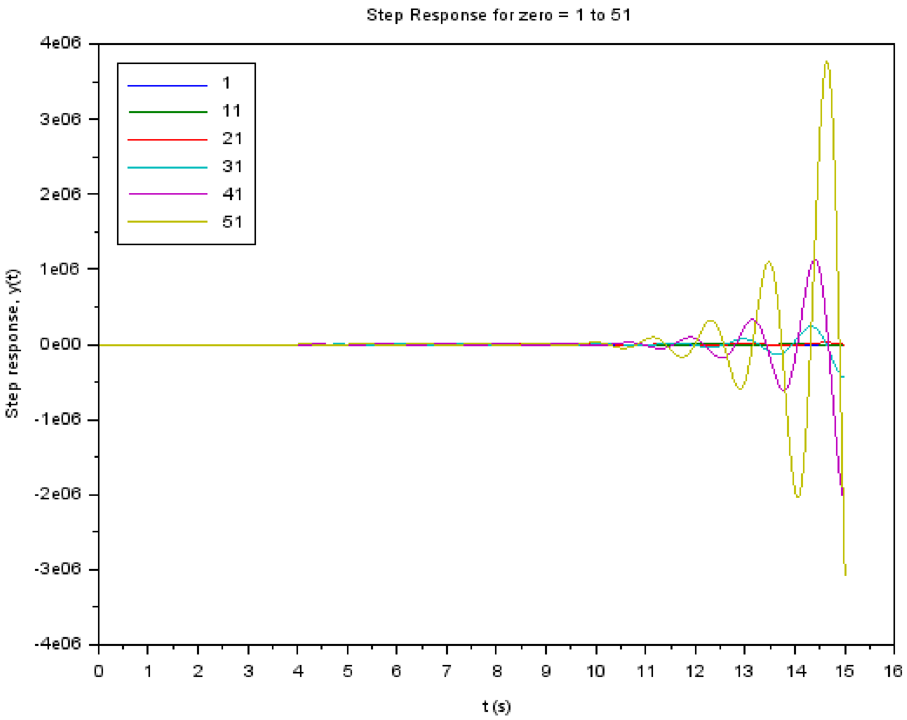
2c

We observe the following graphic when we change the polar-zero position while preserving the ratio. When z rises from 0.1 to 4.1, it will reduce and increase the damping ratio. And the lag compensator takes greater time to come into action as z values increases. All the systems are stable though



Step response for $0.1 \leqslant z \leqslant 4.1$:

When zero is increased from 1 to 51 we observe that system becomes unstable as z increases beyond 5.1. When z is increased from 1 to 51, the %OS increase and system becomes unstable after z crosses 5.1. And the lag compensator takes lesser time to come into action as z values increases. The plot is shown below:



Step response for $1 \leqslant z \leqslant 51$

We thus observe that the system becomes unstable or increases % OS (in case of stable systems), as the z value increases, and when we compromise, the compensator lags provide a better response time.

Scilab code:

```
k = 4.44;
t=0:0.01:15;
fig=scf();
z = 0.1 : 1 : 4.1;
y = zeros(length(t),length(z));
i=1;
for z1 = z
  p = z1/20;
  C = (s+z1)/(s+p);
  H = C*G;
  sysT = syslin('c',k*H/(1+k*H));
  y(:,i) = csim('step', t, sysT);
  i=i+1;
end
plot(t, y);
xlabel("t (s)")
ylabel("Step response, y(t)")
title("Step Response for zero = 0.1 to 4.1");
f = gcf();
```

Question 3
Given transfer function:

$$G(s) = 1 /s^2+ 3s+2$$

3a
Need to design a lead compensator for G(s) to obtain half 2% settling time of that in Q2-a. For 2-a, the settiling time was 2.33 s. Therefore required settling time = halving the settling time we can double the magnitude of Re(pole) as: For 10% OS we obtain damping ratio (ρ) as:

$$\rho = (\ln(10)/\ln(10)+pi^2)^1/2 = 0.43$$
$$2\% \text{ settling time}*Re(pole) = -\ln(0.02(1- \rho^2)^1/2$$
$$Re(pole) = -3.448$$

Therefore we see that Re(pole) reqired is -3.448 (as assumed stable system, while applying the formula). Intersection with required %OS = 10 locus

Therefore we get 2 pole values of the desired system as: -3.448 ± 4.705j
Now upon changing the varying the z and p values in the lead compensator transfer function shown below:

$$C(s) = K(s + z/ s + p)$$

We obtain p = 9.318, z = 4 and K = 41.47 for having pole values as -3.448 ± 4.705j. The plot is shown below:

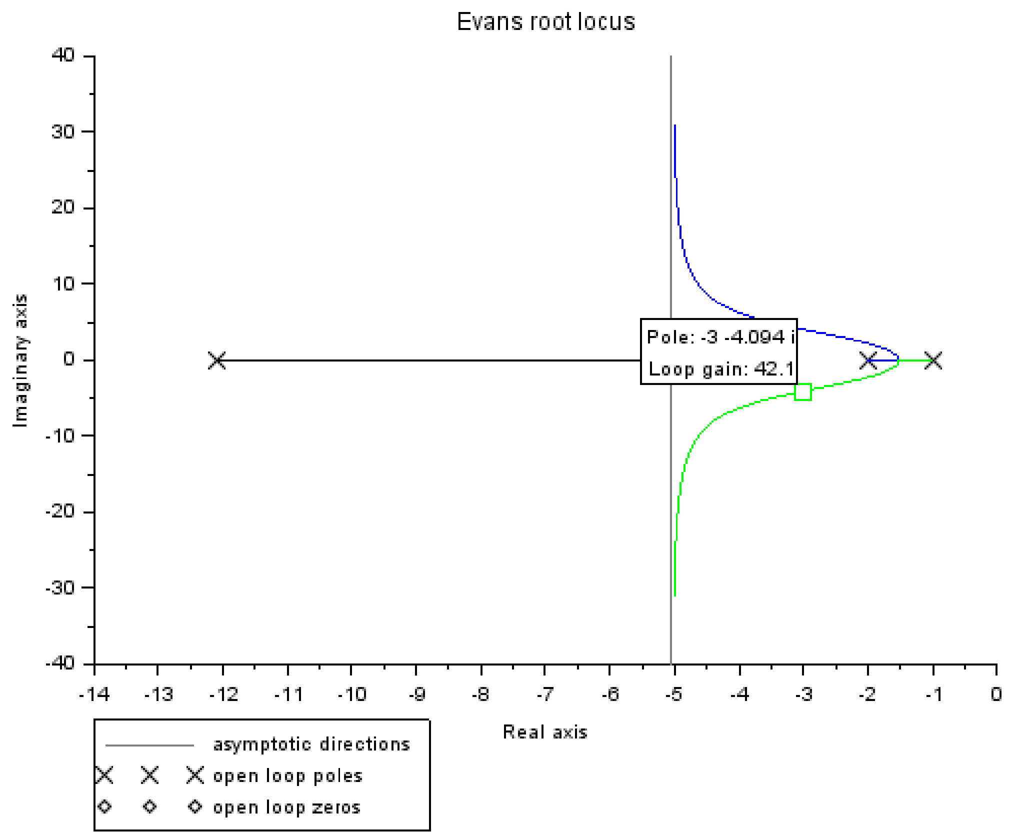


Figure : RL plot of the obtained system

Therefore, required lead compensator is as follows:

C(s) = 41.47(s+4/s+9.318)

Scilab code:

```
s=poly(0,'s');
z = 0.01;
G = 1/((s+1)*(s+2));
sysG = syslin('c',G);
pi = 3.1415;
theta=atan((pi/log(10)));
a=[0:0.01:10];
fig=scf();
evans(sysG, 2000);
x=-cos(theta)*a;
y=sin(theta)*a;
plot(x, y, 'k--');
z = 4;
p = 9.318;
C = (s+z)/(s+p);
H = C*G;
sysH = syslin('c',H);
fig=scf();
evans(sysH,1000);
```

3b

Same specifications as above with a PD controller C(s) specified below:

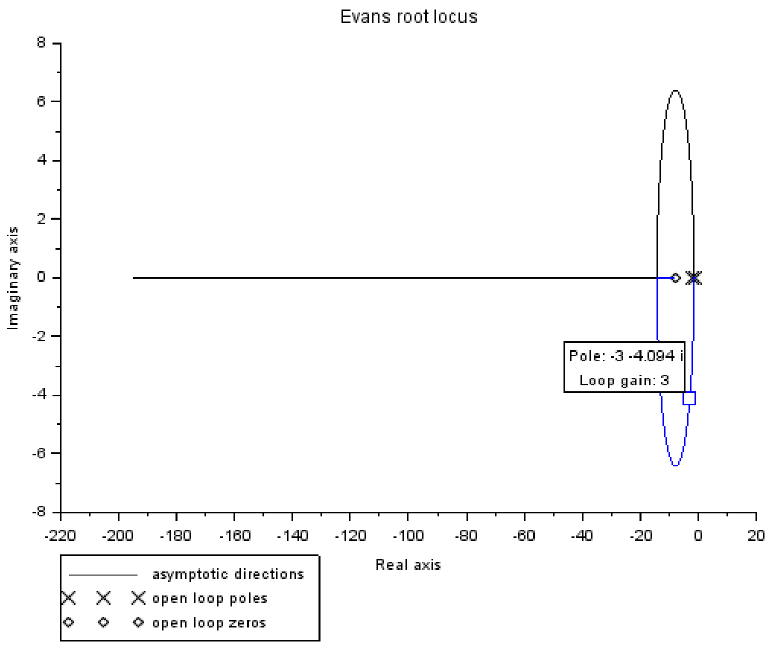
C(s) = K(s+z)

Again 2 pole values of the desired system as: -3.448 ± 4.705j , hence upon varying values of z and K for having the above 2 points on RL plot, we obtain z = 8.223 and K = 3.897.

Therefore, required PD controller has following transfer function:

C(s) = 3.897(s + 8.223)

The plot with required pole labelled is shown below:



Root locus for Obtained system

Scilab code:

```
s=poly(0,'s');
z = 0.01;
G = 1/((s+1)*(s+2));
sysG = syslin('c',G);
pi = 3.1415;

theta=atan((pi/log(10)));
a=[0:0.01:10];
fig=scf();
evans(sysG, 2000);
x=-cos(theta)*a;
y=sin(theta)*a;
plot(x, y, 'k--');

z = 8.223;
C = (s+z);
H = C*G;
sys = syslin('c',H);
evans(sys, 200);
```