Robust Inner Loop Control of a 6-DOF Rotary Wing Unmanned Aircraft System

Riley Kenyon

May 6th, 2020

Abstract

This report describes the process of synthesizing a series of inner loop controllers using H_∞ and μ synthesis techniques for an 11 state rotary wing micro air vehicle with analysis of their robustness properties. The objective is to ensure robust command tracking of the roll and pitch attitude and disturbance rejection of the attitudes in the presence of a structured multiplicative input uncertainty.

1 Topic Discussion

1.1 Introduction

The state vector of the system is given by $x = (u, v, p, q, \phi, \theta, a_s, b_s, w, r, r_{fb})^T$, where u, v, w are the vehicle translational velocities about the longitudinal, lateral, and vertical body axis, p, q, r are the vehicle roll, pitch, and yaw rate respectively, θ and ϕ are the roll and pitch attitude, a_s, b_s are the rotor dynamics states (orientation of the rotor relative to the body), and r_{fb} is the washed out yaw rate, a filtering state associated with the onboard yaw rate feedback. The control input vector is $\delta_{long}, \delta_{lat}, \delta_{ped}$, corresponds to the longitudinal cyclic, the lateral cyclic, and the tail rotor inputs. The outputs available for feedback are the pitch and roll attitude θ and ϕ, and the washed out raw rate r_{fb}. The objective of the project is to ensure robust command tracking of θ and ϕ and output disturbance rejection of θ and ϕ in the presence of a structured multiplicative input uncertainty.

1.2 Generalized Plant for Reference Tracking

The system can be represented in the traditional block diagram with the multiplicative input uncertainty model. The two cases of interest are for output disturbance rejection for the roll and pitch attitude, and reference tracking for the roll and pitch attitude. For reference tracking, the block diagram is shown in figure 1.

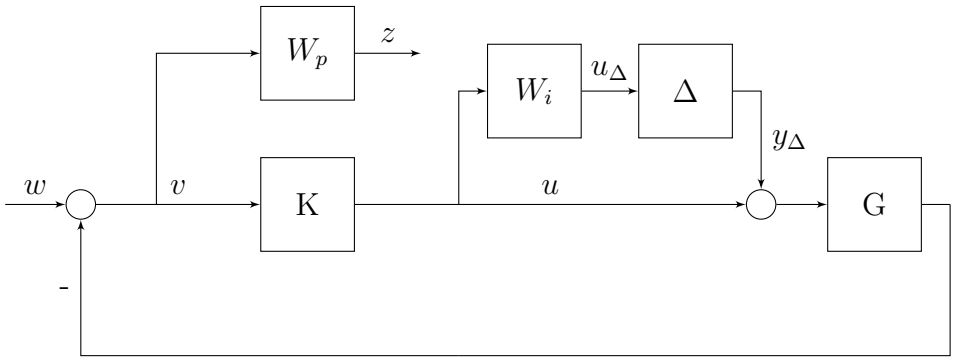

Figure 1: Block diagram for reference tracking with an input multiplicative uncertainty model.

The feedback block diagram representation shown is converted to a generalized plant in the form of figure 2. Writing out the system of equations for the reference tracking problem in terms of the input $(y_\Delta, w, u)^T$ results in a generalized plant from equation 1.

$$\begin{pmatrix} u_\Delta \\ z \\ v \end{pmatrix} = \begin{pmatrix} 0 & 0 & W_i \\ -W_pG & W_p & -W_pG \\ -G & I & -G \end{pmatrix} \begin{pmatrix} y_\Delta \\ w \\ u \end{pmatrix} \tag{1}$$

The generalized plant can be closed in a lower linear fractional transformation (LFT) to obtain the matrix N which is useful for determining performance and stability tests due to the N-Δ form. The w to z transfer function is given by the $N(2,2)$ block which yields information about nominal performance. Information about robust stability is given from the $N(1,1)$ entry, which is a transfer function that relates y_Δ to u_Δ also known as the matrix M in M-Δ form. Using the equation to close a LFT, the generalized plant P and controller K can be combined into the matrix N with the

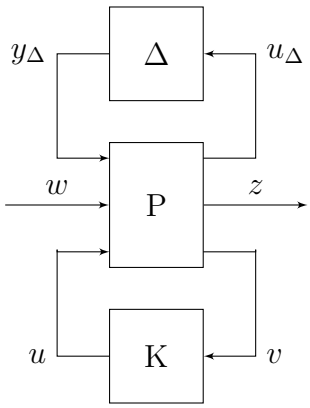

Figure 2: The generalized plant representation of a system with uncertainty.

expression

$$N = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} \tag{2}$$

which results in the matrix N, equal to

$$\begin{pmatrix} u_\Delta \\ z \end{pmatrix} = \begin{pmatrix} -W_i T_i & W_i K S_o \\ -W_p S_o G & -W_p S_o \end{pmatrix} \begin{pmatrix} y_\Delta \\ w \end{pmatrix} \tag{3}$$

From this point, the tests for nominal performance (NP) and robust stability (RS) are determined by the infinity norm $||\cdot||_\infty$ of the entry. The test for nominal performance is then

$$||W_p S_o||_\infty < 1 \tag{4}$$

where in this case W_P is a 3x3 diagonal matrix of performance weighting functions. These entries are multiplied with the three states for feedback $(\theta, \phi, r_{fb})^T$, although the weighting functions of interest are those corresponding to the pitch and roll attitude.

1.3 Generalized Plant for Disturbance Rejection

The standard block diagram can also be formed for a disturbance rejection model, where the output z is a function of the output state vector y and w is the output disturbance. The purpose of looking at the system from this perspective it to design a weighting function W_p and ultimately a controller K through H_∞ synthesis that reduces the effect of w on z.

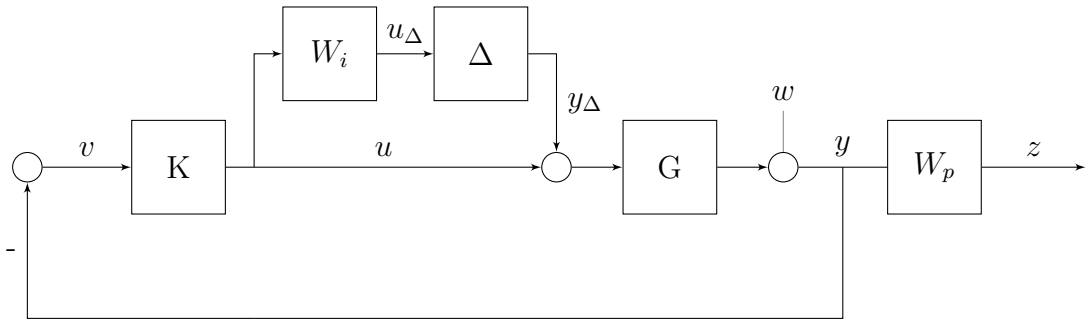

Figure 3: Block diagram for disturbance rejection with an input multiplicative uncertainty model.

For this block diagram, the converted generalized plant results in a matrix of

$$\begin{pmatrix} u_\Delta \\ z \\ v \end{pmatrix} = \begin{pmatrix} 0 & 0 & W_i \\ W_pG & W_p & -W_pG \\ -G & -I & -G \end{pmatrix} \begin{pmatrix} y_\Delta \\ w \\ u \end{pmatrix} \tag{5}$$

following the same procedure as the reference tracking case, the generalized plant P can be closed with the lower LFT to obtain the matrix N.

$$\begin{pmatrix} u_\Delta \\ z \end{pmatrix} = \begin{pmatrix} -W_iT_i & -W_iKS_o \\ W_pS_oG & W_pS_o \end{pmatrix} \begin{pmatrix} y_\Delta \\ w \end{pmatrix} \tag{6}$$

Looking at the entries of the matrix N for the reference tracking case, shown in equation 3, the difference is only a couple of signs on the entries of the matrix. As a result of using the infinity norm, the nominal performance and robust stability test for both the reference tracking and disturbance rejection case is identical.

2 Synthesizing an H_∞ Controller

An H_∞ controller can be made for the generalized plant P by solving two algebraic Riccati equations and solving for an H_∞ cost. This process is done using a Matlab routine $hinfsyn$ and the $sysic$, or system interconnect

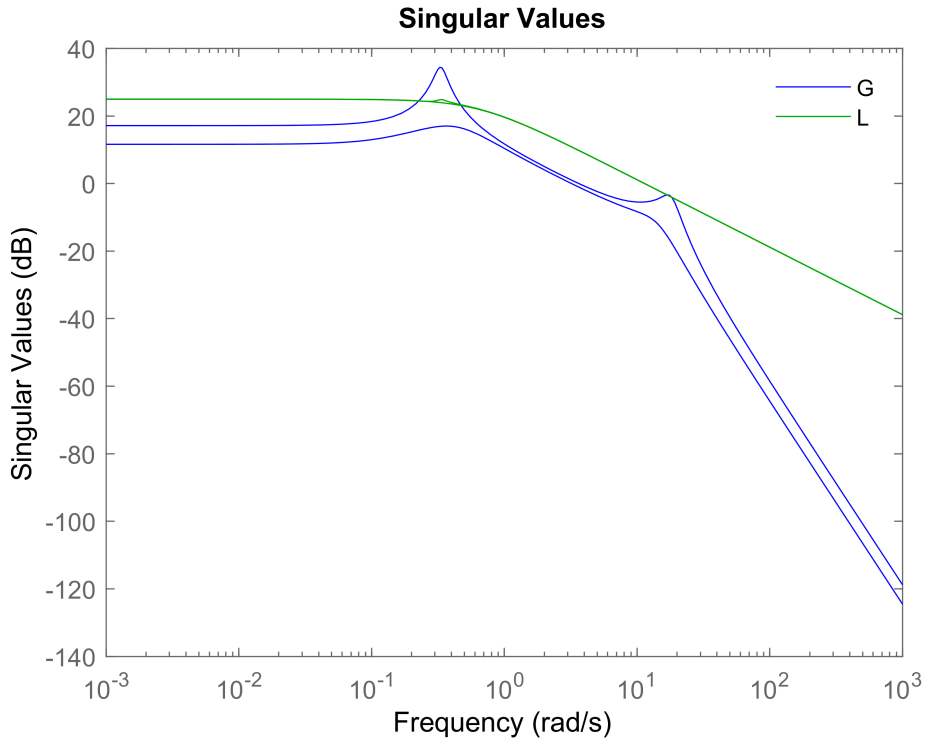

Figure 4: The loop and plant transfer functions using an H_∞ controller. The gain for the loop transfer function is increased in the low frequency range to aid in reducing reference error and increasing disturbance rejection.

command to construct the generalized plant P for larger systems. The initial performance weighting transfer function of the form

$$w_p(s) = \frac{\frac{s}{M} + w_b s}{s + w_b A)} \tag{7}$$

is selected, where M is chosen to be 1.5, A to be 0.08, and a bandwidth w_b of 8. The matrix W_p is a block diagonal matrix of entries w_p used in the robustness analysis. For the values listed above, the plant and loop transfer function singular value plots are shown in figure 4. The gain is increased in the low end of frequencies to shape the system for desired reference an disturbance performance.

The closed loop transfer functions can be used to determine if the system

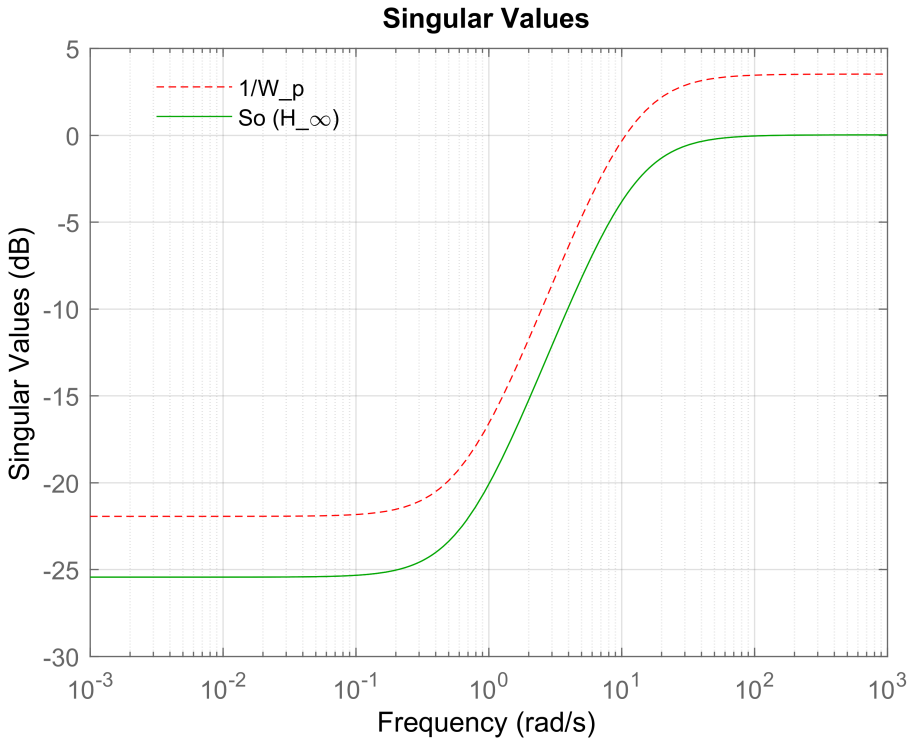

Figure 5: The output sensitivity transfer function used in the nominal performance test of the system is shown to lie beneath the inverse of the performance weighting function representing the system meets nominal performance with the H_∞ controller.

meets nominal performance, the test from equation 4 can be visualized if the singular value plot of the output sensitivity transfer function S_o is less than the inverse of the performance weighting function. The plot of the singular values and the weighting function are shown in figure 5. To visualize the performance of the system, the closed loop step responses for the pitch and roll attitude, as well as the disturbance rejection from an output disturbance to these states at the output of the loop is shown in figure 6.

Under the current controller and performance weighting function, the bandwidth of the system is approximately 18 rad/s with a tracking error of around 8 % for signals under 1 rad/s estimated from the open loop crossover frequency.

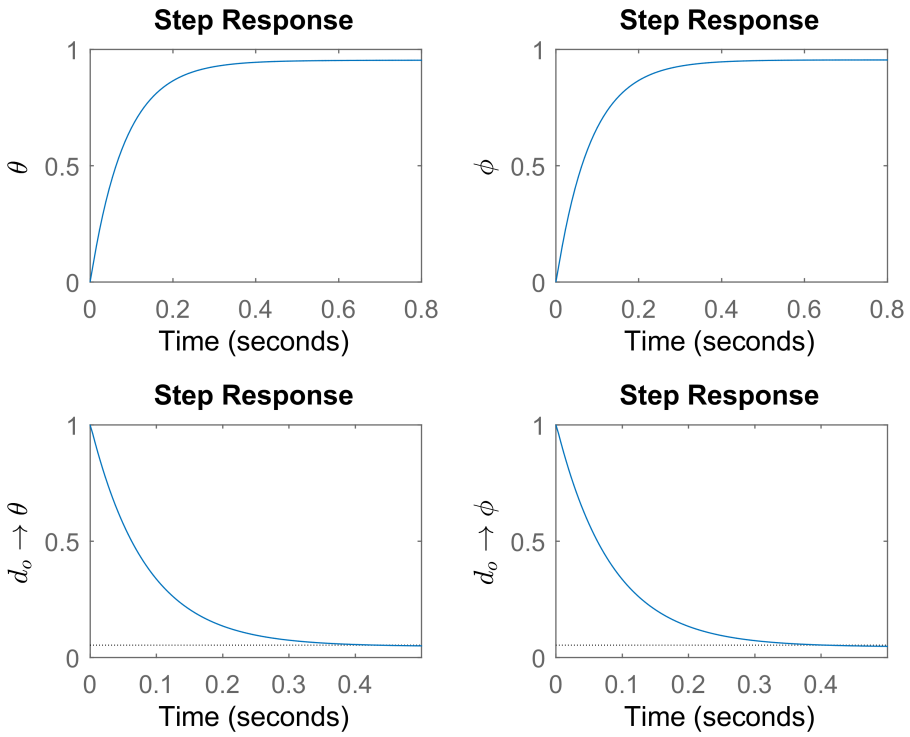

Figure 6: The step response of the closed loop system under H_∞ control and the disturbance rejection to an output disturbance for the roll and pitch attitudes.

3 Performance with Structured Uncertainty

Adding in a structured uncertainty allows for robust stability and performance analysis for the uncertainty model, given that the structure is assumed to be diagonal complex entries. In the case of the aircraft system, the uncertainty model is an input multiplicative uncertainty model with a relative weighting given by the transfer function

$$w_i(s) = \frac{s + 0.2}{0.5s + 1} \tag{8}$$

where the 3x3 diagonal matrix W_i contains the entries of w_i along the diagonal. Adding in the uncertainty to the model requires an additional step to determine robustness. The most conservative approach is assuming no structure (full complex) to the uncertainty and using the infinity norm. However, due to the structure, μ synthesis can be used to obtain a robustness test for performance. The process of μ synthesis requires the system to put into an M-Δ form, but still containing the information form N. This is accomplished by adding in an additional structured Δ block to relate z to w. By doing so, the block diagram has the form of figure 7 with the robust performance test given by

$$||\mu_{\hat{\Delta}}(N)||_\infty < 1 \tag{9}$$

where $\hat{\Delta}$ is the block diagonal matrix from figure 7. In Matlab this is accomplished by setting the block structure of the the uncertainty and using the command *mussv* to obtain the lower and upper bounds of the structured singular value. Using this method on the system results in a new H_∞ controller accounting for the uncertainty in the model. The robust stability singular value plot for the original controller can be seen in figure 8. Notice the system can satisfy nominal performance but not robust stability. With the current controller, the margin on robust stability is 0.61, meaning that the system can only withstand slightly over half of the expected max perturbations.

The robust performance singular value plot for the original controller can be seen in figure 9, where the system also does not satisfy robust performance. For robust performance, the margin is 0.084 suggesting the current controller settings are not robust to performance and require significant adjustment. To develop a better controller and account for some of the uncertainty model, the process is repeated with a different performance weighting function W_p.

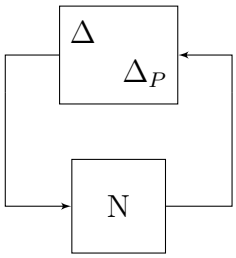

Figure 7: The M-Δ form of the structured singular value with N as M and a block diagonal matrix for Δ.

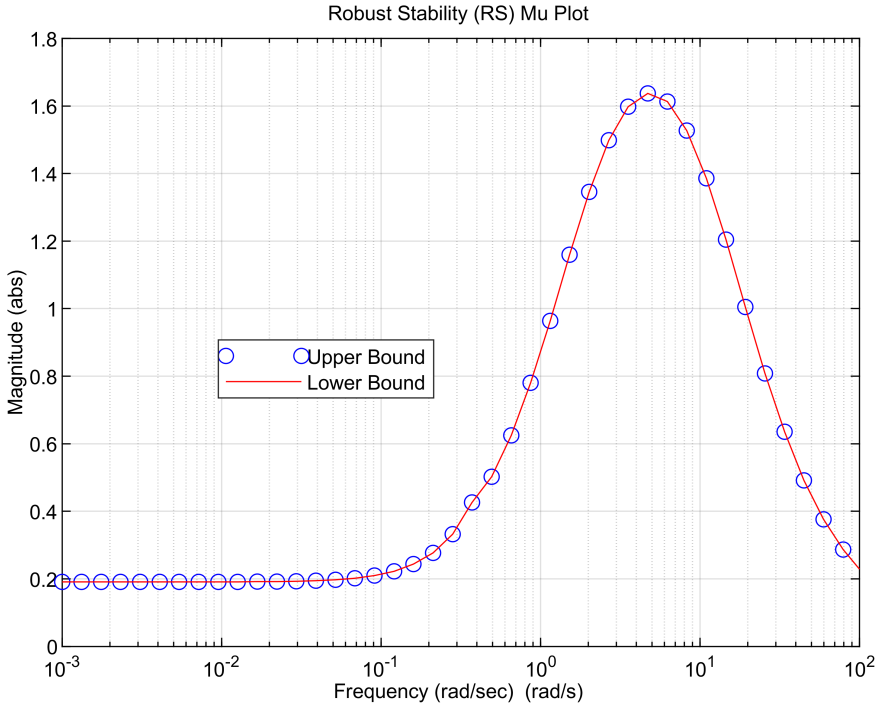

Figure 8: Robust stability plot of the original weighting function and H_∞ controller. The peak of the μ plot is greater than 1, representing robust stability is not met. The robust stability margin in this case is 0.61.

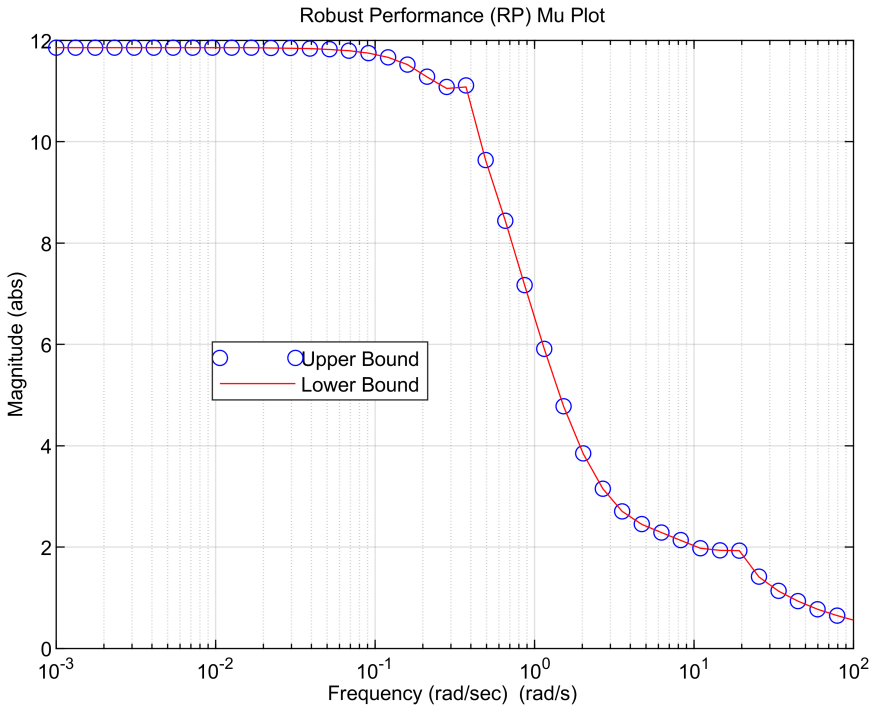

Figure 9: Robust performance plot of the original weighting function and H_∞ controller. In this case, the peak of the robust performance mu plot is much greater than 1 meaning the system does not satisfy robust performance. The robust performance margin with this controller is 0.084, such that the system can withstand approximately 1/12 of the max perturbation from the uncertainty model.

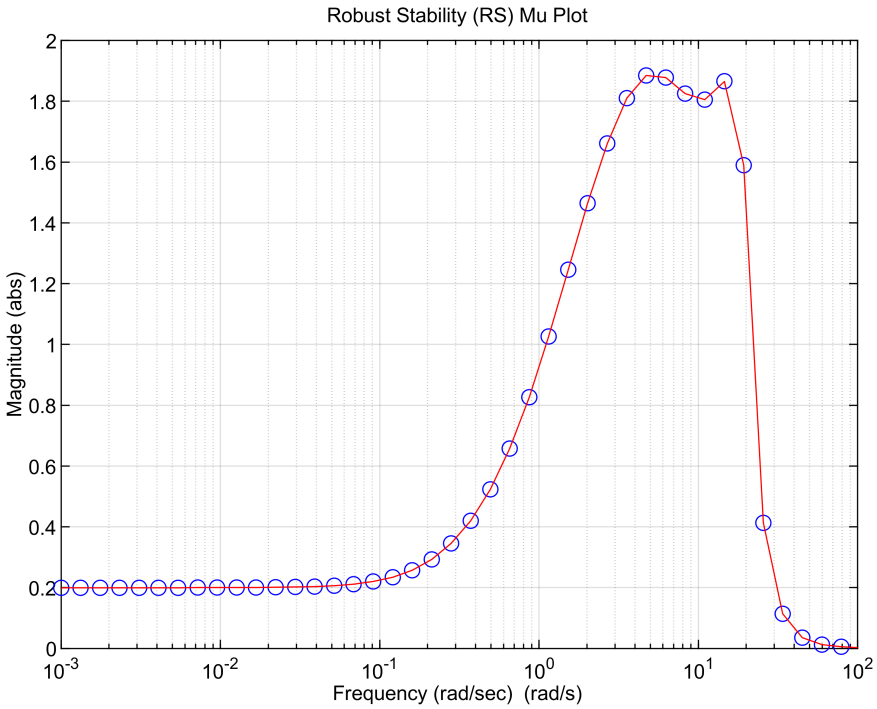

Figure 10: Robust stability plot of the adjusted weighting function and H_∞ controller, the robust stability has been decreased to a robust stability margin of 0.53 from a margin of 0.61.

The values of the performance weighting function were changed such that A = 0.25, M = 1.2, and the bandwidth $w_b = 18$. Using the new performance weightings, the Matlab command $hinfysn$ was used to develop a new controller and the results of the robust stability and performance can be seen in figures 10 and 11 respectively.

with the new controller, the robust stability decreased slightly but robust performance margins were increased. The performance of the new controller with respect to reference tracking and disturbance rejection is shown in figure 12. Although the reference tracking is more oscillatory and bandwidth have decreased, the system is more robustly stable and has a much greater margin. Even so, the system can withstand approximately 1/4 of the max perturbation from the uncertainty model.

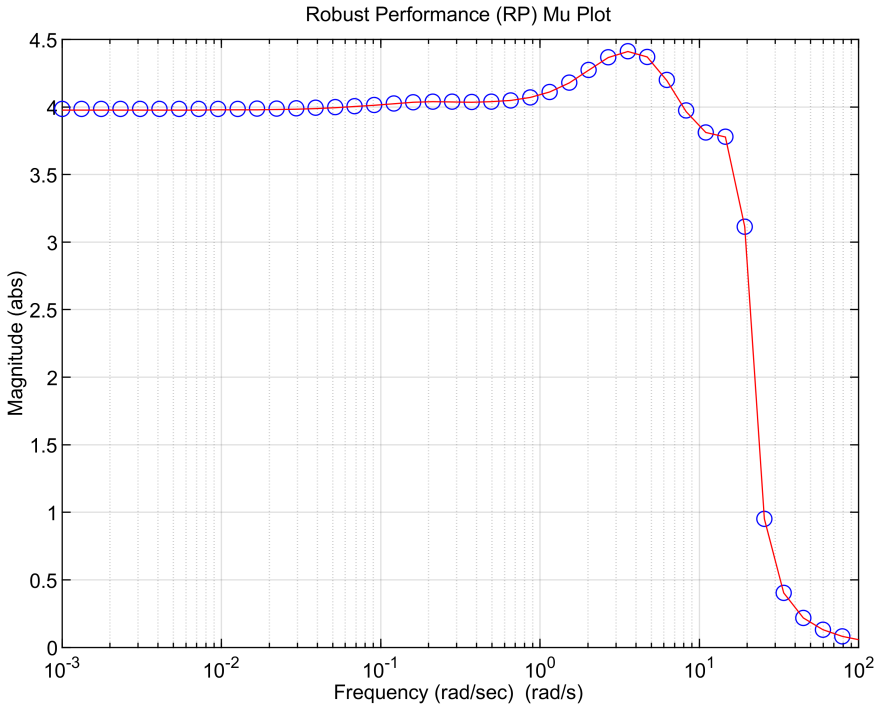

Figure 11: Robust performance plot of the adjusted weighting function and H_∞ controller, the robust performance margin increased almost three times from 0.084 to 0.227.

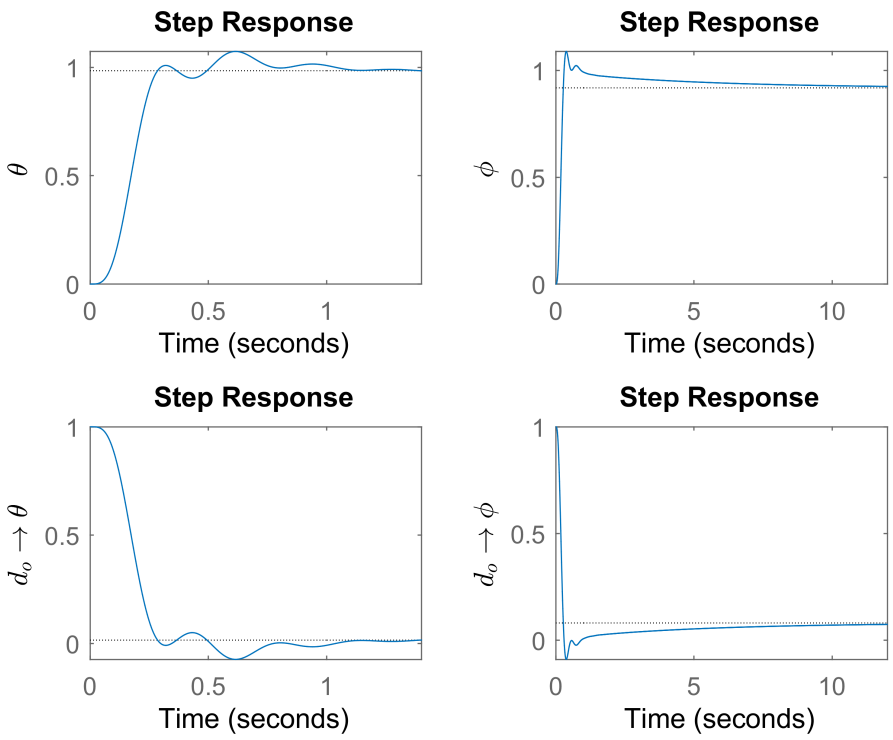

Figure 12: The step response of the closed loop system under the new H_∞ control law and the disturbance rejection to an output disturbance for the roll and pitch attitudes.

4 D-K iteration

To further improve the controller, the design loop of creating an H_∞ controller with robust performance can be cast as an iterative problem by augmenting the generalized plant with a matrix D and D^{-1}. The scaling factors allow for a controller to be recursively solved. The system is fit to scaling factors and the factors are used in the next iteration of the H_∞ controller. For the aircraft, the D-K iteration is done manually by setting a number of iterations, in this case 6, and recursively solved to lower the H_∞ cost and produce better margins for robust stability and performance. The final D-K iteration produces results seen in the open loop with figure 13, closed loop with 14, and with the reference tracking and disturbance rejection plots in figure 15. The difference in the robust performance and robust stability singular value plots is shown in figure 17 and 16.

5 Conclusion

Ultimately, the final controller improved the robust performance and robust stability of the system to a multiplicative input uncertainty model at the expense of bandwidth and the transient response. The robust stability margin with the final controller was 0.4685 which is slightly reduced from the margin of 0.53 from the second controller based off the revised weighting function. The robust performance margin was 0.24 for the final controller. Although not satisfying robust performance, the improvement from the starting margin of 0.084 is three times more robust. Adding in additional states such as roll and pitch rates could improve the margins at the expense of more internal states for the H_∞ controller.

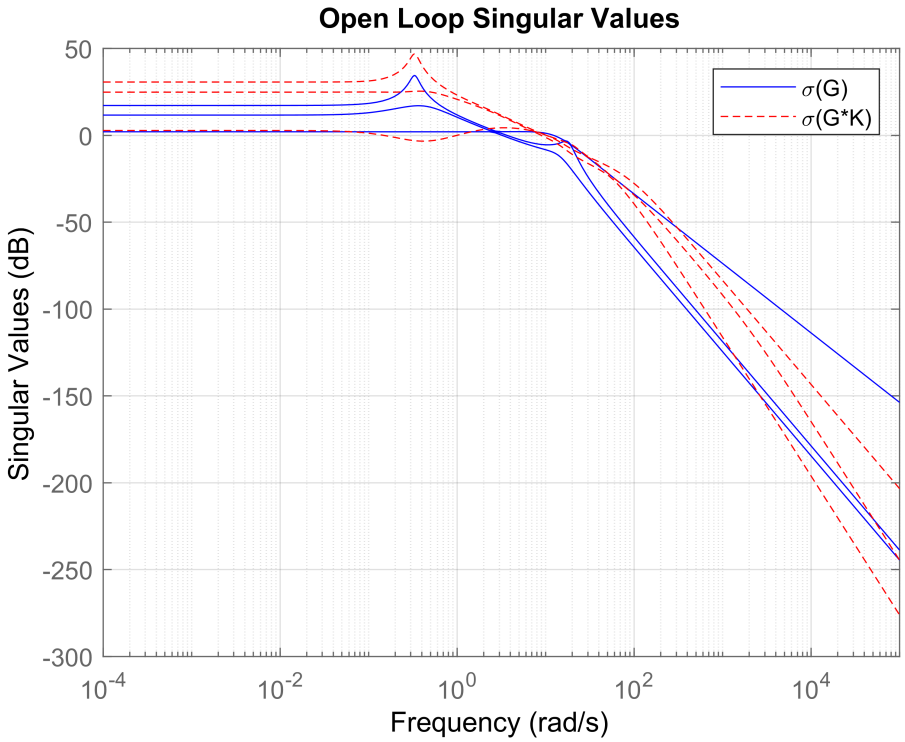

Figure 13: The loop and plant transfer functions using an H_∞ controller. The gain for the loop transfer function is increased in the low frequency range to aid in reducing reference error and increasing disturbance rejection, and is lowered at high frequencies for robustness properties and noise rejection.

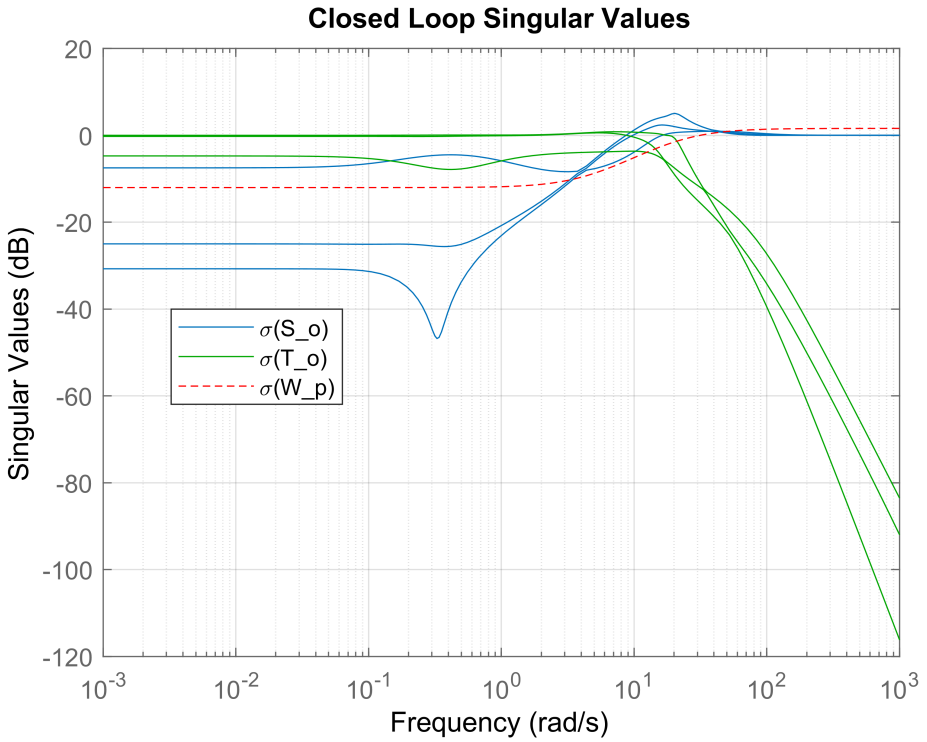

Figure 14: The output sensitivity transfer function used in the nominal performance test of the system. The system does not lie beneath the inverse of the performance weighting function and does not meet nominal performance for all frequencies with the H_∞ controller.

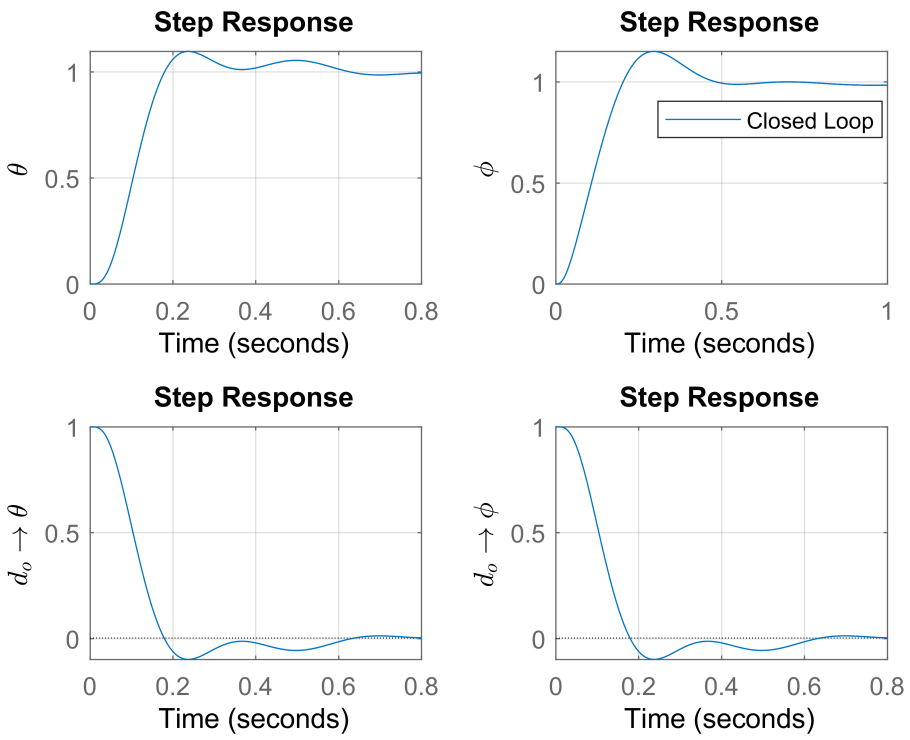

Figure 15: The step response of the closed loop system under the new H_∞ control law after D-K iteration and the disturbance rejection to an output disturbance for the roll and pitch attitudes.

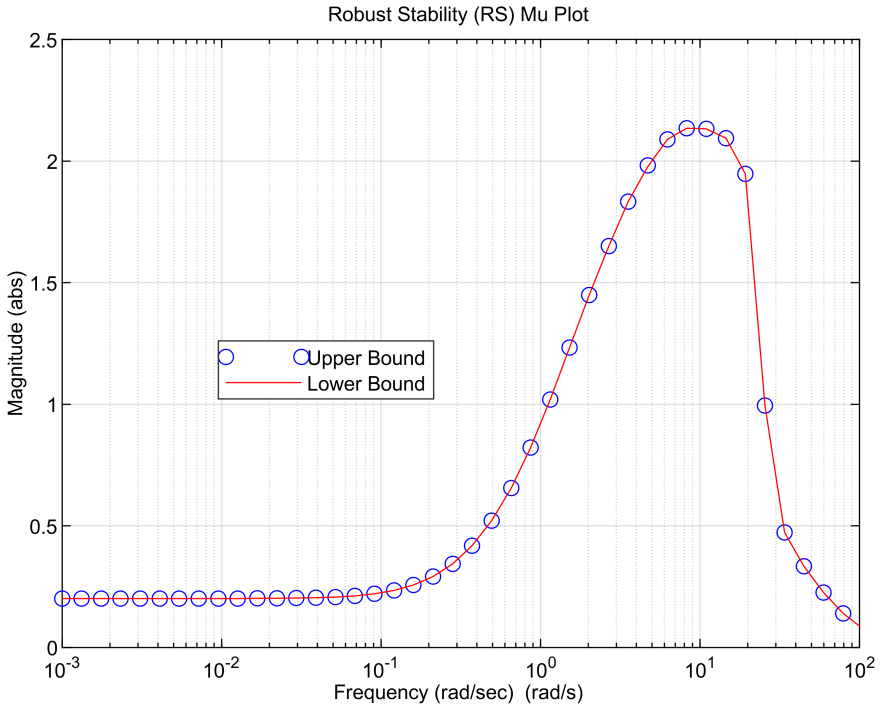

Figure 16: Robust stability plot after the final value from the D-K iteration for the H_∞ controller.

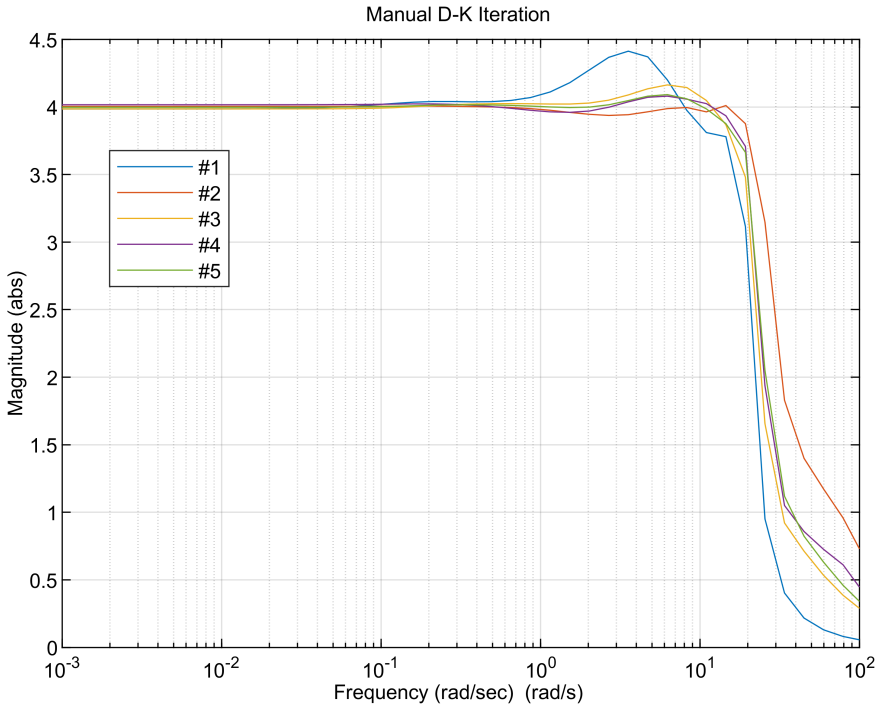

Figure 17: Robust performance plot for each iteration of the D-K iteration for the H_∞ controller, ultimately increasing the robust performance margin.

A Code

A.1 Matlab Analysis

```
%% Robust Inner Loop Control of a 6-DOF Rotary Wing Unmanned Aircraft System
% Riley Kenyon
% 05/07/2020
clear all; close all; clc;

% Load dynamics
run Final_Project_Heli_dynamics.m

% Determine weighting functions
s = tf('s');
A = 0.08; M = 1.5; wb = 8;
% A = 3; M = 5; wb = 10;
wp = (s/M+wb)/(s+wb*A); Wp = blkdiag(wp,wp,wp);

% Frequency range requirements
w_rng = {10^-3,10^3};
w_r = wb/10;                    % Track up low frequencies
w_d = wb*10;                    % Reject disturbances in high frequencies

% Setup P w/o uncertainty
systemnames = 'G Wp';           % performance weights
inputvar = '[w(3); u(3)]';      % (w, u)'
outputvar = '[Wp; w-G]';        % (z, v)', assumes output signal of block
input_to_G = '[u]';             % Inputs from block diagram
input_to_Wp = '[w-G]';
sysoutname = 'P';
sysic;
P = minreal(ss(P));

% Define size of u and v for controller
n_meas = 3;
n_ctrl = 3;

% Synthesize H_inf controller
```

```
[K,CL,gamma(1),info] = hinfsyn(P,n_meas,n_ctrl,'method','ric','Tolgam',1e-3,'DISPL

% Construct CL transfer functions
I = eye(3);
omega = logspace(-3,3);
So = minreal(inv(I+G*K));
To = minreal(G*K*inv(I+G*K));
Si = minreal(inv(I+K*G));
Ti = minreal(K*G*inv(I+K*G));

% Open-loop nominal performance
L = G*K;
figure(1); clf;
sigma(G(1:2,1:2),'b',L(1:2,1:2),'g',w_rng);
legend('G','L')
% R = rectangle('position',[w_rng{1},0,w_r-w_rng{1},db(1/A)]);
% R.LineStyle = '--';
% R.LineWidth = 1.5;
% R.EdgeColor = 'r';
% D = rectangle('position',[w_d,db(1/M),w_rng{2} - w_d, -db(1/M)]);
% D.LineStyle = '--';
% D.LineWidth = 1.5;
% D.EdgeColor = 'r';
%
% Closed-loop nominal performance
figure(2), clf;
sigma(inv(Wp(1,1)),'r--',So(1:2,1:2),'g',{10^-3,10^3});
legend('1/{W_p}','So (H_{\infty})'); grid on;

% Step responses
figure(3), clf;
subplot(2,2,1)
step(To(1,1));
ylabel('$\theta$','interpreter','latex')

subplot(2,2,2)
step(To(2,2));
ylabel('$\phi$','interpreter','latex')
```

```
% Disturbance rejection
subplot(2,2,3)
step(So(1,1));
ylabel('$d_o \rightarrow \theta$','interpreter','latex')

subplot(2,2,4)
step(So(2,2));
ylabel('$d_o \rightarrow \phi$','interpreter','latex')

%% With modeled uncertainty
% Same performance weighting
A = 0.08; M = 1.5; wb = 8;
wp = (s/M+wb)/(s+wb*A); Wp = blkdiag(wp,wp,wp);

% Define structure
wi = (s+0.2)/(0.5*s+1); Wi = eye(3)*wi;

% Setup P with delta
systemnames = 'G Wp Wi';            % added uncertainty weighting function (Wi)
inputvar = '[ydel(3); w(3); u(3)]'; % (y_delta, w, u)'
outputvar = '[Wi; Wp; w-G]';        % (u_delta, z, v)', assumes output signal of b
input_to_G = '[u + ydel]';          % Inputs from block diagram
input_to_Wp = '[G]';
input_to_Wi = '[u]';
sysoutname = 'P';
sysic;
P = minreal(ss(P));

% Mu plot parameters
LinMagopt = bodeoptions;
LinMagopt.PhaseVisible = 'off';
LinMagopt.XLim = [1e-3 1e2];
LinMagopt.MagUnits = 'abs';
omega = logspace(-3,3);

% Define N
clear N;
```

```
%N = [-Wi*Ti Wi*K*So; -Wp*So*G -Wp*So];
N = lft(P,K);

% Robust stability
figure(5); clf;
wiTi_frd = frd(N(1:3,1:3),omega);
clear BlkStruct; BlkStruct = [1 1; 1 1; 1 1];
[mu_wiTi] = mussv(wiTi_frd,BlkStruct);
bodeplot(mu_wiTi(1,1),'bo',mu_wiTi(1,2),'r-',LinMagopt)
legend('Upper Bound','Lower Bound')
xlabel('Frequency (rad/sec)'); grid on;
ylabel('Mu upper/lower bounds (abs)');
title('Robust Stability (RS) Mu Plot');

% Robust performance
figure(6); clf;
N_frd = frd(N,omega);
clear BlkStruct; BlkStruct = [1 1; 1 1; 1 1; 3 3];
[mu_N] = mussv(N_frd,BlkStruct);
bodeplot(mu_N(1,1),'bo',mu_N(1,2),'r-',LinMagopt)
legend('Upper Bound','Lower Bound')
xlabel('Frequency (rad/sec)'); grid on;
ylabel('Mu upper/lower bounds (abs)');
title('Robust Performance (RP) Mu Plot');

%% Iterate on design weighting functions
% Neither robust stability or performance is met with current controller
% A = 3; M = 5; wb = 10;
% A = 0.08; M = 1.5; wb = 8;
% A = 1.8; M = 2; wb = 10;
% A = 0.5; M = 1.8; wb = 1;
A = 0.25; M = 1.2; wb = 18;

clear wp Wp
wp = (s/M+wb)/(s+wb*A); Wp = blkdiag(wp,wp,wp); % cannot have third weight zero
sysic;
P = minreal(ss(P));
```

```
% Synthesize H_inf controller
clear K
[K,CL,gamma(1),info] = hinfsyn(P,n_meas,n_ctrl,'method','ric','Tolgam',1e-3,'DISPL
clear So To Si Ti
So = minreal(inv(I+G*K));
To = minreal(G*K*inv(I+G*K));
Si = minreal(inv(I+K*G));
Ti = minreal(K*G*inv(I+K*G));

% Define N
clear N;
% N = [-Wi*Ti Wi*K*So; -Wp*So*G -Wp*So];
N = lft(P,K);

% Robust stability
figure(7); clf;
clear wiTi_frd, wiTi_frd = frd(N(1:3,1:3),omega);
clear BlkStruct; BlkStruct = [1 1; 1 1; 1 1];
[mu_wiTi] = mussv(wiTi_frd,BlkStruct);
bodeplot(mu_wiTi(1,1),'bo',mu_wiTi(1,2),'r-',LinMagopt)
xlabel('Frequency (rad/sec)'); grid on;
ylabel('Mu upper/lower bounds (abs)');
title('Robust Stability (RS) Mu Plot');

% Robust performance
figure(8); clf;
N_frd = frd(N,omega);
clear BlkStruct; BlkStruct = [1 1; 1 1; 1 1; 3 3];
[mu_N] = mussv(N_frd,BlkStruct);
bodeplot(mu_N(1,1),'bo',mu_N(1,2),'r-',LinMagopt)
xlabel('Frequency (rad/sec)'); grid on;
ylabel('Mu upper/lower bounds (abs)');
title('Robust Performance (RP) Mu Plot');

% Step Responses
figure(9), clf;
subplot(2,2,1)
step(To(1,1));
```

```
ylabel('$\theta$','interpreter','latex')

subplot(2,2,2)
step(To(2,2));
ylabel('$\phi$','interpreter','latex')

% Disturbance rejection
subplot(2,2,3)
step(So(1,1));
ylabel('$d_o \rightarrow \theta$','interpreter','latex')

subplot(2,2,4)
step(So(2,2));
ylabel('$d_o \rightarrow \phi$','interpreter','latex')

% The design is much slower but achieves robust stability and performance

%% Use D-K iteration to redesign controller to achieve improved RP
% D-K iteration (manual/automated)
d0 = 1;
D = append(d0,d0,d0,tf(I),tf(I));    % Define initial scalings

% Initial K-step: Compute initial H-infinity controller
[K,CL,gamma(1),info] = hinfsyn(D*P*inv(D),n_meas,n_ctrl,'method','ric','Tolgam',1e

% Initial D-step: Compute robust performance level
Nf = frd(lft(P,K),omega);
[mu_Nf,mu_Info] = mussv(Nf,BlkStruct);
mu_RP(1) = norm(mu_Nf(1,1),inf,1e-6)

% Generate mu plot
figure(10); clf;
bodeplot(mu_Nf(1,1),LinMagopt)
hold on; grid on;

% D-K iteration loop
N = 5;      % Number of iterations
for i = 2:N
```

```
 % Fit resulting D-scales
 [dsysl,dsysr] = mussvunwrap(mu_Info);          % Extract scalings
 dsysl = dsysl/dsysl(3,3);                      % Normalize
 di = fitfrd(genphase(dsysl(1,1)),4);           % Fit 4th order systems to scaling
 Di = append(di,di,di,tf(I),tf(I));             % Generate new scaling matrix

 % K-step
 [Ki,CL,gamma(i),info] = hinfsyn(Di*P*inv(Di),n_meas,n_ctrl,'method','ric','Tolgam

 % D-step
 Nf = frd(lft(P,Ki),omega);
 [mu_Nf,mu_Info] = mussv(Nf,BlkStruct);
 mu_RP(i) = norm(mu_Nf(1,1),inf,1e-6)

 % Add to plot
 bodeplot(mu_Nf(1,1),LinMagopt); grid on;
    if i == N,
        Kfinal = Ki;
    end;
end;
figure(10);
xlabel('Frequency (rad/sec)');
ylabel('Mu upper/lower bounds (abs)');
legend('#1','#2','#3','#4','#5')
title('Manual D-K Iteration');

figure(11); clf;
wiTi_frd = frd(Nf(1:3,1:3),omega);
clear BlkStruct; BlkStruct = [1 1; 1 1; 1 1];
[mu_wiTi] = mussv(wiTi_frd,BlkStruct);
bodeplot(mu_wiTi(1,1),'bo',mu_wiTi(1,2),'r-',LinMagopt)
legend('Upper Bound','Lower Bound')
xlabel('Frequency (rad/sec)'); grid on;
ylabel('Mu upper/lower bounds (abs)');
title('Robust Stability (RS) Mu Plot');


% Performance plots for final controller
```

```
% Closed loop TFs
So = inv(I+G*Kfinal);
To = G*Kfinal*inv(I+G*Kfinal);

% Open loop singular values
figure(12); clf;
sigma(G,'b',G*Kfinal,'r--',{1e-4,1e5}); grid on;
legend('\sigma(G)','\sigma(G*K)');
title('Open Loop Singular Values')

% Closed loop singular values and Wp
figure(13); clf;
sigma(So,To,'g-',inv(Wp(1,1)),'r--',{1e-3,1e3}); grid on;
legend('\sigma(S_o)','\sigma(T_o)','\sigma(W_p)');
title('Closed Loop Singular Values')

% Closed loop step response and disturbance rejection
figure(14); clf;
subplot(2,2,1),step(To(1,1)); grid on; %step(G(1,1),'b'
ylabel('$\theta$','interpreter','latex')
subplot(2,2,2), step(To(2,2)); grid on; %G(2,2),'b',
ylabel('$\phi$','interpreter','latex')
legend('Closed Loop');
subplot(2,2,3), step(So(1,1)); grid on;
ylabel('$d_o \rightarrow \theta$','interpreter','latex')
subplot(2,2,4), step(So(1,1)); grid on;
ylabel('$d_o \rightarrow \phi$','interpreter','latex')
```