1: 30 points

Uncertainty modeling

(a) Consider a "true" plant $G(s) = \frac{3e^{-0.1s}}{(2s+1)(0.1s+1)^2}$. Derive and plot the additive uncertainty weight when the nominal model is $G(s) = \frac{3}{2s+1}$.

(b) Assume we have derived the following detailed model:

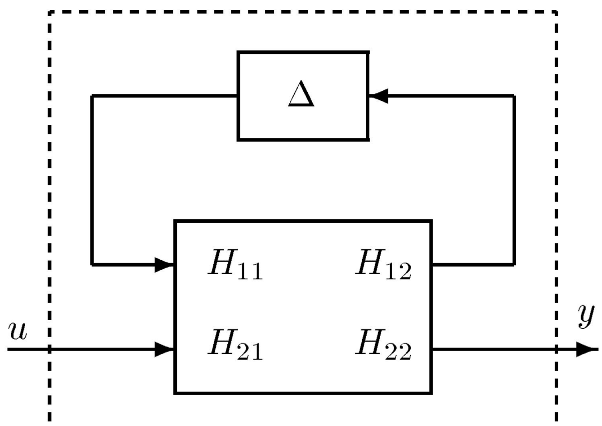
$$G_{\text{actual}}(s) = \frac{10(-0.5s+1)}{(6s+1)(0.2s+1)(20s+1)}$$

and we want to use the simplified nominal model $G(s) = \frac{10}{6s+1}$ with multiplicative uncertainty. Find an appropriate weighting function $w_I(s)$.

(c) A fairly general way of representing an uncertain plant G_p is in terms of an LFT in Δ as shown in the figure, i.e.

$$G_p = F_u\left(\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}, \Delta\right) = H_{22} + H_{21}\Delta(I - H_{11}\Delta)^{-1}H_{12},$$

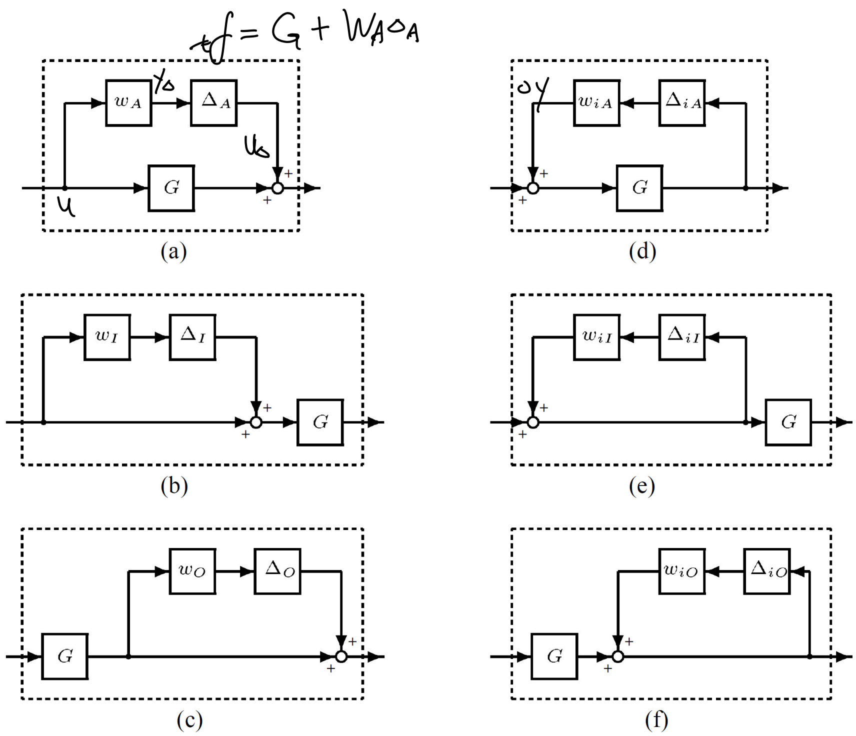
where $G = H_{22}$ is the nominal plant model. Find H for each of the six uncertainty forms shown below.



$$\begin{bmatrix} y_\Delta \\ z \\ v \end{bmatrix} = \begin{bmatrix} W_A & 0 & 0 \\ & & \\ & & \end{bmatrix} \begin{bmatrix} u_\Delta \\ w \\ u \end{bmatrix}$$

1×1 $\quad$ 1×1

(a) $W_A \cancel{\Delta_A} = H_{21} \cancel{\Delta_A} (I - H_{11}\Delta_A)^{-1} H_{12}$



2: 30 points

Disk Drive Control Application

The file HDDModel_DS_Uncertain.m contains a dual-stage HDD model that includes uncertainty from various sources.

(a) The file contains 2 uncertain models - VCM and PZT. Use Matlab to fit a 2nd order multiplicative uncertainty weight that best approximates the uncertainty for each model. Report the final weight for each, and plot $\frac{G_P - G}{G}$ for various perturbed plants G_p vs. the uncertainty weight for each plant.

(b) Perform single stage robust controller design for the VCM plant using *mixsyn*. Maximize the crossover frequency such that the low frequency disturbances are rejected by a factor of 1000, the sensitivity peak is below 2, and $\gamma < 1$. A first order performance weight is fine. Compute $\left\| \begin{bmatrix} W_P S \\ W_T T \end{bmatrix} \right\|_\infty$ for your final design and plot the Bode magnitude plot of the uncertain sensitivity function vs. the performance weight.

(c) Perform dual stage robust controller design for the dual stage system $G = \begin{bmatrix} VCM & PZT \end{bmatrix}$. Use the same performance criteria from part b, and again maximize the crossover frequency such that $\gamma < 3.5$. For each step of your iteration, capture γ. Plot the value of γ vs. iteration count and plot the Bode magnitude plot of the uncertain sensitivity function vs. the performance weight for the final design. Does your final design satisfy robust performance?

3: 20 points

Aircraft Control Application

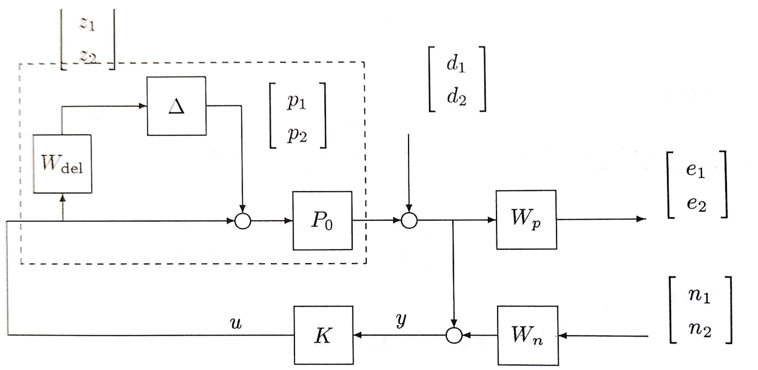
The nominal plant model for a highly maneuverable aircraft is given by

$$A = \begin{bmatrix} -0.0226 & -36.6 & -18.9 & -32.1 \\ 0 & -1.9 & 0.983 & 0 \\ 0.0123 & -11.7 & -2.63 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ -0.414 & 0 \\ -77.8 & 22.4 \\ 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 57.3 & 0 & 0 \\ 0 & 0 & 0 & 57.3 \end{bmatrix} \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Consider the block diagram below with

$$W_p = \begin{bmatrix} \frac{s+3}{s+0.03} & 0 \\ 0 & \frac{0.5(s+3)}{s+0.03} \end{bmatrix} \quad W_n = \begin{bmatrix} \frac{2(s+1.28)}{s+320} & 0 \\ 0 & \frac{2(s+1.28)}{s+320} \end{bmatrix}$$



(a) The file *responses.mat* gives a vector of responses for the system. Fit a multiplicative uncertainty weight W_{del} to the response. Create a Bode magnitude plot that shows the quality of your fit.

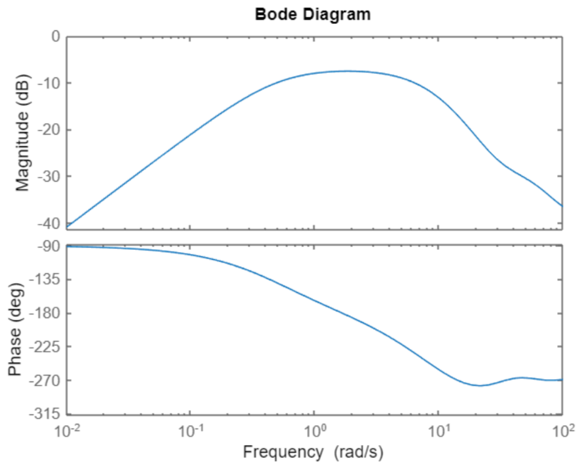
(b) Design an H_∞ optimal controller considering the uncertainty. Plot the Bode magnitude of the sensitivity function for 10 samples of the uncertain plant. Do you meet robust performance specs? What about robust stability?

1(a)

$$G_p = G_A + W_I \Delta_A$$

$$\frac{3e^{-0.1s}}{(2s+1)(0.1s+1)^2} = \frac{3}{2s+1} + W_I \Delta_A$$

$$W_I = \frac{3}{2s+1}\left(\frac{e^{-0.1s}}{(0.1s+1)^2} - 1\right)$$



(b) $G_p = G(1 + W_I \Delta)$

$$\frac{10(-0.5s+1)}{(6s+1)(0.2s+1)(20s+1)} = \frac{10}{6s+1}(1+W_I\Delta)$$

$$1 + W_I\Delta = \frac{-0.5s+1}{(0.2s+1)(20s+1)}$$

$$W_I\Delta = \frac{-0.5s+1}{(0.2s+1)(20s+1)} - 1$$

(c) $$G_p = F_u\left(\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}, \Delta\right) = H_{22} + H_{21}\Delta(I - H_{11}\Delta)^{-1}H_{12},$$

$H_{22} = G$

$$H_{22} + \frac{H_{21}\Delta}{I - H_{11}\Delta} H_{22}$$

(a) $G_p = G + W_A \Delta_A$

$H_{22} = G$ $\quad H_{12} = I$ $\quad H_{11} = 0$ $\quad H_{21} = W_A$

$$H = \begin{bmatrix} 0 & I \\ W_A & G \end{bmatrix}$$

(b) $G_p = G(I + W_I \Delta_I)$

$$G_p = G + G W_I \Delta_I$$

$H_{11} = 0$, $H_{12} = I$, $H_{21} = G W_I$, $H_{22} = G$

$$H = \begin{bmatrix} 0 & I \\ G W_I & G \end{bmatrix}$$

(c) $G_p = (I + W_o \Delta_o) G$

$$G_p = G + W_o \Delta_o G$$

$H_{11} = 0$ $\quad H_{12} = G$ $\quad H_{21} = W_o$ $\quad H_{22} = G$

$$H = \begin{bmatrix} 0 & G \\ W_o & G \end{bmatrix}$$

(d) $G_p = \dfrac{G}{I - G W_{iA} \Delta_{iA}} = G(I - W_{iA} \Delta_{iA} G)^{-1}$

$$H_{21} \Delta (I - H_{11} \Delta)^{-1} H_{12} = \frac{G}{I - W_{iA} \Delta_{iA} G} - G$$

$$= \frac{G - G(I - W_{iA} \Delta_{iA} G)}{I - W_{iA} \Delta_{iA} G}$$

$$= \frac{G - G + G W_{iA} \Delta_{iA} G}{I - W_{iA} \Delta_{iA} G}$$

$$= \frac{G W_{iA} \Delta_{iA} G}{I - W_{iA} \Delta_{iA} G}$$

$$H_{11} = G W_{iA}, \quad H_{12} = G, \quad H_{21} = G W_{iA}, \quad H_{22} = G$$

$$H = \begin{bmatrix} G W_{iA} & G \\ G W_{iA} & G \end{bmatrix}$$

(e) $$G_p = \frac{I}{I - W_{iI} \Delta_{iI}} \cdot G$$

$$H_{21} \Delta (I - H_{11} \Delta)^{-1} H_{12} = \frac{G}{I - W_{iI} \Delta_{iI}} - G$$

$$= \frac{G - G(I - W_{iI} \Delta_{iI})}{I - W_{iI} \Delta_{iI}}$$

$$= \frac{G - G + G W_{iI} \Delta_{iI}}{I - W_{iI} \Delta_{iI}}$$

$$= \frac{G W_{iI} \Delta_{iI}}{I - W_{iI} \Delta_{iI}}$$

$H_{21} = GW_{iI}, \quad H_{22} = G \quad H_{11} = W_{iI}, \quad H_{12} = I$

$$H = \begin{bmatrix} W_{iI} & I \\ GW_{iI} & G \end{bmatrix}$$

(f) $$G_p = G\left(\frac{I}{I - W_{io}\Delta_{io}}\right)$$

$$= G + W_{io}(I - W_{io}\Delta_{io})^{-1}\Delta_{io}G$$

$$= G + W_{io}\Delta_{io}(I - W_{io}\Delta_{io})^{-1}G$$

$H_{22} = G \quad H_{21} = W_{io} \quad H_{11} = W_{io} \quad H_{12} = G$

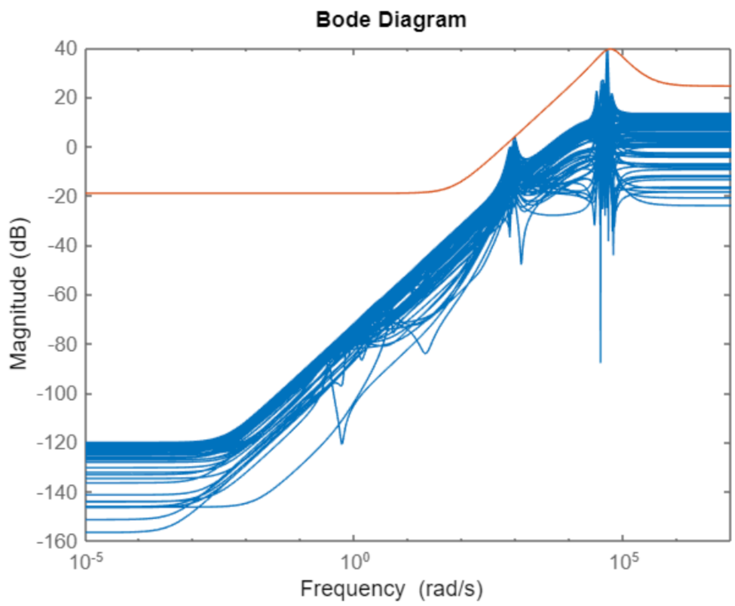
$$H = \begin{bmatrix} W_{io} & G \\ W_{io} & G \end{bmatrix}$$

2,a

```
Gvec1 = usample(VCM,100);  %Create 100 Monte Carlo samples of G
[P1,info] = ucover(Gvec1,VCM.NominalValue,2);  %Find the best 2nd order multiplicative uncertainty
% P1
info.W1
```

```
ans =

  A =
               x1          x2
   x1  -5.978e+04  -7.285e+04
   x2   4.627e+04        3659

  B =
          u1
   x1   3609
   x2  -1805

  C =
          x1     x2
   y1  376.4  -1639

  D =
          u1
   y1  16.74

Continuous-time state-space model.
```

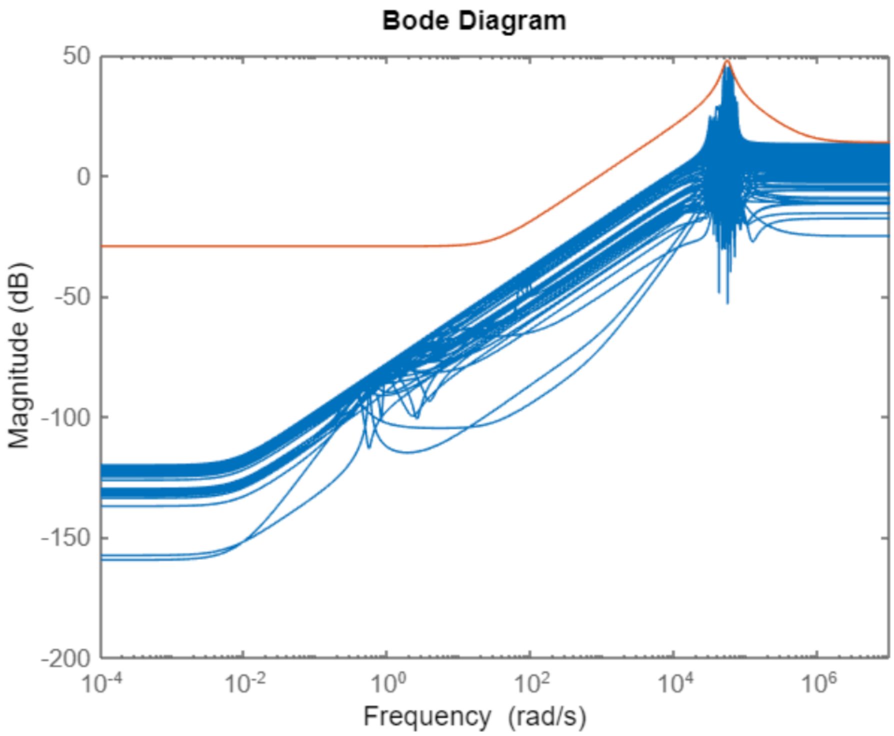
```
bodemag((Gvec1-VCM.NominalValue)/VCM.NominalValue,info.W1) %Gvec: 100 s
```




```
Gvec2 = usample(PZT,100);  %Create 100 Monte Carlo samples of G
[P2,info] = ucover(Gvec2,PZT.NominalValue,2);  %Find the best 4th order multipl
% P2
info.W1
```

```
ans =

  A = 
                x1          x2
   x1   -1.333e+05  -1.313e+05
   x2    1.442e+05   1.204e+05

  B = 
          u1
   x1   7735
   x2  -7735

  C = 
           x1      x2
   y1   119.3  -284.2

  D = 
          u1
   y1  4.855

Continuous-time state-space model.
```

```
bodemag((Gvec2-PZT.NominalValue)/PZT.NominalValue,info.W1) %Gvec: 100 samples
```

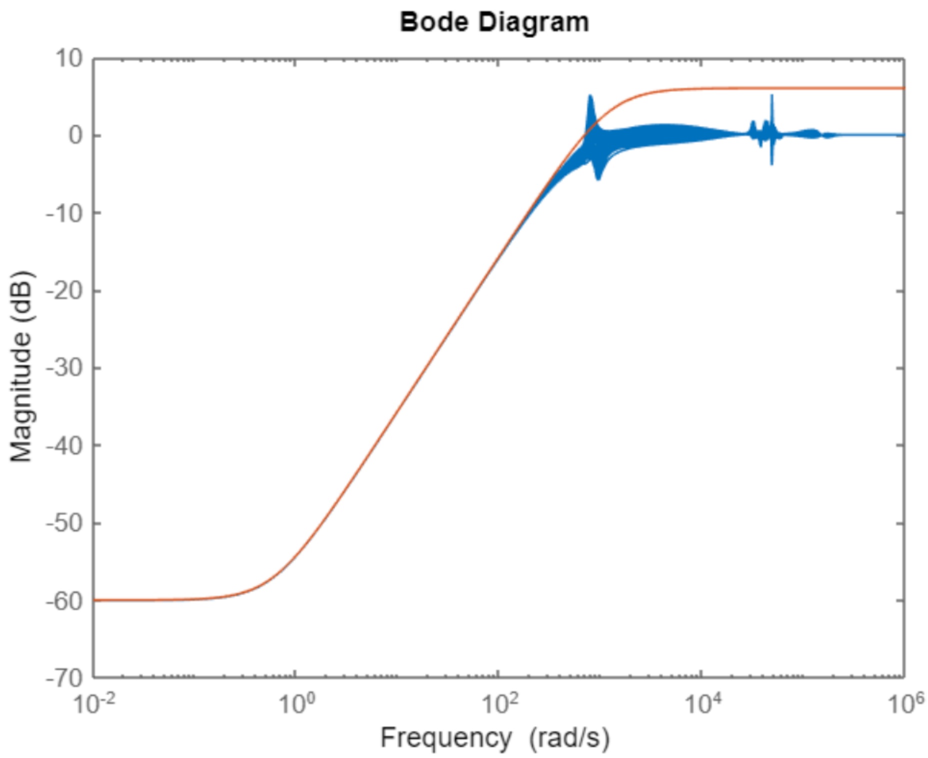


2b

```
VCMvec = usample(VCM,100);
[P_VCM,VCM_info] = ucover(VCMvec,VCM.NominalValue,2);
Wt = VCM_info.W1;  %Extract the weight
wh = 1000;  %Just guessing at this...we'll see if it fails
wl = 0;
w_try = wh;  %Just for simplicity
w_new = 1/2*(wh+wl);

while(abs(w_new-w_try)>.001)  %Stopping criterion
    w_try = w_new;
    Wp = makeweight(1000,w_try,1/2);
    [K,CL,GAM] = mixsyn(VCM,Wp,[],Wt);
    if GAM<1%/sqrt(2)
        wl = w_try;  %We're not agressive enough
    else
        wh = w_try;  %We're too agressive
    end
    w_new = 1/2*(wh+wl);
end
S = 1/(1+VCM*K);
T = 1-S;
Svec = usample(S,100);
```

Now plot the achieved sensitivity functions for the uncertain plant vs the sensitivity weight.

```
bodemag(Svec,1/Wp)
```



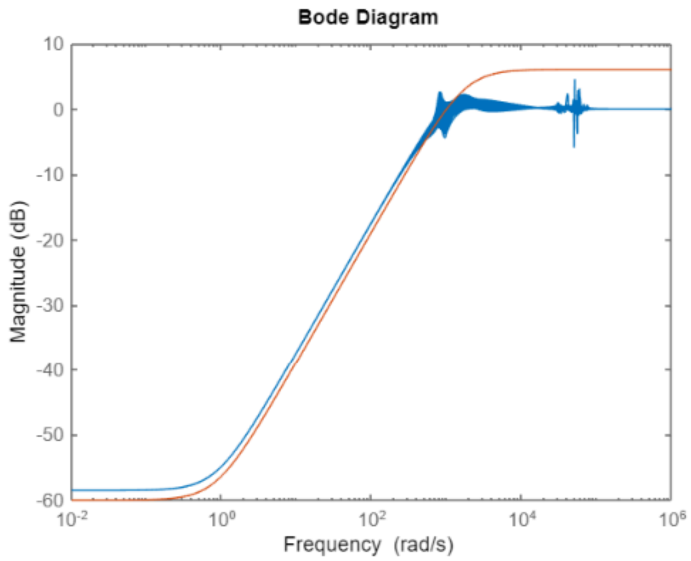

```
norm([Wp*S; Wt*T],"inf")
```

```
ans = 1.0000
```

2C

```
G = [VCM PZT];
VCM_nom = VCM.NominalValue;
VCMvec = usample(VCM,100);
[P_VCM, VCM_info] = ucover(VCMvec, VCM.NominalValue, 2);
VCM_Wt = VCM_info.W1;  %Extract the weight

PZT_nom = PZT.NominalValue;
PZTvec = usample(PZT,100);
[P_PZT, PZT_info] = ucover(PZTvec, PZT.NominalValue, 2);
PZT_Wt = PZT_info.W1;  %Extract the weight

Wt = [VCM_Wt; PZT_Wt];

wh = 1000;  %Just guessing at this...we'll see if it fails
wl = 0;
w_try = wh;  %Just for simplicity
w_new = 1/2*(wh+wl);
gam_list = [];
i = 1;
while(abs(w_new-w_try)>.001)  %Stopping criterion
    w_try = w_new;
    Wp = makeweight(1000,w_try,1/2);
    systemnames = 'VCM_nom PZT_nom Wp VCM_Wt PZT_Wt'; %Block name only
    inputvar = '[d1;d2;r;u1;u2]';
    outputvar = '[VCM_Wt;PZT_Wt;Wp;VCM_nom+PZT_nom+d1+d2;r-VCM_nom-PZT_nom-d1-d2]'; %Strangely, the system outputs are just the name
    input_to_VCM_nom = '[u1]';
    input_to_PZT_nom = '[u2]';
    input_to_Wp = '[r-VCM_nom-PZT_nom-d1-d2]';
    input_to_VCM_Wt = '[u1]';
    input_to_PZT_Wt = '[u2]';
    cleanupsysic = 'yes';  %This drops all the useless variables from workspace
    P = sysic;
    [K,CL,GAM] = hinfsyn(P,1,2);
    gam_list(i) = GAM;
```

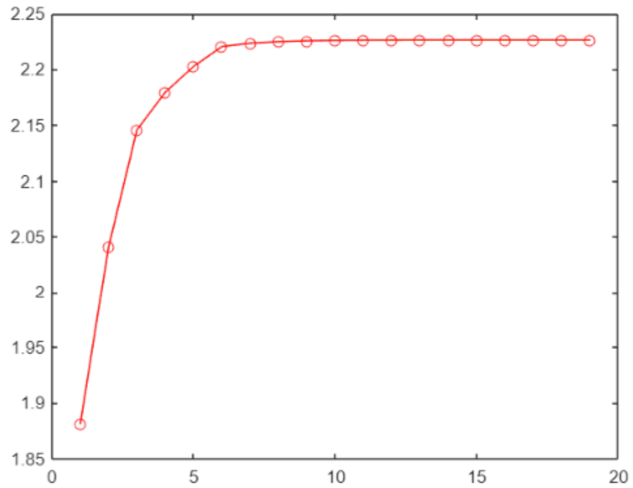
```
    if GAM<3.5 %/sqrt(2)
        wl = w_try;  %We're not agressive enough
    else
        wh = w_try;  %We're too agressive
    end
    w_new = 1/2*(wh+wl);
    i = i+1;
end
S = inv(1+G*K);
T = 1-S;
Svec = usample(S,100);
bodemag(Svec,1/Wp)
```




```
%Check Robust Perforamnce
perfmarg_inf = robustperf(P);
%perfmarg_inf = robgain(Wp*S,1); %wrap uncertainty
mu_inf = 1/perfmarg_inf.LowerBound
```

```
mu_inf = 1.0149e+04
```

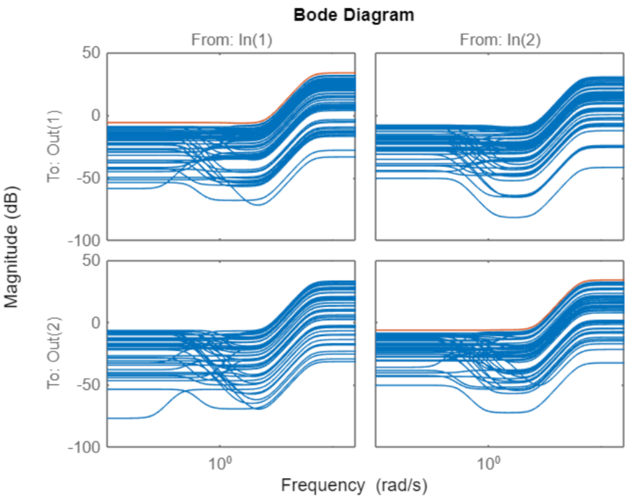
```
% figure
plot(1:i-1,gam_list, '-or');
```



Don't have Robust Performance

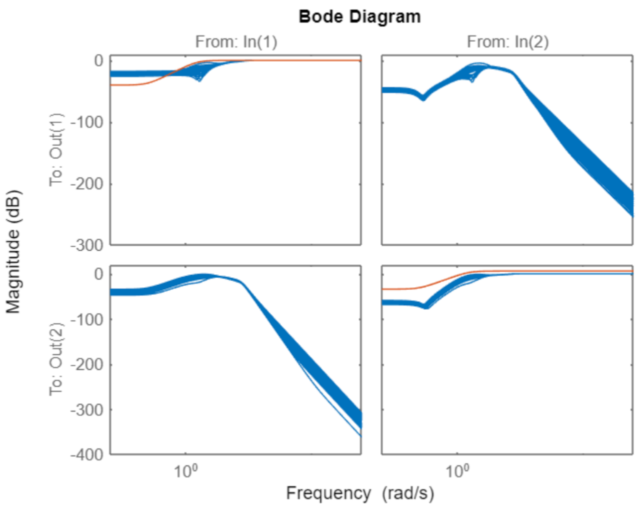
3a

```
close all;
clc;
clear;
A = [-0.0226 -36.6 -18.9 -32.1; 0 -1.9 0.983 0; 0.0123 -11.7 -2.63 0; 0 0 1 0];
B = [0 0;-0.414 0;-77.8 22.4;0 0];
C = [0 57.3 0 0; 0 0 0 57.3];
D = zeros(2,2);
load('responses.mat','Gp_samples');
rsp_vec = Gp_samples; %sample
rsp_nom = tf(ss(A,B,C,D));
%rsp_vec = usample(rsp,100);  %Create 100 Monte Carlo samples of G
[P1,info_rsp] = ucover(rsp_vec,rsp_nom,[2,2]);  %Find the best 2nd order multiplicative uncertainty
% P1
W_del = info_rsp.W1;
bodemag(inv(rsp_nom)*rsp_vec-eye(2), W_del) %Gvec: 100 samples
```



3b

```
s = tf('s');
%W_del = tf(W_del);
Wp = [(s+3)/(s+0.03) 0;0 0.5*(s+3)/(s+0.03)];
Wn = [2*(s+1.28)/(s+320) 0;0 2*(s+128)/(s+320)];
P0 = rsp_nom;

systemnames = 'P0 Wp Wn W_del'; %Block name only
inputvar = '[p{2};n{2};d{2};u{2}]';
outputvar = '[W_del;Wp;n+P0+d]'; %Strangely, the system outputs are just the name
input_to_P0 = '[u+p]';
input_to_Wp = '[d+P0]';
input_to_Wn = '[n]';
input_to_W_del = '[u]';
cleanupsysic = 'yes';  %This drops all the useless variables from workspace
P = sysic;
% P = [zeros(2) zeros(2) zeros(2) W_del; Wp*rsp_nom zeros(2) Wp Wp*rsp_nom; rsp_nom Wn eye(2) rsp_nom];
[K,CL,GAM] = hinfsyn(P,2,2);
G_vec = usample(rsp_vec,10);
Svec = inv(eye(2)+G_vec*K);
bodemag(Svec,1/Wp)
```



```
N = lft(P,K);
%Svec = usample(S,10);

%Check Robust Stability
[STABMARG,WCU] = robstab(N);
```

The returns of this command are weird, effectively giving 1/mu as the main output.

```
mu = 1/STABMARG.LowerBound
```

```
mu = 0
```

Have Robust Stability

```
%Check Robust Perforamnce
perfmarg_inf = robustperf(N); %wrap uncertainty
mu_inf = 1/perfmarg_inf.LowerBound
```

```
mu_inf = 96.8558
```

Don't have Robust Performance