E L E V E N

Design via

Frequency Response

SOLUTIONS TO CASE STUDIES CHALLENGES

**Antenna Control: Gain Design**

**a.** The required phase margin for 25% overshoot ( = 0.404), found from Eq. (10.73), is 43.49o.

From the solution to the Case Study Challenge problem of Chapter 10, G(s) = .

Using the Bode plots for K = 1 from the solution to the Case Study Challenge problem of Chapter 10, we find the required phase margin at  = 1.35 rad/s, where the magnitude response is -14 dB. Hence, K = 5.01 (14 dB).

**b.**

**Program:**

%Input system

numg=50.88;

deng=poly([0 -1.32 -100]);

G=tf(numg,deng);

%Percent Overshoot to Damping Ratio to Phase Margin

Po=input('Type %OS ');

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

Pm=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi);

fprintf('\nPercent Overshoot = %g',Po)

fprintf(', Damping Ratio = %g',z)

fprintf(', Phase Margin = %g',Pm)

%Get Bode data

bode(G)

pause

w=0.01:0.05:1000;%Step size can be increased if memory low.

[M,P]=bode(G,w);

M=M(:,:);

P=P(:,:);

Ph=-180+Pm;

for i=1:1:length(P);

if P(i)-Ph<=0;

M=M(i);

K=1/M;

fprintf(', Frequency = %g',w(i))

fprintf(', Phase = %g',P(i))

fprintf(', Magnitude = %g',M)

fprintf(', Magnitude (dB) = %g',20\*log10(M))

fprintf(', K = %g',K)

break

end

end

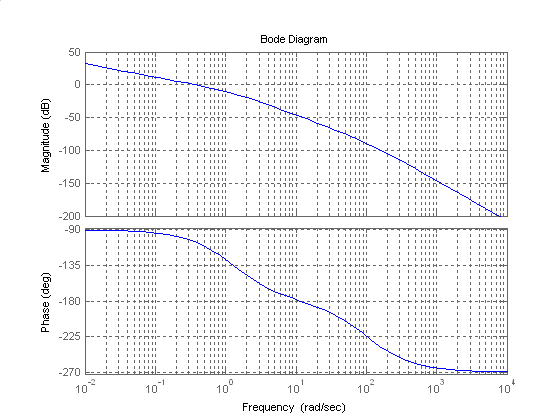
T=feedback(K\*G,1);

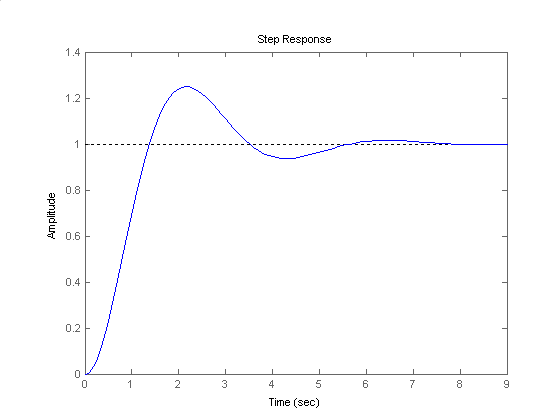
step(T)

**Computer response:**

Type %OS 25

Percent Overshoot = 25, Damping Ratio = 0.403713, Phase Margin = 43.463, Frequency = 1.36, Phase = -136.634, Magnitude = 0.197379, Magnitude (dB) = -14.094, K = 5.06641





**Antenna Control: Cascade Compensation Design**

**a.** From the solution to the previous Case Study Challenge in this chapter, G(s) = .

For Kv = 20, K = 51.89. Hence, the gain compensated system is

G(s) =

Using Eq. (10.73), 15% overshoot (i.e.  = 0.517) requires a phase margin of 53.18o. Using the Bode plots for K = 1 from the solution to the Case Study Challenge problem of Chapter 10, we find the required phase margin at  = 0.97 rad/s where the phase is -126.82o.

To speed up the system, we choose the compensated phase margin frequency to be 4.6 \* 0.97 = 4.46 rad/s. Choose the lag compensator break a decade below this frequency, or  = 0.446 rad/s.

At the phase margin frequency, the phase angle is -166.067o, or a phase margin of 13.93o. Using 5o leeway, we need to add 53.18o - 13.93o + 5o = 44.25o. From Figure 11.8,  = 0.15, or  = = 6.667. Using Eq. (11.15), the lag portion of the compensator is

GLag (s) = = .

Using Eqs. (11.9) and (11.15), T2 = = 0.579. From Eq. (11.15), the lead portion of the compensator is

GLead (s) =

The final forward path transfer function is

G(s)GLag (s)GLead(s) =

**b.**

**Program:**

%Input system \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

K=51.89;

numg=50.88\*K;

deng=poly([0 -1.32 -100]);

G=tf(numg,deng);

Po=15;

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

%Determine required phase margin\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Pmreq=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi)

phreq=Pmreq-(180)%required phase

w=0.1:0.01:10;

[M,P]=bode(G,w);

for i=1:1:length(P);%search for phase angle

if P(i)-phreq<=0;

ph=P(i)

w(i)

break

end

end

wpm=4.6\*w(i)

[M,P]=bode(G,wpm);%Find phase at wpm

Pmreqc=Pmreq-(180+P)+5%Find contribution required from compensator+5

beta=(1-sin(Pmreqc\*pi/180))/(1+sin(Pmreqc\*pi/180))

%Design lag compensator\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

zclag=wpm/10;

pclag=zclag\*beta;

Kclag=beta;

%Design lead compensator\*\*\*\*\*\*\*\*\*\*

zclead=wpm\*sqrt(beta);

pclead=zclead/beta;

Kclead=1/beta;

%Create compensated forward path\*\*\*\*\*\*\*\*\*

numgclag=Kclag\*[1 zclag];

dengclag=[1 pclag];

'Gclag(s)'

Gclag=tf(numgclag,dengclag);

Gclagzpk=zpk(Gclag)

numgclead=Kclead\*[1 zclead];

dengclead=[1 pclead];

'Gclead(s)'

Gclead=tf(numgclead,dengclead);

Gcleadzpk=zpk(Gclead)

Gc=Gclag\*Gclead;

'Ge(s)=G(s)\*Gclag(s)\*Gclead(s)'

Ge=Gc\*G;

Gezpk=zpk(Ge)

%Test lag-lead compensator\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

T=feedback(Ge,1);

bode(Ge)

title('Lag-lead Compensated')

[Gm,Pm,wcp,wcg]=margin(Ge);

'Compensated System Results'

fprintf('\nResulting Phase Margin = %g',Pm)

fprintf(', Resulting Phase Margin Frequency = %g',wcg)

pause

step(T)

title('Lag-lead Compensated')

**Computer response:**

Pmreq =

53.1718

phreq =

-126.8282

ph =

-126.8660

ans =

0.9700

wpm =

4.4620

Pmreqc =

44.2468

beta =

0.1780

ans =

Gclag(s)

Zero/pole/gain:

0.17803 (s+0.4462)

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

(s+0.07944)

ans =

Gclead(s)

Zero/pole/gain:

5.617 (s+1.883)

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

(s+10.58)

ans =

Ge(s)=G(s)\*Gclag(s)\*Gclead(s)

Zero/pole/gain:

2640.1632 (s+1.883) (s+0.4462)

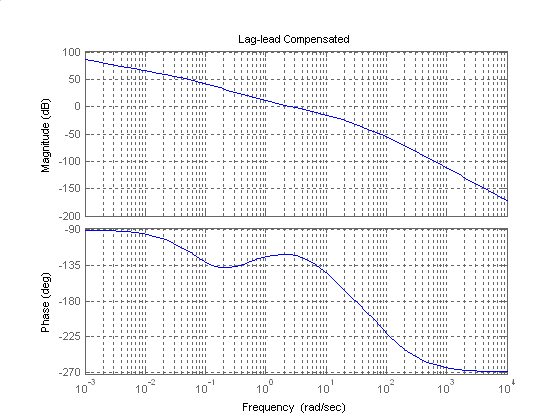
----------------------------------------

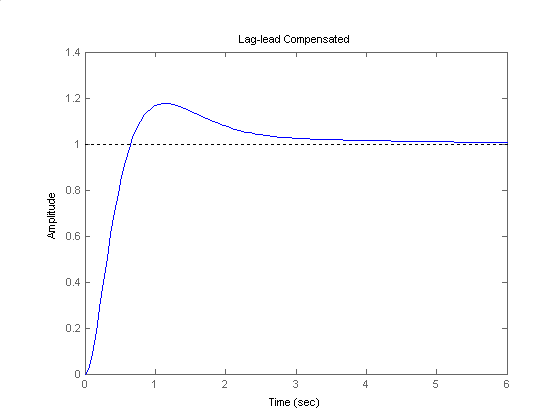
s (s+100) (s+10.58) (s+1.32) (s+0.07944)

ans =

Compensated System Results

Resulting Phase Margin = 57.6157, Resulting Phase Margin Frequency = 2.68618»





Answers to Review Questions

**1.** Steady-state error requirements can be designed simultaneously with transient response requirements.

**2.** Via the phase margin

**3.** The lag compensator is a low pass filter. Thus, while the low frequency gain is increased, the high-frequency gain at 180 o is decreased to make the system stable.

**4.** The lag network affects the phase angle at low frequencies, but not at high frequencies. For the compensated system, the phase plot is about the same as that of the uncompensated system around and above the phase margin frequency yielding the same transient response.

**5.** To compensate for the slight negative angle that the lag compensator has near the phase margin frequency

**6.** Compensated system has higher low-frequency gain than the uncompensated system designed to yield the same transient response; compensated and uncompensated system have the same phase margin frequency; the compensated system has lower gain around the phase margin frequency; the compensated and uncompensated system's have approximately the same phase values around the phase margin frequency.

**7.** The lead network is a high pass filter. It raises the gain at high frequencies. The phase margin frequency is increased.

**8.** Not only is the magnitude curve increased at higher frequencies, but so is the phase curve. Thus the 180o point moves up in frequency with the increase in gain.

**9.** To correct for the negative phase angle of the uncompensated system

**10.** When designing the lag portion of a lag-lead compensator, we do not worry about the transient design. The transient response will be considered when designing the lead portion of a lag-lead compensator.

**Solutions to Problems**

**1.**

**a.** Using a Bode plot one can find that the frequency response has when rad/sec. Since when , , 75dB must be added to the open loop response, so .

**b.** Using a Bode plot one can find that the frequency response has when rad/sec. Since when , , 53.4dB must be added to the open loop response, so .

**c.** Using a Bode plot one can find that the frequency response has when rad/sec. Since when , , 52.1dB must be added to the open loop response, so .

**2.**

**a.** For a 40o phase margin, the phase must be -140o when the magnitude plot is zero dB. The phase is -140o at  = 9.12 rad/s. At this frequency, the magnitude curve is –67.48 dB. Thus a 67.48 dB increase (K = 2365) will yield a 40o phase margin.

**b.** For a 40o phase margin, the phase must be -140o when the magnitude plot is zero dB. The phase is -140o at  = 2.76 rad/s. At this frequency, the magnitude curve is – 42.86 dB. Thus a 42.86 dB increase (K = 139) will yield a 40o phase margin.

**c.** For a 40o phase margin, the phase must be -140o when the magnitude plot is zero dB. The phase is -140o at  = 5.04 rad/s. At this frequency, the magnitude curve is – 54.4 dB. Thus a 54.4 dB increase (K = 525) will yield a 40o phase margin.

**3.**

10% OS corresponds to or from Eq (10.73) a

**a.** Looking at the phase diagram, where M = 59.7o (i.e. -120.3o), the phase margin frequency = 1.78 rad/s. At this frequency, the magnitude curve is -39.6 dB. Thus, the magnitude curve has to be raised by 39.6 dB (K = 95.5).

**b**. Looking at the phase diagram, where M = 59.7o (i.e.  = -120.3o), the phase margin frequency = 3.44 rad/s. At this frequency, the magnitude curve is – 50.5 dB. Thus the magnitude curve has to be raised by 50.5 dB (K = 335).

**c.** Looking at the phase diagram, where M = 59.7o (i.e.  = -120.3o), the phase margin frequency = 5.93 rad/s. At this frequency, the magnitude curve is – 62.5 dB. Thus the magnitude curve has to be raised by 62.5 dB (K = 1333.5).

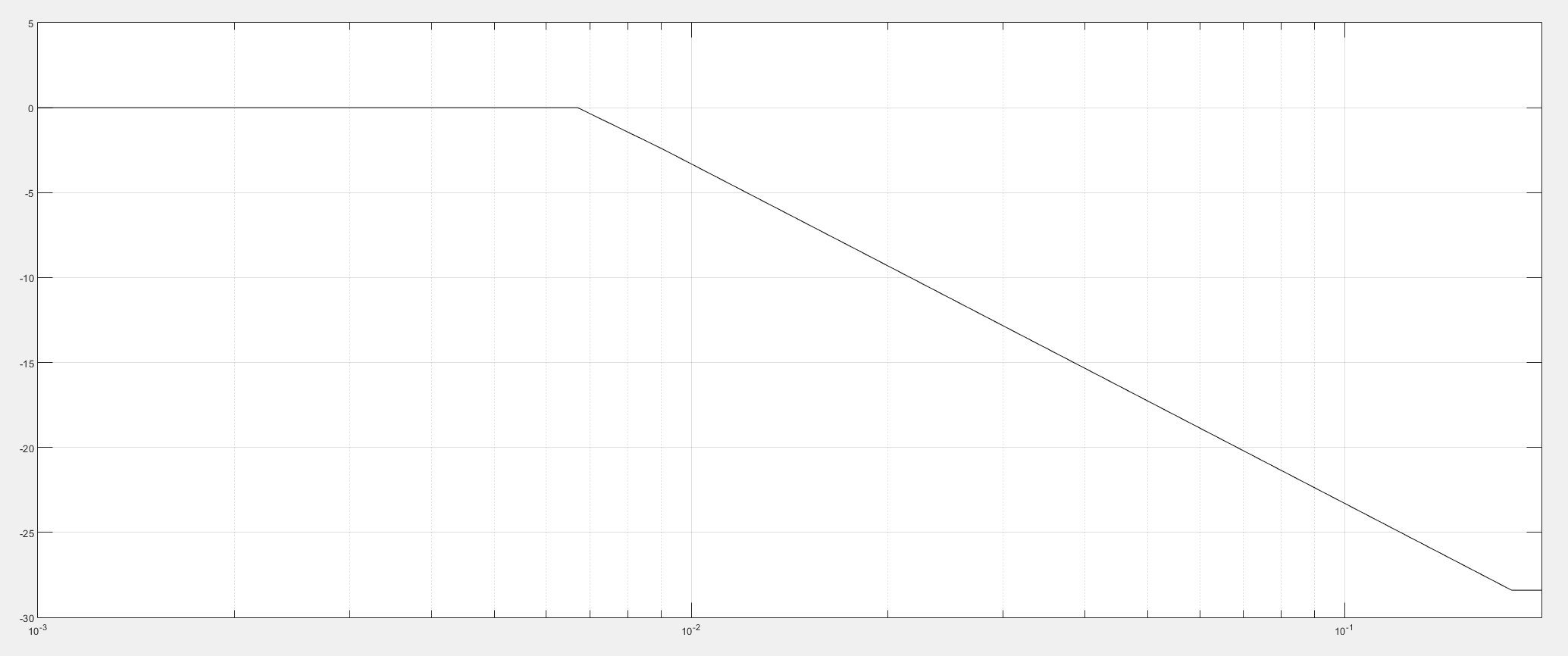
**4.**

For Kv = 50, K = 250. Plot the Bode plots for this gain.



Also, since %OS = 10%, . Using Eq. (10.73), M = 59.7o. Increasing M by 10o we will design for a phase margin of 69.7o. The phase margin frequency is where the phase angle is

69.7 - 180o = -110.3o, or M = 1.8 rad/s. At this frequency, the magnitude is 28.4 dB. Start the magnitude of the compensator at – 28.4 dB and draw it to 1 decade below M.



Then begin +20 dB/dec until zero dB is reached. Read the break frequencies as 0.0067 rad/s and 0.18 rad/s from the Bode plot and form a lag transfer function that has unity dc gain:

The compensated forward path is

**5.**

The MATLAB M-file given in Appendix B for Example 11.2 was modified as follows to assist in solving this problem:

'(ch11p7) Problem 11.7' % Display label.

numg = 1; % Set numerator of G(s)to unity.

deng = poly([-2 -8 -15]);

Gu = tf(numg, deng); % G(s)with numerator set to unity.

rlocus(Gu); % Plot the root locus of Gu(s).

title('Full Root Locus')

pause

axis ([- 4, 0, 0, 10]); % Zoom-in the root locus of Gu(s).

pos=10; % Desired percent overshoot.

z=(-log(pos/100))/(sqrt(pi^2+log(pos/100)^2));

% Calculate required damping ratio,z.

sgrid(z,0)

title('Root Locus Zoomed-in for Dominant Poles at 10% O.S.')

[K1,p]=rlocfind(Gu);

pause

Kpo = K1/(2\*8\*15) % Find uncompensated error constant.

ess = 1/(1+Kpo); % Find uncompensated step error.

ec = ess/5; % Required compensated step error.

Kpn = (1/ec)-1; % Required compensated error constant.

K = (2\*8\*15)\*Kpn; % Calculate required open-loop gain

numg=[K]; % Define numerator of G(s).

'G(s)' % Display label.

G=tf(numg,deng) % Create and display G(s).

Pm=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi)+10;

% Calculate required phase margin and

% add 10 degrees to compensate for the

% effect of the lag compensator.

w=0.01:0.01:100; % Set range of frequency from 0.01 to

% 100 in steps of 0.01.

bode(G) % Bode plot for G(s).

grid

pause

[M,P]=bode(G,w); % Get Bode data.

Ph=-180+Pm; % Calculate required phase angle.

for k=1:1:length(P); % Search Bode data for required phase

% angle.

if P(k)-Ph<=0; % If required phase angle is found,

% find the value of

M=M(k); % magnitude at the same frequency.

wf=w(k); % At this frequency the magnitude

% plot must go through 0 dB.

break % Stop the loop.

end % End if.

end % End for.

wh=wf/10; % Calculate the high-frequency break

% of the lag compensator.

wl=(wh/M); % Calculate the low-frequency break

% of the lag compensator; found from

% lag compensator, Gc(s)=Kc(s+wh)/(s+wl),

% high & low frequency gain requirements.

% At low w, gain=1. Thus, Kc\*wh/wl=1.

% At high w, gain=1/M. Thus Kc=1/M. Hence

% Kc=wl/wh=1/M, or wl=wh/M.

numc=[1 wh]; % Generate numerator of lag

% compensator, Gc(s).

denc=[1 wl]; % Generate denominator of lag

% compensator, Gc(s).

Kc=wl/wh; % Generate K for Gc(s).

'Lag compensator' % Display label.

Kc % Display lag compensator K.

'Gc(s)' % Display label.

Gc=tf(Kc\*numc,denc) % Create and display Gc(s).

'Gc(s)G(s)' % Display label.

GcG=Gc\*G % Create and display Gc(s)G(s).

T=feedback(GcG,1); % Create T(s).

step(T) % Generate a closed-loop, lag-

% compensated step response.

title('Closed-Loop Step Response for Lag-Compensated System')

% Add title to step response.

pause

**Uncompensated system:**  The full root locus for the uncompensated system, *Gu*(s), with the gain set to unity, is shown below.

****

Searching the zoomed-in root locus (shown below) along the 143.8o line (10% overshoot; e.g., ), the dominant poles were found to be at – 3.6161 ± j4.9831 with *K* = 425.3. Therefore, the uncompensated static error constant is and the steady-state error is which may be also obtained in the command window as: ess = 0.3608.



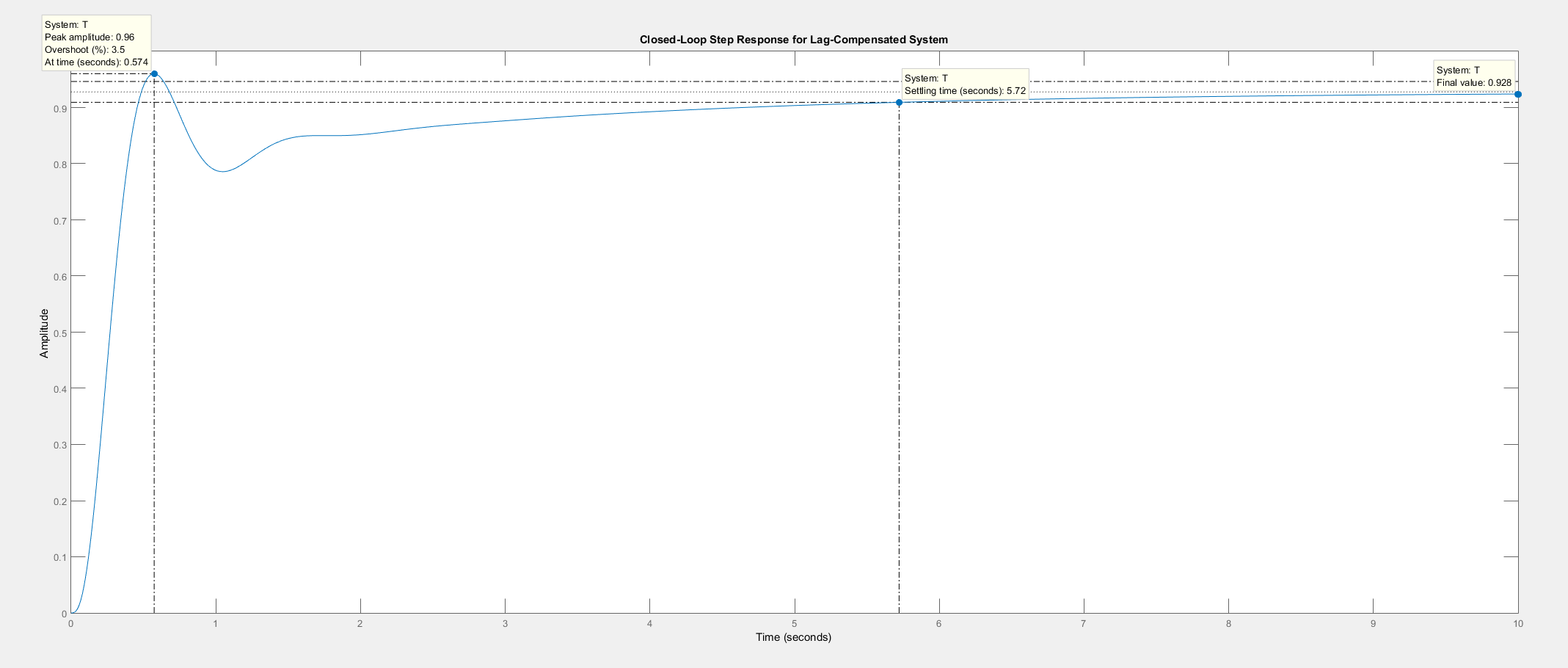
**Compensated system:**

For a 5 times improvement in steady-state error: , yielding,  and *K* = 3088.7**.** TheBodeplot obtained for the uncompensated *G(s)* at that gain is shown below. Its data along with the desired phase margin (calculated as 58.6o & adding 10o to compensate for the effect of the lag compensator) were used to generate the transfer function of the compensator as: *Gc = ( 0.2264 s + 0.1008)/( s + 0.1008)* e.g.;

****

Thus, the compensated forward path was found to be:

From the step response, shown below, we see that the closed-loop lag-compensated system has a final value of 0.928); e.g. a steady-state error of 0.072, and an overshoot of less than 3%.



**6.**

A 15% overshoot corresponds to . Using Eq. (10.73), M = 53.17o. In the uncompensated system this is achieved when K=72444. The uncompensated system is

Allowing for a 10o contribution from the PI controller, we want a phase margin of 63.18o, or a phase angle of -180o + 63.18o = -116.82o. This phase angle occurs at  = 12.7 rad/s where the magnitude is 3.44 dB. Thus, the PI controller should contribute – 3.44 dB at  = 12.7 rad/s. Selecting a break frequency a decade below the phase margin frequency,

This function has a high-frequency gain of zero dB. Since we want a high-frequency gain of -3.44 dB (a gain of 0.673),

The compensated forward path is

**7.**

**Program:**

%PI Compensator Design via Frequency Response

%Input system

G=zpk([],[-5 -10],1);

G=tf(G);

%Percent Overshoot to Damping Ratio to Phase Margin

Po=input('Type %OS ');

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

Pm=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi)+10;

fprintf('\nPercent Overshoot = %g',Po)

fprintf(', Damping Ratio = %g',z)

fprintf(', Phase Margin = %g',Pm)

%Get Bode data

bode(G)

title('Uncompensated')

pause

%Find frequency at desired phase margin and the gain at this frequency

w=logspace(-1,2,10000);

%w=.1:0.1:100;

[M,P,w]=bode(G,w);

Ph=-180+Pm

for i=1:1:length(P);

if P(i)-Ph<=0

Mag=M(i)

wf=w(i);

fprintf(', Frequency = %g',wf)

fprintf(', Phase = %g',P(i))

fprintf(', Magnitude = %g',Mag)

fprintf(', Magnitude (dB) = %g',20\*log10(Mag))

break

end

end

%Design PI compensator

%Break frequency is a decade below phase margin frequency

wh=wf/10;

%Magnitude is reciprocal of magnitude of G at the phase margin frequency

%so net magnitude is 0 dB at the phase margin frequency

Kc=1/Mag

'PI Compensator'

Gpi=tf(Kc\*[1 wh],[1 0])

bode(Gpi)

title(['PI compensator'])

pause

'G(s)Gpi(s)'

Ge=series(G,Gpi);

Ge=zpk(Ge)

bode(Ge)

title('PI Compensated')

[Gm,Pm,Wcp,Wcg]=margin(Ge);

'Gain margin(dB); Phase margin(deg.); 0 dB freq. (r/s);'

'180 deg. freq. (r/s)'

margins=[20\*log10(Gm),Pm,Wcg,Wcp]

pause

T=feedback(Ge,1);

step(T)

title('PI Compensated')

**Computer response:**

Type %OS 25

Percent Overshoot = 25, Damping Ratio = 0.403713, Phase Margin = 53.463

Ph =

-126.5370

Mag =

0.0037

, Frequency = 14.5518, Phase = -126.54, Magnitude = 0.00368082, Magnitude (dB) = -48.6811

Kc =

271.6786

ans =

PI Compensator

Transfer function:

271.7 s + 395.3

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

s

ans =

G(s)Gpi(s)

Zero/pole/gain:

271.6786 (s+1.455)

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

s (s+10) (s+5)

ans =

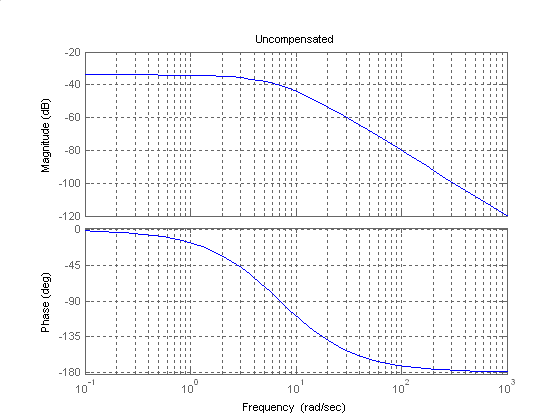
Gain margin(dB); Phase margin(deg.); 0 dB freq. (r/s);

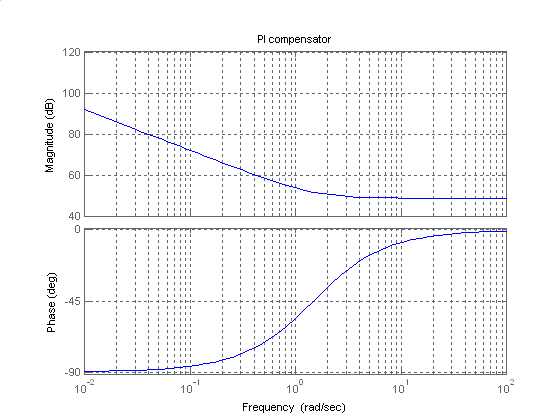
ans =

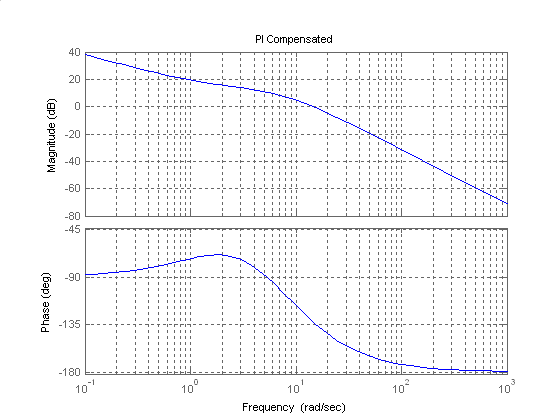
180 deg. freq. (r/s)

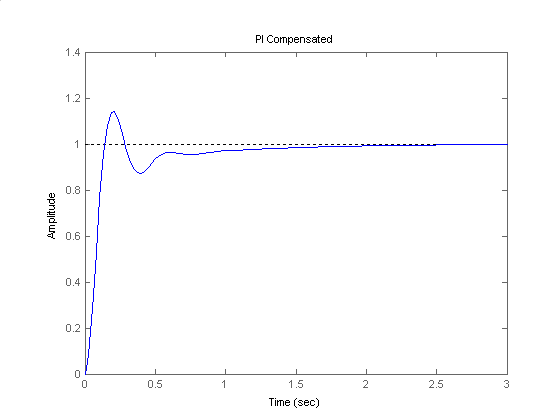
margins =

Inf 47.6277 14.5975 Inf









7. • Sp 11-6 (Lead Compensation) (Solution Complete)

**7. Design a compensator for the unity feedback system of Figure P.11.1 with   
G(s) = to yi**e**ld a Kv = 2 and a phase margin of 30** degr

**8ees.8.**

 or . We obtain the Bode diagram:



The uncompensated system shows a at a frequency of  rad/sec. To achieve the phase margin specification we need  and we add an arbitrary  to compensate for the fact that the phase margin will increase. So we will specify  of phase lead. Using equations (11.11) and (11.12) we get that  and  dB. Searching on the Bode diagram we find that  dB, so we chose  rad/sec as the frequency for the compensator. Using equations (11.6) and (11.9) we get that the compensator is

****

Applying this compensator it is found that the resulting . Adjusting slightly the poles and zeros of the compensator one gets that

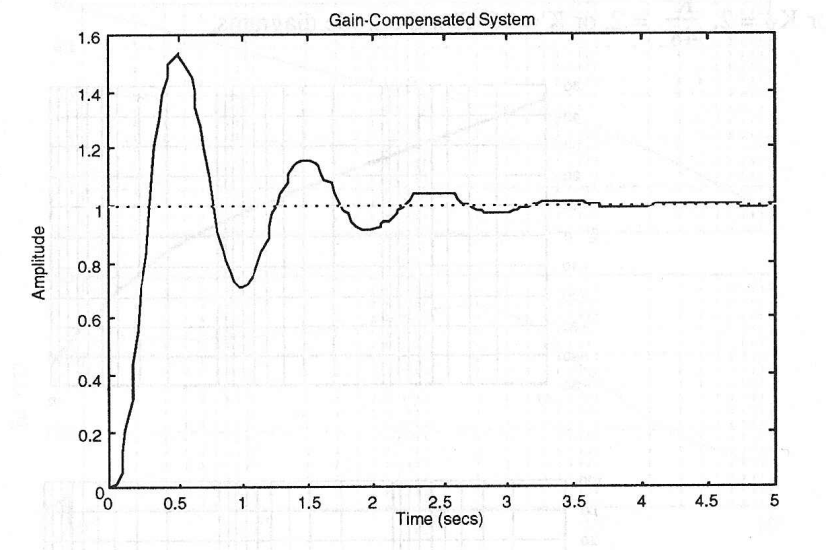
****

for a . The resulting open loop transfer function is:

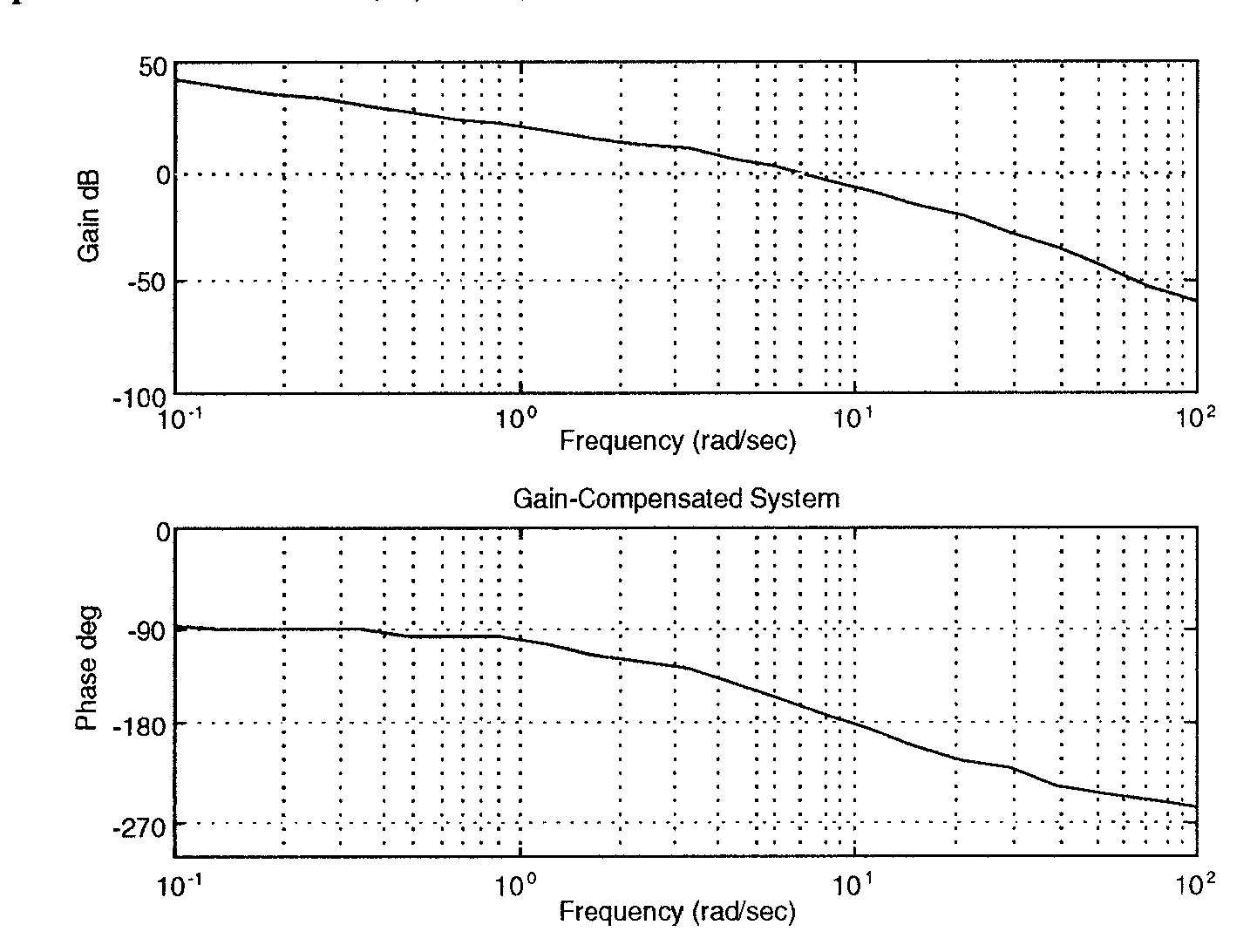
****

**9.**

a. Gain-compensated time response:



Bode plots for K = 1000 (Kv = 10):



The specifications for the gain compensated system are: K = 1000, percent overshoot = 10,  = 0.591155, peak time = 0.5 s, current phase margin = 22.5362o.

To meet the requirements: required phase margin (Eq. 10.73) = 58.5931o, required phase margin with correction factor of 20o = 78.5931, required bandwidth (Eq. 10.56) = 9.03591, required phase contribution from compensator = 78.5931o - 22.5362o = 56.0569o, compensator beta (Eq. 11.11) = 0.0931398, new phase margin frequency (Eq. 11.12) = 11.51.

Now design the compensator: Compensator gain Kc = 1/ = 10.7366, compensator zero (Eq. 11.12) = -3.51272, compensator pole = zc/ = -37.7144.

**Lead-compensated Bode plots:**



Lead-compensated phase margin = 50.2352.

**b.**

**Program:**

numg=1000;

deng=poly([0 -5 -20]);

G=tf(numg,deng);

numc=[1 3.51272];

denc=[1 37.7144];

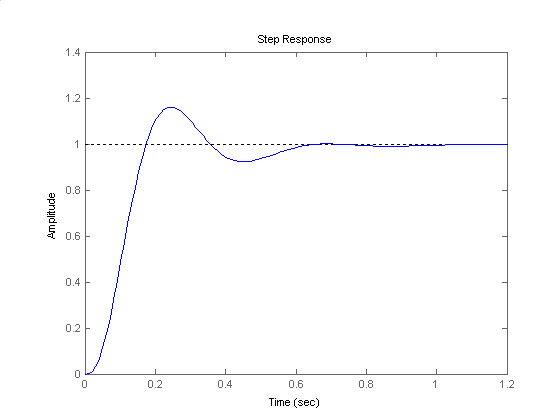
Gc=tf(numc,denc);

Ge=G\*10.7366\*Gc;

T=feedback(Ge,1);

step(T)

**Computer response:**



**10.**

The following MATLAB M-file was written to assist in solving this problem:

'G(s)' % Display label.

G=zpk(K\*Gp) % Define G(s), put K into it,

% convert

% to factored form, and display.

Pm=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi);

% Calculate phase margin.

Tsd= input('Type Desired Settling Time, Tsd=(Ts/2) = ');

% Input Desired Ts.

wn=4/(z\*Tsd); % Calculate required natural

% frequency.

wBW=wn\*sqrt((1-2\*z^2)+sqrt(4\*z^4-4\*z^2+2));

% Determine required bandwidth.

w=0.01:0.5:1000; % Set range of frequency from 0.01

% to 1000 in steps of 0.5.

bode(G) % Display the Bode plots.

pause

[M,P]=bode(G,w); % Get Bode data.

[Gm,Pm,Wcg,Wcp]=margin(G); % Find current phase margin.

Pmreq=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi);

% Calculate required phase margin.

Pmreqc=Pmreq+10; % Add a correction factor of 10

% degrees.

Pc=Pmreqc-Pm; % Calculate phase contribution

% required from lead compensator.

% Design lead compensator

beta=(1-sin(Pc\*pi/180))/(1+sin(Pc\*pi/180));

% Find compensator beta.

magpc=1/sqrt(beta); % Find compensator peak

% magnitude.

for k=1:1:length(M); % Find frequency at which

% uncompensated system has a

% magnitude of 1/magpc.

% This frequency will be the new

% phase margin frequency.

if M(k)-(1/magpc)<=0; % Look for peak magnitude.

wmax=w(k); % This is the frequency at the

% peak magnitude.

break % Stop the loop.

end % End if.

end % End for.

% Calculate lead compensator zero, pole, and gain.

zc=wmax\*sqrt(beta); % Calculate the lead compensator’s

% low break frequency.

pc=zc/beta; % Calculate the lead compensator’s

% high break frequency.

Kc=1/beta; % Calculate the lead compensator’s

% gain.

'Gc(s)' % Display label.

Gc=tf(Kc\*[1 zc],[1 pc]); % Create Gc(s).

Gc=zpk(Gc) % Convert Gc(s) to factored form

% and display.

'Ge(s)=G(s)Gc(s)' % Display label.

Ge=G\*Gc % Form Ge(s)=Gc(s)G(s).

Ge=minreal(Ge); % Cancel common factors.

Kp=dcgain(Ge) % Calculate Kp.

T=feedback(Ge,1); % Find T(s).

step(T) % Generate closed-loop, lead-

% compensated step response.

title('Lead-Compensated Step Response')

% Add title to lead-compensated

% step response.

pause

**Uncompensated System:** The full root locus for the uncompensated system, *Gp*(s), with the gain set to unity, is shown below. Searching the zoomed-in locus (shown below the full locus) along the ζ = 0.456 line (20% O.S.), find the dominant pole Q = ̶ 7.54 + j 14.7 with a gain of 239.

Hence:

1. Ts = 4 / ζ ωn = 4 / (0.456×16.5) = 0.532 sec;
2. .





1. The Bode plot for the uncompensated system is shown below. From that plot we see that the phase margin and the phase-margin frequency are 30.8 degrees and 26.4 rad/sec, respectively.



1. As could be seen from the above M-file, frequency response techniques were used to design a compensator that would yield a threefold improvement in *KP*and a twofold reduction in settling time while keeping the overshoot at 20%. That lead compensator was found to have a transfer function:

*GC* (s) =

**Compensated system:**

Thus, the compensated forward path was found to be:

*Ge* (s) = *G*(s) *GC* (s) =

The step response of the lead-compensated system, shown below, indicates that all requirements are met: We obtained more than threefold improvement in the error constant (*KP*was increased more than 3.2 times to 25.84) and the settling time was reduced to 0.186 seconds (almost 2.9 fold reduction) while the overshoot was kept almost at 20%.



**11.**

**a.** Bode plots and specifications for gain compensated system are the same as Problem 13. Required phase margin and required bandwidth is the same as Problem 13. Select a phase margin frequency 7 rad/s higher than the bandwidth = 9 + 7 = 16 rad/s. The phase angle at the new phase-margin frequency is -201.60. The phase contribution required from the compensator is –1800 + 201.60 + 58.590 = 80.20 at the phase-margin frequency. Using the geometry below:



tan (80.2) = . Therefore, zc = 2.76. Thus, Gc (s) = .

The PD compensated Bode plots:



Compensated phase margin is 62.942o.

**b.**

**Program:**

numg=1000;

deng=poly([0 -5 -20]);

G=tf(numg,deng);

numc=(1/2.76)\*[1 2.76];

denc=1;

Gc=tf(numc,denc);

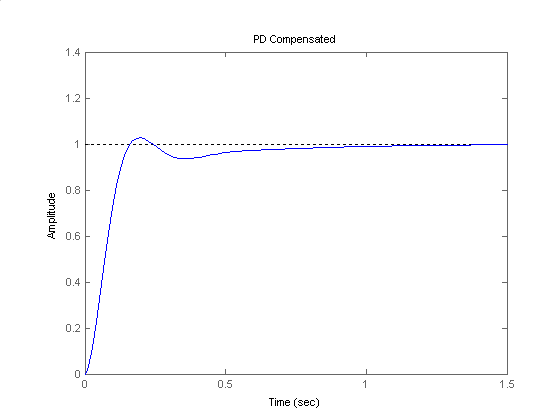
Ge=G\*Gc;

T=feedback(Ge,1);

step(T)

title('PD Compensated')

**Computer response:**



**12.**

**Program:**

%Lead Compensator Design via Frequency Response

%Input system

K=input('Type K to meet steady-state error ');

numg=K\*[1 1];

deng=poly([0 -2 -6]);

'Open-loop system'

'G(s)'

G=tf(numg,deng)

%Generate uncompensated step response

T=feedback(G,1);

step(T)

title('Gain Compensated')

%Input transient response specifications

Po=input('Type %OS ');

%Ts=input('Type settling time ');

Tp=input('Type peak time ');

%Determine required bandwidth

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

%wn=4/(z\*Ts);

wn=pi/(Tp\*sqrt(1-z^2));

wBW=wn\*sqrt((1-2\*z^2)+sqrt(4\*z^4-4\*z^2+2));

%Make a Bode plot and get Bode data

%Get Bode data

bode(G)

title('Gain Compensated')

w=0.01:0.1:100;

[M,P]=bode(numg,deng,w);

%Find current phase margin

[Gm,Pm,wcp,wcg]=margin(M,P,w);

%Calculate required phase margin

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

Pmreq=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi);

%Add a correction factor of 10 degrees

Pmreqc=Pmreq+10;

%Calculate phase required from compensator

Pc=Pmreqc-Pm;

%Design lead compensator

%Find compensator beta and peak compensator magnitude

beta=(1-sin(Pc\*pi/180))/(1+sin(Pc\*pi/180));

magpc=1/sqrt(beta);

%Find frequency at which uncompensated system has a magnitude of 1/magpc

%This frequency will be the new phase margin frequency

for i=1:1:length(M);

if M(i)-(1/magpc)<=0;

wmax=w(i);

break

end

end

%Calculate the lead compensator's break frequencies

zc=wmax\*sqrt(beta);

pc=zc/beta;

Kc=1/beta;

numc=[1 zc];

denc=[1 pc];

'Gc(s)'

Gc=tf(numc,denc)

%Display data

fprintf('\nK = %g',K)

fprintf(' Percent Overshoot = %g',Po)

fprintf(', Damping Ratio = %g',z)

%fprintf(', Settling Time = %g',Ts)

fprintf(', Peak Time = %g',Tp)

fprintf(', Current Phase Margin = %g',Pm)

fprintf(', Required Phase Margin = %g',Pmreq)

fprintf(', Required Phase Margin with Correction Factor = %g',Pmreqc)

fprintf(', Required Bandwidth = %g',wBW)

fprintf(', Required Phase Contribution from Compensator = %g',Pc)

fprintf(', Compensator Beta = %g',beta)

fprintf(', New phase margin frequency = %g',wmax)

fprintf(' Compensator gain, Kc = %g',Kc)

fprintf(' Compensator zero,= %g',-zc)

fprintf(' Compensator pole,= %g',-pc)

'G(s)Gc(s)'

Ge=G\*Kc\*Gc

pause

%Generate compensated Bode plots

%Make a Bode plot and get Bode data

%Get Bode data

bode(Ge)

title('Lead Compensated')

w=0.01:0.1:1000;

[M,P]=bode(Ge,w);

%Find compensated phase margin

[Gm,Pm,wcp,wcg]=margin(M,P,w);

fprintf('\nCompensated Phase Margin,= %g',Pm)

pause

%Generate step response

T=feedback(Ge,1);

step(T)

title('Lead Compensated')

**Computer response:**

Type K to meet steady-state error 360

ans =

Open-loop system

ans =

G(s)

Transfer function:

360 s + 360

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

s^3 + 8 s^2 + 12 s

Type %OS 10

Type peak time 0.1

ans =

Gc(s)

Transfer function:

s + 11.71

---------

s + 77.44

K = 360 Percent Overshoot = 10, Damping Ratio = 0.591155, Peak Time = 0.1, Current Phase Margin = 21.0851, Required Phase Margin = 58.5931, Required Phase Margin with Correction Factor = 68.5931, Required Bandwidth = 45.1795, Required Phase Contribution from Compensator = 47.508, Compensator Beta = 0.151164, New phase margin frequency = 30.11 Compensator gain, Kc = 6.61532 Compensator zero,= -11.7067 Compensator pole,= -77.4437

ans =

G(s)Gc(s)

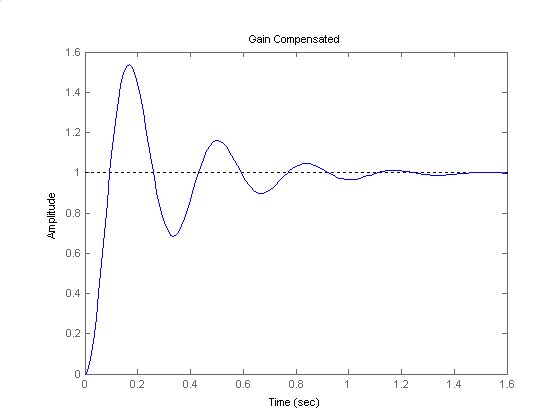
Transfer function:

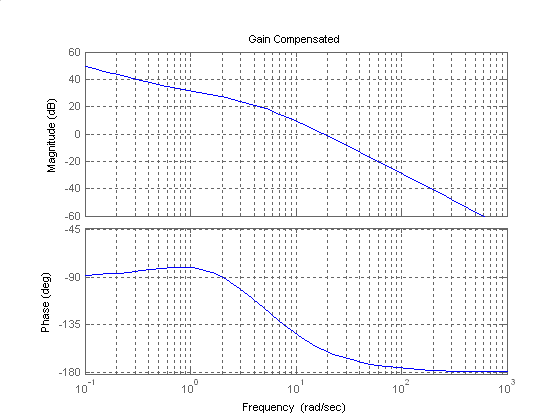
2382 s^2 + 3.026e004 s + 2.788e004

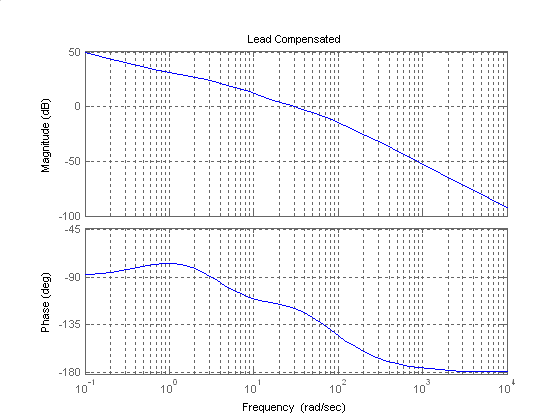
-------------------------------------

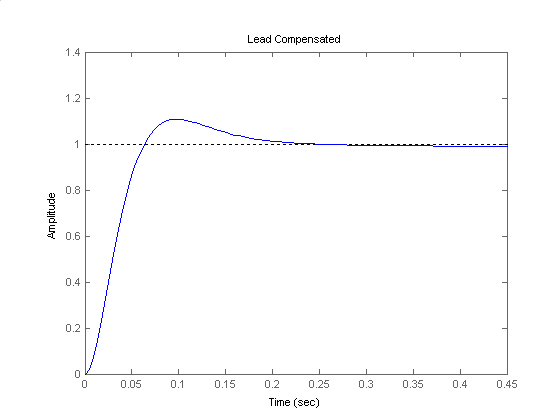
s^4 + 85.44 s^3 + 631.5 s^2 + 929.3 s

Compensated Phase Margin,= 60.676»









**13.**

, so . A 10% overshoot corresponds to a  damping factor. Using equation (4.42) . The required bandwidth is obtained using equation (10.54)  rad/sec. The phase margin is obtained using equation (10.73) with an additional  ; . A new phase margin frequency is chosen close to the bandwidth frequency  rad/sec. The Bode plot for  is



At the new phase margin frequency the phase angle is  . The contribution from the lead compensator must be . Using equation (11.11) .

Lag compensator design:  ;  ;  ; 

Lead compensator design: Using equations (11.6), (11.9) and (11.12)  ;  ;  ;  ; 

A unit step input simulation is performed:



Note that the design does not satisfy specs. Redesign if required.

**14.**

**Program:**

%Lag-Lead Compensator Design via Frequency Response

%Input system

K=input('Type K to meet steady-state error ');

numg=K\*[1 5];

deng=poly([0 -2 -10]);

G=tf(numg,deng);

'G(s)'

Gzpk=zpk(G)

%Input transient response specifications

Po=input('Type %OS ');

Ts=input('Type settling time ');

%Tp=input('Type peak time ');

T=feedback(G,1);

step(T)

title('Gain Compensated')

pause

%Determine required bandwidth

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

wn=4/(z\*Ts);

%wn=pi/(Tp\*sqrt(1-z^2));

wBW=wn\*sqrt((1-2\*z^2)+sqrt(4\*z^4-4\*z^2+2));

%wBW=(4/(Ts\*z))\*sqrt((1-2\*z^2)+sqrt(4\*z^4-4\*z^2+2));

%wBW=(pi/(Tp\*sqrt(1-z^2)))\*sqrt((1-2\*z^2)+sqrt(4\*z^4-4\*z^2+2));

%Determine required phase margin

Pmreq=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi)+5;

%Choose new phase margin frequency

wpm=0.8\*wBW;

%Determine additional phase lead required at the

%new phase margin frequency from the lead compensator

[M,P]=bode(G,wpm);

Pmreqc=Pmreq-(180+P);

beta=(1-sin(Pmreqc\*pi/180))/(1+sin(Pmreqc\*pi/180));

%Display data

fprintf('\nPercent Overshoot = %g',Po)

fprintf(', Settling Time = %g',Ts)

%fprintf(', Peak Time = %g',Tp)

fprintf(', Damping Ratio = %g',z)

fprintf(', Required Phase Margin = %g',Pmreq)

fprintf(', Required Bandwidth = %g',wBW)

fprintf(', New Phase Margin Frequency = %g',wpm)

fprintf(', Required Phase from Lead Compensator = %g',Pmreqc)

fprintf(', Beta = %g',beta)

bode(numg,deng)

title('Gain compensated')

pause

%Design lag compensator

zclag=wpm/10;

pclag=zclag\*beta;

Kclag=beta;

'Lag compensator'

'Gclag'

Gclag=tf(Kclag\*[1 zclag],[1 pclag]);

Gclagzpk=zpk(Gclag)

%Design lead compensator

zclead=wpm\*sqrt(beta);

pclead=zclead/beta;

Kclead=1/beta;

'Lead compensator'

'Gclead'

Gclead=tf(Kclead\*[1 zclead],[1 pclead]);

Gcleadzpk=zpk(Gclead)

%Create compensated forward path

'Gclag(s)Gclead(s)G(s)'

Ge=G\*Gclag\*Gclead;

Gezpk=zpk(Ge)

%Test lag-lead compensator

T=feedback(Ge,1);

bode(Ge)

title('Lag-lead Compensated')

[M,P,w]=bode(Ge);

[Gm,Pm,wcp,wcg]=margin(M,P,w);

'Compensated System Results'

fprintf('\nResulting Phase Margin = %g',Pm)

fprintf(', Resulting Phase Margin Frequency = %g',wcg)

pause

step(T)

title('Lag-lead Compensated')

Computer Response:

Type K to meet steady-state error 4000

ans =

G(s)

Gzpk =

4000 (s+5)

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

s (s+10) (s+2)

Continuous-time zero/pole/gain model.

Type %OS 10

Type settling time 0.2

Percent Overshoot = 10, Settling Time = 0.2, Damping Ratio = 0.591155, Required Phase Margin = 63.5931, Required Bandwidth = 39.2424, New Phase Margin Frequency = 31.394, Required Phase from Lead Compensator = 51.3288, Beta = 0.123126

ans =

Lag compensator

ans =

Gclag

Gclagzpk =

0.12313 (s+3.139)

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

(s+0.3865)

Continuous-time zero/pole/gain model.

ans =

Lead compensator

ans =

Gclead

Gcleadzpk =

8.1218 (s+11.02)

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

(s+89.47)

Continuous-time zero/pole/gain model.

ans =

Gclag(s)Gclead(s)G(s)

Gezpk =

4000 (s+11.02) (s+5) (s+3.139)

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

s (s+89.47) (s+10) (s+2) (s+0.3865)

Continuous-time zero/pole/gain model.

ans =

Compensated System Results

Resulting Phase Margin = 55.8196, Resulting Phase Margin Frequency = 41.1584



Note that the design does not satisfy specs. Redesign if required.

**15.**

The required bandwidth for a peak time of 1.8 seconds and ζ = 0.456 (i.e. 20% overshoot) is 2.588 rad/s. From the Bode diagram plotted for *K* = 1 and 20% overshoot, we get a ΦM = 48.15o, or a phase angle of −180o + 48.15o = −131.85o. We also find that this occurs at 1.08 rad/s and that at *K* = 13.5, the magnitude curve will intersect zero dB at 1.08 rad/s. Thus, the following function yields 20% overshoot:

The Bode diagram plotted for *K* = 13.5 is shown below.



**PI controller design:** Allowing for a 5o margin, we want ΦM = 48.1o + 5o = 53.1o or a phase angle of −180o + 53.1o = −126.9o. This angle occurs at ω = 0.92 rad/s where the magnitude is at 1.75 dB. The controller should contribute − 1.75 dB so that the magnitude curve passes through 0 dB at ω = 0.92 rad/s. Choosing the break frequency one decade below the phase margin frequency of 0.92 rad/s, and adjusting the controller's gain to yield − 1.75 dB at high frequencies, the ideal integral controller is

*GcPI(s)*

and the PI compensated forward path is

*GPI(s)* = *G(s)GcPI(s)*

Plotting the Bode diagram for the PI compensated system yields,



The magnitude is zero dB at ω = 1.6 rad/s. The phase at this frequency is −150.6o. Thus, we have a phase margin of −29.4o.

**PID controller design:** Let us increase the phase margin frequency to 4 rad/s, at which the phase is −191o. To obtain the required phase margin of 48.1o, the phase curve must be raised an additional 59.1o. Assume the following form for the proportional derivative part of the controller:

*GcPD(s)*

The angle contributed by *GcPD* is: *φc*

Letting ω = 4 rad/s → *KD* = 0.418. Hence, the PD controller is *GcPD(s)*

The final PID compensated forward path is:

*GPID(s)* = *GPI(s) GcPD(s)*

Letting , the magnitude of this function at 4 rad/s is -8.841 dB. Thus, K' must be adjusted to bring the magnitude to zero dB. Hence, K' = 2.767 (8.841 dB).

Thus,

*GPID(s)*

The PID compensated Bode plot follows:



The PID compensated time response is shown below:



**16.**

**Program:**

%Input system

numg1=1;

deng1=poly([0 -3 -6]);

G1=tf(numg1,deng1);

[numg2,deng2]=pade(0.5,5);

G2=tf(numg2,deng2);

'G(s)=G1(s)G2(s)'

G=G1\*G2;

Gzpk=zpk(G)

Tu=feedback(G,1);

step(Tu)

title('K = 1')

%Percent Overshoot to Damping Ratio to Phase Margin

Po=input('Type %OS ');

z=(-log(Po/100))/(sqrt(pi^2+log(Po/100)^2));

Pm=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi);

fprintf('\nPercent Overshoot = %g',Po)

fprintf(', Damping Ratio = %g',z)

fprintf(', Phase Margin = %g',Pm)

%Get Bode data

bode(G)

title('K = 1')

pause

w=0.1:0.01:100;

[M,P]=bode(G,w);

Ph=-180+Pm;

for i=1:1:length(P);

if P(i)-Ph<=0;

M=M(i);

K=1/M;

fprintf(', Frequency = %g',w(i))

fprintf(', Phase = %g',P(i))

fprintf(', Magnitude = %g',M)

fprintf(', Magnitude (dB) = %g',20\*log10(M))

fprintf(', K = %g',K)

break

end

end

T=feedback(K\*G,1);

step(T)

title('Gain Compensated')

**Computer response:**

ans =

G(s)=G1(s)G2(s)

Zero/pole/gain:

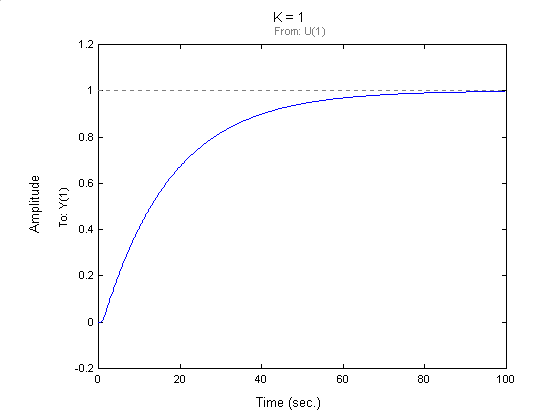
- (s-14.59) (s^2 - 26.82s + 228.4) (s^2 - 18.6s + 290.5)

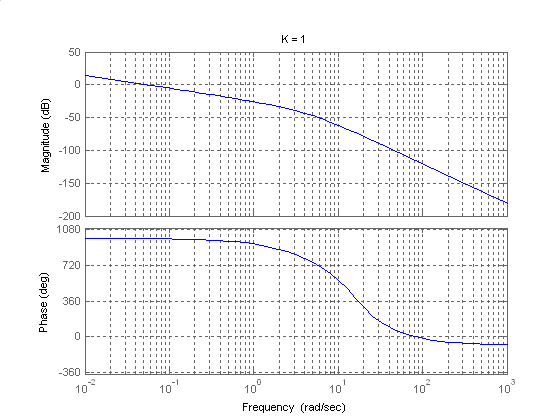
--------------------------------------------------------------------

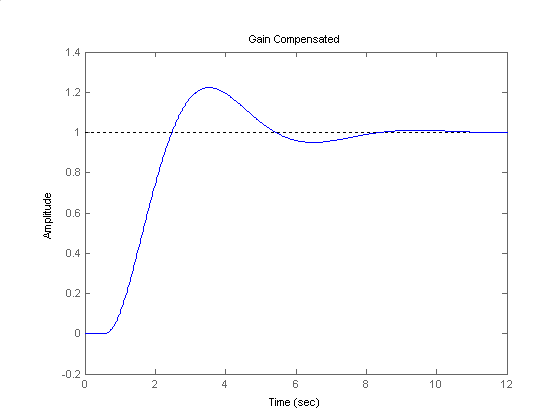
s (s+14.59) (s+6) (s+3) (s^2 + 26.82s + 228.4) (s^2 + 18.6s + 290.5)

Type %OS 20

Percent Overshoot = 20, Damping Ratio = 0.45595, Phase Margin = 48.1477, Frequency = 0.74, Phase = -132.087, Magnitude = 0.0723422, Magnitude (dB) = -22.8122, K = 13.8232»







Second-order approximation not valid.

**SOLUTIONS TO DESIGN PROBLEMS**

**17.**

1. We calculate for step inputs in the uncompensated system. , so . The uncompensated phase margin can also be obtained through a bode plot and is of 77.3°.

A tenfold improvement in steady state error requires multiplying the open loop gain by 10. The resulting open loop transfer function frequency response is shown in the following ode plot. The desired phase margin for the design will be 77.3°+10°=87.3°. It can be seen there that this phase value is achieved when . At this point the magnitude of the open loop transmission is 25.8db. So the lag compensator must provide -25.8db at .



The compensator is designed by postulating -25.8db at high frequencies with a higher break frequency of 10.1/10=1.01 rad/sec. The phase lag asymptote predicts -5.8 db one decade earlier at 0.101 rad/sec, and approximately 0db one octave before that at 0.101/2=0.0505 rad/sec. Maintaining unity dc gain for the compensator we get: 

**b.**

>> syms s

>> s=tf('s');

>> P=1361/(s^2+69\*s+70.85);

>> Gc=10\*0.05\*(s+1.01)/(s+0.0505);

>> T=P/(1+P);

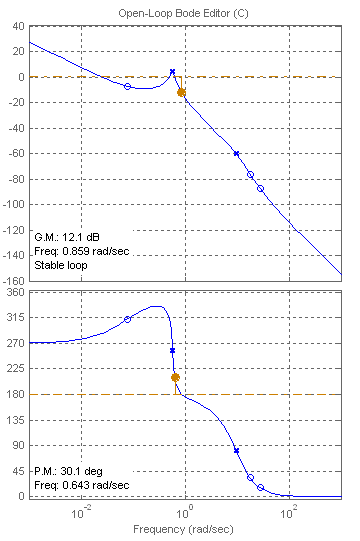
>> Tc=Gc\*P/(1+Gc\*P);

>> step(T,Tc,0.7)



**18.**

**a.** For a phase margin of 30° the gain is as shown in the following bode plot



1. The gain margin is 12.1db
2. From figure 10.48 a phase margin of 30° corresponds to a damping factor. This in turn corresponds to a %OS=37%. The bode plot above shows three crossovers of -7db. We use the largest one as the estimate for ; from equation 10.55 we get 

>>syms s

>>s=tf(‘s’);

>>P=(-34.16\*s^3-144.4\*s^2+7047\*s+557.2)/(s^5+13.18\*s^4+95.93\*s^3+14.61\*s^2+31.94\*s);

>> L=0.00135\*P;

>> T=L/(1+L);

>> t=linspace(0,350,5000);

>> step(T,t)



The estimate for the %OS is very inaccurate, the settling time estimate is reasonable.

1. The reason the estimate of %OS is very inaccurate is due to the multiple crossovers of the magnitude response. The hypothesis of a second order approximation is invalid.

**19.**

1. The %OS spec required a damping factor of , which in turn requires a phase margin of . The bandwidth requirement is obtained from . To obtained the compensator gain requirement to achieve this bandwidth obtain

. The compensator’s gain can be obtained from  or .

The required open loop crossover frequency is obtained by solving , giving . This is the frequency at which the lead compensator should provide maximum lead phase. So for the design of the compensator from Figure 11.8 let  with giving . So the designed compensator is . The gain of the compensator is now adjusted so that the maximum phase lead is provided at the crossover frequency giving. However a time domain simulation shows that although the settling time spec is satisfied the resulting %OS=22%. The parameters of the phase lead compensator are slightly adjusted to provide more phase lead giving 

**b**.

>> syms s

>> s=tf('s');

>> P=3.3333e4/s^2;

>> G=173.67\*15\*(s+1000)/(s+15000);

>> L=G\*P;

>> T=L/(1+L);

>> step(T,4e-3)

****

**20.**

We follow the step outlined in the chapter.

**a.**

* 1. We calculate the required . The required bandwidth is 
  2. The uncompensated system is  so . The steady state requirement requires  or 
  3. The bode plot for the uncompensated system is



Uncompensated

Uncompensated

* 1. The required phase margin for the design is 
  2. We arbitrarily select 
  3. It can be seen from the uncompensated bode plot that at the phase margin is 29°. The requirement in the phase lead network is going to be 59.2°-29°+5° (for the lag compensator contribution)=35.2°
  4. The lag compensator design is done by schooseng the higher break frequency as . From the lead compensator graphs, figure 11.8 in the text let or . So the lag compensator is given by 
  5. For the lead compensator design  and .  and Resulting in 
  6. The resulting lag-lead compensator is . The resulting compensated bode plot is shown above.

1. The step response simulation is

>> syms s

>> s=tf('s');

>> G=13.6\*5.8333e-4/(s+0.02)/(s+0.25)/(s+0.0833);

>> Gc = (s+0.01)\*(s+0.05)/(s+0.0025)/(s+0.2);

>> L=G\*Gc;

>> T=L/(1+L);

>> step(T)



The system exhibits a “long tail” response because compensation adds a pole-zero pair very close to the origin. However it can be seen that after 60sec the response is very close to steady state.

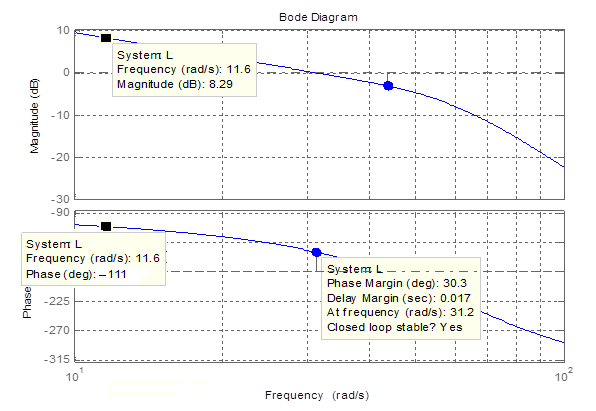


**21.**

**a**.

 from which we obtain  . Also 10% overshoot corresponds to a damping factor of , which in turn requires (equation(10.73)) a phase margin of , but since the lag compensator will contribute some phase lag the requirement is augmented by  to .

A bode plot with shows that the uncompensated phase margin is  at a frequency of  rad/sec, thus the need for compensation.



We look on the uncompensated Bode plot for the frequency at which the open loop transfer function has  of phase. This occurs when  rad/sec. This will be the new phase margin frequency. Thus at this frequency we want the gain to be  dB. The current gain is  dB.

To design the compensator we select the high break frequency as  rad/sec. We use dB as the high frequency asymptote of the compensator and we draw a -20 db/sec asymptote from the intersection of these points towards 0dB. We find that the 0dB line is intersected when rad/sec. Thus the lag compensator is



The step response is shown below:



**22.**

The controlled plant of Problem 10.52 was given by:

**.**

The following MATLAB M-file was written:

clc % Clear the Command window.

clf % Clear the Figure.

numg = 0.1111\*[4 5 1];

deng = [1 3.1 0.85 0.87 0.1111];

Gp = tf(numg, deng);

pos = 10; % Desired percent overshoot.

z=(-log(pos/100))/(sqrt(pi^2+log(pos/100)^2));

% Calculate required damping ratio,z.

Pm=atan(2\*z/(sqrt(-2\*z^2+sqrt(1+4\*z^4))))\*(180/pi);

% Calculate required phase margin.

w=0.01:0.01:100; % Set range of frequency from 0.01 to

% 100 in steps of 0.01.

bode(Gp) % Bode plot for Gp(s).

grid

pause

[M,P]=bode(Gp,w); % Get Bode data.

Ph=-180+Pm; % Calculate required phase angle.

for k=1:1:length(P); % Search Bode data for required phase

% angle.

if P(k)-Ph<=0; % If required phase angle is found,

% find the value of the magnitude at

M=M(k); % the same frequency.

wf=w(k); % At this frequency the magnitude

% plot must go through 0 dB.

break % Stop the loop.

end % End if.

end % End for.

'G(s)' % Display label for G(s).

G=zpk(Gp/M) %

bode(G) % Bode plot for G(s).

grid

pause

numgcpi = 0.9613\*poly([-0.0557]);

% Numerator of PI controller's TF.

dengcpi = poly([0]); % Denominator of PI controller's TF.

Gcpi=tf(numgcpi,dengcpi);

Gpi=zpk(G\*Gcpi)

bode(Gpi) % Bode plot for PI-controlled system.

grid

pause

numgcpd = 12.8463\*poly([-3.593]);

% Numerator of PD controller's TF.

dengcpd = 1; % Denominator of PD controller's TF.

Gcpd=tf(numgcpd,dengcpd);

Gpid=zpk(Gpi\*Gcpd)

bode(Gpid) % Bode plot for PID-controlled system.

grid

pause

T=feedback(Gpid,1); % Find T(s).

step(T) % Generate closed-loop, PID-Controlled

% step response.

title('PID-Controlled Step Response')

% Add title

pause

The required bandwidth for a settling time of 50 sec. and ζ = 0.456 (i.e. 10% overshoot) is **BW = 0.157 rad/s. The Bode plot (for *K* = 1 and 10% overshoot) gives a ΦM = 24.1o, or a phase angle = −180o + 24.1o = −155.9o. This occurs at 0.68 rad/s at which *K* = 0.2588, the magnitude curve will intersect zero dB at 0.68 rad/s. Thus, the following transfer function (in pole-zero form) yields 10% overshoot:



The Bode diagram plotted for *G(s)* is shown below.

**PI controller design:** Allowing for a 5o margin, we want ΦM = 50.1o + 5o = 55.1o or a phase angle of −180o + 55.1o = −124.9o. This angle occurs at ω = 0.557 rad/s where the magnitude is at 0.343 dB. The controller should contribute − 0.343 dB so that the magnitude curve passes through 0 dB at ω = 0.557 rad/s. Choosing the break frequency one decade below the phase margin frequency and adjusting the controller's gain to yield − 0.343 dB at high frequencies, the ideal integral controller is:



*GcPI(s)*

Thus, the PI compensated forward path is:



From the Bode diagram for the PI compensated system (shown below), we see that the phase margin is 46oat a frequency ω = 0.558 rad/s.



**PID controller design:** Let us increase the phase margin frequency to 2 rad/s, at which the phase is −154o. To obtain the required phase margin of 55.1o, the phase curve must be raised an additional 29.1o. Assume the following form for the proportional derivative part of the controller:

*GcPD(s)*

The angle contributed by *GcPD* is:

Letting ω = 2 rad/s → *KD* = 0.2783. Hence, the PD controller is *GcPD(s)*

The final PID compensated forward path is:

*GPID(s)* = *GPI(s) GcPD(s)*

Letting , the magnitude of this function at 2 rad/s is -33.28 dB. Thus, K' must be adjusted to bring the magnitude to zero dB. Hence, K' = 46.16 (33.2846 dB).

Thus,

*GPID(s)*

The PID compensated Bode plot follows:



The PID compensated time response is shown below. Since it satisfies all requirements, the design is acceptable.

.

****

**23.**

1. For an overdamped system . So for this system 
2. The bode plot is:



At -90° of open loop phase lag the crossover frequency of the open loop transmission equals the bandwidth of the closed loop system, so it can be seen that we have to drop the magnitude 73.8db. So. Note that in the low range of frequencies the phase of the loop transmission is approximately -90°, lowering the gain will maintain closed loop stability.

**c.**

>> syms s

>> s=tf(‘s’);

>> G=-2e-4\*(s^2+0.04\*s+0.0048)/(s+0.02)/s;

>> L=G\*P;

>> T=L/(1+L);

>> step(T)



No further gain adjustments are necessary.

**24.**

1. For a zero steady-state error for step inputs and a steady-state error for ramp inputs ≤2 %, the required value of *K* may be found from:

, where .

Hence: 

The following MATLAB file was used to plot the Bode magnitude and phase plots for that system and to obtain the response of the system, c(t), to a step input, *r(t)* = 4 *u(t)*.

numg = [1 0.6];

deng = poly ([0 -0.5858]);

G = 48.82\*tf(numg, deng);

bode (G);

grid

pause

T = feedback(G,1); %T is the closed-loop TF of the PI-controlled system

T = minreal(T);

step(4\*T);

axis

grid

xlabel ('Time')

ylabel ('Speed Sensor Output, c(t) in volts')

title ('PI-controlled Systems Response to a 4 volt Input Step')

The Bode magnitude and phase plots obtained are shown below with the minimum stability margins displayed on the phase plot.

Fora *%OS* ≤ 4.32 %, the damping ratio is . Using Eq. (10.73) we find the phase margin needed to meet the damping ratio requirement:





The phase margin found from the Bode plot obtained in step (1) is greater than the required value. Therefore, the response of the system, c(t), to a step input, *r(t)* = 4 *u(t)*, has been plotted and is shown below. The settling time, *Ts*, and the final value of the output are noted.



As could be seen from the graph and the analysis presented above all requirements are met. Therefore, the design has been completed.

1. 1) When the PI-controller zero, *ZI*, moves to – 0.01304:

.

The phase margin found from the Bode plot obtained is still greater than the required value. Therefore, c(t) was plotted and is shown below with the settling time, *Ts*, and the final value of the output noted on the graph. As could be seen from the graph, the settling and rise times are less by ~ 20% than the respective values obtained for *ZI* at – 0.0163.



2) When the PI-controller zero, *ZI*, moves to – 0.01956:

.

The phase margin found from the Bode plot obtained is still greater than the required value. Therefore, c(t) was plotted and is shown below with the settling time, *Ts*, and the final value of the output noted on the graph. In this case, however, the settling and rise times are higher by ~ 20% than the respective values obtained for *ZI* at – 0.0163. Nevertheless, all requirements are still met.



The responses obtained in all cases here are closer to the response of a first-order system rather than a second order system. Note that when *ZI* is at – 0.01956, for example, two of the closed-loop poles (at – 0.6002 and – 0.01956) are very close, respectively, to the closed-loop zeros located at – 0.5999 and – 0.0196. Therefore, the system behaves as if it has only one closed-loop pole, which is at – 40.66.

Since the PI-controller designed meets all requirements (even when pole-zero cancellation is not achieved and the controller’s zero changes by ± 20%), there no need to add a derivative mode.

**25.**

1. The resulting Nyquist diagram is



1. Using the gain margin, the new range of closed loop stability for K is 0<K<2.86.
2. We use equation (10.73) to find that a damping factor corresponds to a phase margin of

 , which corresponds to . Examination of the frequency response shows that this phase is achieved when  rad/sec with a magnitude of -0.973db (when K=1). Thus to achieve this damping factor K=1.12 .

1. The corresponding unit step response for the closed system is shown below:

