Assignment 3

Collaborated with: Aditya Rathi

Ruchita Sinha

Q1. (a)

Initialize transfer function

```
warning('off','all')
s = tf('s');
G = exp(-0.005*s)*(-s+3)/((s+2)*(s+1000));
```

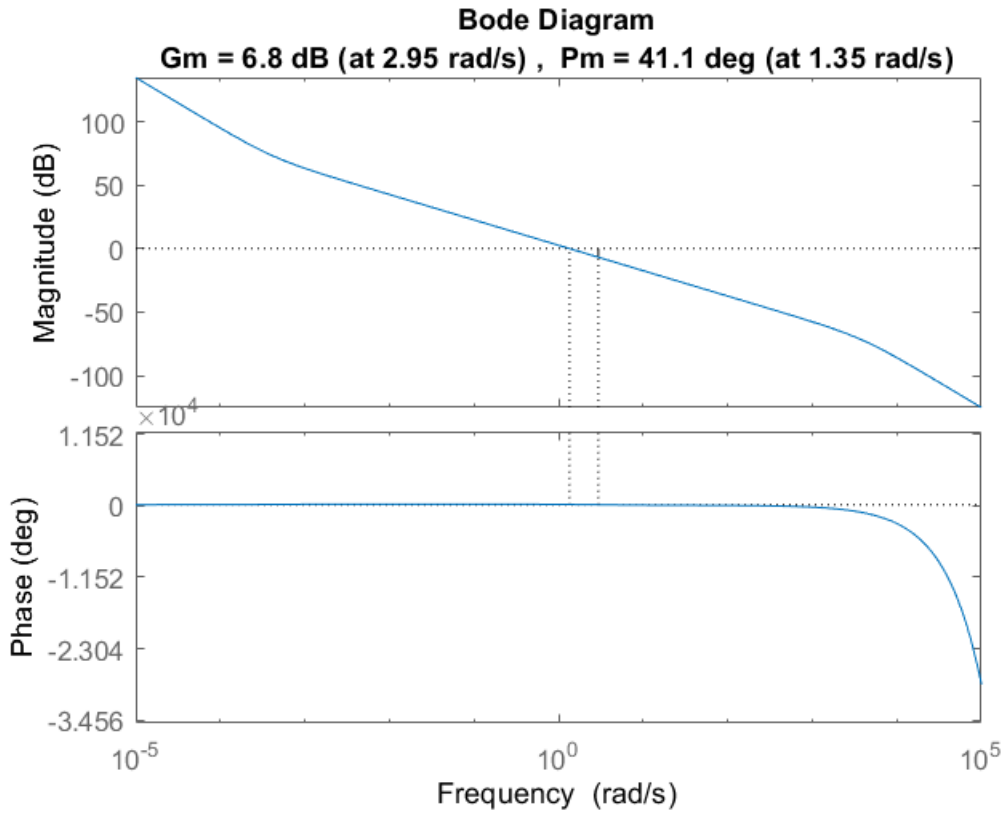
Set random intial values for wc, Gm, and Pm. Set 'a' and 'b' a bit wide apart from the wc

```
wc = 10;
a = 0.0003;
b = 3000;
Gm = 1;
Pm = 20;
```

Create a loop to update the value of wc so that we get the maximum value of wc where the required system parameters are satisfied. The value of K has been calculated by hand to eliminate the time delay error in MATLAB.

```
while(Pm<40 || Gm < 10^(6/20))
    L = ((-s+3)/(s+3))*exp(-0.005*s)*(s+a*wc)/((s^2)*(s+b*wc));
    K = (s+a*wc)*(s+2)*(s+1000)/((s+b*wc)*(s+3)*s^2);
    [mag2,phase] = bode(G*K,wc);
    K = K/mag2;
    [Gm,Pm,Wcg,Wcp] = margin(G*K);
    if (Pm > 40 && Gm > 10^(6/20))
        [Gm,Pm,Wcg,Wcp] = margin(G*K)
        margin(G*K)
        wc
        break
    end
    wc = wc - 0.05;
end
```

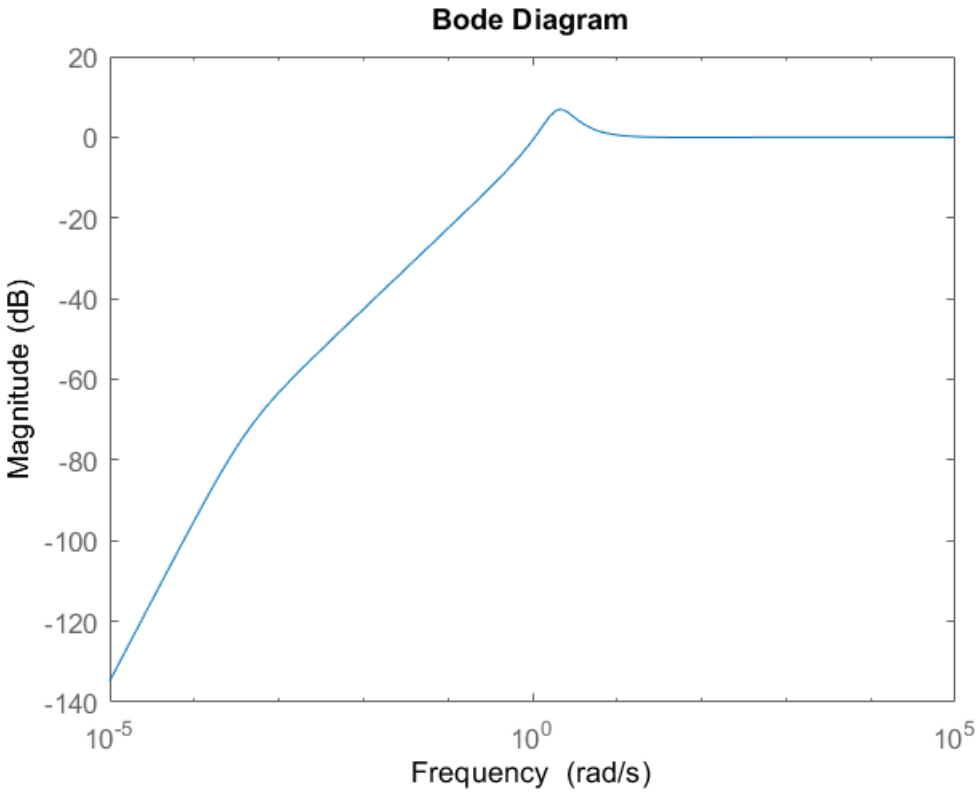
```
Gm = 2.1877
Pm = 41.1215
Wcg = 2.9535
Wcp = 1.3500
```




```
wc = 1.3500
```

magnitude bode plot of the sensitivity function.

```
bodemag(1/(1+G*K))
```

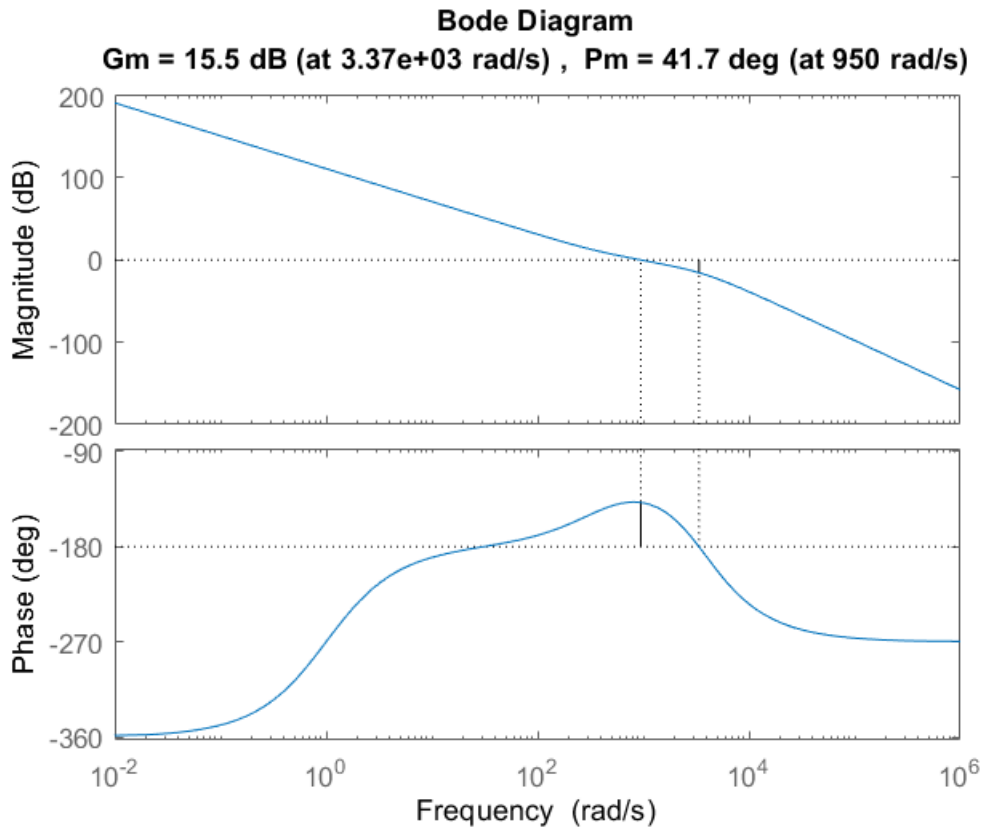

Bode Diagram
Magnitude (dB)
20
0
-20
-40
-60
-80
-100
-120
-140
10^-5
10^0
10^5
Frequency (rad/s)

Initializa transfer function

```
s = tf('s');
G = (s+5)/((-s+1)*(s+1000));
wc = 1100;
```

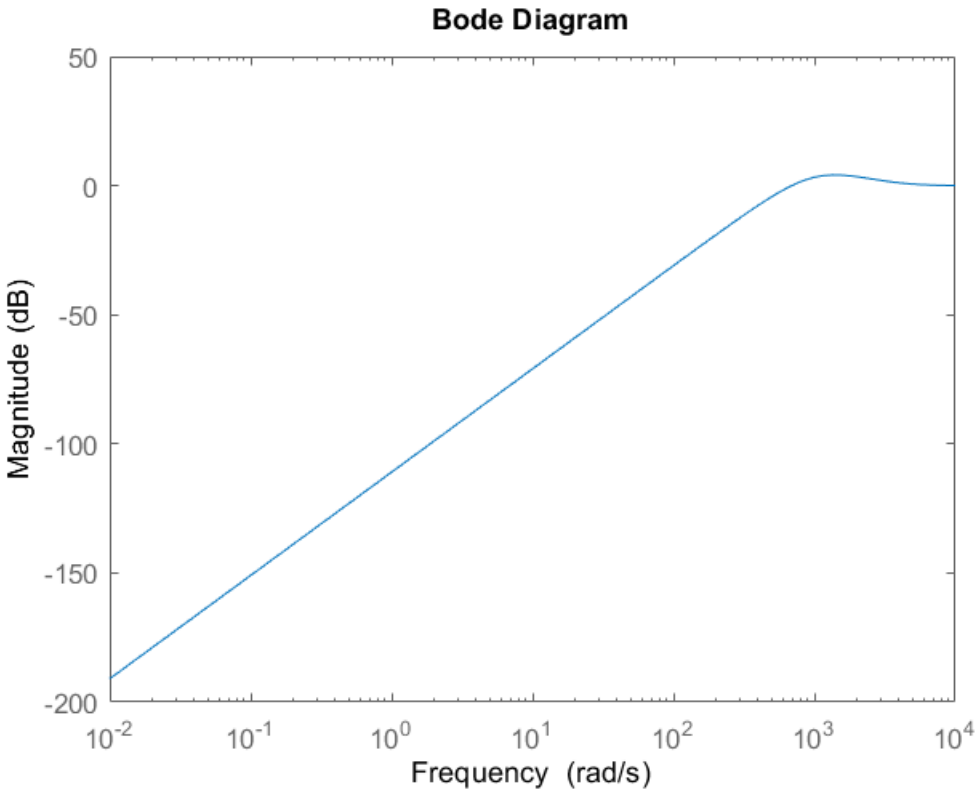
Obtain the desired loop shape using loopsyn.

```
[K,CL,GAM] = loopsyn(G,wc^2/s^2);
[Gm,Pm,Wcg,Wcp] = margin(G*K);
margin(G*K)
```



bode magnitude plot for the sensitivity function.

```
bodemag(1-CL)
```


Bode Diagram
Magnitude (dB)
50
0
-50
-100
-150
-200
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
Frequency (rad/s)

Q2. (A)

Given System,

```
Gv = 10;  %DC Gain
Gv1 = tf((2*pi*135)^2,[1 2*0.1*2*pi*135 (2*pi*135)^2]);
Gv2 = tf((2*pi*5500)^2,[1 2*0.03*2*pi*5500 (2*pi*5500)^2]);
Gv3 = tf((2*pi*8640)^2,[1 2*0.05*2*pi*8640 (2*pi*8640)^2]);
Gv4 = 7300^2/7650^2*tf([1 2*.015*2*pi*7650 (2*pi*7650)^2],[1 2*.03*2*pi*7300 (2*pi*7300)^2]);
P = Gv*Gv1*Gv2*Gv3*Gv4;
```

Initialize transfer funciton

```
s = tf('s');

M = 10^(6/20);
A = 1000;
BW = 3000*2*pi;
```

Initialize weights of 1st and 2nd order.

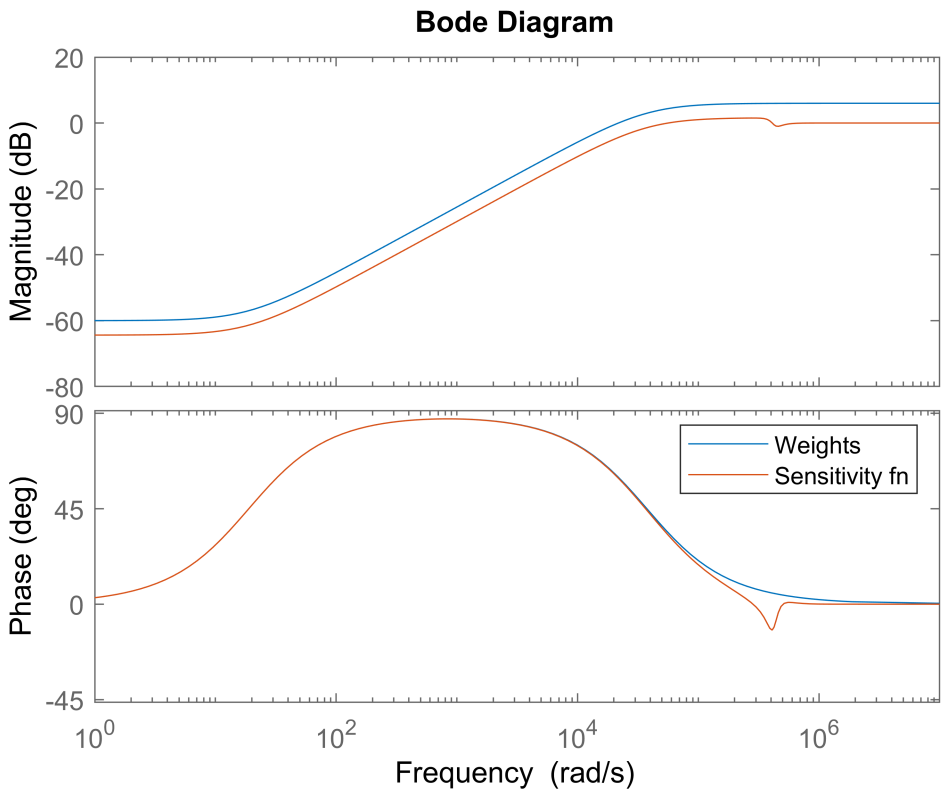
```
Wp_1 = (s/M+BW)/(s+BW/A);
Wp_2 = (s/sqrt(M)+BW)^2/(s+BW/sqrt(A))^2;
```

Generate the controller using mixsym for first order. Also, computed the Loop shape, Sensitivity, and the complementary sensitivity functions.

```
[K_1,CL,GAM_1,info] = mixsyn(P,Wp_1,[],[]);

L1 = P*K_1;
S1 = 1/(1+L1);
T1 = 1-S1;
```

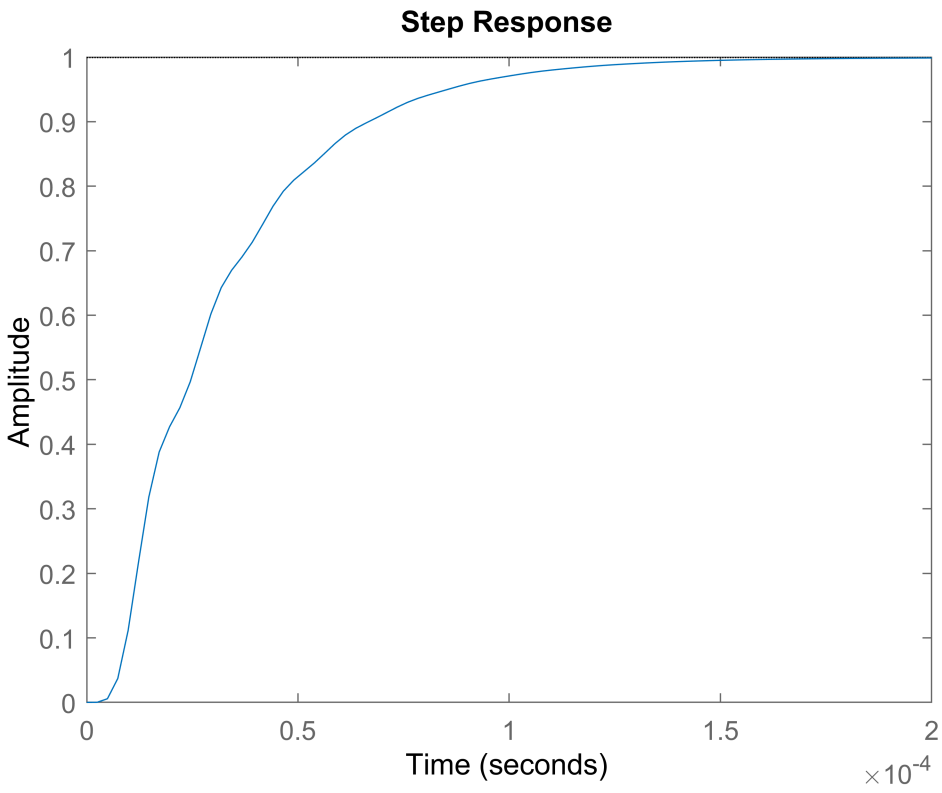
Sensitivity funciton plot

```
figure
bode(1/Wp_1,S1)
legend('Weights','Sensitivity fn',"Location","best")
```



Step response of the closed loop system (1st order)

```
figure
step(T1)
```

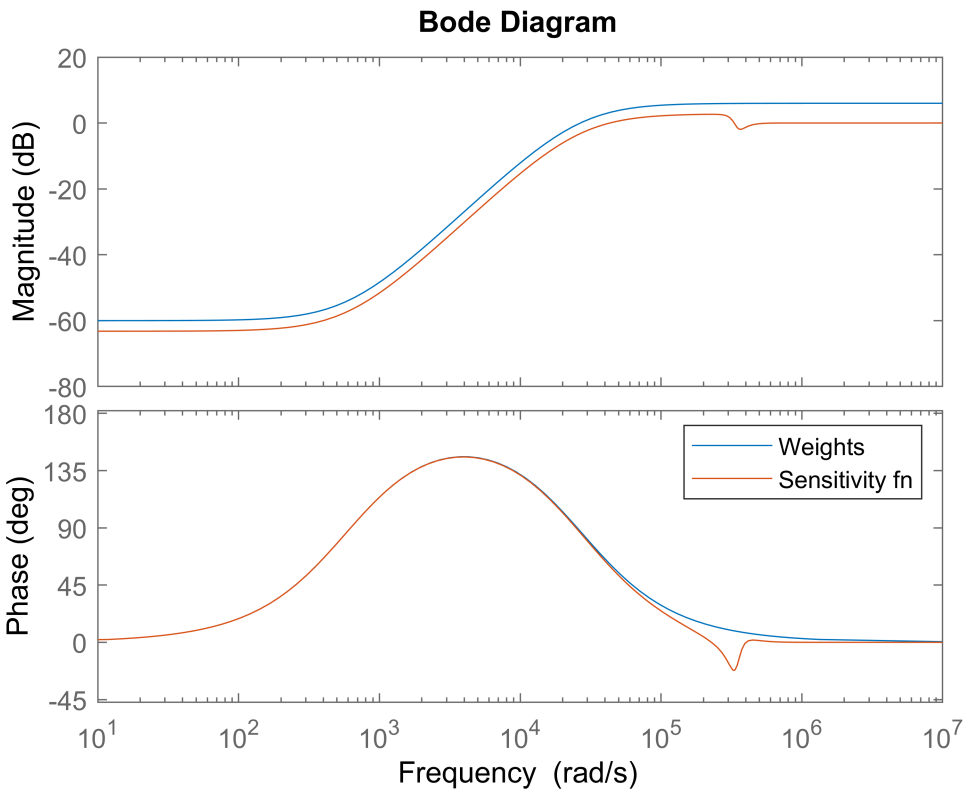


Generate the controller using mixsym for second order. Also, computed the Loop shape, Sensitivity, and the complementary sensitivity functions.

```
[K_2,CL,GAM_2,info] = mixsyn(P,Wp_2,[],[]);  %This is the magic synthesis command.  Much of thi
L = P*K_2;
S = 1/(1+L);
T = 1-S;
```

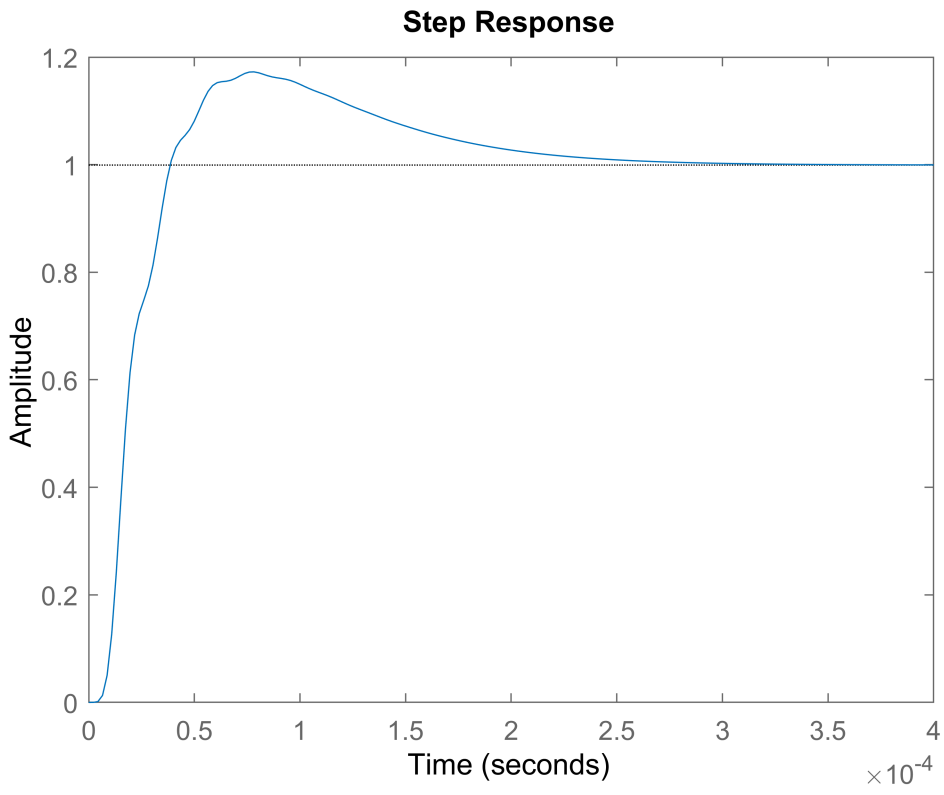
Sensitivity plot

```
figure
bode(1/Wp_2,S)
legend('Weights','Sensitivity fn')
```



Step response of the closed loop system (2nd order)

```
figure
step(T)
```



Check the limit of bandwidth for second order weights

```
GAM_wp2 = 0.5;
while GAM_wp2 < 1
    Wp_wp2 = (s/sqrt(M)+BW)^2/(s+BW/sqrt(A))^2;
    [K_2,CL,GAM_wp2,info] = mixsyn(P,Wp_wp2,[],[]);
    BW = BW + 50;
end
BW
```

```
BW = 3.9050e+04
```

As we can see that we have a threshold for bandwidth above which mixsyn does not produce a controller which has Gamma <1. Therefore, there exists a limit to sensitivity crossing frequency.

Q2. (B)

Initialize the weight parameters

```
At = 80;
Mt = 40;
BW = 10000*2*pi;
Wt = makeweight(1/At,BW,Mt);
```

Generate controller using mixsyn. Also, computed the Loop shape, Sensitivity, and the complementary sensitivity functions.

```
[K_t,CL,GAM_t,info] = mixsyn(P,Wp_2,[],Wt);
```

```
L_t = P*K_t;
S_t = 1/(1+L_t);
T_t = 1-S_t;
```

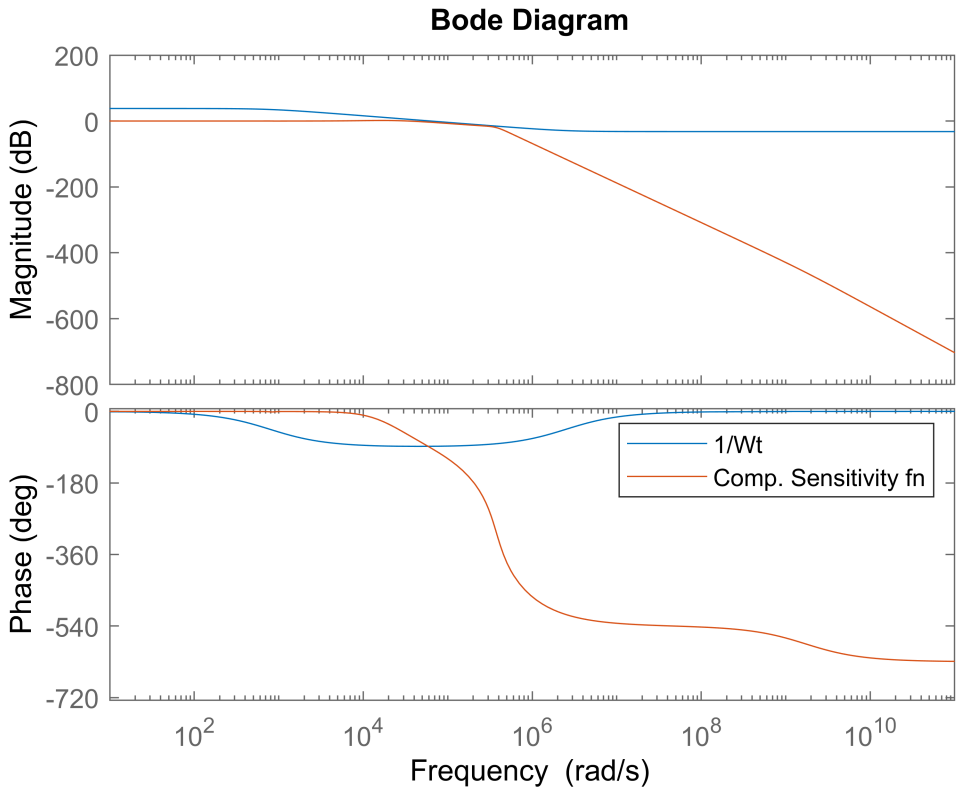
Gamma Value

```
GAM_t
```

```
GAM_t = 0.9679
```

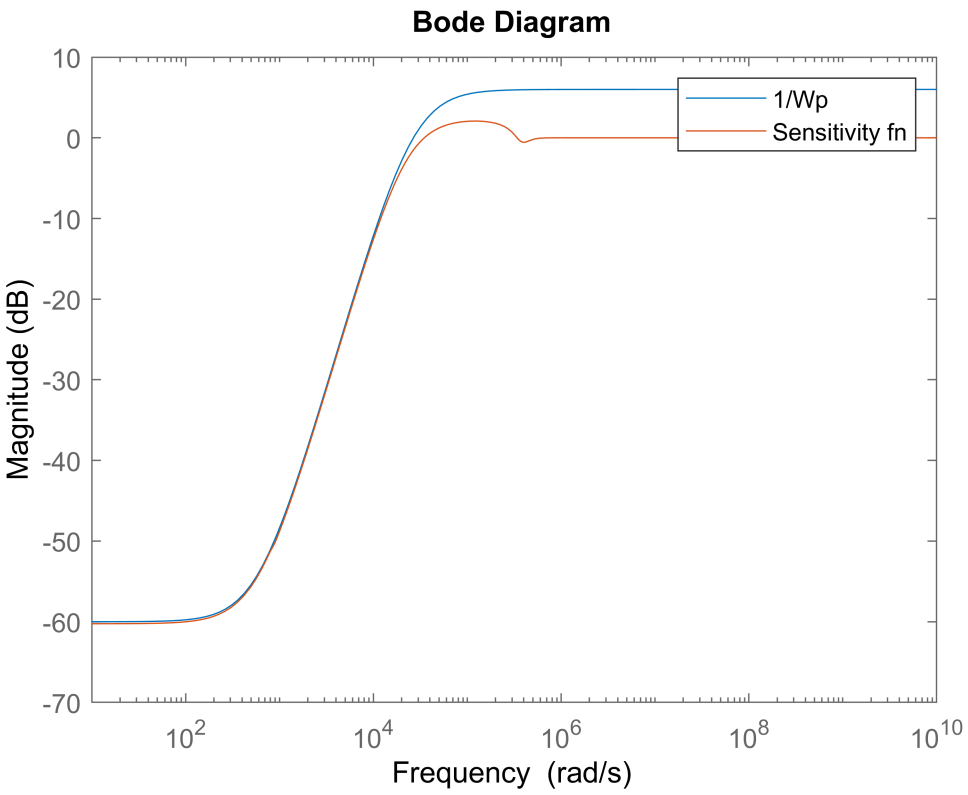
Complementary Sensitivity function

```
bode(1/Wt,T_t)
legend('1/Wt','Comp. Sensitivity fn')
```

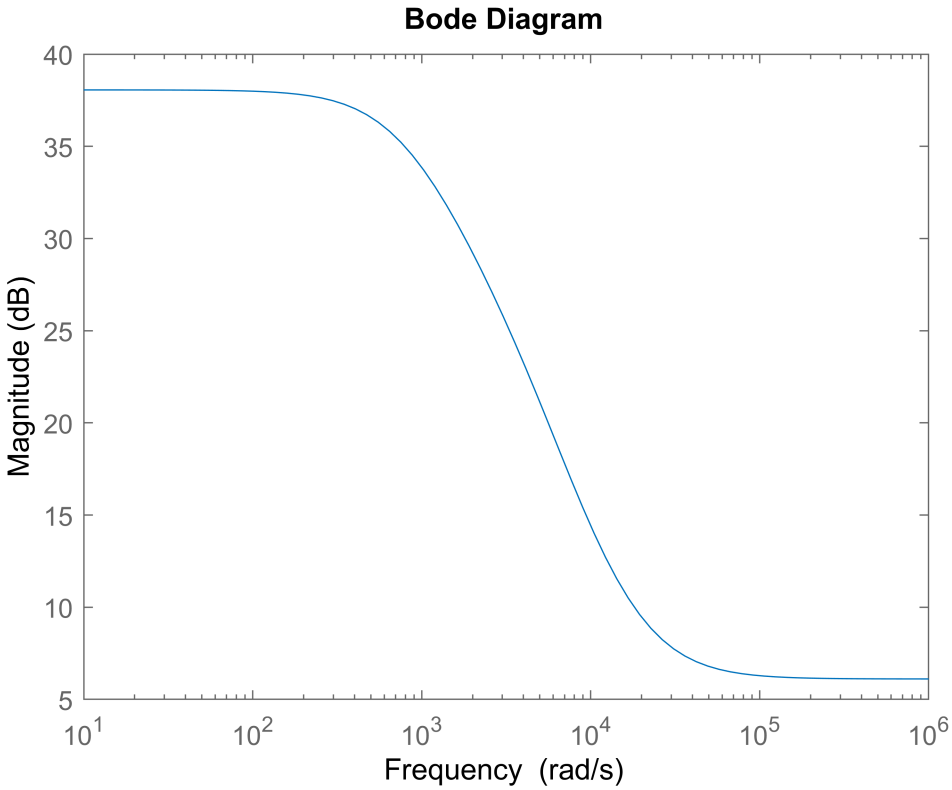


Sensitivity function

```
bodemag(1/Wp_2,S_t);
legend('1/Wp','Sensitivity fn')
```




```
bodemag(1/Wt + 1/Wp)
```



We can see that 1/Wt + 1/Wp > 1. The constraint holds true.

Q2. (C)

Initialize system parameters

```
s = tf('s');
Wu = 1/100;
GAM = 10;
BW = 3000*2*pi;
BW_step = 25;
```

Create loop to achieve the desired Gamma value.

```
while GAM>1
    Wp = (s/sqrt(M)+BW)^2/(s+BW/sqrt(A))^2;
    Wt = makeweight(0.5,5*BW,1000);
    [K_u,CL,GAM,info] = mixsyn(P,Wp,Wu,Wt);
    BW = BW - BW_step;
end
```

Gamma Value

```
GAM
```

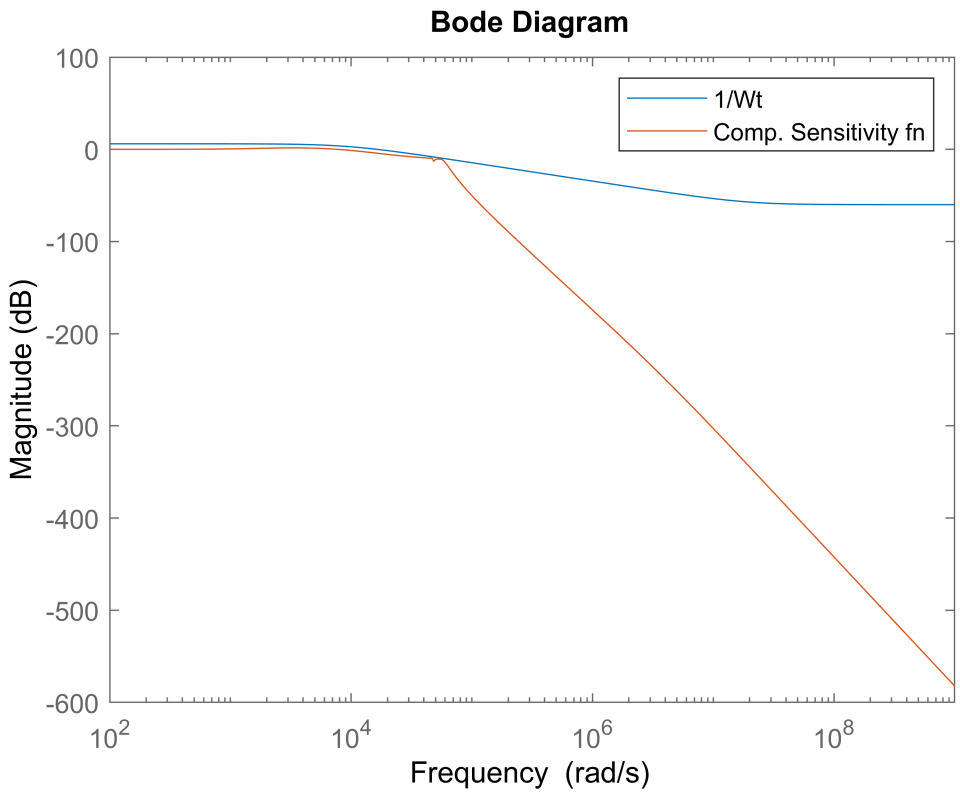
```
GAM = 0.9919
```

Generate controller using mixsyn. Also, computed the Loop shape, Sensitivity, and the complementary sensitivity functions.

```
L_u = P*K_u;
S_u = 1/(1+L_u);
T_u = 1-S_u;
```

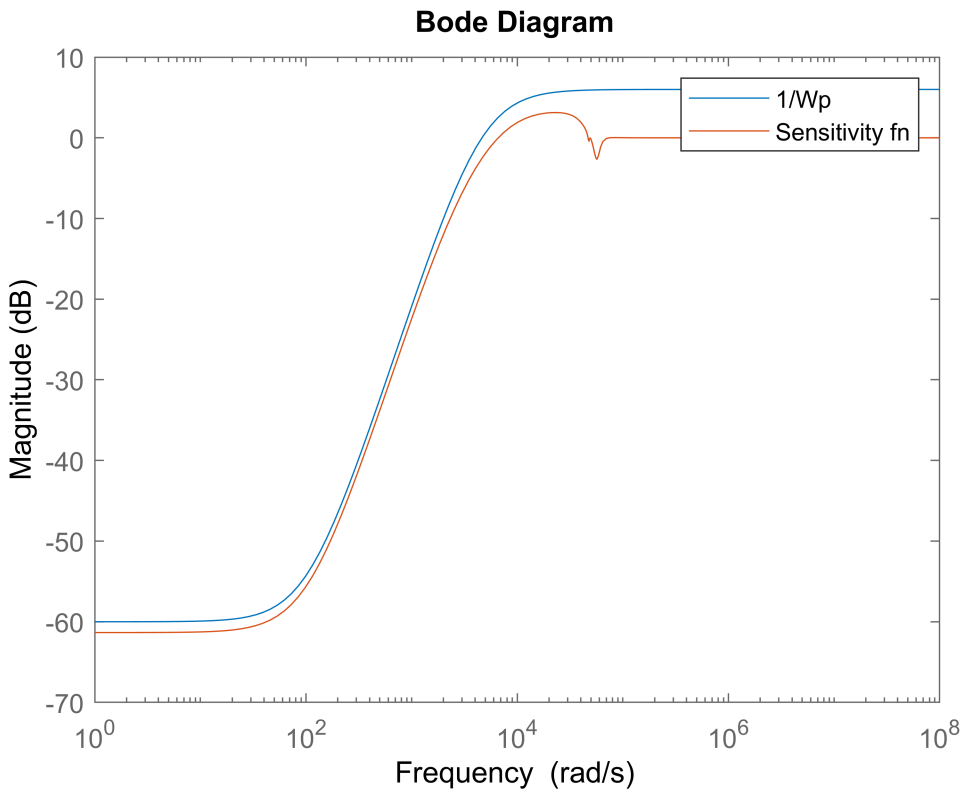
Complementary Sensitivity Plot

```
bodemag(1/Wt,T_u)
legend('1/Wt','Comp. Sensitivity fn')
```



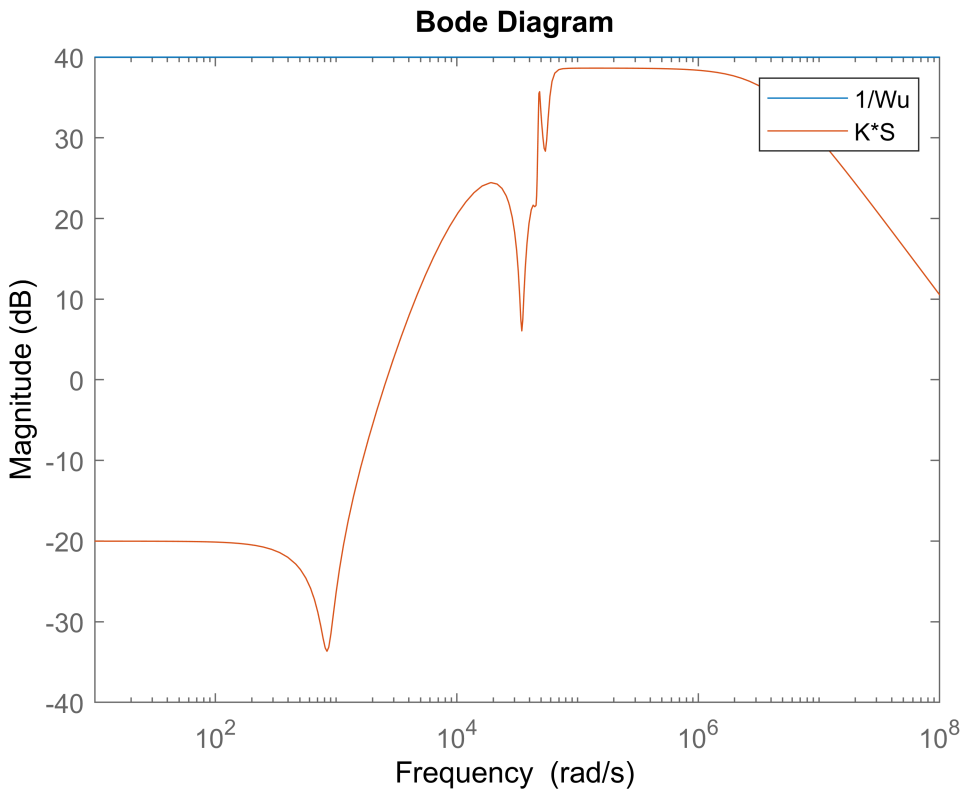
Sensitivity Plot

```
bodemag(1/Wp,S_u);
legend('1/Wp','Sensitivity fn')
```



Controller weight plot

```
bodemag(tf(1/Wu,1), K_u*S_u)
legend('1/Wu','K*S')
```




```
norm(CL,'inf')
```

```
ans = 0.9919
```

Q3. (a)

Given system,

```
%Create VCM Plant Model
%Note:  This model is from "Design and testing of track-following
%controllers for dual-state servo systems with PZT actuated suspensions" by
%Li and Horowitz, Microsystem Technologies, 2002
Gv = 10;  %DC Gain
Gv1 = tf((2*pi*135)^2,[1 2*0.1*2*pi*135 (2*pi*135)^2]);
Gv2 = tf((2*pi*5500)^2,[1 2*0.03*2*pi*5500 (2*pi*5500)^2]);
Gv3 = tf((2*pi*8640)^2,[1 2*0.05*2*pi*8640 (2*pi*8640)^2]);
Gv4 = 7300^2/7650^2*tf([1 2*.015*2*pi*7650 (2*pi*7650)^2],[1 2*.03*2*pi*7300 (2*pi*7300)^2]);
VCM = Gv*Gv1*Gv2*Gv3*Gv4;

Gp = 0.1;  %DC gain
Gp1 = tf((2*pi*8460)^2,[1 2*0.01*2*pi*8460 (2*pi*8460)^2]);
Gp2 = 5500^2/5650^2*tf([1 2*.03*2*pi*5650 (2*pi*5650)^2],[1 2*.03*2*pi*5500 (2*pi*5500)^2]);
Gp3 = 7300^2/7650^2*tf([1 2*.015*2*pi*7650 (2*pi*7650)^2],[1 2*.03*2*pi*7300 (2*pi*7300)^2]);
Gp4 = 8070^2/8250^2*tf([1 2*.02*2*pi*8250 (2*pi*8250)^2],[1 2*.015*2*pi*8070 (2*pi*8070)^2]);
Gp5 = 10650^2/10530^2*tf([1 2*.015*2*pi*10530 (2*pi*10530)^2],[1 2*.01*2*pi*10650 (2*pi*10650)^
PZT = Gp*Gp1*Gp2*Gp3*Gp4*Gp5;
```

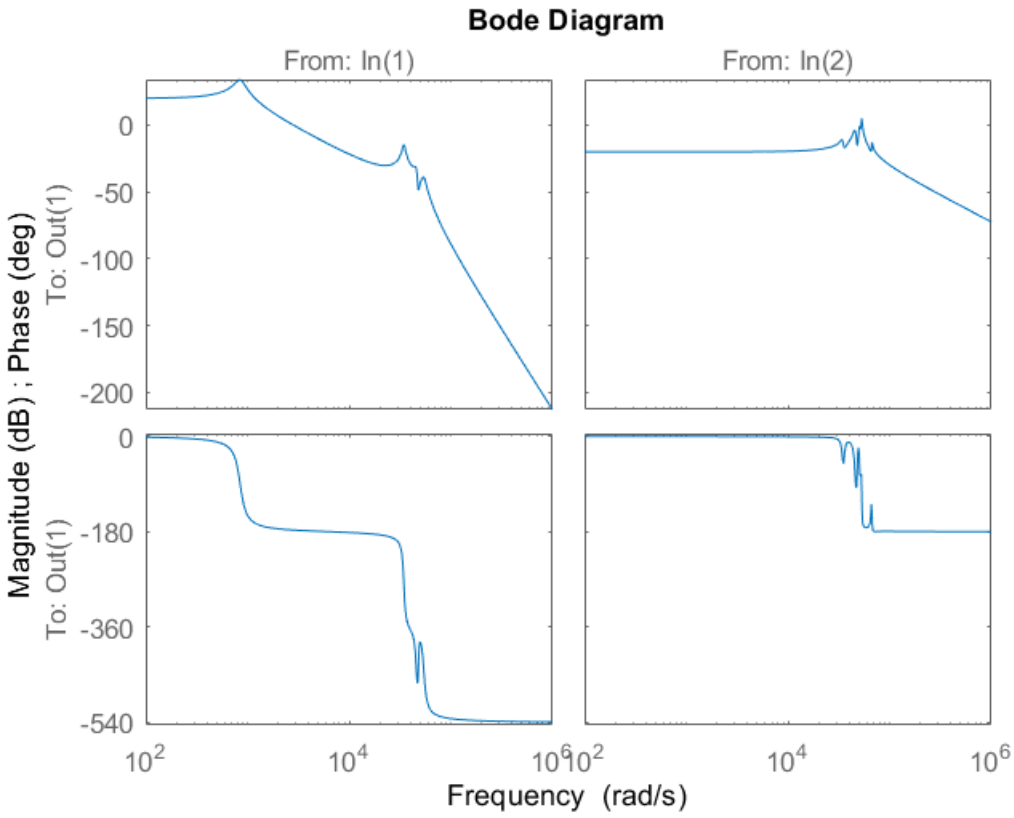
Create a DISO model

```
G = [VCM,PZT]
```

```
G =

  From input 1 to output:

                              2.306e25 s^2 + 3.325e28 s + 5.327e34

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

  s^8 + 1.042e04 s^7 + 6.279e09 s^6 + 4.088e13 s^5 + 1.23e19 s^4 + 3.829e22 s^3 + 7.419e27 s^2

                                                                          + 1.282e30 s + 5.327e33


  From input 2 to output:

  2.386e08 s^8 + 1.821e12 s^7 + 2.543e18 s^6 + 1.453e22 s^5 + 9.549e27 s^4 + 3.65e31 s^3

                                                              + 1.497e37 s^2 + 2.893e40 s + 8.173e45

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

  s^10 + 8748 s^9 + 1.32e10 s^8 + 9.411e13 s^7 + 6.675e19 s^6 + 3.637e23 s^5 + 1.616e29 s^4

                                          + 5.985e32 s^3 + 1.868e38 s^2 + 3.523e41 s + 8.173e46


Continuous-time transfer function.
```

Bode plot of the diso model

```
bode(G)
```



Q3. (b)

Initialize transfer function and the weight parameters as done in Q2 (a), (c)

```
s = tf('s');
Wu_V = 1/100;
Wu_P = 1/10;
Wu = [Wu_V 0;0 Wu_P];
GAM = 10;
BW = 3000*2*pi;
BW_step = 25;
M = 10^(6/20);
A = 1000;
```

Optimization loop

```
while GAM>1
    Wp = (s/sqrt(M)+BW)^2/(s+BW/sqrt(A))^2;
    Wt = makeweight(0.5,5*BW,1000);
    [K,CL,GAM,info] = mixsyn(G,Wp,Wu,Wt);
    BW = BW - BW_step;
end
```

Gamma Value

```
GAM
```

```
GAM = 0.9985
```

Semsitivity function

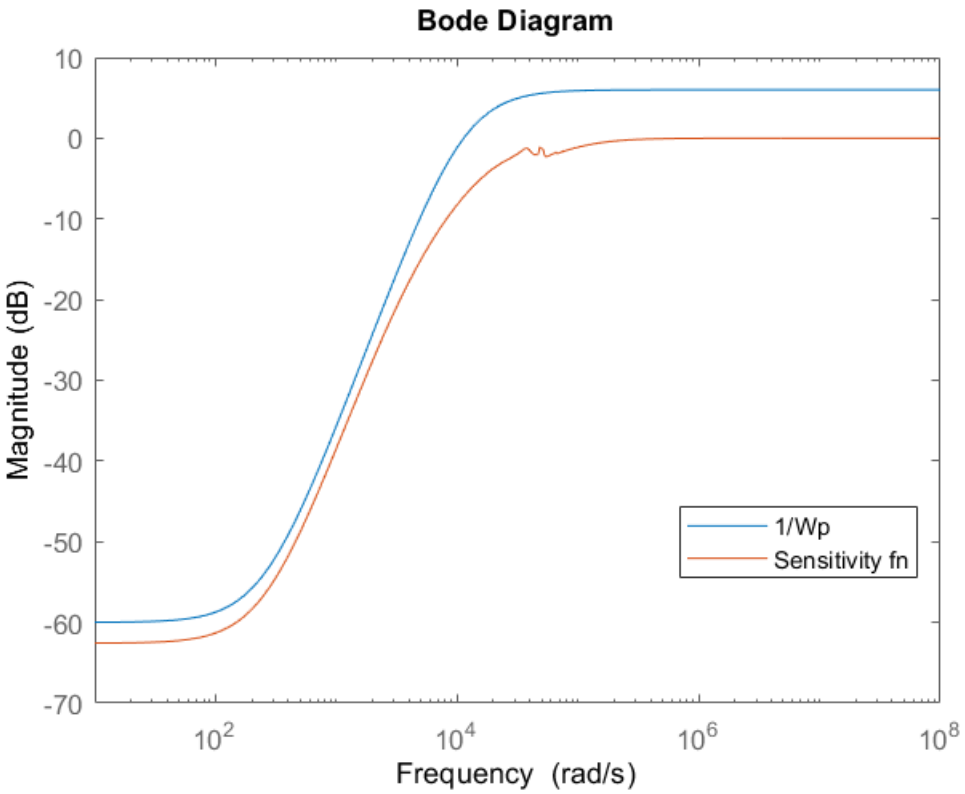
```
S = inv(eye(1)-G*K);
```

Somplementary sensitivity function

```
T = 1-S;
```

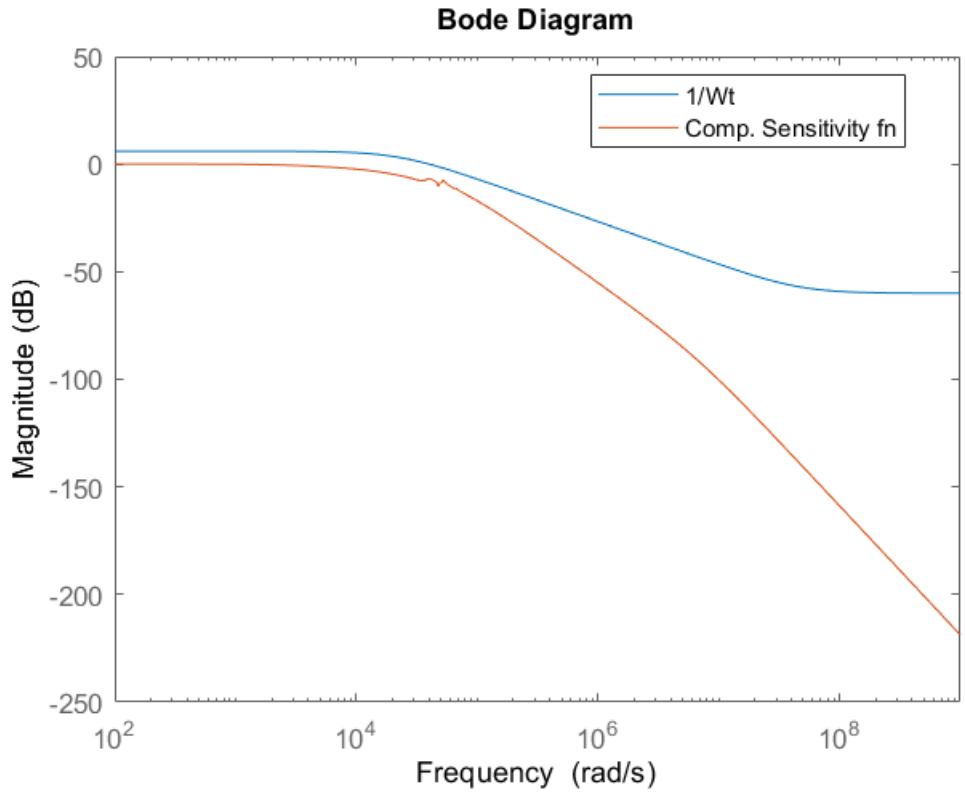
Bode plot of sensitivity funciton

```
bodemag(1/Wp,S)
legend('1/Wp','Sensitivity fn',"Location",'best')
```



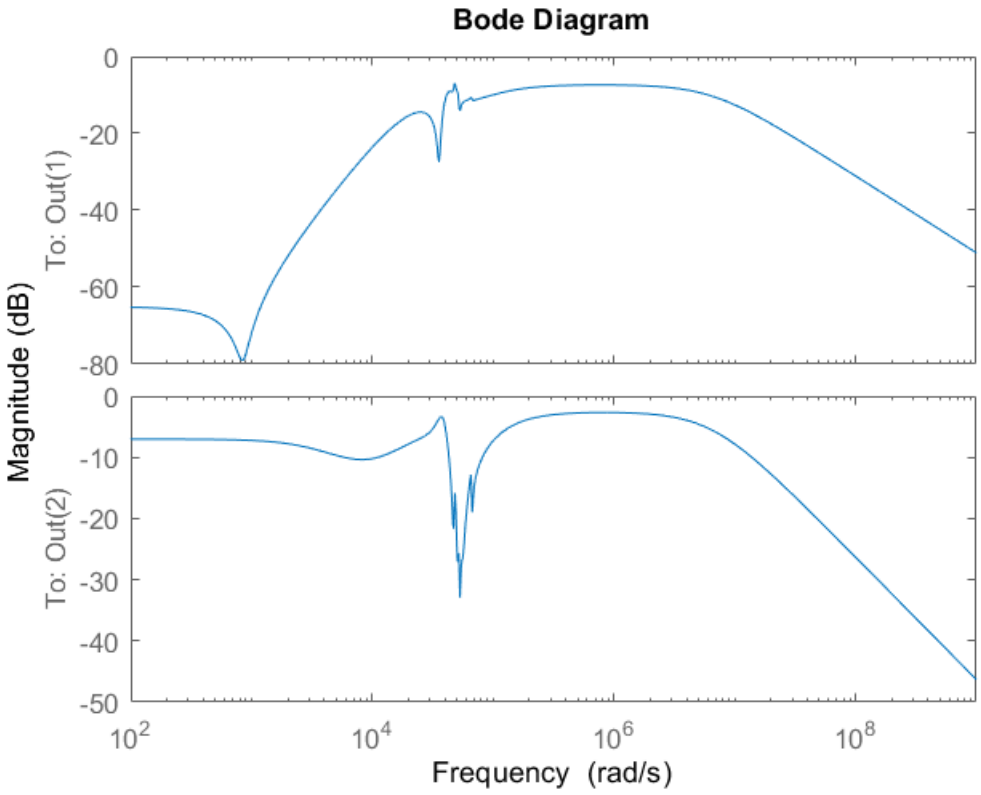
Bode plot of complementary sensitivity funciton

```
bodemag(1/Wt,T)
legend('1/Wt','Comp. Sensitivity fn',"Location",'best')
```



Bode magnitude plot of Wu*K*S

```
bodemag(Wu*K*S)
```


Bode Diagram
Magnitude (dB)
To: Out(1)
To: Out(2)
0
-20
-40
-60
-80
0
-10
-20
-30
-40
-50
10^2
10^4
10^6
10^8
Frequency (rad/s)