1: 10 points

Algebraic Ricatti Equations

Write your own function to find the stabilizing solution to an algebraic Riccati equation. Your function should check the conditions for solution existence and provide meaningful error messages if no stabilizing solution exists. The function inputs should be the A, Q, and R matrices of the equation

$$A^T X + XA + XRX + Q = 0.$$

Use your function to find the solution to

$$A = \begin{bmatrix} -3 & 2 & 5 \\ -2 & 1 & 6 \\ 4 & 5 & 6 \end{bmatrix}, R = I_{3\times 3}, Q = \begin{bmatrix} 1 & 4 & 4 \\ 4 & 7 & 7 \\ 4 & 7 & 10 \end{bmatrix}.$$

Show also the result of your function when A is scaled by 0.1.

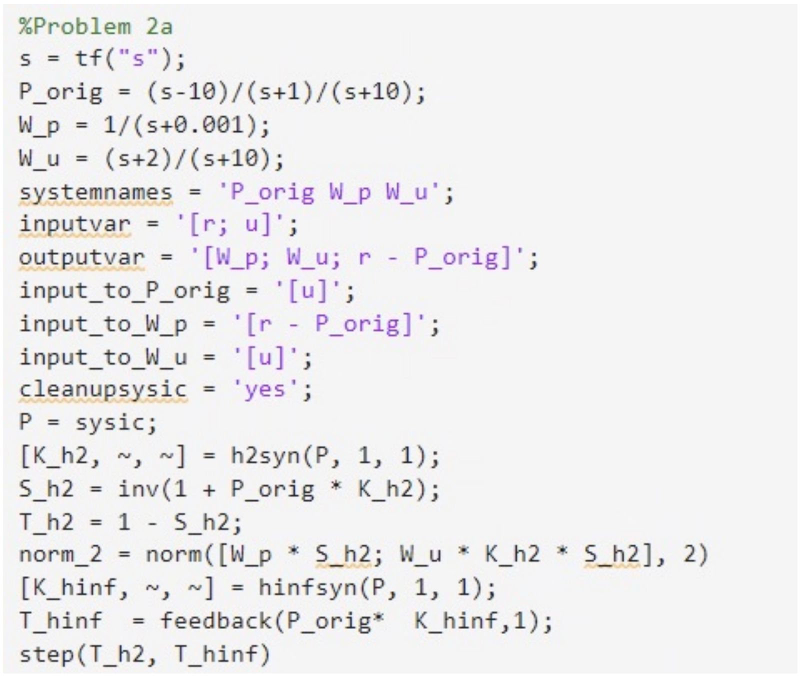
```
%Problem 1
A = [-3 2 5;-2 1 6;4 5 6];
Q = [1 4 4;4 7 7;4 7 10];

R = eye(3);
H = [A R;-Q -A'];
[V, J] = jordan(H)

x_1 = [0.6733 -0.7244 0.0219; -2.3268 -0.8485 -0.0122; 1.1303 0.5645 -0.0652];
x_2 = [0.3318 0.2336 0.2024; -0.7239 0.1501 0.5655; 1 1 1];
X_mine = x_2 * inv(x_1)
X_matlab = are(A, -R, Q)

A_scaled = 0.1 * A;
H_scaled = [A_scaled R;-Q -A_scaled'];
[V_scaled, J_scaled] = jordan(H_scaled)
x_1_scaled = [8.2416+0.0000i -0.0565-0.0957i 0.1352-0.1745i; -1.9374+0.0000i -0.0591-0.1916i -0.0709-0.9477i;
              -1.9243+0.0000i -0.0658-0.223i 0.0061+0.8944i];
x_2_scaled = [-5.4862+0.0000i 0.4174-0.1090i -0.1478-0.4645i; 5.1848+0.0000i 0.8146-0.1068i -1.0527-0.3958i;
              1.0000+0.0000i 1.0000+0.0000i 1.0000+0.0000i];
X_mine_scaled = x_2_scaled * inv(x_1_scaled)
disp('My Function has no Stable Solution')
X_matlab_scaled = are(A_scaled, -R, Q)
```

```
X_mine = 3x3

     -0.5828   -1.7548   -2.9717
     -1.7551   -4.3051   -8.4573
     -2.9719   -8.4565  -14.7533

X_matlab = 3x3

     -0.5831   -1.7561   -2.9740
     -1.7561   -4.3085   -8.4633
     -2.9740   -8.4633  -14.7651


X_mine_scaled = 3x3 complex

    -0.6116 + 0.4002i   0.4238 + 0.7733i ...
     0.4238 + 0.7734i  -0.2224 + 0.9229i
    -0.1952 + 0.9354i  -0.6553 + 2.3829i

My Function has no Stable Solution

Error using are
No solution: (A,B) may be uncontrollable or no solution
exists
```

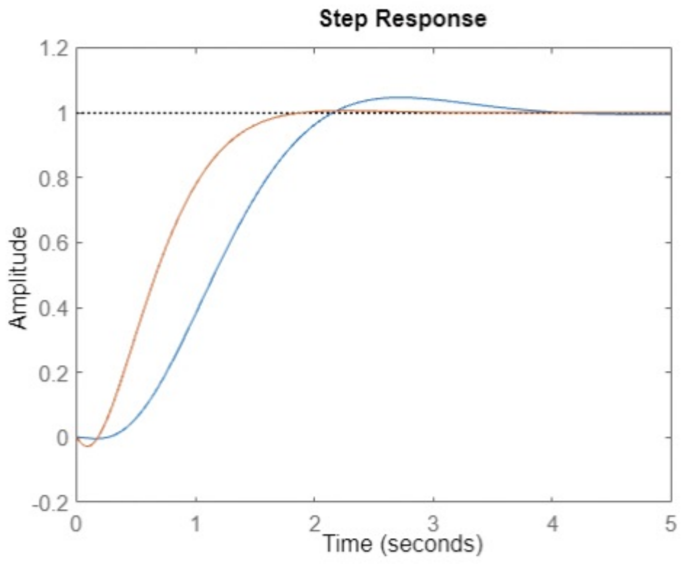
2: 20 points

H_2 Synthesis

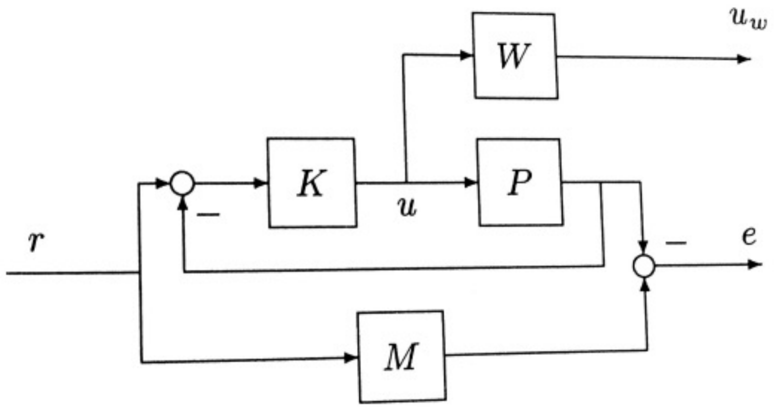
(a) Consider the standard mixed sensitivity control configuration with plant model $P = \frac{s-10}{(s+1)(s+10)}$ and set $W_p = \frac{1}{s+0.001}$, $W_u = \frac{s+2}{s+10}$, and W_T empty. Design a controller that minimizes $\| \begin{bmatrix} W_p S & W_u K S \end{bmatrix}^T \|_2$. Solve the same problem using H_∞ synthesis and plot the step response of T for both designs on the same graph.



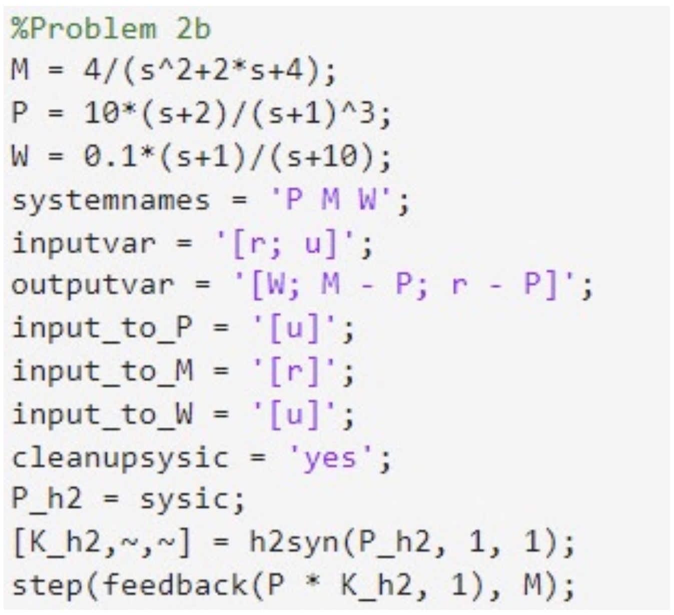
```
%Problem 2a
s = tf("s");
P_orig = (s-10)/(s+1)/(s+10);
W_p = 1/(s+0.001);
W_u = (s+2)/(s+10);
systemnames = 'P_orig W_p W_u';
inputvar = '[r; u]';
outputvar = '[W_p; W_u; r - P_orig]';
input_to_P_orig = '[u]';
input_to_W_p = '[r - P_orig]';
input_to_W_u = '[u]';
cleanupsysic = 'yes';
P = sysic;
[K_h2, ~, ~] = h2syn(P, 1, 1);
S_h2 = inv(1 + P_orig * K_h2);
T_h2 = 1 - S_h2;
norm_2 = norm([W_p * S_h2; W_u * K_h2 * S_h2], 2)
[K_hinf, ~, ~] = hinfsyn(P, 1, 1);
T_hinf  = feedback(P_orig*  K_hinf,1);
step(T_h2, T_hinf)
```



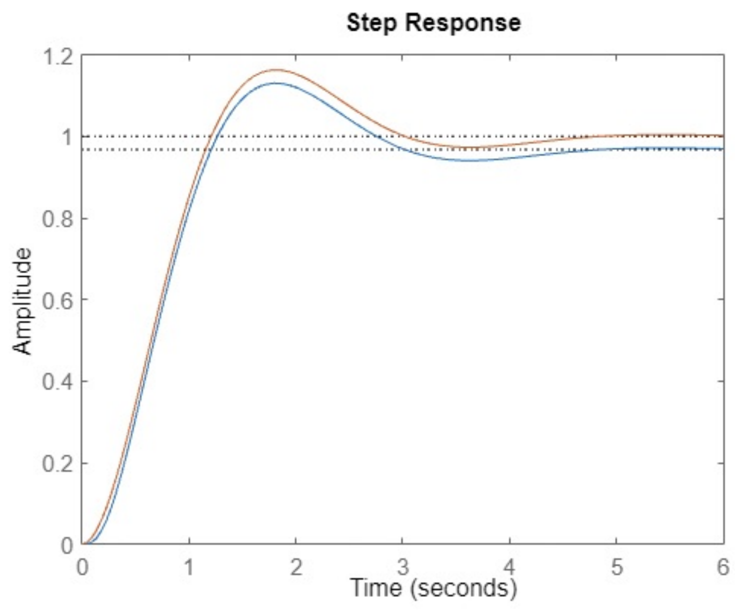
(b) Consider the model matching control problem shown here:



Setup and solve the H_2 optimal control problem the minimizes u_w and e for $M(s) = \frac{4}{s^2+2s+4}$, $P(s) = \frac{10(s+2)}{(s+1)^3}$, and $W(s) = \frac{0.1(s+1)}{s+10}$. Plot the step response of the closed loop system on the same plot as the step response of M to demonstrate the performance of your model matcher.

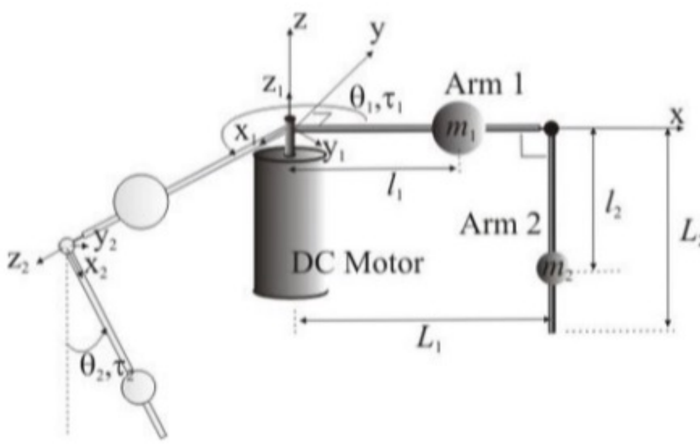


```
%Problem 2b
M = 4/(s^2+2*s+4);
P = 10*(s+2)/(s+1)^3;
W = 0.1*(s+1)/(s+10);
systemnames = 'P M W';
inputvar = '[r; u]';
outputvar = '[W; M - P; r - P]';
input_to_P = '[u]';
input_to_M = '[r]';
input_to_W = '[u]';
cleanupsysic = 'yes';
P_h2 = sysic;
[K_h2,~,~] = h2syn(P_h2, 1, 1);
step(feedback(P * K_h2, 1), M);
```



3: 40 points

Furata Pendulum Control
The Furata pendulum is a classical system that is both non-minimum phase and unstable when the pendulum is balanced upright.



For a specific design the linearized model is given by

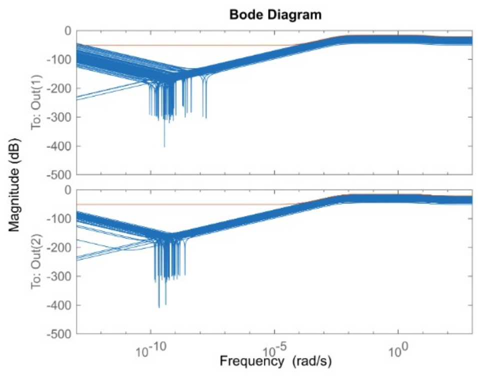
$$\dot{x} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 149.3 & -0.01 & 0 \\ 0 & 261.6 & -0.01 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 49.73 \\ 49.15 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

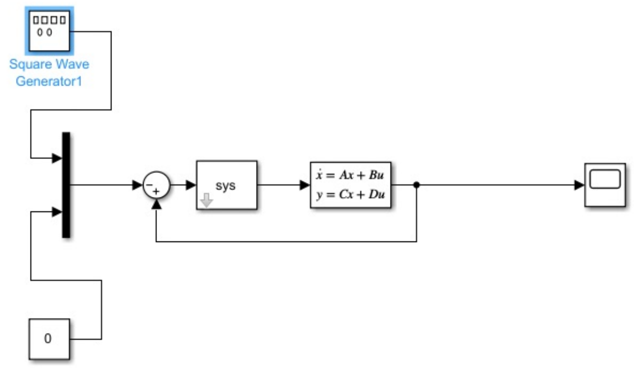
.

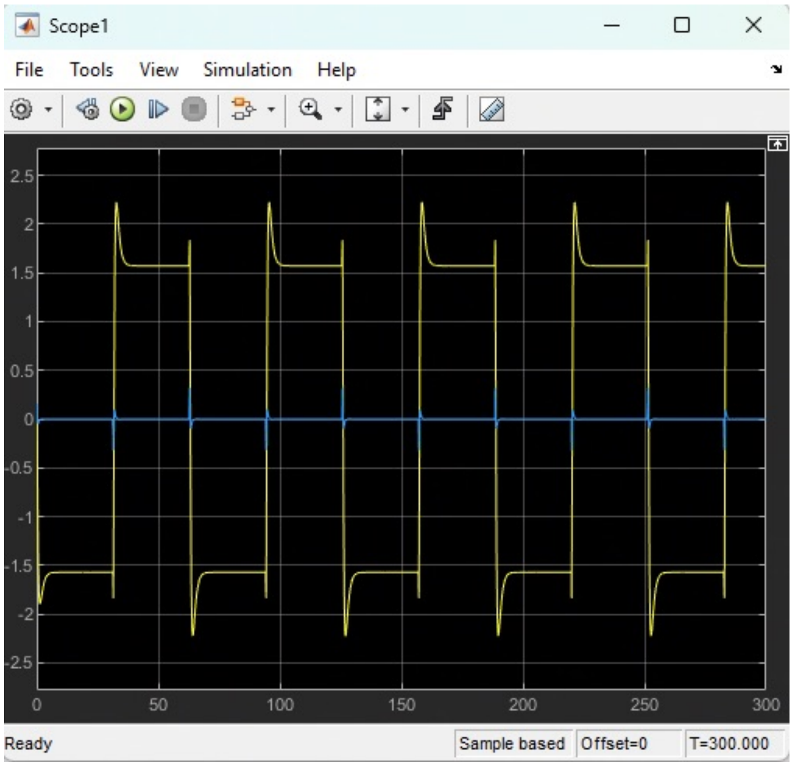
(a) In terms of parameters and uncertainty, the file *plant.m* includes an uncertain linear model that includes 20% parametric uncertainty in the mass of the pendulum. Fit multiplicative output uncertainty weights to the uncertain plant. Plot the Bode plot of your plant with multiplicative uncertainty on top of the plant with parametric uncertainty to demonstrate the quality of your fit.

```
%Problem 3a
%From File
%% Motor
% Resistance
Rm = 8.4;
% Current-torque (N-m/A)
kt = 0.042;
% Back-emf constant (V-s/rad)
km = 0.042;
%
%% Rotary Arm
% Mass (kg)
Mr = 0.095;
% Total length (m)
Lr = 0.085;
% Moment of inertia about pivot (kg-m^2)
Jr = Mr*Lr^2/12;
% Equivalent Viscous Damping Coefficient (N-m-s/rad)
Dr = 0;
%
%% Pendulum Link
% Mass (kg)
Mp = ureal('M',0.024,'Percentage',20);
% Total length (m)
Lp = 0.129;
% Moment of inertia about pivot (kg-m^2)
Jp = Mp*Lp^2/12;
% Equivalent Viscous Damping Coefficient (N-m-s/rad)
Dp = 0;
% Gravity Constant
g = 9.81;

% State Space Representation
Jt = Jr*Jp + Mp*(Lp/2)^2*Jr + Jp*Mp*Lr^2;
A = [0 0 1 0;
     0 0 0 1;
     0 Mp^2*(Lp/2)^2*Lr*g/Jt -Dr*(Jp+Mp*(Lp/2)^2)/Jt -Mp*(Lp/2)*Lr*Dp/Jt;
     0 Mp*g*(Lp/2)*(Jr+Mp*Lr^2)/Jt -Mp*(Lp/2)*Lr*Dr/Jt -Dp*(Jr+Mp*Lr^2)/Jt];

B = [0; 0; (Jp+Mp*(Lp/2)^2)/Jt; Mp*(Lp/2)*Lr/Jt];
C = eye(2,4);  %Measure only the angles
D = zeros(2,1);

% Add actuator dynamics
B = kt * B / Rm;
A(3,3) = A(3,3) - kt*kt/Rm*B(3);
A(4,3) = A(4,3) - kt*kt/Rm*B(4);

% Load into state-space system
Gunc = ss(A,B,C,D); %2 outputs 1 input
G = Gunc.NominalValue; %2 outputs 1 input

%My Code
Gunc_samples = usample(Gunc,100);
[P_11, info_11] = ucover(Gunc_samples(1, 1), G(1, 1), 2);
[P_21, info_21] = ucover(Gunc_samples(2, 1), G(2, 1), 2);
G_11 = (Gunc_samples(1, 1)-  G(1, 1))/G(1, 1);
G_21 = (Gunc_samples(2, 1) - G(2, 1))/G(2, 1);
bodemag([G_11;G_21], [info_11.W1; info_21.W1])
```



(b) For unstable, non-minimum phase systems it is often difficult to know where to start with our weights on S. LQG control can be used to get a starting point from which we can add robustness. Build a Simulink model for the system that tracks a square wave reference of amplitude $\frac{\pi}{2}$ in motor angle while balancing the pendulum. Design an LQG controller that achieves "good" simulation performance. NOTE: Don't spend a ton of time tuning this! If the step response looks OK, just move on. We are not grading the quality of your LQG design.





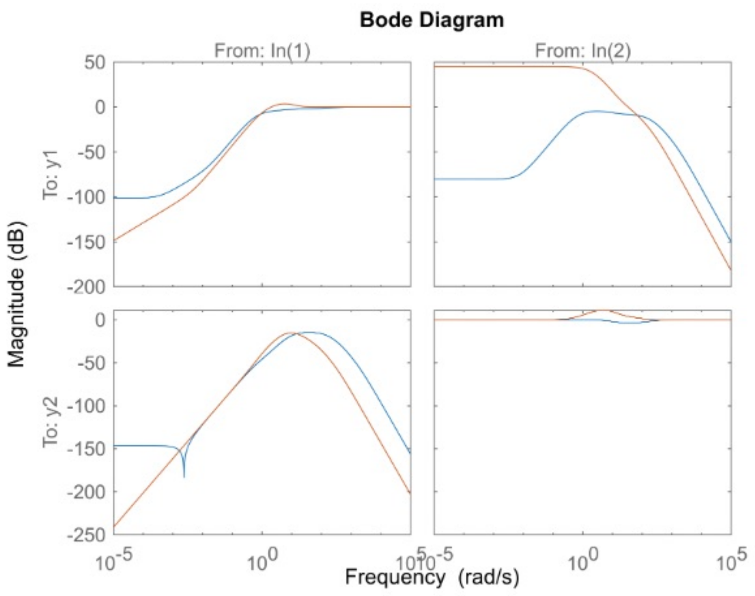
```
%Problem 3b
QXU =  blkdiag(diag([10 10 1 1]),1);
QWV = blkdiag(100000 * eye(4), 1 * eye(2));
%KLQG = lqg(SYS,QXU,QWV) computes an optimal linear-quadratic Gaussian (lqg) regulator KLQ
K_lqg = lqg(G, QXU, QWV);
```

(c) Using your final LQG design as your guide, develop a diagonal performance weight for the system. Design an H_∞ optimal controller for the uncertain system using the uncertainty found in Part a. Iterate on the controller design, retuning a constant control weight to achieve good performance. For the final controller, plot the sensitivity function against the sensitivity function achieved by the LQG controller.

NOTE1: The inverse of the sensitivity function is often unstable. You can use the *fitmagfrd* function to find stable approximations for the sensitivity weights if you choose to use the sensitivity function from Part b directly.

NOTE2: H_∞ synthesis is generally unhappy about imaginary axis poles. You can move them by multiplying the appropriate transfer function by $\frac{s}{s+\epsilon}$ and being sure to take a minimal realization.

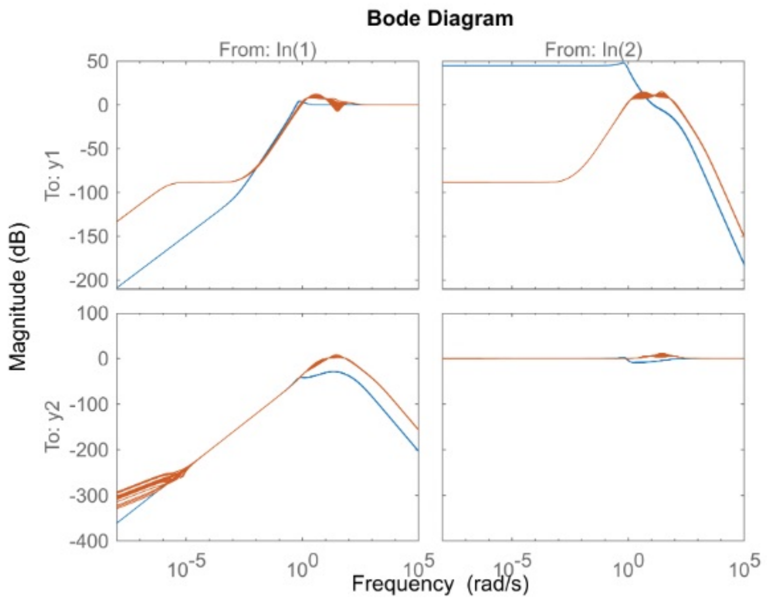
```
%Problem 3c
s = tf("s");
S_lqg = inv(eye(2) - G * K_lqg);
W_p_11 = fitmagfrd(frd(1/S_lqg(1, 1), logspace(-2, 4)), 2);
W_p_22 = fitmagfrd(frd(1/S_lqg(2, 2), logspace(-2, 4)), 2);
W_p = blkdiag(W_p_11, W_p_22);
W_I = [info11.W1 0; 0 info21.W1];
W_u = 0.01;
[K_hinf, ~, ~] = mixsyn(G, W_p, W_u, W_I);
G = G*s/(s + 1e-6);
S_inf = minreal(inv(eye(2) - G * K_hinf));
bodemag(S_inf, S_lqg);
```



(d) Test the robust stability of your LQG and H_∞ designs and plot the uncertain sensitivity functions on top of each other. Has the H_∞ design succeeded in adding robustness?

```
%Problem 3d
S_unc_lqg = inv(eye(2) + Gunc*K_lqg);
[stabmarg, ~] = robstab(S_unc_lqg);
mu_lqg = 1/stabmarg.LowerBound
S_unc_hinf = inv(eye(2) + Gunc*K_hinf);
[stabmarg, ~] = robstab(S_unc_hinf);
mu_inf = 1/stabmarg.LowerBound
bodemag(S_unc_lqg, S_unc_hinf);
```

```
mu_lqg = Inf

mu_inf = 0.4885
```


Bode Diagram
From: In(1)
From: In(2)
Magnitude (dB)
To: y1
To: y2
10^-5
10^0
10^5
Frequency (rad/s)