Class: Robust Multivariate Control
Professor: Dr. Sean Humbert
TAs: Santosh Chaganti
Student: Steve Gillet
Date: May 6, 2025
Assignment: Final Project

Project Overview

The original project was for the robust inner loop control of a 6-DOF Micro-Helicopter described in the problem statement below.

For this project you will synthesize a series of inner loop controllers using H_∞ and μ-synthesis techniques for an 11-state rotary wing micro-helicopter and analyze their robustness properties. The state vector is given by $x = [u, v, p, q, \phi, \theta, a_s, b_s, w, r, r_{fb}]^T$. The variables u, v, w are the vehicle translational velocities along the vehicle's longitudinal, lateral, and vertical body axis, p, q, r are the vehicle roll, pitch, and yaw rate respectively, ϕ and θ are the roll and pitch attitude, a_s and b_s are the rotor dynamics states (flap of the rotor relative to the body), and r_{fb} is the washed out yaw rate, a filtering state associated with on-board yaw rate feedback. The control input vector is $u = [\delta_{long}, \delta_{lat}, \delta_{ped}]^T$, corresponding 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 yaw rate r_{fb}. The equations of motion linearized about hover are given in the MATLAB m-file on the Canvas website.

The objective of the project is to design a controller to ensure robust command tracking of $\theta_r \to \theta$ and $\phi_r \to \phi$ for the reference $r = \begin{bmatrix}\theta_r & \phi_r\end{bmatrix}^T$ in the presence of a structured multiplicative input uncertainty.

I chose to do a 2-wheeled self-balancing robot with different drive instead because I had attempted one as an undergrad and I've been thinking about it a while.

The dynamics are actually a bit more simple for the self-balancing robot, essentially a cart inverted pendulum with the differential drive, but I believe that I can accomplish the same objectives by changing a few of the specifics such as the input to output channels of interest. I'll go into detail of the changes as they come up. Below is an overview of the system and a derivation of the state-space model.

Dynamics of a 2-Wheeled Self-Balancing Robot with Differential Drive

This section derives the equations of motion for a 2-wheeled self-balancing robot capable of moving and turning in a 2D plane. The robot is modeled as an inverted pendulum mounted on a wheeled base, with separate torque inputs applied to each wheel to enable differential drive. The nonlinear dynamics are derived using Newtonian mechanics, linearized around the upright equilibrium, and expressed in state-space form.

1. System Description and Variables

The robot consists of a body with mass M, whose center of mass is located at a distance l above the wheel axle, and two wheels with total mass m, each of radius R. The wheels are separated by a distance d. Separate torques τ_l and τ_r are applied to the left and right wheels, respectively, enabling both translation and rotation.

The state variables are:

- θ: Tilt angle of the body from the vertical (upright at $\theta = 0$).
- $\dot{\theta}$: Angular velocity of the tilt.
- ψ: Yaw angle, defining the robot's orientation in the 2D plane.
- $\dot{\psi}$: Yaw rate (angular velocity about the vertical axis).
- v: Forward velocity of the robot along its heading (direction of ψ).

The state vector is:

$$z = \begin{bmatrix} \theta & \dot{\theta} & \psi & \dot{\psi} & v \end{bmatrix}^T \tag{1}$$

The control inputs are the torques:

$$u = \begin{bmatrix} \tau_l & \tau_r \end{bmatrix}^T \tag{2}$$

Alternatively, inputs can be expressed as average and differential torques:

$$\tau_{\text{avg}} = \frac{\tau_l + \tau_r}{2}, \quad \tau_{\text{diff}} = \frac{\tau_r - \tau_l}{2} \tag{3}$$

Constants include gravitational acceleration $g = 9.81\,\text{m/s}^2$ and the moment of inertia I_ψ of the robot about the vertical axis.

2. Nonlinear Equations of Motion

The dynamics are derived by considering the translational motion of the system, the rotational motion of the pendulum-like body, and the yaw motion due to differential torques.

2..1 Translational Motion

The robot moves in the direction of its heading (aligned with ψ). The force from each wheel is $F_l = \frac{\tau_l}{R}$ and $F_r = \frac{\tau_r}{R}$. The total force along the heading is:

$$F = F_l + F_r = \frac{\tau_l + \tau_r}{R} \tag{4}$$

Summing forces in the direction of motion, accounting for the pendulum's effect:

$$(M + m)\dot{v} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2 \sin\theta = \frac{\tau_l + \tau_r}{R} \tag{5}$$

Here:

- $(M + m)\dot{v}$: Total mass times acceleration along the heading.
- $ml\ddot{\theta}\cos\theta$: Horizontal component of the pendulum's acceleration.
- $-ml\dot{\theta}^2 \sin\theta$: Centripetal force term.

2..2 Rotational Motion (Pendulum)

For the body, summing torques about its center of mass (or using force balance in the horizontal direction):

$$ml\ddot{\theta} + mg\sin\theta + m\dot{v}\cos\theta = 0 \tag{6}$$

Here:

- $ml\ddot{\theta}$: Rotational acceleration.
- $mg\sin\theta$: Gravity torque.
- $m\dot{v}\cos\theta$: Coupling with the base's acceleration.

2..3 Yaw Motion

The differential torque causes rotation about the vertical axis. The torque about the center of the robot is:

$$\tau_\psi = \frac{d}{2}(F_r - F_l) = \frac{d}{2}\left(\frac{\tau_r}{R} - \frac{\tau_l}{R}\right) = \frac{d}{2R}(\tau_r - \tau_l) \tag{7}$$

The yaw dynamics are:

$$I_\psi\ddot{\psi} = \frac{d}{2R}(\tau_r - \tau_l) \tag{8}$$

where I_ψ is the moment of inertia about the vertical axis.

3. Linearization Around the Equilibrium Point

Linearize around the equilibrium: $\theta = 0$, $\dot{\theta} = 0$, small $\dot{\psi}$, and constant v. Approximations:

- $\sin\theta \approx \theta$
- $\cos\theta \approx 1$
- $\dot{\theta}^2 \approx 0$

The equations become:
1. **Translational**:

$$(M+m)\dot{v} + ml\ddot{\theta} = \frac{\tau_l + \tau_r}{R} \tag{9}$$

2. **Rotational**:

$$ml\ddot{\theta} + mg\theta + m\dot{v} = 0 \tag{10}$$

3. **Yaw**:

$$I_\psi\ddot{\psi} = \frac{d}{2R}(\tau_r - \tau_l) \tag{11}$$

4. Solving for Accelerations

Solve for $\ddot{\theta}$, $\dot{v}$, and $\ddot{\psi}$:
From (11):

$$\dot{v} = -l\ddot{\theta} - g\theta \tag{12}$$

Substitute into (10):

$$(M+m)(-l\ddot{\theta} - g\theta) + ml\ddot{\theta} = \frac{\tau_l + \tau_r}{R} \tag{13}$$

$$-(M+m)l\ddot{\theta} - (M+m)g\theta + ml\ddot{\theta} = \frac{\tau_l + \tau_r}{R} \tag{14}$$

$$-Ml\ddot{\theta} - (M+m)g\theta = \frac{\tau_l + \tau_r}{R} \tag{15}$$

$$Ml\ddot{\theta} + (M+m)g\theta = -\frac{\tau_l + \tau_r}{R} \tag{16}$$

$$\ddot{\theta} = -\frac{(M+m)g}{Ml}\theta - \frac{1}{MlR}(\tau_l + \tau_r) \tag{17}$$

Substitute back into (13):

$$\dot{v} = -l\left(-\frac{(M+m)g}{Ml}\theta - \frac{1}{MlR}(\tau_l + \tau_r)\right) - g\theta \tag{18}$$

$$\dot{v} = \frac{(M+m)g}{M}\theta + \frac{1}{MR}(\tau_l + \tau_r) - g\theta \tag{19}$$

$$\dot{v} = \frac{mg}{M}\theta + \frac{1}{MR}(\tau_l + \tau_r) \tag{20}$$

From (12):

$$\ddot{\psi} = \frac{d}{2I_\psi R}(\tau_r - \tau_l) \tag{21}$$

Using $\tau_{\text{avg}} = \frac{\tau_l+\tau_r}{2}$, $\tau_{\text{diff}} = \frac{\tau_r-\tau_l}{2}$:

$$\tau_l + \tau_r = 2\tau_{\text{avg}}, \quad \tau_r - \tau_l = 2\tau_{\text{diff}} \tag{22}$$

Rewrite (18) and (20):

$$\ddot{\theta} = -\frac{(M+m)g}{Ml}\theta - \frac{2}{MlR}\tau_{\text{avg}} \tag{23}$$

$$\dot{v} = \frac{mg}{M}\theta + \frac{2}{MR}\tau_{\text{avg}} \tag{24}$$

$$\ddot{\psi} = \frac{d}{I_\psi R}\tau_{\text{diff}} \tag{25}$$

5. State-Space Model

The state-space representation is:

$$\dot{z} = Az + Bu \tag{26}$$

where $z = [\theta, \dot{\theta}, \psi, \dot{\psi}, v]^T$, $u = [\tau_{\text{avg}}, \tau_{\text{diff}}]^T$, and:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -\frac{(M+m)g}{Ml} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{mg}{M} & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ -\frac{2}{MlR} & 0 \\ 0 & 0 \\ 0 & \frac{d}{I_\psi R} \\ \frac{2}{MR} & 0 \end{bmatrix} \tag{27}$$

This model captures the unstable balancing dynamics (θ) and the decoupled yaw dynamics (ψ), driven by the average and differential torques.

(a)

Construct the corresponding block diagram and formulate the generalized plant with the relevant uncertainty and performance weighting functions to achieve the desired command tracking behavior. Note that the output as prescribed has three states; you will include the r_{fb} state in the reference input as well so that the G and K matrix transfer functions are square. However from the design perspective we are only interested in the performance in the $\theta_r \to \theta$ and $\phi_r \to \phi$ channels.

I will do the same thing for my model except that there are just the two measurable, feedback-available, states of interest θ and ψ, so we will look at the performance in the $\theta_r \to \theta$ and $\psi_r \to \psi$ channels.

Here is the block diagram for a multiplicative input uncertainty model:

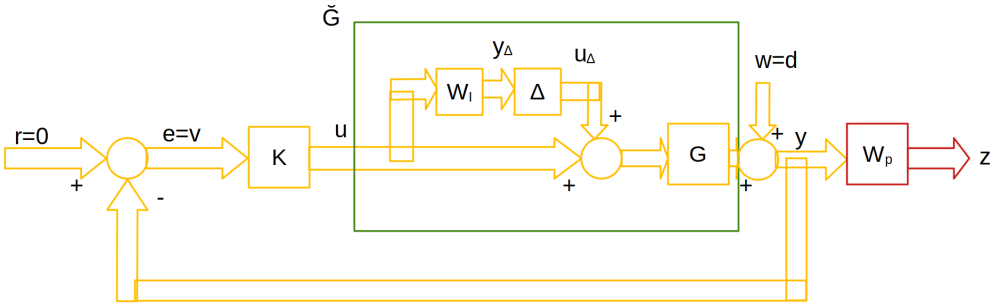

Figure 1: Block diagram of the system with multiplicative input uncertainty.

Here is the generalized plant formulation:
Putting y_Δ, z, and v in terms of u_Δ, w, and u:

$$y_\Delta = W_I u \tag{28}$$
$$z = W_P[w + G(u_\Delta + u)] \tag{29}$$
$$v = -[w + G(u_\Delta + u)] \tag{30}$$

$$\begin{bmatrix} y_\Delta \\ z \\ v \end{bmatrix} = \begin{bmatrix} 0 & 0 & W_I \\ W_P G & W_P & W_P G \\ -G & -I & -G \end{bmatrix} \begin{bmatrix} u_\Delta \\ w \\ u \end{bmatrix}$$

(b)

Synthesize a H_∞ controller and analyze its nominal performance via singular value plots of the open loop transfer functions G, GK and closed loop transfer functions S_o, T_b and performance weight W_p. Provide plots of step responses in $\theta_r \to \theta$ and $\phi_r \to \phi$ and the output disturbance quantities $d_\theta \to \theta$ and $d_\phi \to \phi$ for a disturbance vector $d_o = \begin{bmatrix} d_\theta & d_\phi \end{bmatrix}^T$. Comment on the performance of your design in terms of its bandwidth, tracking error, and disturbance rejection capabilities.

I designeed the H_∞ controller using 'sysic' and 'hinfsyn' in Matlab as shown in the code below.

```
systemnames = 'G Wp';
inputvar = '[w(2); u(2)]';
outputvar = '[Wp; -G-w]';
input_to_G = '[u]';
input_to_Wp = '[w + G]';
sysoutname = 'P';
sysic;
P = minreal(ss(P));


[K, CL, gamma, info] = hinfsyn(P,nOut,nIn,'method','ric','Tolgam',1e-3,'DISPLAY','on');
```

I used a generic weighting function that we had used in class before, it basically weighs each channel the same and decays as the frequency increases.

```
s = tf('s');
Wp_theta = (s/100 + 1)/(s + 1*0.01);  % Low-frequency gain ~100, bandwidth ~1 rad/s
Wp_psi = (s/100 + 1)/(s + 1*0.01);    % Adjust parameters later
Wp = blkdiag(Wp_theta, Wp_psi);
Wp = ss(Wp);
```

Then I plotted G, GK, S_O, T_O, and W_P using the 'sigma' function.

Then I plotted the responses of the closed loop system to step inputs in the reference and disturbance channels.

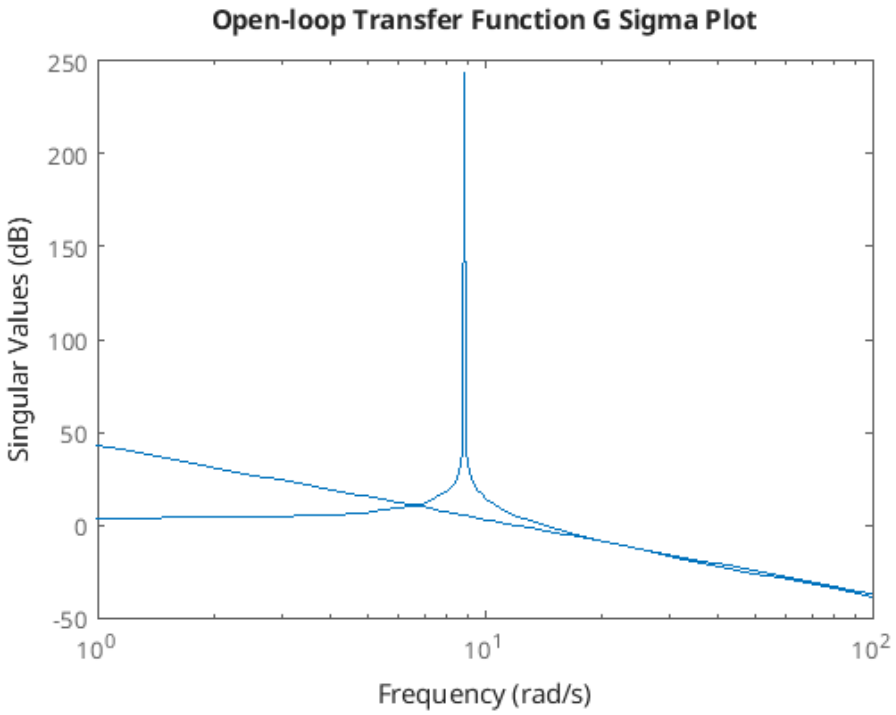

Figure 2: Singular values of G.

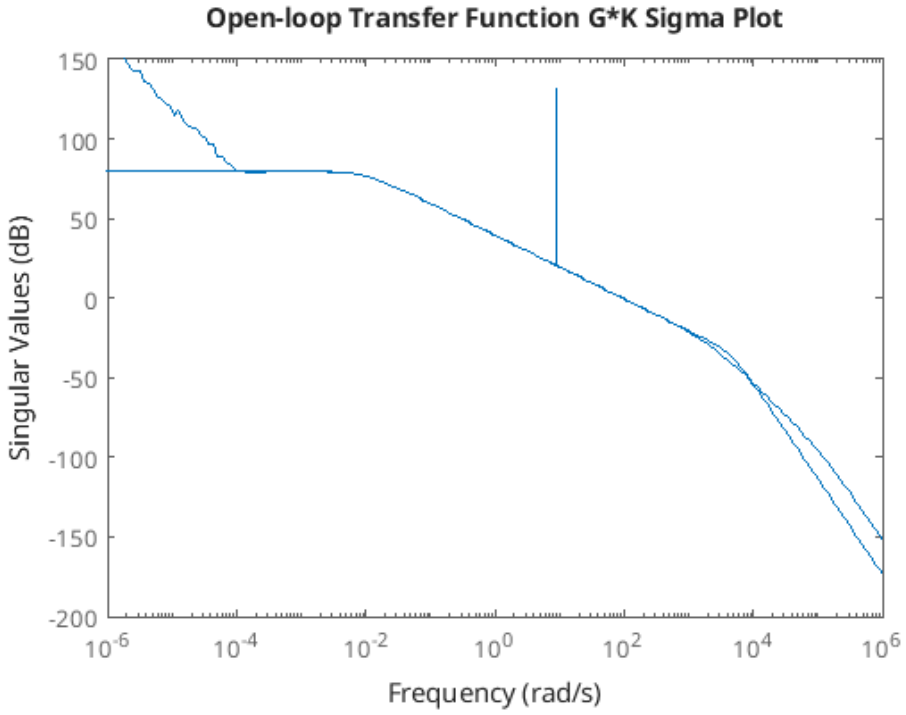

Figure 3: Singular values of GK.

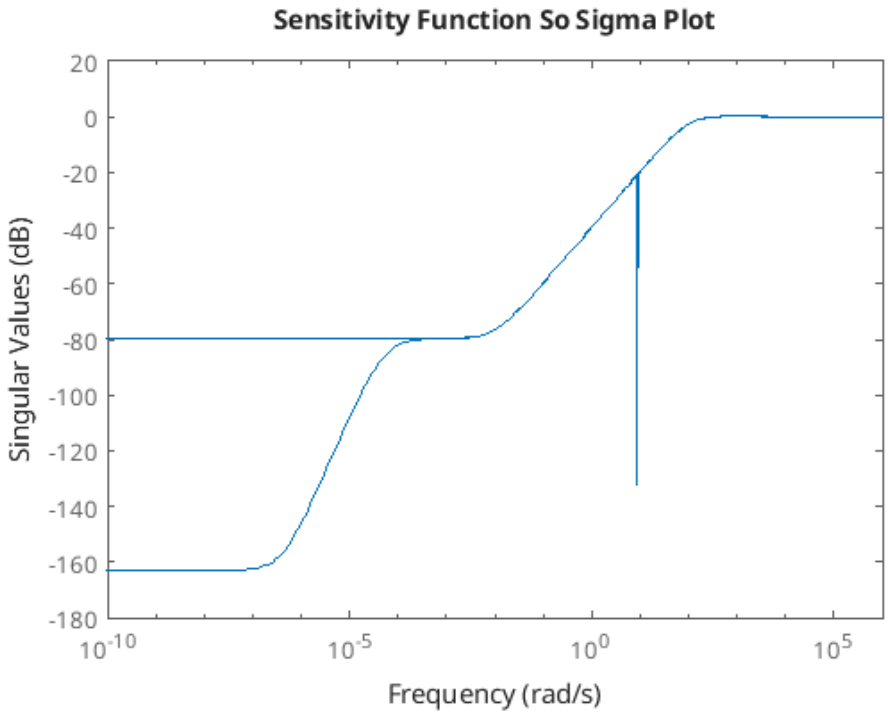

Figure 4: Singular values of So.

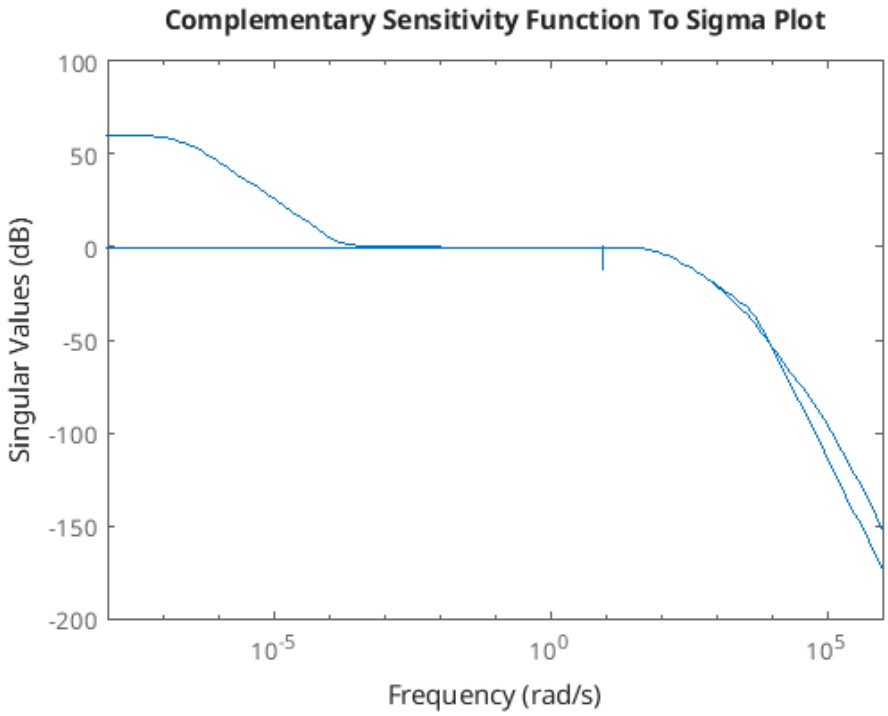

Figure 5: Singular values of To.

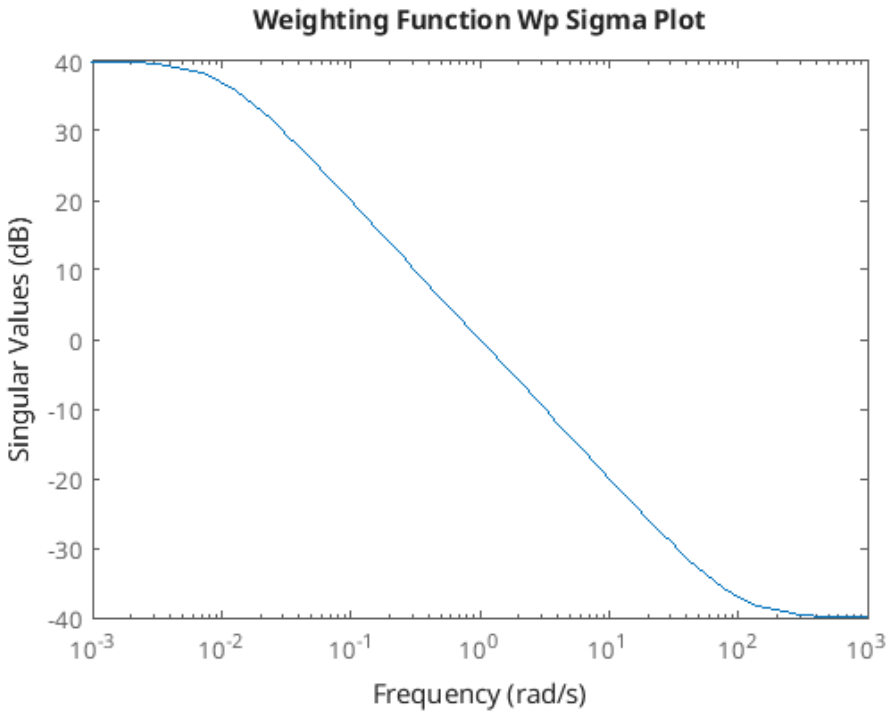

Figure 6: Singular values of Wp.

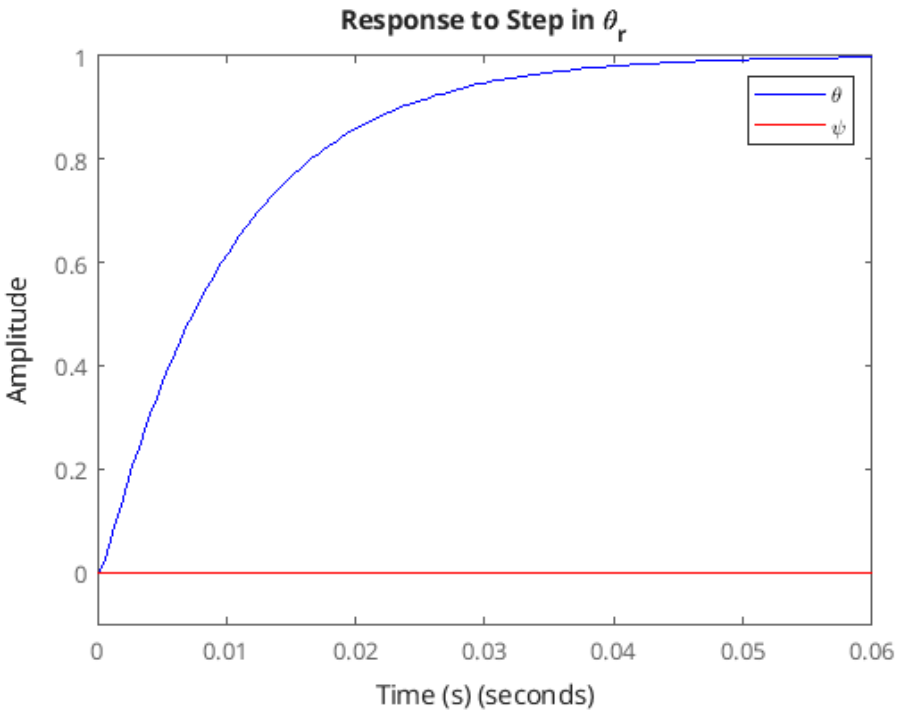

Figure 7: Step response for theta.

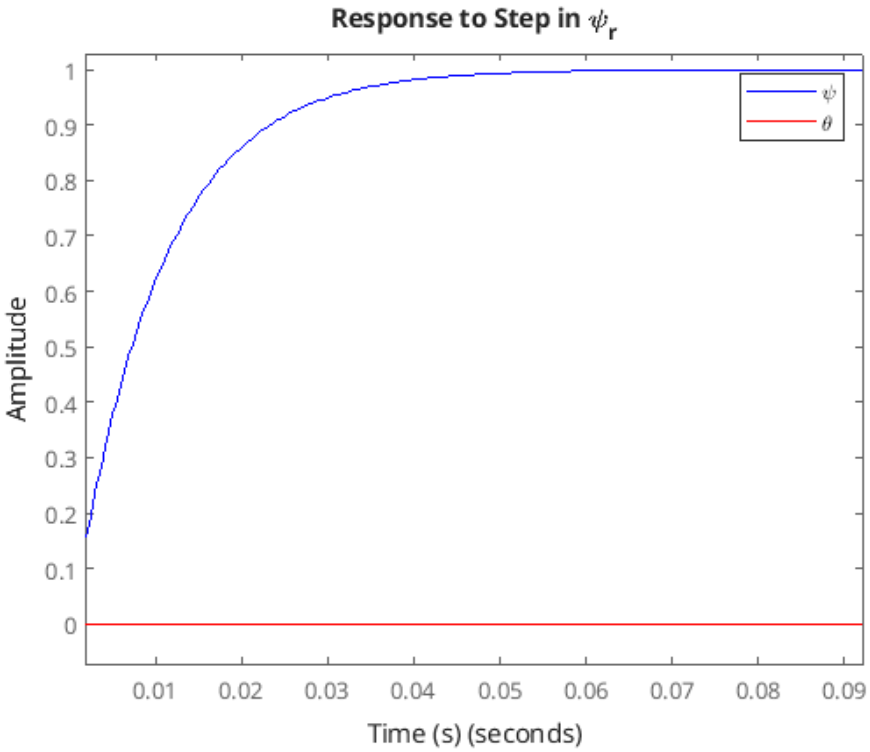

Figure 8: Step response for psi.

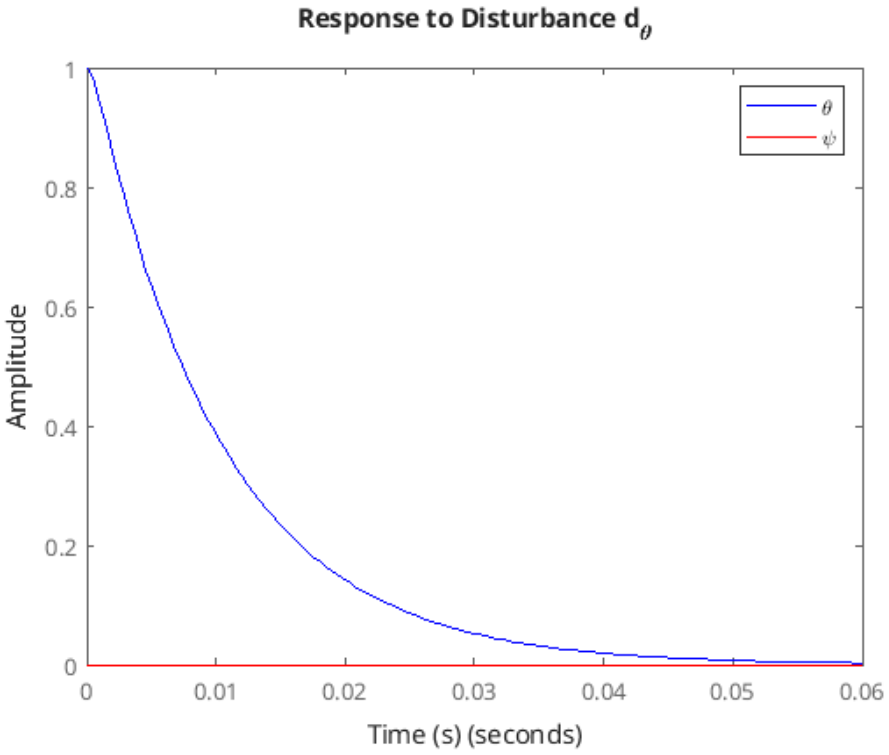

Figure 9: Step response for theta disturbance.

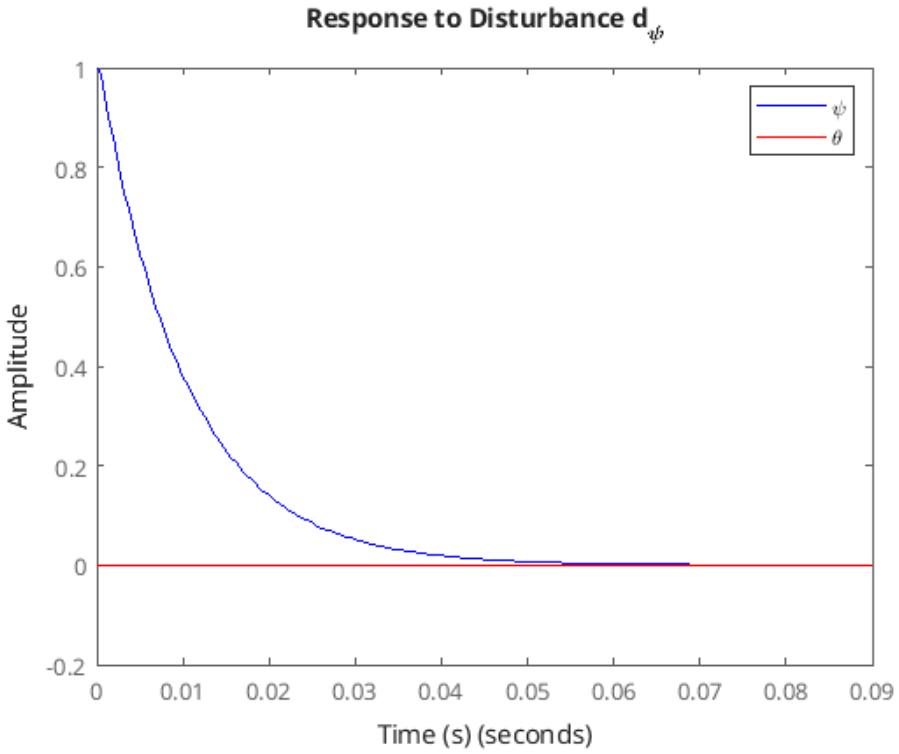

Figure 10: Step response for psi disturbance.

The closed loop bandwidth is about 90 rad/s, the tracking error is 6% up to 6.28 rad/s, and the settling time is 0.03 seconds for disturbance rejection.

(c)

Update the generalized plant to include a diagonal structured complex uncertainty with a relative weight of $w_i(s) = \frac{s+0.2}{0.5s+1}$ *in each channel. Assume* $\|\Delta\|_\infty \leq 1$*, and compute the upper and lower bounds on the structured singular value for the controller obtained above, and determine the margins* $(1/\mu_{peak})$ *on robust stability and robust performance. Provide an iteration on the performance weighting function to obtain improved margins. Note that if you are using only* θ*,* ϕ*, and* r_{fb} *as outputs, I don't expect you to achieve robust performance for the given uncertainty profiles, but rather use these tools to assess how much uncertainty your design can tolerate. Again provide plots of step responses in* θ *and* ϕ *and the disturbance channels and comment on the performance of your design.*

I used the following code to update the generalized plant with the input uncertainty.

```
Wi = (s+0.2)/(0.5*s+1)*eye(nIn);

systemnames = 'G Wp Wi';
inputvar = '[ydel(2); w(2); u(2)]';
outputvar = '[Wi; Wp; -G-w]';
input_to_G = '[u + ydel]';
input_to_Wp = '[G+w]';
input_to_Wi = '[u]';
sysoutname = 'P';
sysic;
P = minreal(ss(P));


[K, CL, gamma, info] = hinfsyn(P,nOut,nIn,'method','ric','Tolgam',1e-3,'DISPLAY','on');
```

```
15
16 omega = logspace(-3,3);
17 blk = [1 0; 1 0];
18
19 N = frd(lft(P,K),omega);
20 [muRS, infoRS] = mussv(N(1:2,1:2), blk);
21
22 bodeOptions = bodeoptions;
23 bodeOptions.PhaseVisible = 'off';
24 bodeOptions.XLim = [1e-3 1e3];
25 bodeOptions.MagUnits = 'abs';
26
27 figure(11);
28 bodeplot(muRS, bodeOptions);
29 hold on; grid on;
30 title('Robust Stability Plot');
31
32 blk = [1 0; 1 0; 2 2];
33 [muRP, infoRP] = mussv(N, blk);
34 figure(12);
35 bodeplot(muRP, bodeOptions);
36 hold on; grid on;
37 title('Robust Performance Plot');
38
39 muRSpeak = max(sigma(muRS));
40 disp('Robust Stability Margin:');
41 disp(1/muRSpeak);
42
43 muRPpeak = max(sigma(muRP));
44 disp('Robust Performance Margin:');
45 disp(1/muRPpeak);
```

Then I plotted the structured singular values and computed the margins for robust stability and performance.

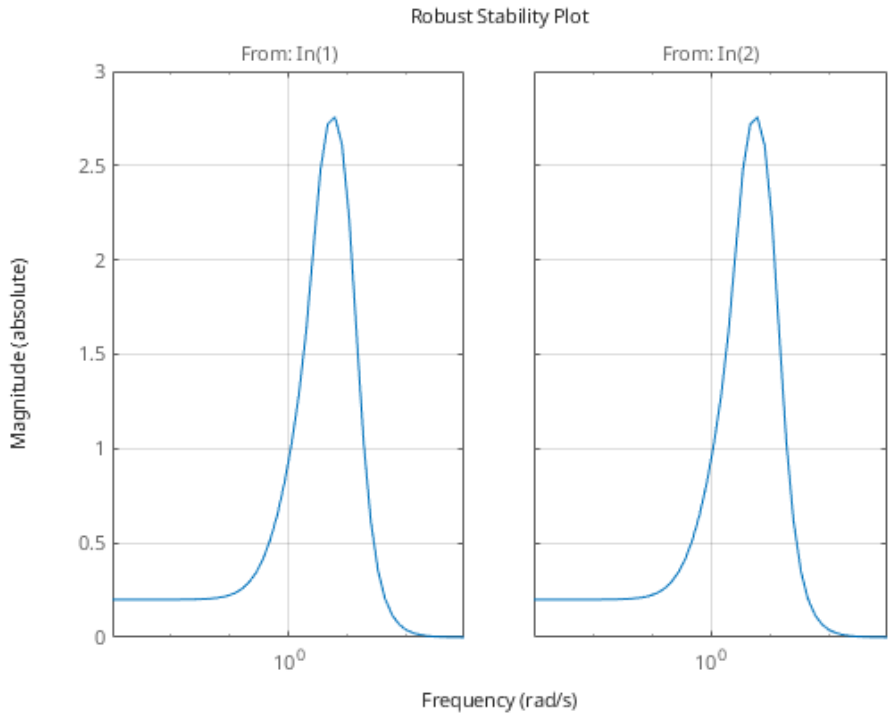

Figure 11: Robust stability plot.

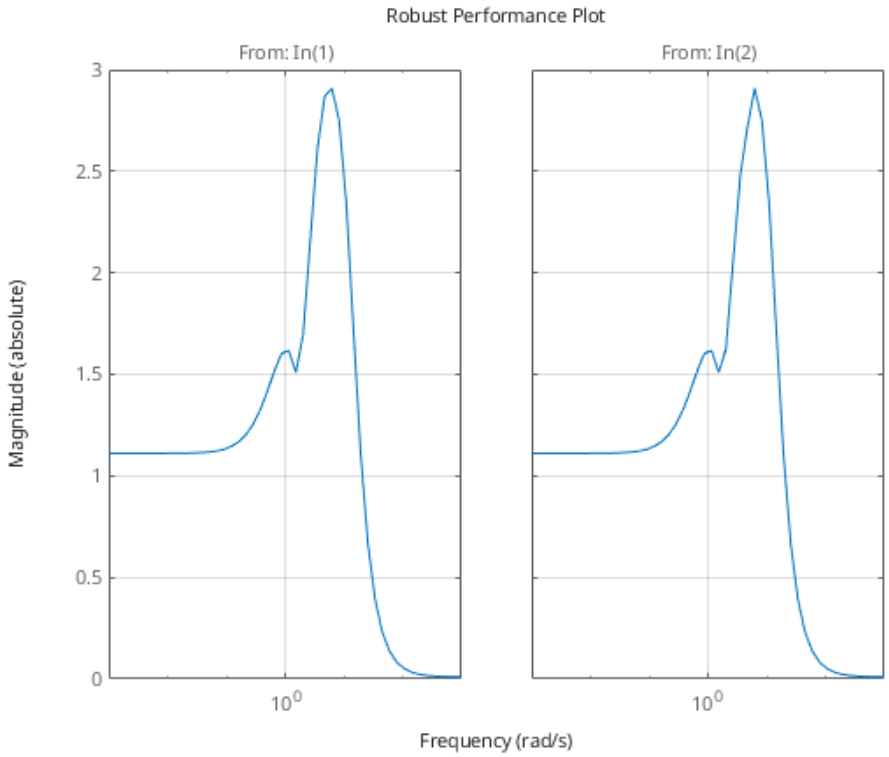

Figure 12: Robust performance plot.

The robust stability margin is 0.2564 and the robust performance margin is 0.2431. I then decreased the ω_B of the weighting function to 0.01 to push down on the closed loop bandwidth.

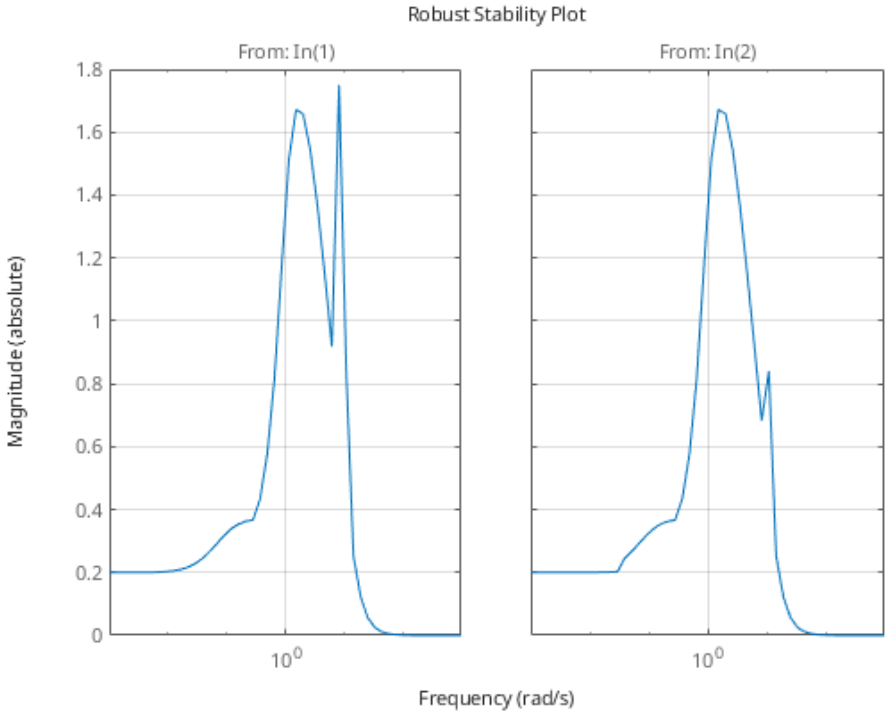

Figure 13: Robust stability plot.

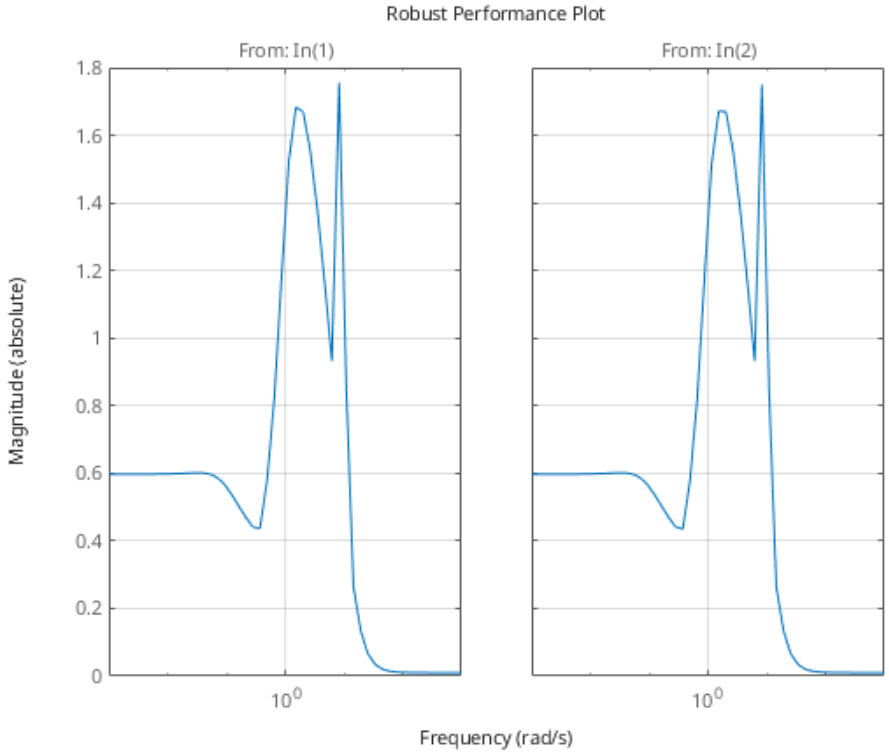

Figure 14: Robust performance plot.

That put the robust stability margin at 0.4230 and robust performance at 0.4034
Then I plotted the same step responses as before in the reference input and disturbance inputs.

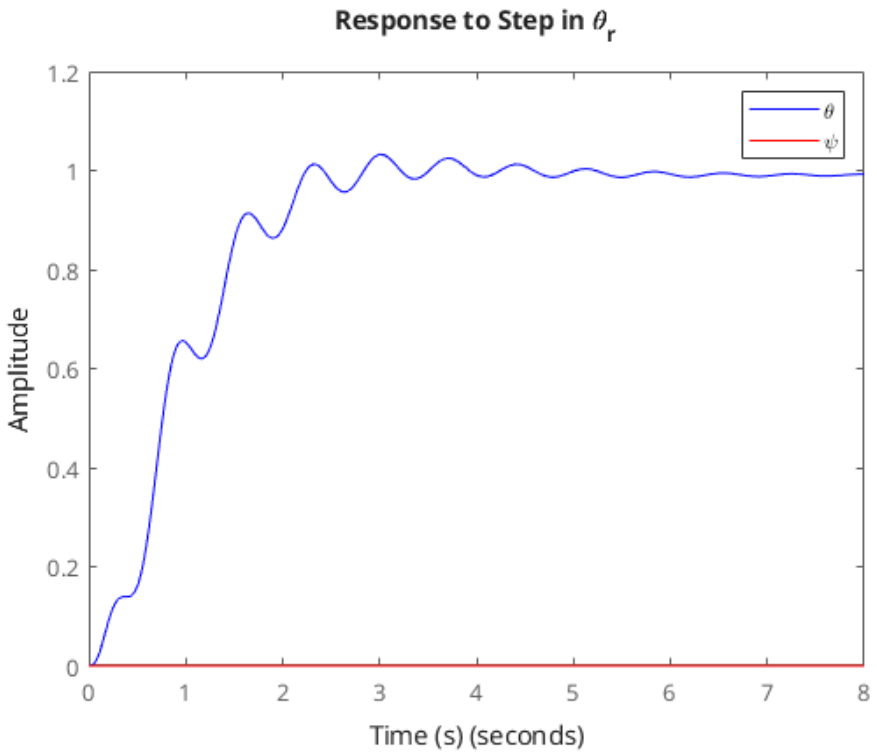

Figure 15: Step response for theta with uncertainty controller.

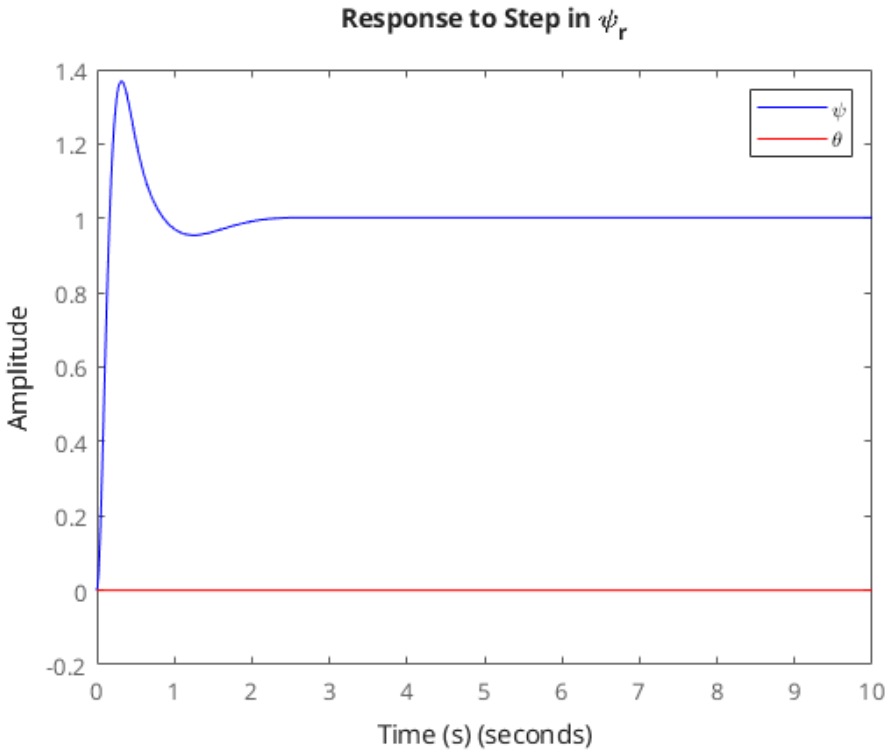

Figure 16: Step response for psi with uncertainty controller.

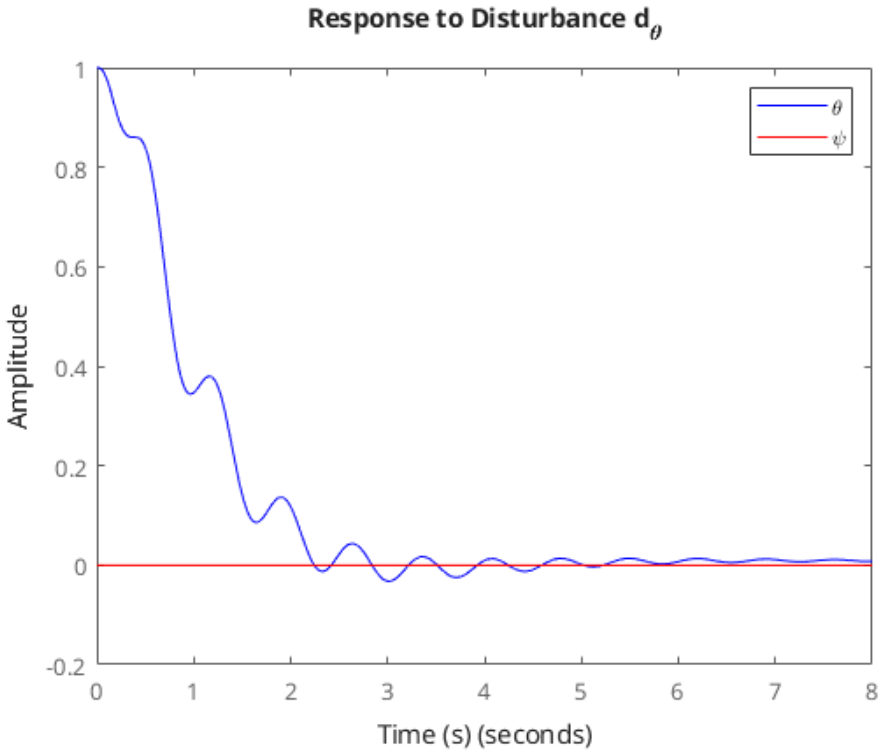

Figure 17: Step response for theta disturbance with uncertainty controller.

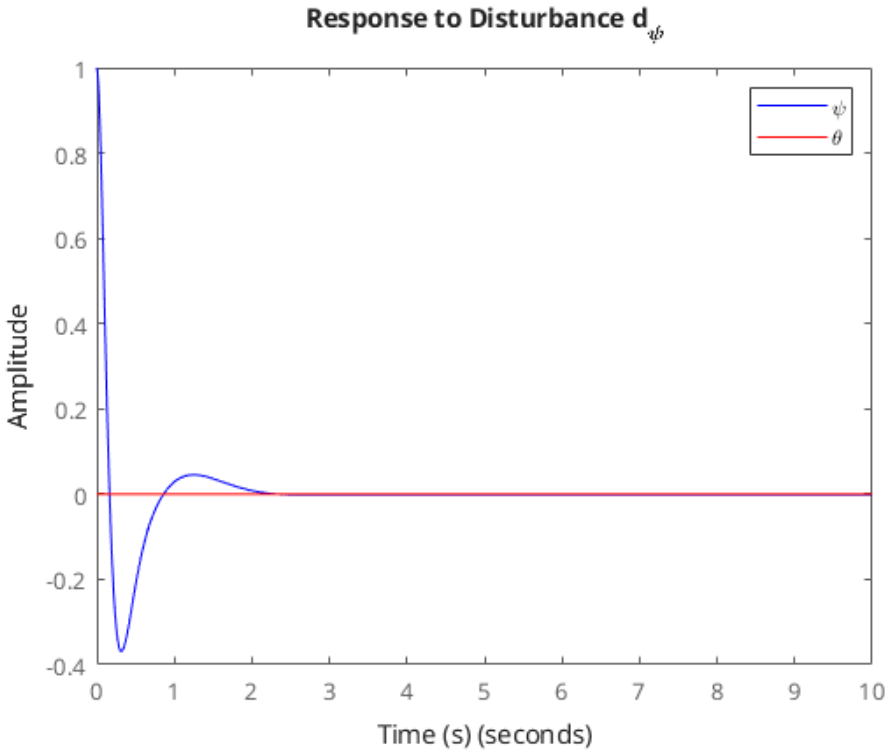

Figure 18: Step response for psi disturbance with uncertainty controller.

And you can see the decreased performance. The settling time is more like 2 seconds and there is some overshoot now.

(d)

Use D-K iteration to redesign your controller to achieve improved robust performance. Re-generate all the appropriate plots and comment on your final design, in particular the tradeoff between high bandwidth and robust performance in the presence of uncertainty.

I did the D-K iteration using the 'musyn' function in Matlab.

```
Delta = [ultidyn('D_1', [1 1]) 0 ; 0 ultidyn('D_2', [1 1])];
Punc = lft(Delta,P);
[Kauto,clp,dkinfo] = musyn(Punc, nOut, nIn);

N = frd(lft(P,Kauto), omega);
[muAuto, infoAuto] = mussv(N, blk);
muRP = norm(muAuto(1,1),inf,1e-6);
```

The robust performance margin was 0.5126 and I generated the plots for the open and closed loop singular values as well as the step responses again.

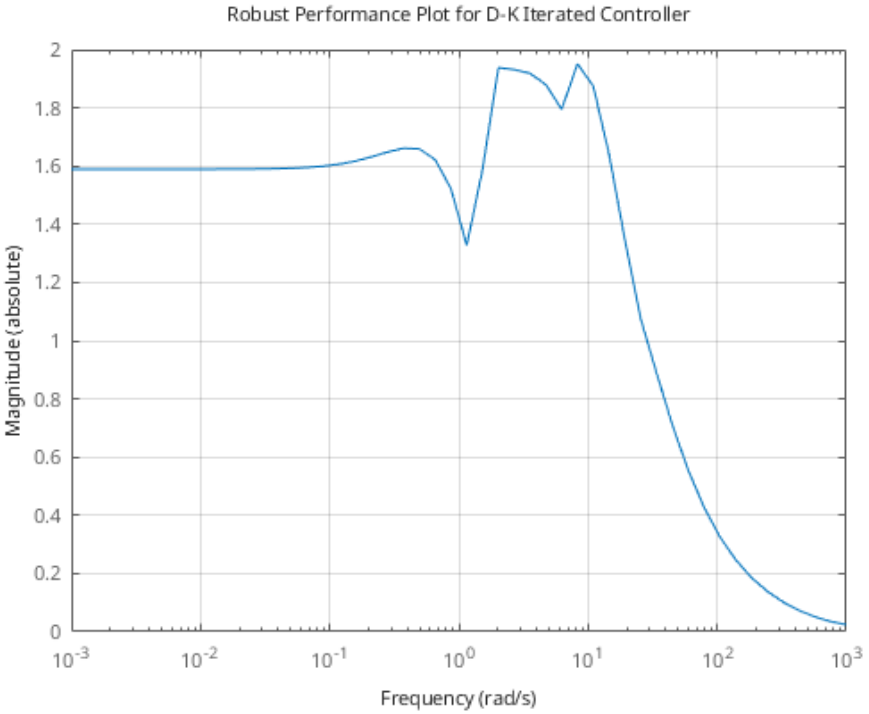

Figure 19: Robust performance plot with D-K iteration.

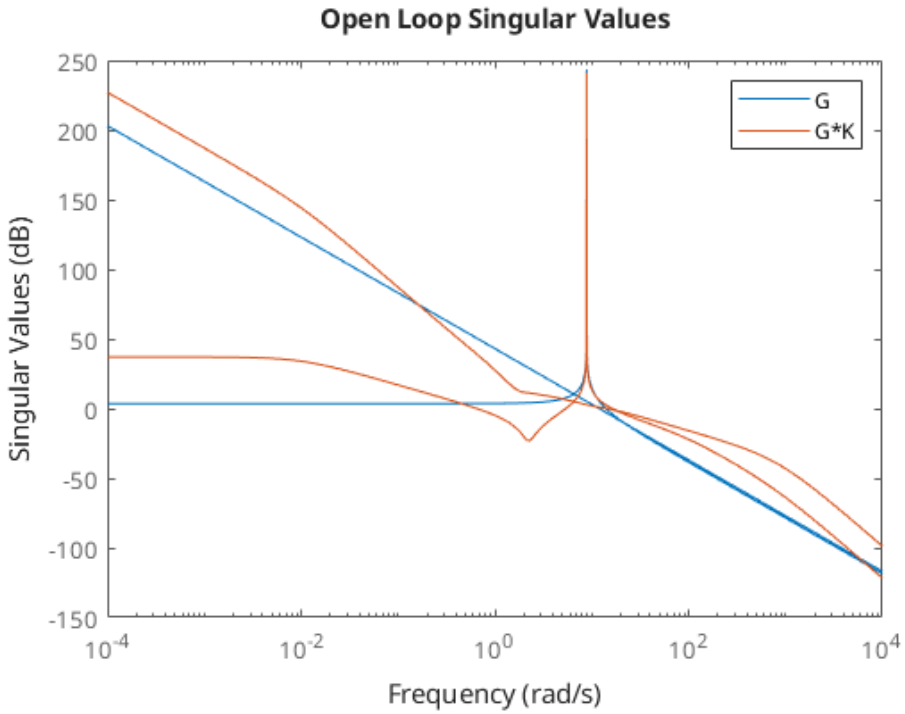

Figure 20: Open Loop D-K iteration singular values.

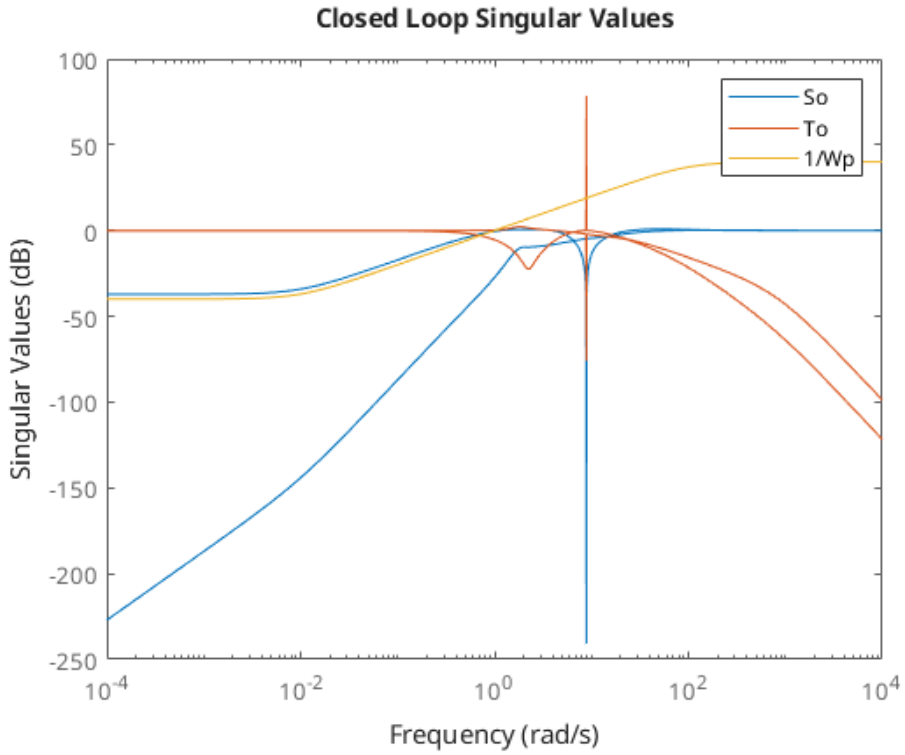

Figure 21: Closed Loop D-K iteration singular values.

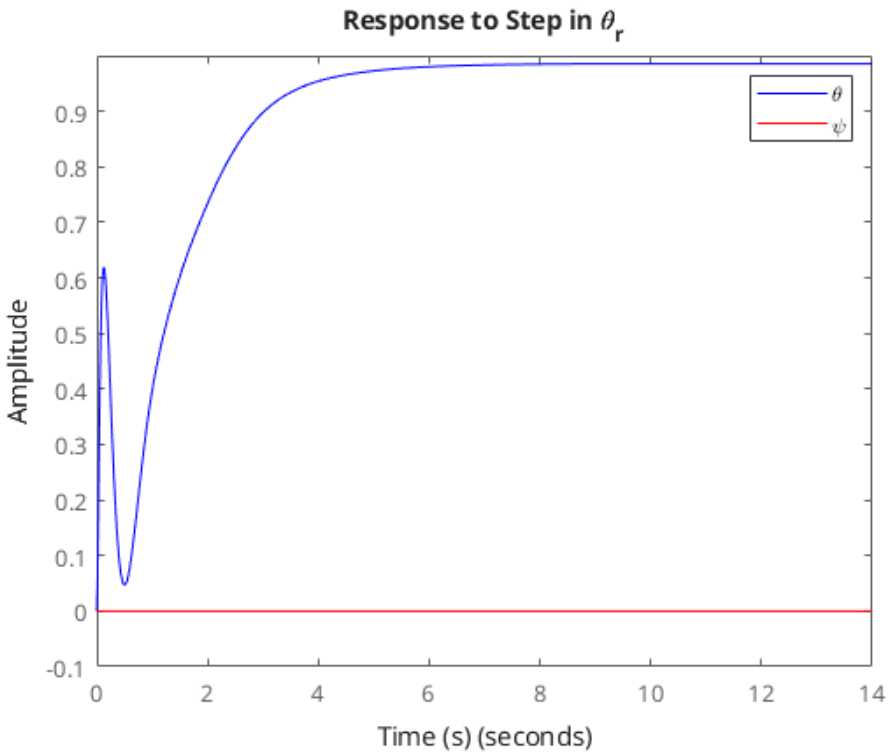

Figure 22: Step response for theta with D-K iteration.

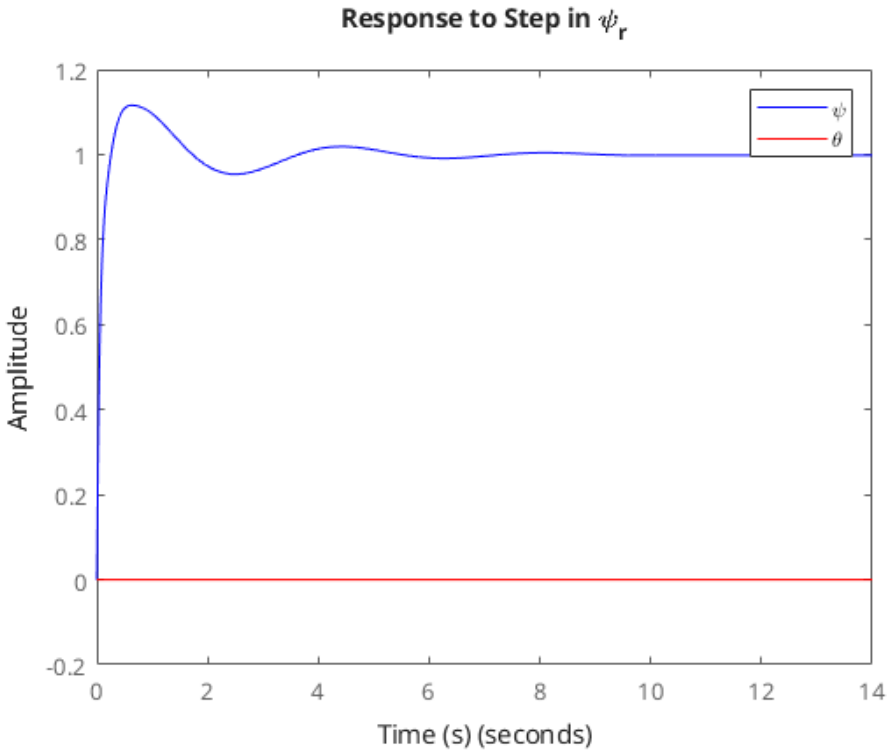

Figure 23: Step response for psi with D-K iteration.

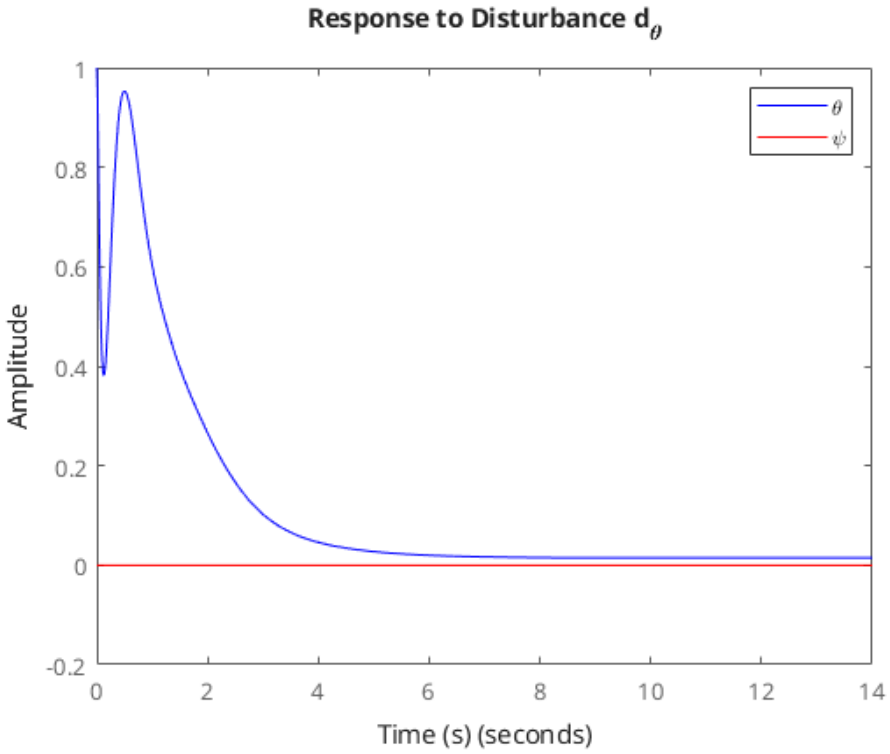

Figure 24: Step response for theta disturbance with D-K iteration.

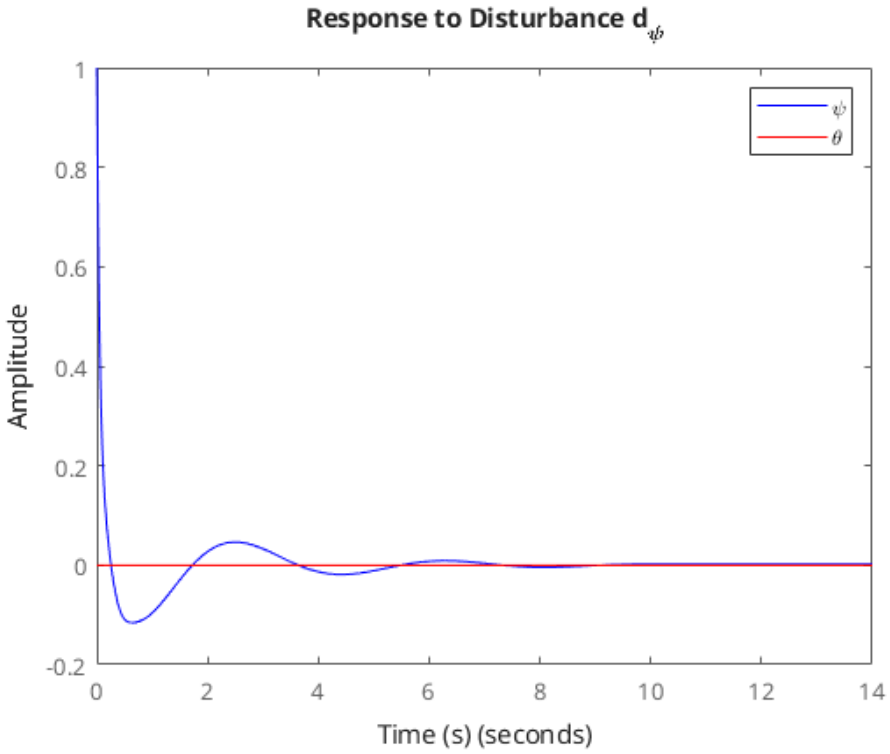

Figure 25: Step response for psi disturbance with D-K iteration.

You can see the performance is about the same with improved stability margins. The settling time is about 2 seconds and there is 10% overshoot. A mild improvement over the first controller in terms of performance. I mentioned it with the weighting function iteration but there appears to be a direct tradeoff between bandwidth and performance.