ELEC 341

MATLAB Project

Submitted to Prof. Jane Wang

Apr 10, 2020

Jian Gao

[redacted]

Note that all MATLAB code is attached to the end of this report.
Problems are in Modern Control Systems 12th edition--just realized right before the deadline--
The numbers given in Problem 2, 5, 10 may be slightly different in 12th&13th versions.

Problem 1. CP7.10

a)

Characteristic Equation:

$$det[sI - A] = s^3 + (2 + k)s^2 + 5s + 1 = 0$$

b)

The Routh Array:

$$\begin{array}{c|cc} s^3 & 1 & 5 \\ s^2 & 2+k & 1 \\ s^1 & 5 - \dfrac{1}{2+k} & 0 \\ s^0 & 1 & 0 \end{array}$$

To make the system stable:

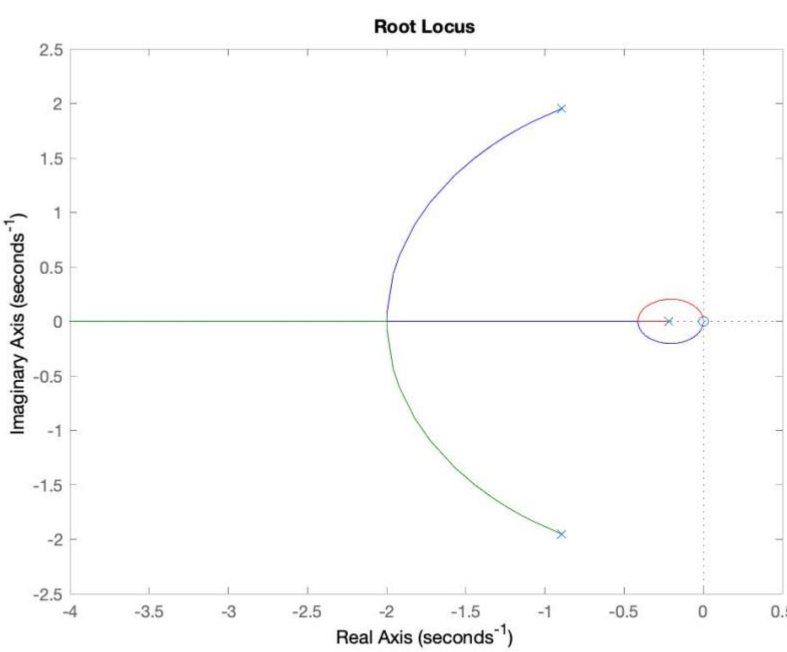
$$2 + k > 0, \qquad 5 - \frac{1}{2+k} > 0$$

Thus,

$$k > -\frac{9}{5}$$

c)

Root Locus:



For $k > 0$, the system is always stable since the plot is on the left-hand side of jw-axis.

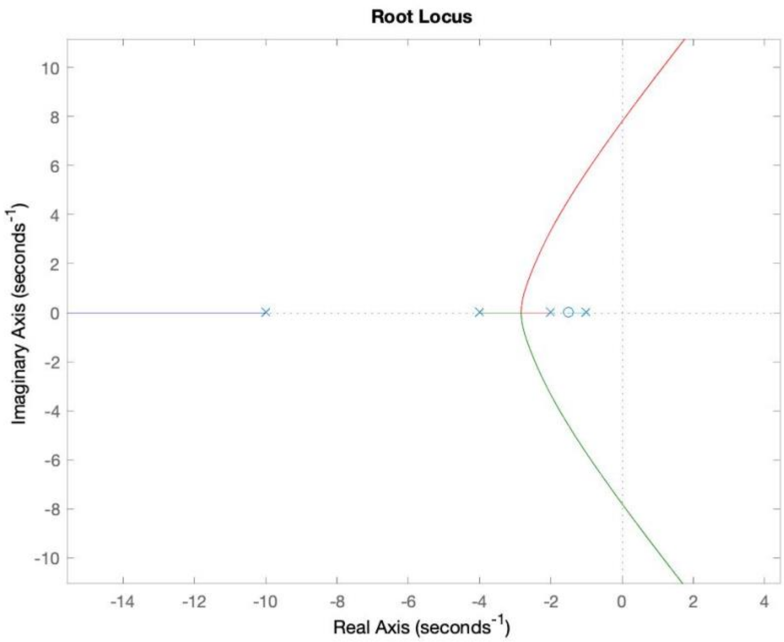
Problem 2. DP7.12

a)

Characteristic Equation:

$$1 + K * \frac{s + 1.5}{(s+1)(s+2)(s+4)(s+10)} = 0$$

Root Locus:



When $K = 100$, roots are:

$$s = -11.38, -1.45, -2.09 \pm j3.10$$

When $K = 300$, roots are:

$$s = -12.94, -1.48, -1.29 \pm j5.10$$

When $K = 600$, roots are:

$$s = -14.44, -1.49, -0.53 \pm j6.72$$

b)

When K=100,

$$T_s = \frac{4}{\zeta\omega_n} = \frac{4}{2.09} \cong 1.91\ s$$

$$\omega_n = \sqrt{2.09^2 + 3.1^2} = 3.74$$

$$\zeta = \frac{2.09}{3.74} = 0.56$$

Thus,

$$P.O.\% = 13\%$$

Steady-state error:

$$e_{ss} = \lim_{s \to 0} s * \frac{1}{s} * T(s) = \frac{\frac{1.5}{80} * 100}{1 + \frac{1.5}{80} * 100} = 0.65$$

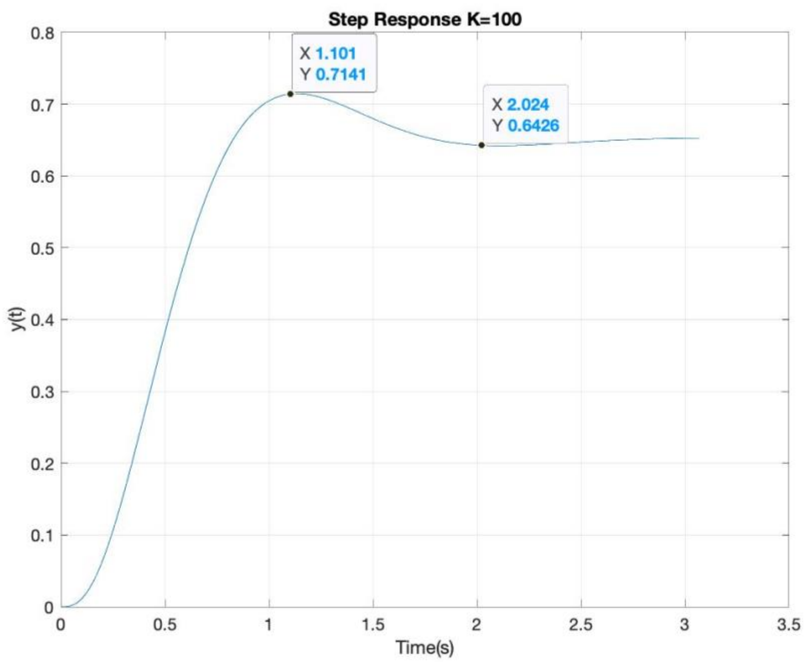
We then apply the same calculation to K=300, 600.

c)

$$T(s) = \frac{K * (s + 1.5)}{(s + 1)(s + 2)(s + 4)(s + 10) + K * (s + 1.5)}$$

$$= \frac{Ks + 1.5K}{s^4 + 17s^3 + 84s^2 + (148 + K)s + 80 + 1.5K}$$

When K=100,

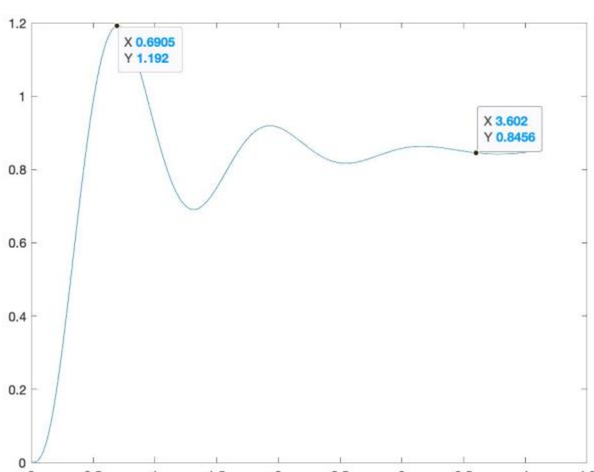


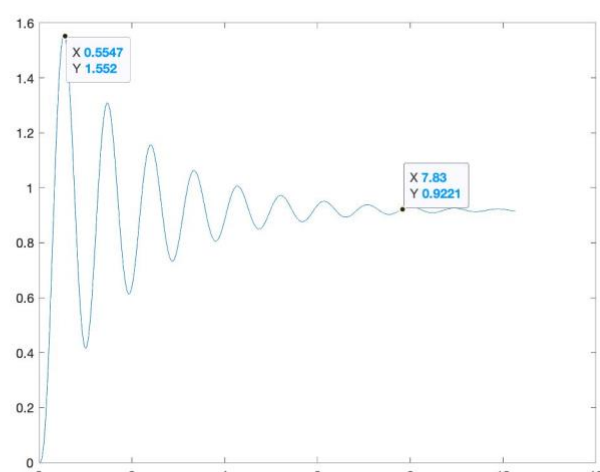
From the graph:

$$O.P.\% \cong 11.12\%, \qquad T_s \cong 2\ s$$

The results are pretty close to the predicted values from part b.

Here are also the plots for K=300, 600. Again, the O.P. and Ts are similar to what we obtain from part b.



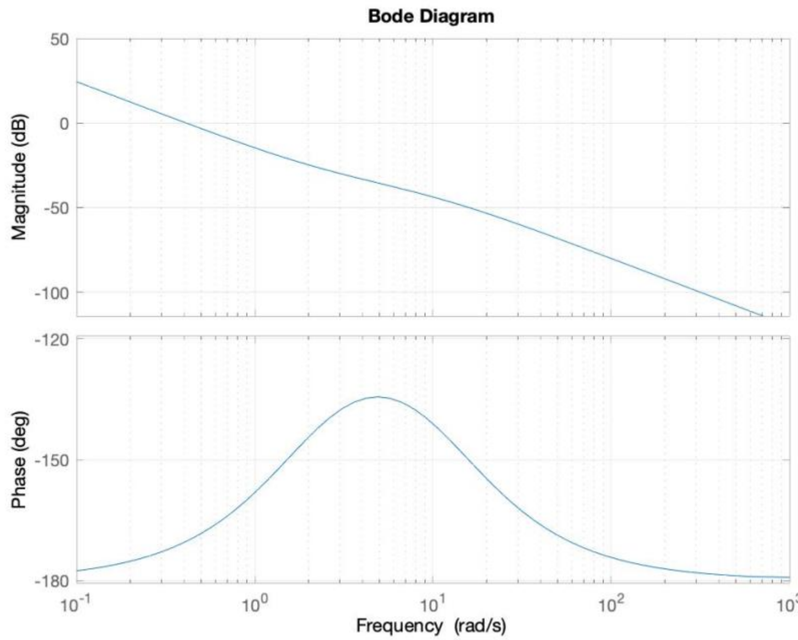


Problem 3. DP8.1

a)

$$G_c(s) * G(s) = \frac{s+2}{s^2(s+12)}$$

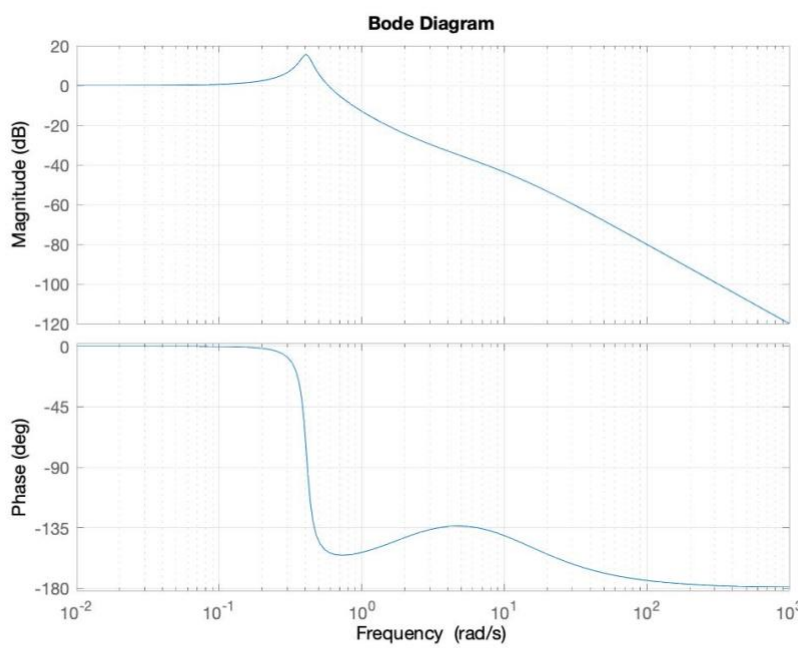
Bode Plots:



b)

$$T(s) = \frac{s+2}{s^3 + 12s^2 + s + 2}$$

Bode Plots:

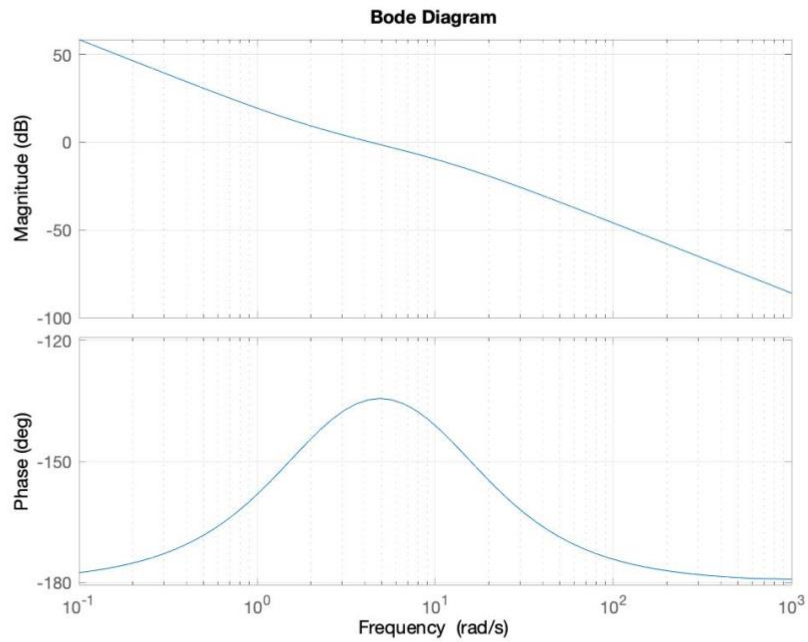


c)

K=50:

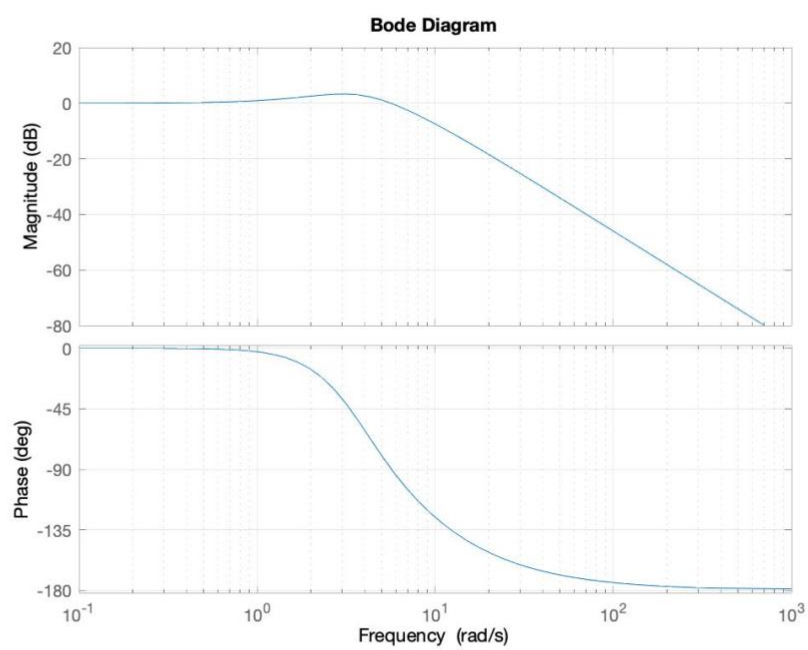
$$G_c(s) * G(s) = \frac{50s + 100}{s^2(s + 12)}$$

Bode Plots:



$$T(s) = \frac{50s + 100}{s^3 + 12s^2 + 50s + 100}$$

Bode Plots:



d)

$$M_p \leq 2$$

So that:

$$K \geq 14$$

Due to the fact that the system has to be stable:

$$14 \leq K \leq 350$$

We choose:

$$K = 350, \qquad \omega_B = 29\, rad/sec$$

e)

Steady-state Error for a ramp input:

$$e_{ss} = \lim_{s \to 0} s * R(s) * T(s) = 0$$

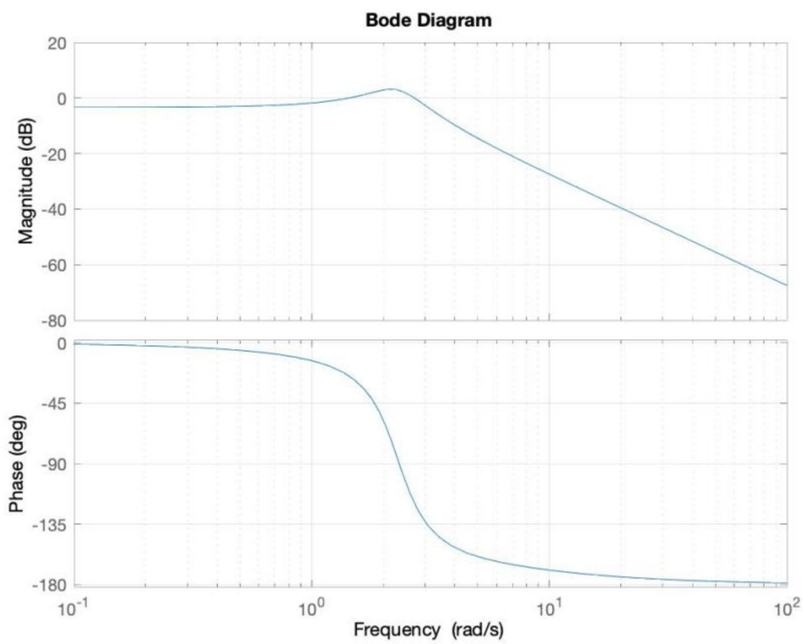
Problem 4. DP8.3

$$T(s) = \frac{K(s+5)}{s^3 + 7s^2 + 12s + 10 + 5K}$$

From the question (20lgM=3), we get

$$K = 4.2$$

Body Plot:



We can tell from the graph that the system bandwidth is:

$$3.72\, rad/sec$$

Steady-state error:

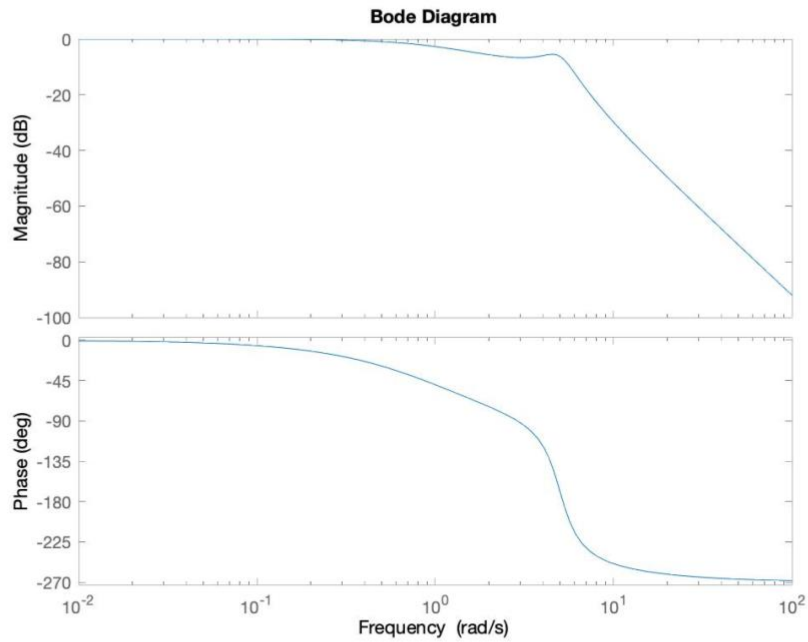
$$e_{ss} = \lim_{s\to 0} s * E(s) = \lim_{s\to 0}\big(1 - T(s)\big) = 0.32$$

Problem 5. CP8.6

Open-loop system:

$$L_s = \frac{25}{s^3 + 3s^2 + 27s + 25}$$

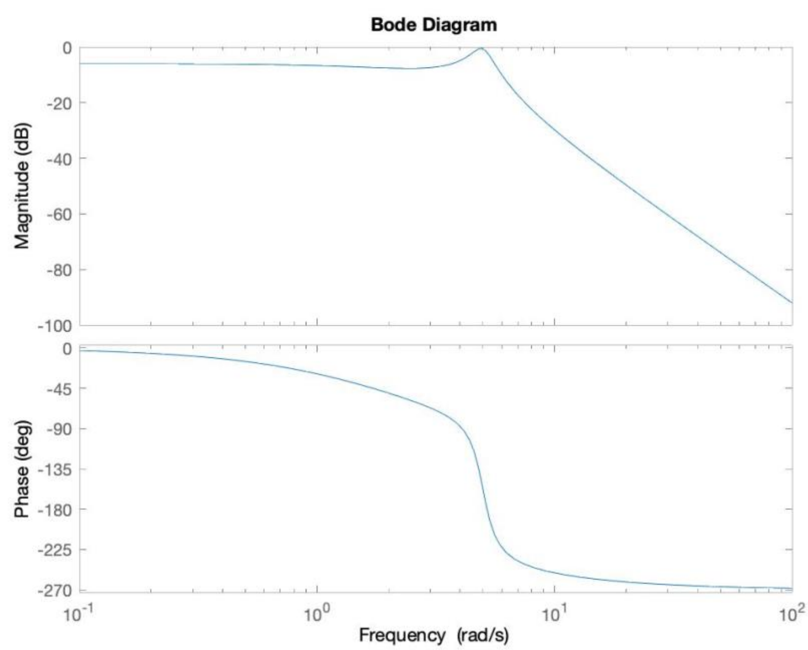
Bode Plot:



Closed-loop system:

$$L_s = \frac{25}{s^3 + 3s^2 + 27s + 50}$$

Bode Plot:



Problem 6. DP10.1

$$G(s) = \frac{20}{s(s+2)}$$

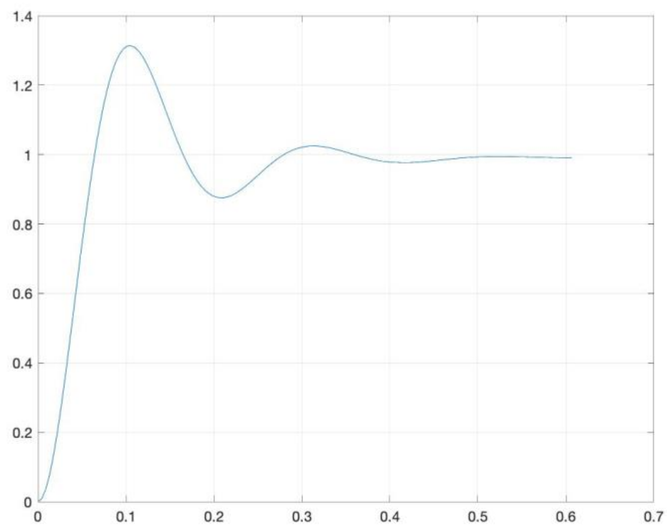
The possible lead and lag compensators can be:

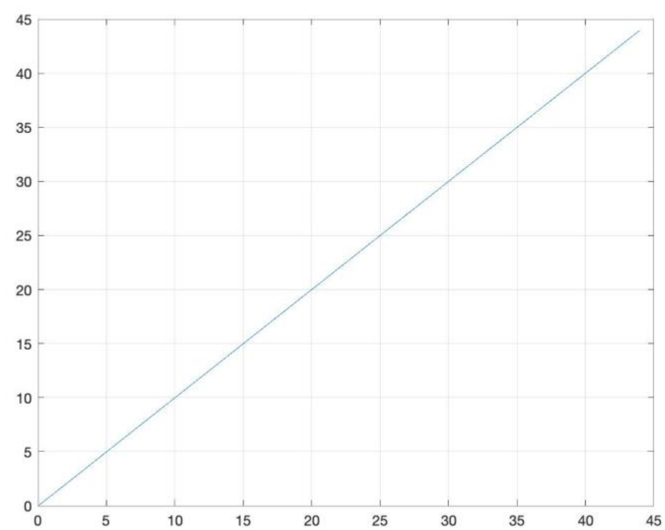
$$G_{lead}(s) = \frac{50(s+1)}{s+20}, \qquad G_{lag}(s) = \frac{(s+0.1)}{s+0.022}$$

Thus,

$$L_s = \frac{1000(s+1)\,(s+0.1)}{s(s+2)(s+20)(s+0.022)}$$

Step and ramp responses are shown as follows:



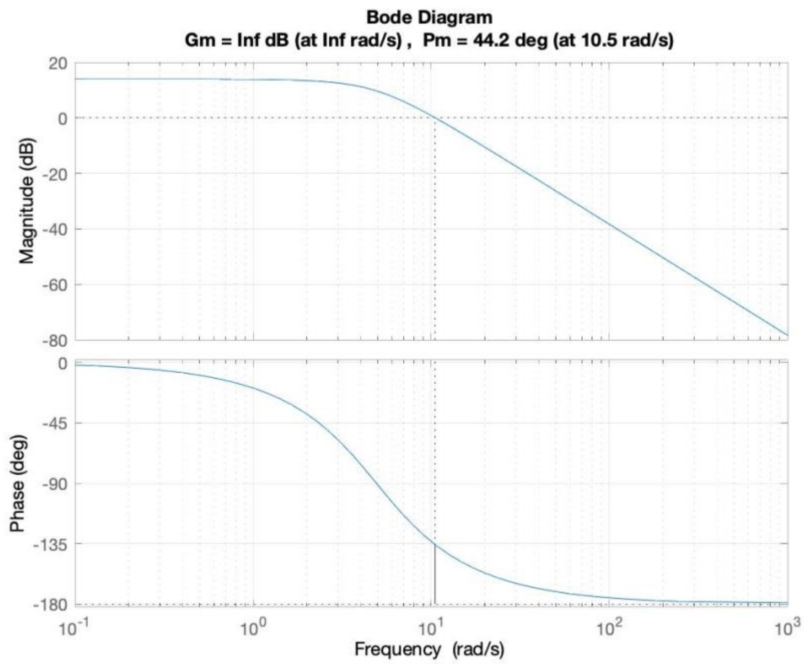


Problem 7. CP10.2

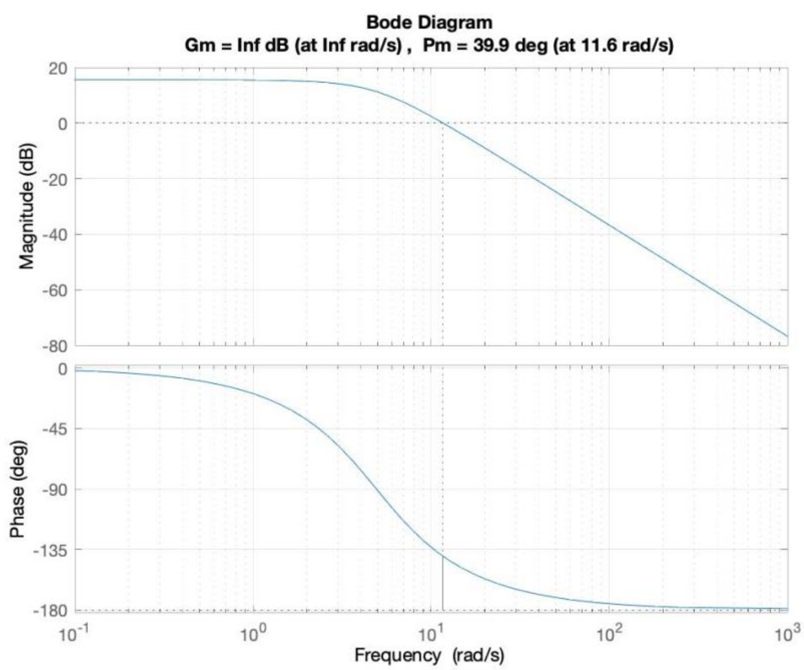
Open-loop system:

$$L_s = \frac{24.2}{s^2 + 8s + 24.2}$$

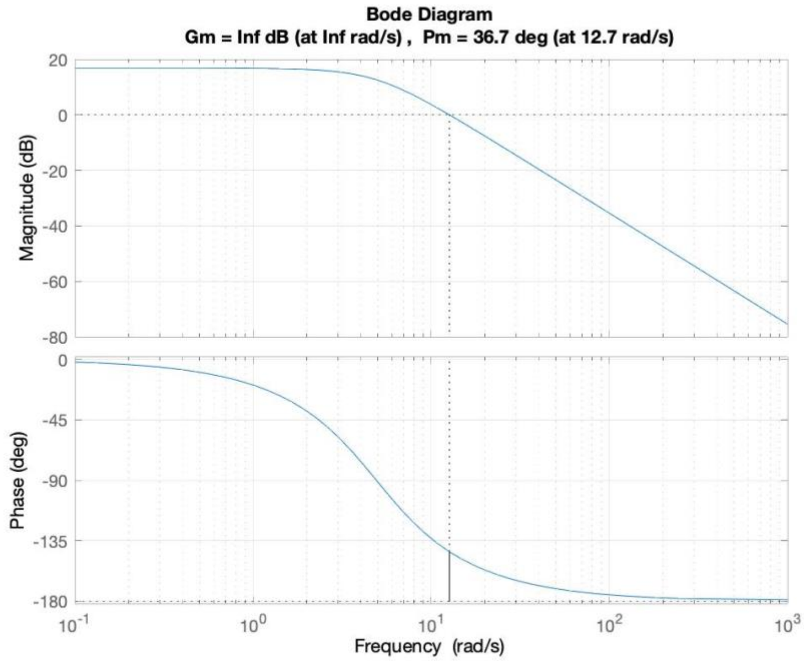
Bode Plot (K=5):



Bode Plot (K=6):



Bode Plot (K=7):



By iterating different values of K and checking phase margin in MATLAB, we conclude that:

$$K = 6$$

K=6 gives us a phase margin of about 40 degrees.

Problem 8. CP10.4

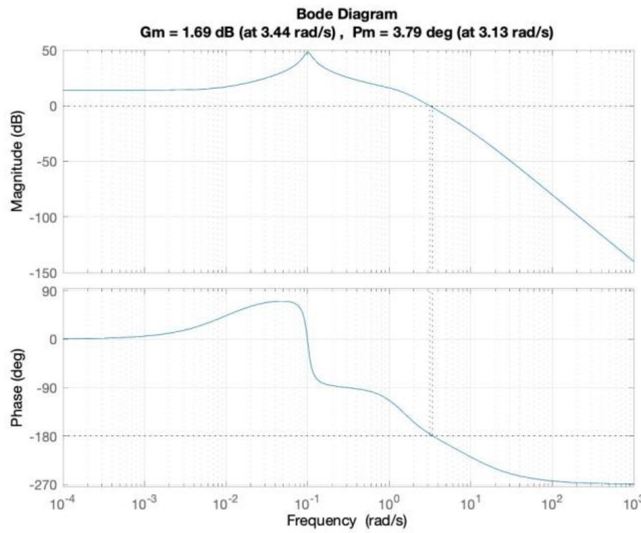
a)

In order to have $O.P.\% < 10\%$, we have:

$$\zeta > 0.6, \ \ phase\ margin > 60^{\circ}$$

Using MATLAB, we found out the phase margin of the uncompensated system is 3.79° which is far less than the expected value. Meanwhile, α is found out to be 10.8 here.

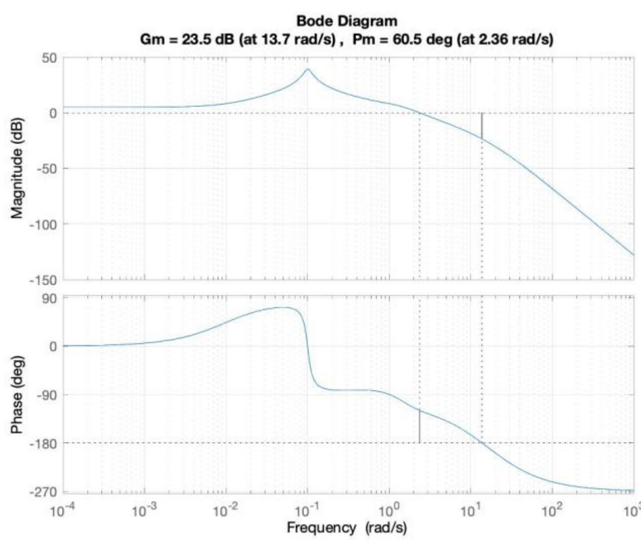
Bode Plot:



From the graph above, $G(j\omega)$ is $0\ dB$ at about $2\ rad/sec$. We can also find out K converges to 4, and $p = \alpha z = 21.7$. So that we can design the phase lead compensator to be as follows:

$$G_c(s) = \frac{4(s+2)}{s+22}$$

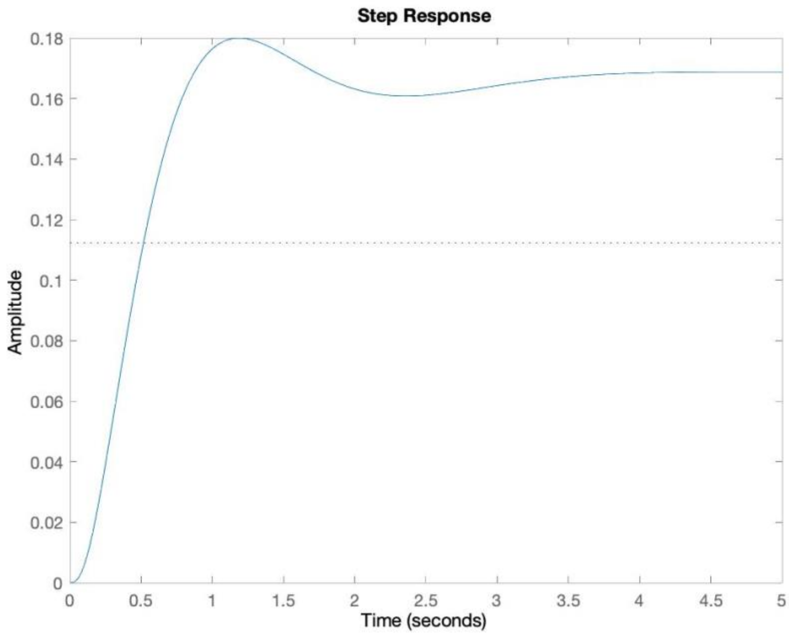
New Bode Plot:



This system now satisfies the requirement.

b)

In MATLAB, we can draw the step response for the input of $60^{\circ}/sec$:



Problem 9. CP11.3

Given:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad C = [1 \quad -1], \qquad D = [0],$$

$$s_1 = -1, \qquad s_2 = -2$$

The gain matrix of the closed-loop system is easily calculated in MATLAB:

$$K = [0.5\ 0.5]$$

Problem 10. DP11.7

1)

$$A - BK - LC = \begin{bmatrix} -L_1 & 1 & 0 \\ -L_2 & 0 & 1 \\ -2 - K_1 - L_3 & -5 - K_2 & -10 - K_3 \end{bmatrix}$$

Here we choose:

$$K_1 = -2, \qquad K_2 = -5, \qquad K_3 = -10$$

Thus,

$$\det\big(\lambda I - (A - BK - LC)\big) = 0$$

Plugging in, we have:

$$\lambda^3 + L_1 * \lambda^2 + L_2 * \lambda + L_3 = 0$$

According to R-H table, to make the system stable:

$$\begin{cases} L_2 > \dfrac{L_3}{L_1} \\ L_3 > 0 \end{cases}$$

Since $\omega_B \geq 10\ dB$, we can also choose:

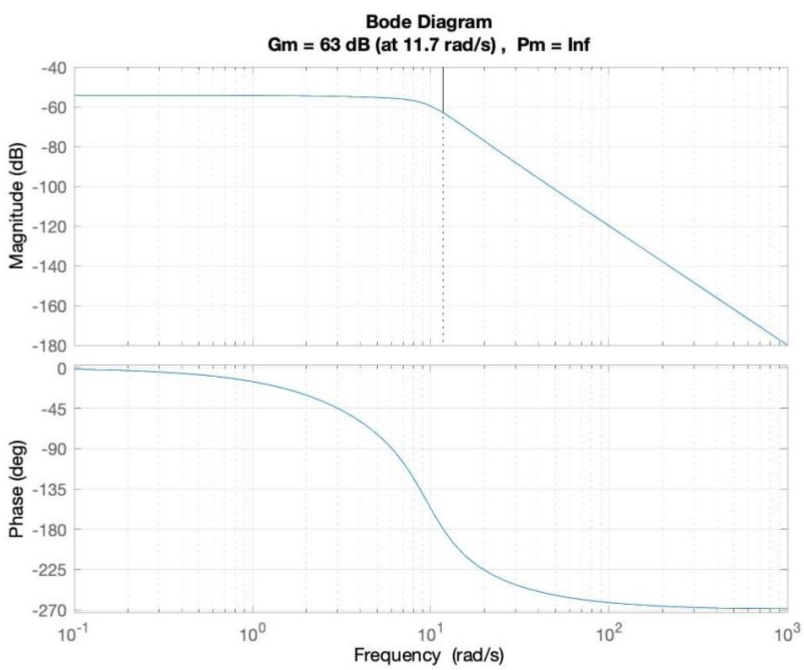
$$\omega_n = 8\ rad/sec$$

Thus,

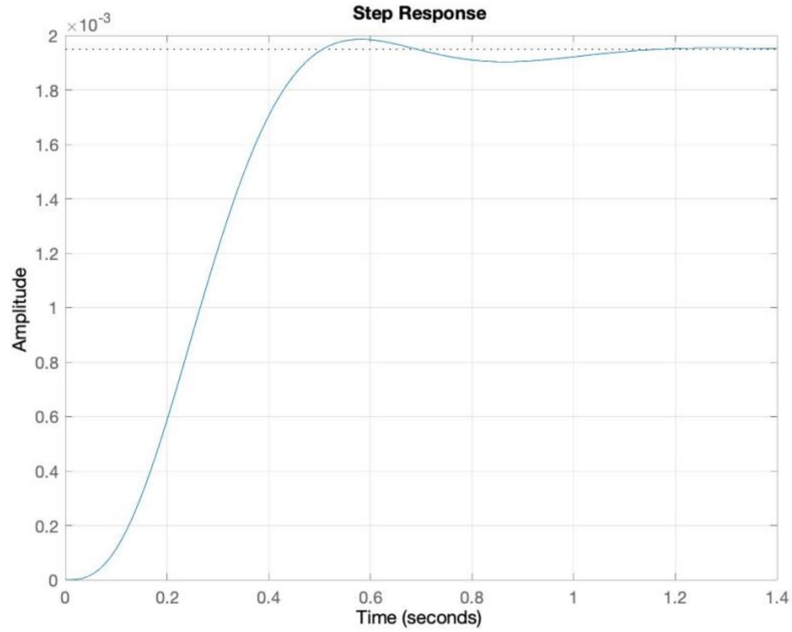
$$L = \begin{bmatrix} 14 \\ 138 \\ 512 \end{bmatrix}$$

In MATLAB:

Bode Plot w/ gain margin greater than 6 dB:



Step Response:



Steady-state error meets the requirement.

Problem 11

1)

$$Q(s) = 1 + \frac{s + \alpha}{s^3 + (1+\alpha)s^2 + (\alpha - 1)s + (1 - \alpha)} = 0$$

$$s^3 + (1+\alpha)s^2 + (\alpha - 1)s + (1 - \alpha) + s + \alpha = 0$$

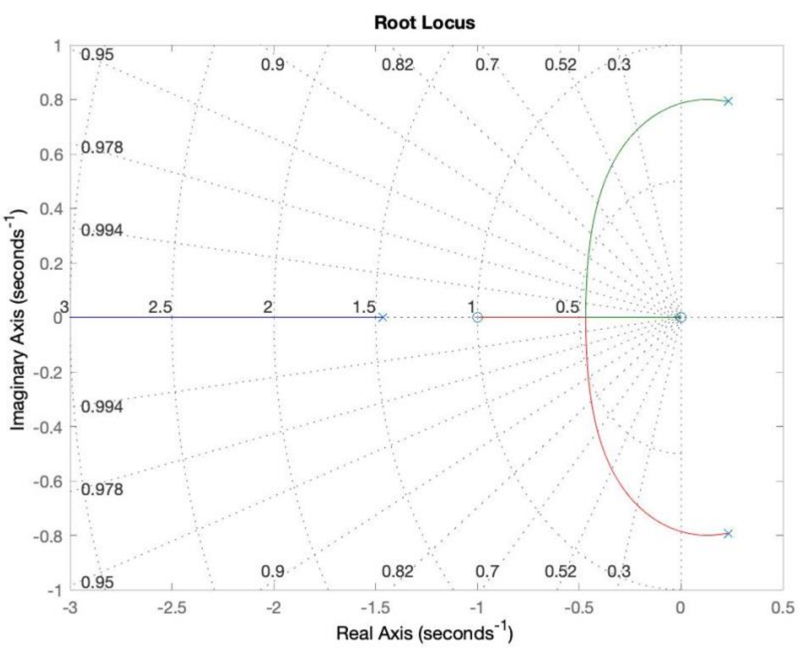
$$s^3 + s^2 + 1 + \alpha(s^2 + s) = 0$$

$$1 + \frac{\alpha(s^2 + s)}{s^3 + s^2 + 1} = 0$$

Thus,

$$L_s = \frac{(s^2 + s)}{s^3 + s^2 + 1}$$

Root Locus:



2)

$$Q(s) = s^3 + (1+\alpha)s^2 + \alpha s + 1 = 0$$

The Routh Array:

s^3	1	α
s^2	$1+\alpha$	1
s^1	$\alpha - \frac{1}{1+\alpha}$	0
s^0	1	0

To make the system stable:

$$1 + \alpha > 0, \qquad \alpha - \frac{1}{1+\alpha} > 0$$

Thus,

$$\alpha > 0.61803$$

3)

$$E(s) = \frac{R(s)}{1 + L_s} = \frac{\frac{1}{s}}{1 + \frac{s + \alpha}{s^3 + (1+\alpha)s^2 + (\alpha - 1)s + (1 - \alpha)}}$$

$$e_{ss} = \lim_{s \to 0} s * E(s) = \frac{1}{1 + \frac{\alpha}{1 - \alpha}} \leq 10\%$$

$$\alpha \geq 0.9$$

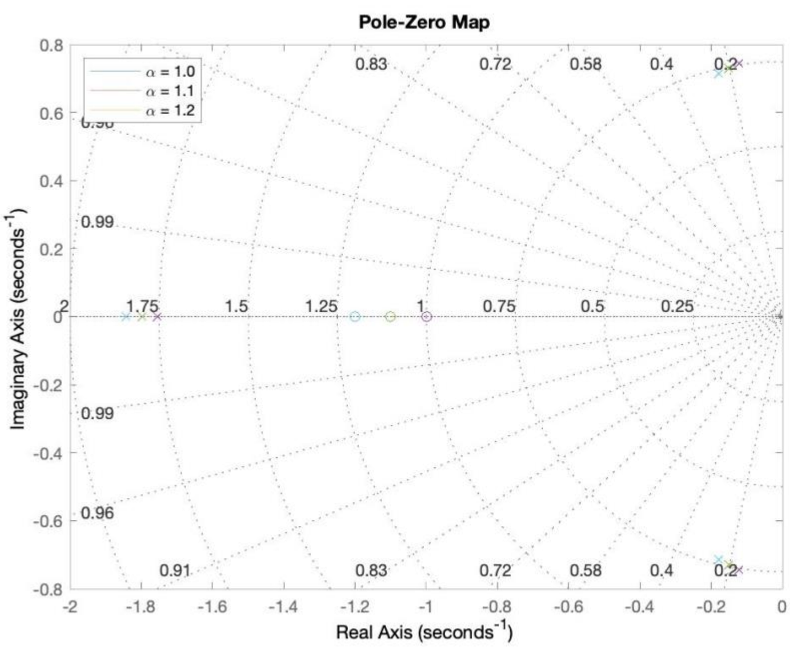
4)

$$T(s) = \frac{s + \alpha}{s^3 + (1+\alpha)s^2 + \alpha s + 1}$$

To satisfy the requirement in part 3, we choose:

$$\alpha = 1.0, \qquad 1.1, \qquad 1.2$$

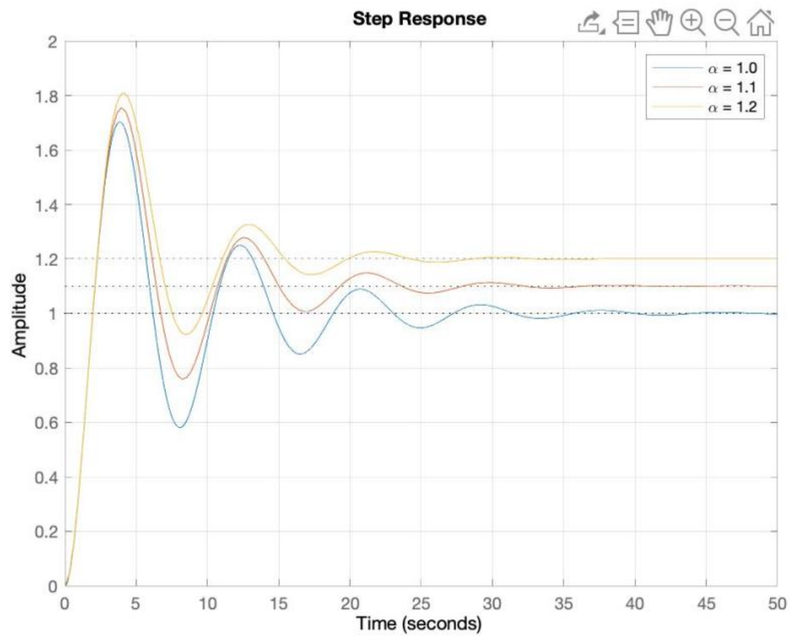
The roots are plotted as follows:



For a specific α, we have one real zero, one real pole, and two complex poles. This matches the root locus plot we obtained in part 1.

5)

Step response is printed as:



$\alpha = 1.2$ gives us the best result since the settling time would be less than the other two, and the same is the steady-state error.

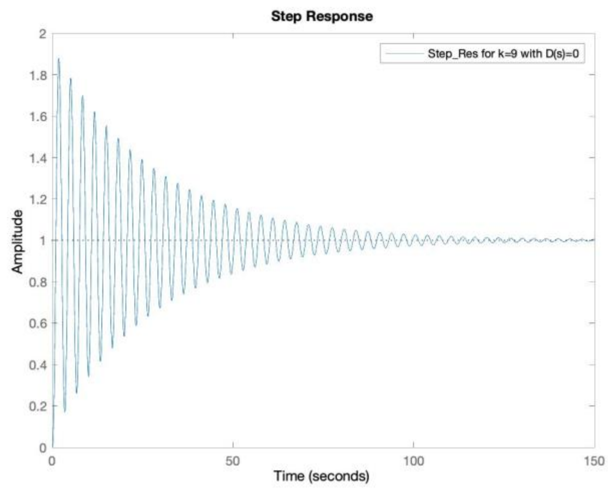
Problem 12

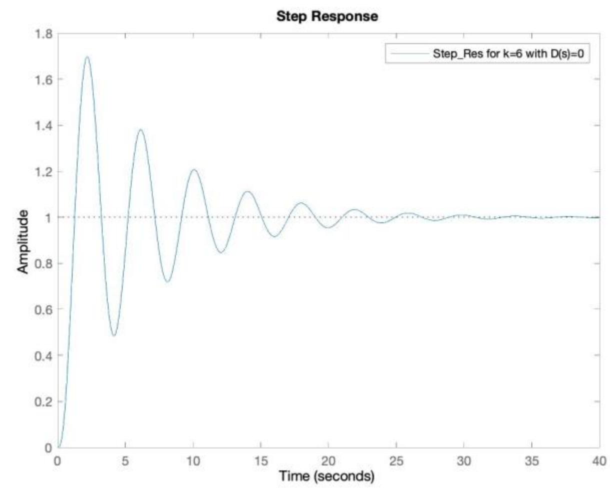
1.a)

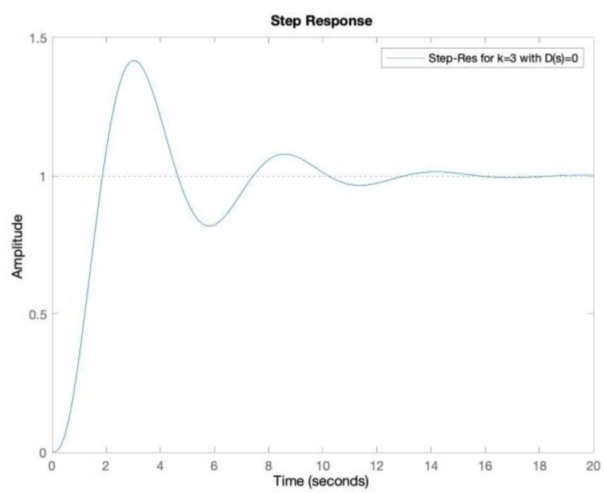
$$L_s = G_c * G = \frac{2K}{s(s+1)(s+4)}$$

$$T_D(s) = \frac{\frac{2}{s(s+1)(s+4)}}{1 + \frac{2K}{s(s+1)(s+4)}} = \frac{2}{s(s+1)(s+4) + 2K}$$

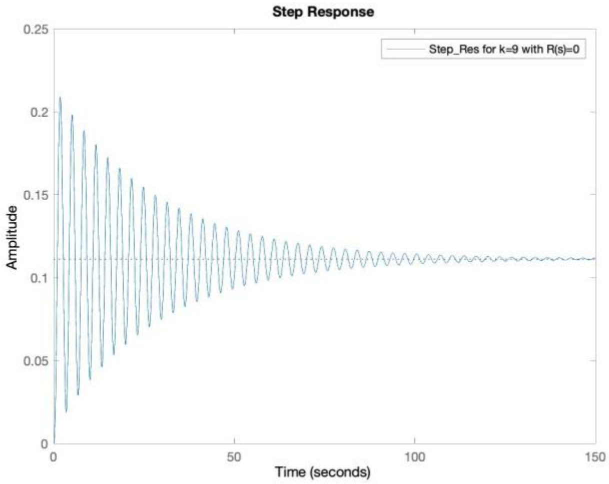
Input Step Response:

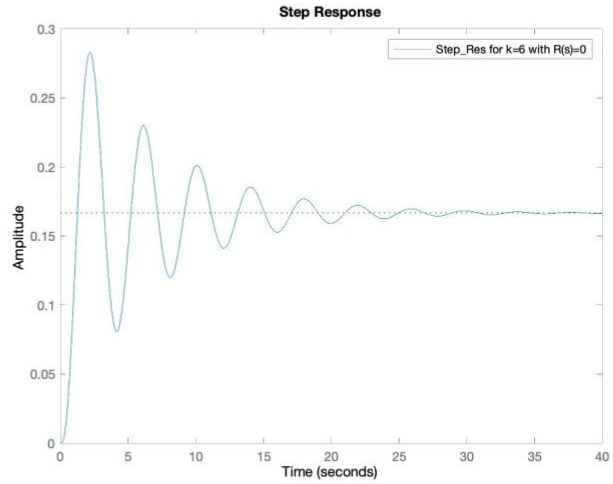


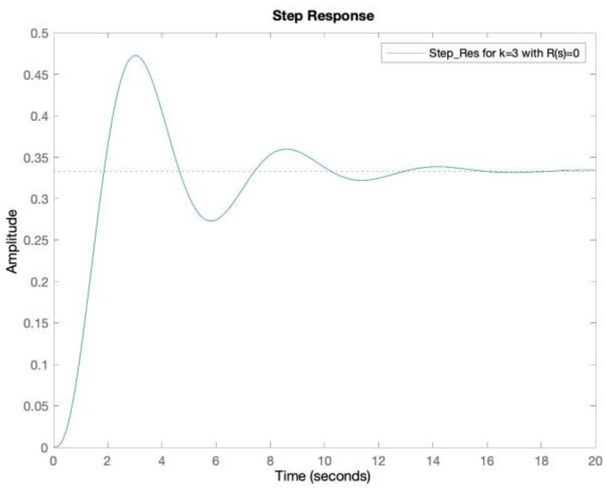




Noise Step Response:







Steady-state Error for a step input:

$$e_{ss} = \lim_{s \to 0} s * \frac{1}{s} * T_D(s) = \frac{1}{K}$$

1.b)

$$T_R(s) = \frac{2K}{s(s+1)(s+4)+2K} = \frac{2K}{s^3+5s^2+4s+2K}$$

The Routh Array:

s^3	1	4
s^2	5	$2K$
s^1	$4-\frac{2K}{5}$	0
s^0	$2K$	0

To make the system stable:

$$4-\frac{2K}{5} > 0, \qquad 2K > 0$$

Thus,

$$0 < K < 10$$

To minimize steady-state error:

$$K = 10$$

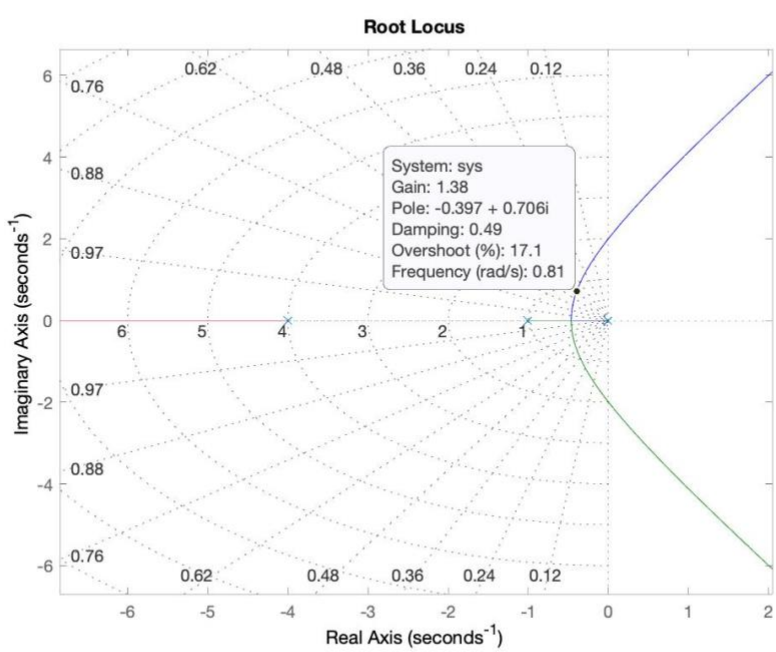
1.c)

$$G(s) = \frac{2}{s(s+1)(s+4)}$$

$$n = 3, \qquad m = 0, \qquad n-m = 3$$

$$\phi_{Asym} = \pm 60^o, 180^o$$

Root Locus:



When $\zeta = 0.5$,

$$\beta = \cos^{-1} 0.5 = 60^o, \qquad s = -0.4 + j0.7$$

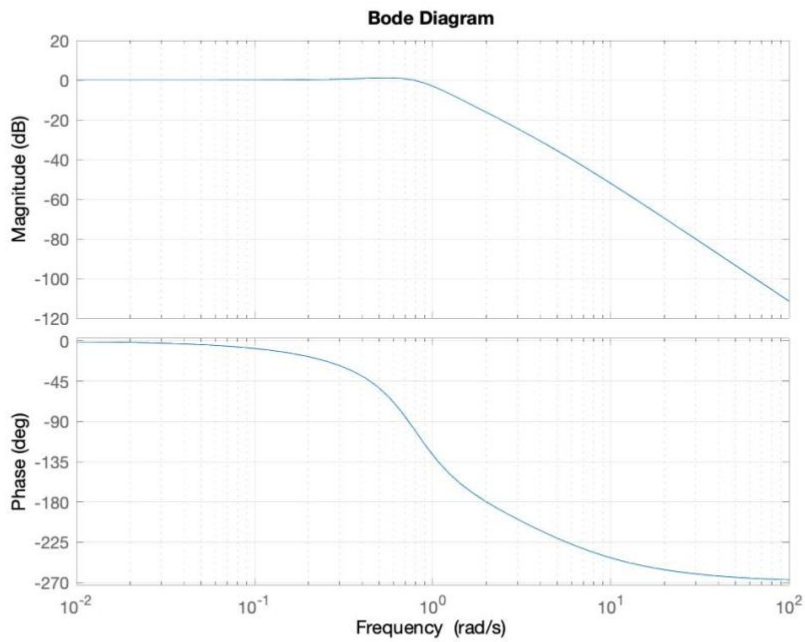
Since we have $\left|\frac{2K}{s(s+1)(s+4)}\right| = 1$,

$$K = 1.34$$

1.d)

$$T_R(s) = \frac{2 * 1.34}{s(s+1)(s+4) + 2 * 1.34} = \frac{2.68}{s^3 + 5s^2 + 4s + 2.68}$$

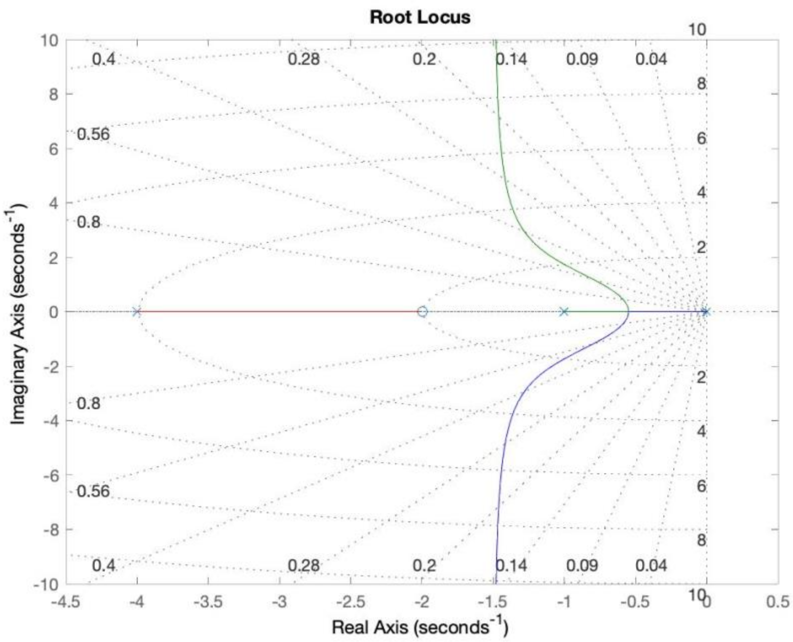
Bode plots:



2.a)

$$L_s = \frac{2K(s+2)}{s(s+1)(s+4)}$$

Root Locus:

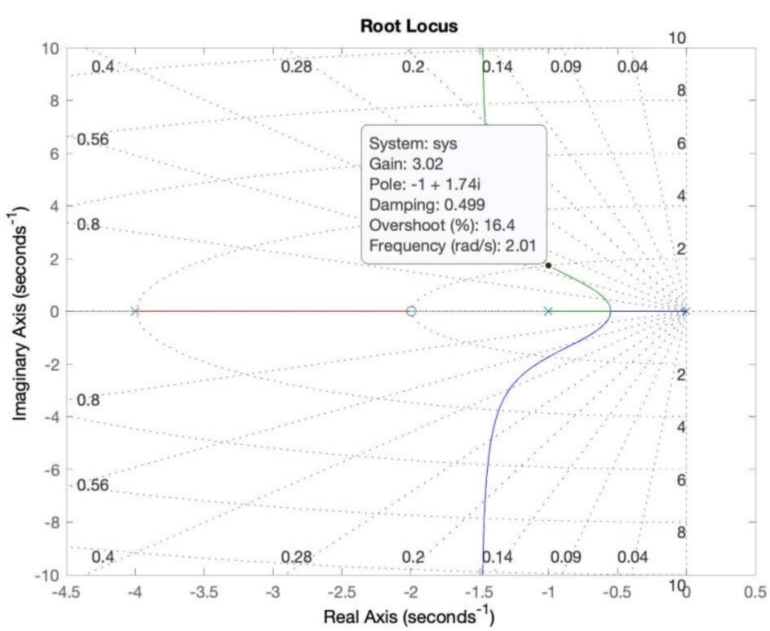


It is noted that we now have $\phi_{Asym} = \pm 90^o$ since $n - m = 2$. Meanwhile, the crossing point of asymptotes is shifted to the left due to the new zero.
PD controller makes the system faster and more stable.

2.b)

$$T_s = 4\ s, \qquad \zeta\omega_n = 1$$

From rlocus in 2.a):

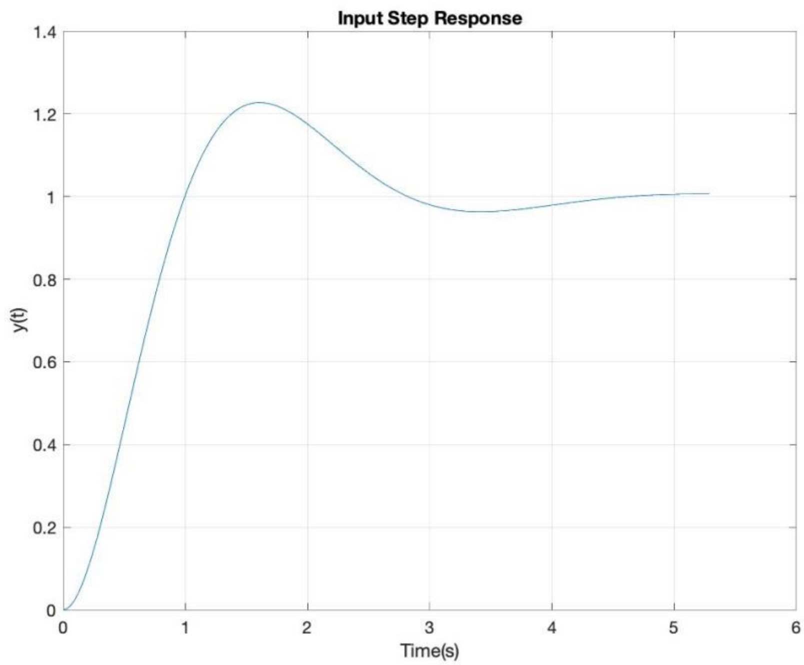


$$K = 3.02$$

2.c)

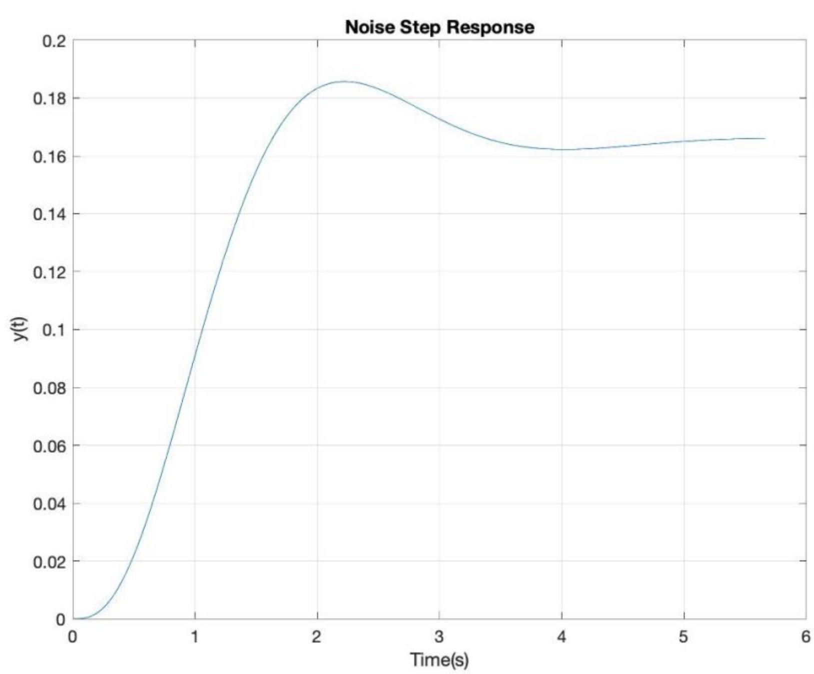
$$T_R(s) = \frac{2K(s+2)}{s^3 + 5s^2 + 4s + 2K(s+2)} = \frac{6.04s + 12.08}{s^3 + 5s^2 + 10.04s + 12.08}$$

Input Step Response:



$$T_D(s) = \frac{2}{s^3 + 5s^2 + 4s + 2K(s+2)} = \frac{2}{s^3 + 5s^2 + 10.04s + 12.08}$$

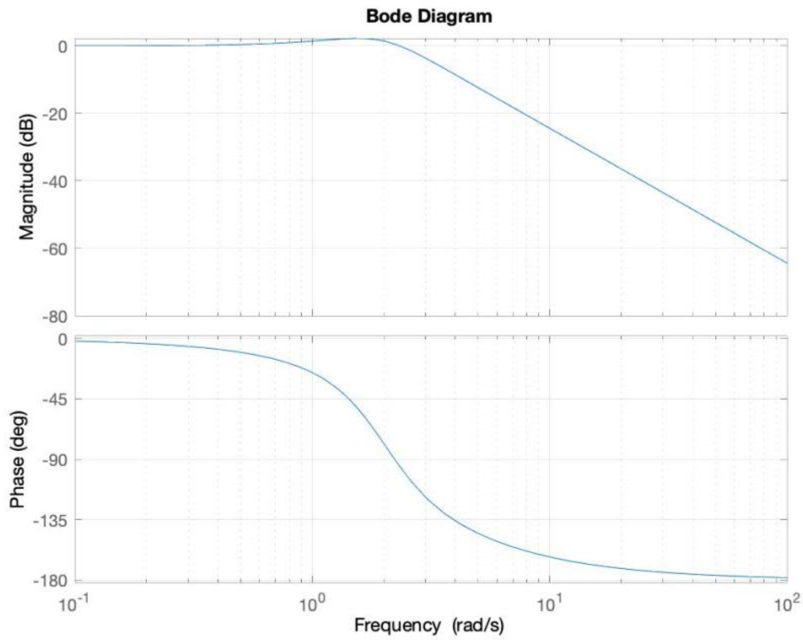
Nosie Step Response:



2.d)

$$T_R(s) = \frac{6.04s + 12.08}{s^3 + 5s^2 + 10.04s + 12.08}$$

Bode Plot:



<u>This is the end of the first section. Thanks for reading it.</u>

Code

Problem 1. CP7.10

```
>> sys=tf([0 1 0 0],[1 2 5 1])

sys =

        s^2
  ---------------------
  s^3 + 2 s^2 + 5 s + 1

Continuous-time transfer function.

>> rlocus(sys)
```

Problem 2. DP7.12

```
>> sys=tf([0 0 0 1 1.5],[1 17 84 148 80])

sys =

             s + 1.5
  ----------------------------------
  s^4 + 17 s^3 + 84 s^2 + 148 s + 80

Continuous-time transfer function.

>> rlocus(sys)

>> K=100

K =

   100
>> vpasolve(1+K*(s+1.5)/((s+1)*(s+2)*(s+4)*(s+10))==0)

ans =

                                 -11.376224942507200415181627931047
                                 -1.4460198425824747094898010247309
 - 2.0888776074551624376642855221109 - 3.1013131610271484903489641911771i
 - 2.0888776074551624376642855221109 + 3.1013131610271484903489641911771i

>> K=300

K =

   300
```

```
>>  vpasolve(1+K*(s+1.5)/((s+1)*(s+2)*(s+4)*(s+10))==0)
ans =

                              -12.939320814675075614406929484264
                              -1.4821505578182696046435102197956
 - 1.2892643137533273904747801479703 + 5.0964300428123383917409735291857i
 - 1.2892643137533273904747801479703 - 5.0964300428123383917409735291857i

>> K=600

K =

   600

>>  vpasolve(1+K*(s+1.5)/((s+1)*(s+2)*(s+4)*(s+10))==0)

ans =

                              -14.444522307111860227899924413701
                              -1.4911078580399334553177469171597
 - 0.53218491742410315839116433456944 - 6.7243612669765139076940964681778i
 - 0.53218491742410315839116433456944 + 6.7243612669765139076940964681778i

>> sys=tf([100 150], [1 17 84 248 230])

sys =

           100 s + 150
  -----------------------------------
  s^4 + 17 s^3 + 84 s^2 + 248 s + 230

Continuous-time transfer function.

>> [y,t]=step(sys)
>> plot(t,y)
>> grid
>> title('Step Response K=100')
>> xlabel('Time(s)')
>> ylabel('y(t)')
>> sys2=tf([300 450], [1 17 84 448 530])
```

Problem 3. DP8.1

```
>> sys=tf([1 2],[1 12 0 0])

sys =

     s + 2
  ------------
  s^3 + 12 s^2

Continuous-time transfer function.

>> bode(sys)
>> grid
>> sys1=tf([1 2],[1 12 1 2])

sys1 =

         s + 2
  --------------------
  s^3 + 12 s^2 + s + 2

Continuous-time transfer function.

>> bode(sys1)
>> sys=tf([50 100],[1 12 0 0])

sys =

   50 s + 100
  ------------
  s^3 + 12 s^2

Continuous-time transfer function.

>> bode(sys)

>> sys1=tf([50 100],[1 12 50 100])

sys1 =

         50 s + 100
  -------------------------
  s^3 + 12 s^2 + 50 s + 100

Continuous-time transfer function.

>> bode(sys1)
```

Problem 4. DP8.3

```
>> sys=tf([0 0 4.2 21],[1 7 12 31])

sys =

      4.2 s + 21
  ----------------------
  s^3 + 7 s^2 + 12 s + 31

Continuous-time transfer function.

>> bode(sys)
>> grid
```

Problem 5. CP8.6

```
>> sys=tf([0 0 0 25],[1 3 27 25])

sys =

          25
  ----------------------
  s^3 + 3 s^2 + 27 s + 25

Continuous-time transfer function.

>> bode(sys)
>> sys=tf([0 0 0 25],[1 3 27 50])

sys =

          25
  ----------------------
  s^3 + 3 s^2 + 27 s + 50

Continuous-time transfer function.

>> bode(sys)
```

Problem 6. DP10.1

```
>> comp=(sys/(1+sys))

comp =


  1000 s^6 + 23122 s^5 + 6.481e04 s^4 + 4.761e04 s^3 + 5016 s^2

                                            + 88 s

  ------------------------------------------------------------

  s^8 + 44.04 s^7 + 1566 s^6 + 2.491e04 s^5 + 6.649e04 s^4

                      + 4.769e04 s^3 + 5017 s^2 + 88 s


Continuous-time transfer function.

>> [y,t]=step(comp)
>> plot(t,y)
>> grid

>> s = tf('s');
>> [y,t]=step(comp/s)
>> plot(t,y)
>> grid
```

Problem 7. CP10.2

```
>> sys=tf([0 0 24.2],[1 8 24.2])

sys =

      24.2
  ----------------
  s^2 + 8 s + 24.2

Continuous-time transfer function.

>> margin(sys*5),grid;
>> margin(sys*6),grid;
>> margin(sys*7),grid;
```

Problem 8. CP10.4

a)

```
>> sys=tf([0 0 -10 -10.1 -0.1],[1 2.02 2.0501 0.0602 0.0202])

sys =

        -10 s^2 - 10.1 s - 0.1
  ------------------------------------------
  s^4 + 2.02 s^3 + 2.05 s^2 + 0.0602 s + 0.0202

Continuous-time transfer function.

>> margin(sys)
>> clear
>> sys=tf([0 0 0 100 101 1],[1 12.02 22.2501 20.5612 0.6222 0.202])

sys =

              100 s^2 + 101 s + 1
  -------------------------------------------------------
  s^5 + 12.02 s^4 + 22.25 s^3 + 20.56 s^2 + 0.6222 s + 0.202

Continuous-time transfer function.

>> margin(sys)
>> grid
>> sys1=(sys*tf([4 8],[1 22]))

sys1 =


           400 s^3 + 1204 s^2 + 812 s + 8

  ------------------------------------------------------------

  s^6 + 34.02 s^5 + 286.7 s^4 + 510.1 s^3 + 453 s^2 + 13.89 s

                                                    + 4.444


Continuous-time transfer function.

>> margin(sys1);grid
```

b)

```
>> closed_sys=feedback(sys1,1)

closed_sys =
```

```
            400 s^3 + 1204 s^2 + 812 s + 8
  ------------------------------------------------------------------
  s^6 + 34.02 s^5 + 286.7 s^4 + 910.1 s^3 + 1657 s^2 + 825.9 s  + 12.44
```

Continuous-time transfer function.

```
>> t=[0:0.02:5]
>> step(n*closed_sys,t)
```

Problem 9. CP11.3

```
a=[0 1; -1 -2];
b=[1; 1];
c=[1 -1];
d=[0];
p=[-1; -2];
K=acker(a, b, p);
```

Problem 10. DP11.7

```
>> sys=tf([0 0 0 1],[1 14 138 512])
sys =

           1
  ----------------------------
  s^3 + 14 s^2 + 138 s + 512
```

Continuous-time transfer function.

```
>> margin(sys)
>> grid

>> sys1=feedback(sys,1)

sys1 =

          1
  --------------------------
  s^3 + 14 s^2 + 138 s + 513
```

Continuous-time transfer function.

```
>> step(sys1)
>> grid
```

Problem 11

```
>> sys=tf([0 1 1 0],[1 1 0 1])

sys =

   s^2 + s
 -------------
 s^3 + s^2 + 1

Continuous-time transfer function.

>> rlocus(sys)
>> grid
```

```
alpha=[1 1.1 1.2];
for k=1:3
sys=tf([1 alpha(k)],[1 1+alpha(k) alpha(k) 1]);
pzmap(sys);
hold on
end
legend({'\alpha = 1.0','\alpha = 1.1','\alpha = 1.2'},'Location','northwest');
grid;

alpha=[1 1.1 1.2];
for k=1:3
sys=tf([1 alpha(k)],[1 1+alpha(k) alpha(k) 1]);
step(sys);
hold on
end
legend('\alpha = 1.0','\alpha = 1.1','\alpha = 1.2');
grid;
```

Problem 12

```
% gains
k=[3 6 9];
% plant
g=tf([0 0 0 2],[1 5 4 0]);

% input STEP RESPONSE

step(feedback(g*k(1),1));
legend('Step-Res for k=3 with D(s)=0');
figure;
grid;

step(feedback(g*k(2),1));
legend('Step_Res for k=6 with D(s)=0');
figure;
grid;

step(feedback(g*k(3),1));
legend('Step_Res for k=9 with D(s)=0')
figure;
```

```
grid;

% noise STEP RESPONSE

step(feedback(g,k(1)));
legend('Step_Res for k=3 with R(s)=0')
figure;
grid;

step(feedback(g,k(2)));
legend('Step_Res for k=6 with R(s)=0')
figure;
grid;

step(feedback(g,k(3)));
legend('Step_Res for k=9 with R(s)=0')
figure;
grid;
```

```
>> sys=tf([0 0 0 2],[1 5 4 0])

sys =

        2
  -----------------
  s^3 + 5 s^2 + 4 s

Continuous-time transfer function.

>> rlocus(sys)
>> grid

>> sys=tf([0 0 0 2.68],[1 5 4 2.68])

sys =

          2.68
  ------------------------
  s^3 + 5 s^2 + 4 s + 2.68

Continuous-time transfer function.

>> bode(sys)
>> grid

>> sys=tf([0 0 2 4],[1 5 4 0])

sys =

      2 s + 4
  -----------------
  s^3 + 5 s^2 + 4 s
```

Continuous-time transfer function.

```
>> rlocus(sys)
>> grid

>> sys=tf([0 0 6.04 12.08],[1 5 10.04 12.08])

sys =

        6.04 s + 12.08
  -----------------------------
  s^3 + 5 s^2 + 10.04 s + 12.08
```

Continuous-time transfer function.

```
>> [y,t]=step(sys)
>> plot(t,y)
>> grid
>> title('Input Step Response')
>> xlabel('Time(s)')
>> ylabel('y(t)')

>> sys1=tf([0 0 0 2],[1 5 10.04 12.08])

sys1 =

              2
  -----------------------------
  s^3 + 5 s^2 + 10.04 s + 12.08
```

Continuous-time transfer function.

```
>> [y,t]=step(sys1)
>> plot(t,y)
>> grid
>> title('Noise Step Response')
>> xlabel('Time(s)')
>> ylabel('y(t)')

>> bode(sys)
>> grid
```