Massachusetts Institute of Technology

Department of Electrical Engineering and Computer Science

6.245: MULTIVARIABLE CONTROL SYSTEMS

by A. Megretski

Problem Set 4 Solutions [1]

Problem 4.1T

For the following transfer matrices G, find co-prime factorization $G = D^{-1}N$, the "natural" state space for the associated transfer matrix model, and the corresponding "natural" state space model.

(a) discrete time transfer matrix

$$G(z) = \begin{bmatrix} \frac{1}{z-1} & \frac{1}{z-1} \\ \frac{1}{z-1} & \frac{1}{z-1} \end{bmatrix}.$$

Answer: one possible co-prime factorization $G = D^{-1}N$ is given by

$$D(z) = \begin{bmatrix} \frac{z-1}{z} & 0 \\ -1 & 1 \end{bmatrix}, \quad N(z) = \begin{bmatrix} \frac{1}{z} & \frac{1}{z} \\ 0 & 0 \end{bmatrix}.$$

The corresponding "canonical" state space $X \subset H_2^2(\mathbb{T})$ consists of all *constant* functions

$$x(z) \equiv \begin{bmatrix} a \\ 0 \end{bmatrix}, \quad a \in \mathbb{R},$$

with update/output equations

$$x^+ = x + \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} w, \quad y = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} x(\infty),$$

[1] Version of December 9, 2011.

which is equivalent to

$$a(t+1) = a(t) + [1 \;\; 1]w(t), \quad y(t) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} a(t).$$

Reasoning: the identity $DG = N$ can be verified by a direct multiplication. To see that the factorization $G = D^{-1}N$ is co-prime, i.e. that there exists $\epsilon > 0$ such that

$$D(z)'D(z) + N(z)'N(z) \geq \epsilon I \quad \forall\, z \in \mathbb{D}^+ = \{z \in \mathbb{C} : \; |z| > 1\},$$

note that $D(z)$ is continuous on $\mathbb{D}^+ \cup \{\infty\}$, and not singular for all $z \in \mathbb{D}^+ \cup \{\infty\}$ except $z = 1$, at which point

$$D(z)'D(z) + N(z)'N(z) = D(1)'D(1) + N(1)'N(1) = 2I > 0.$$

Note that there is no co-prime factorization with $D(z) = (1 - 1/z)I$ (the "easy" choice) because this would yield a *double* pole of $D^{-1}N$ at $z = 1$.

The "canonical" state space X for G is the subset of all functions $V \in H_2^2(\mathbb{T})$ which can be approximated arbitrarily well by functions $V_0 \in H_2^2(\mathbb{T})$ of the form $V_0 = NW_0 - DY_0$, where W_0, Y_0 are Fourier transforms of square summable sequences $w_0, y_0 : \; \mathbb{Z} \to \mathbb{R}^2$ such that $w_0(t) = 0$ and $y_0(t) = 0$ for $t \geq 0$.

Using the fact that $zN(z)$ and $zD(z)$ are polynomials in z while comparing the coefficients at the z-expansions in the equality $V_0 = NW_0 - DY_0$, one can see that V_0 has to be a constant function of the form

$$V_0(z) \equiv \begin{bmatrix} a \\ 0 \end{bmatrix} = \lim_{z \to 0} z\{N(z)W_0(z) - D(z)Y_0(z)\}.$$

To see that a can be an arbitrary real number, use

$$W_0(z) = \begin{bmatrix} az \\ 0 \end{bmatrix}, \quad Y_0(z) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The update/output laws for the canonical state $x(t) \in H_2^2(\mathbb{T})$ defined by

$$x(t)(z) = z^t[N(z)\tilde{W}_t(z) - D(z)\tilde{Y}_t(z) + x(0)(z)],$$

where

$$W_t(z) = \sum_{\tau=0}^{t-1} w(\tau)z^{-\tau}, \quad Y_t(z) = \sum_{\tau=0}^{t-1} y(\tau)z^{-\tau},$$

are given by

$$x^+(t)(z) = z[x(t)(z) - D(z)y(t) + N(z)w(t)],$$

where

$$y(t) = D(\infty)^{-1}[x(t)(\infty) + N(\infty)w(t)].$$

Substituting

$$x(t)(z) \equiv \begin{bmatrix} a(t) \\ 0 \end{bmatrix}, \quad D(\infty) = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \quad N(\infty) = 0$$

yields the answer.

(b) CONTINUOUS TIME TRANSFER MATRIX

$$G(s) = \begin{bmatrix} 1/s & -1/s \\ -1/s & 1/s \end{bmatrix}.$$

Answer: one possible co-prime factorization $G = D^{-1}N$ is given by

$$D(s) = \begin{bmatrix} \frac{s}{s+1} & 0 \\ 1 & 1 \end{bmatrix}, \quad N(s) = \begin{bmatrix} \frac{1}{s+1} & -\frac{1}{s+1} \\ 0 & 0 \end{bmatrix}.$$

The corresponding "canonical" state space $X \subset H_2^2(j\mathbb{R})$ consists of all functions

$$x(s) \equiv \begin{bmatrix} \frac{a}{s+1} \\ 0 \end{bmatrix}, \quad a \in \mathbb{R},$$

with update/output equations

$$\frac{dx(t)(s)}{dt} = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix} w(t), \quad y(t) = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} 2x(t)(1),$$

which is equivalent to

$$\dot{a}(t) = [1 \quad -1]w(t), \quad y(t) = \begin{bmatrix} 1 \\ -1 \end{bmatrix} a(t).$$

Reasoning: the identity $DG = N$ can be verified by a direct multiplication. To see that the factorization $G = D^{-1}N$ is co-prime, i.e. that there exists $\epsilon > 0$ such that

$$D(s)'D(s) + N(s)'N(s) \geq \epsilon I \quad \forall\, s \in \mathbb{C}^+ = \{s \in \mathbb{C} : \ \mathrm{Re}\{s\} > 0\},$$

note that $D(s)$ is continuous on $\mathbb{C}^+ \cup \{\infty\}$, and not singular for all $s \in \mathbb{C}^+ \cup \{\infty\}$ except $s = 0$, at which point

$$D(s)'D(s) + N(s)'N(s) = D(0)'D(0) + N(0)'N(0) = 2I > 0.$$

Note that there is no co-prime factorization with $D(s) = \frac{s}{s+1}I$ (the "easy" choice) because this would yield a *double* pole of $D^{-1}N$ at $s = 0$.

The "canonical" state space X for G is the subset of all functions $V \in H_2^2(j\mathbb{R})$ which can be approximated arbitrarily well by functions $V_0 \in H_2^2(j\mathbb{R})$ of the form $V_0 = NW_0 - DY_0$, where W_0, Y_0 are Fourier transforms of square integrable functions $w_0, y_0 : \mathbb{R} \to \mathbb{R}^2$ such that $w_0(t) = 0$ and $y_0(t) = 0$ for $t \geq 0$.

Using the fact that $(s+1)N(s)$ and $(s+1)D(s)$ are polynomials in s while comparing the inverse Fourier transforms of both sides of $V_0 = NW_0 - DY_0$, one can see that V_0 has to be a rational function of the form

$$V_0(s) = \begin{bmatrix} \frac{a}{s+1} \\ 0 \end{bmatrix}.$$

To see that a can be an arbitrary real number, use

$$W_0(s) = \begin{bmatrix} \frac{2a}{1-s} \\ 0 \end{bmatrix}, \quad Y_0(s) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The canonical state $x(t) \in H_2^2(j\mathbb{R})$ is defined by

$$x(t)(s) = e^{ts}[N(s)W_t(s) - D(s)Y_t(s) + x(0)(s)],$$

where

$$W_t(s) = \int_0^t e^{-\tau s} w(\tau) d\tau, \quad Y_t(s) = \int_0^t e^{-\tau s} y(\tau) d\tau.$$

Substitution

$$x(t)(s) = \begin{bmatrix} \frac{a(t)}{s+1} \\ 0 \end{bmatrix},$$

followed by differentiation with respect to t yields

$$\begin{bmatrix} \dot{a}(t)/(s+1) \\ 0 \end{bmatrix} = s \begin{bmatrix} a(t)/(s+1) \\ 0 \end{bmatrix} + N(s)w(t) - D(s)y(t),$$

which is equivalent to the state space equations given above.

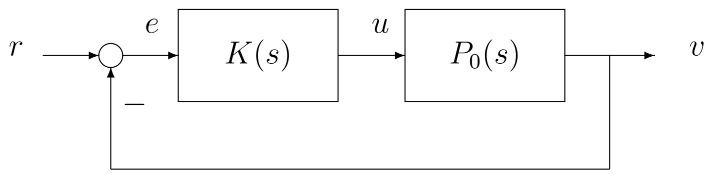

Figure 1: Design objectives of Problem 2.1

Problem 4.2P

Consider the feedback design setup from Figure 1. Let us define *closed loop bandwidth* of the feedback system as the largest $\omega_0 > 0$ such that $|S(j\omega)| \leq 0.1$ for all $\omega \in [0, \omega_0]$, where

$$S = \frac{1}{1 + P_0 K}$$

is the closed loop sensitivity function (in this definition, the threshold 0.1 is a bit arbitrary).

It is frequently claimed that location of unstable zeros of P_0 limits the maximal achievable closed loop bandwidth. While mathematically this is not exactly true, the only way to achieve a substantially larger bandwidth is by making $|S(j\omega)|$ extremely large at other frequencies.

You are asked to verify this for

$$P_0(s) = \frac{s - a}{s(s + 2a)},$$

where $a > 0$ is a real parameter (determining location of the open loop zero), using `hinfsyn.m` to estimate the maximal bandwidth achievable by a stabilizing LTI controller $K = K(s)$ of order not larger than 5, satisfying the closed loop sensitivity magnitude bound $|S(j\omega)| < 20$ at all frequencies ω, as a function of $a \neq 0$. Generate a plot of your estimate, as a function of $a > 0$.

Hint: write an algorithm which attempts to achieve a *given* closed loop bandwidth, using pre-designed low-pass filter, like the Butterworth filter, to incorporate the bandwidth constraint into H-Infinity optimization. Once this is accomplished, use binary search to find (approximately) the maximal bandwidth.

Conclusion: maximal achievable closed loop bandwidth grows *linearly* with a, at a rate of at least $0.43a$. The rate can be made slightly larger at the expense of using much higher controller gains. See the plot on Figure 2.

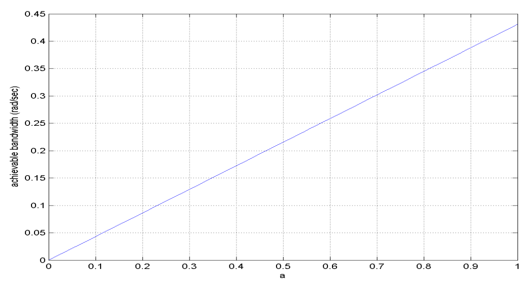

Figure 2: Bandwidth vs. "a" for Problem 4.2

Approach: since

$$aP_0(as) = P_1(s) = \frac{s-1}{s(s+2)}$$

for every $a > 0$, controller $K_0(s)$ stabilizes P_0 and achieves bandwidth ab if and only if controller $K_1(s) = K_0(as)/a$ stabilizes P_1 and achieves bandwidth b. Hence it is sufficient to find the best achievable bandwidth for $a = 1$.

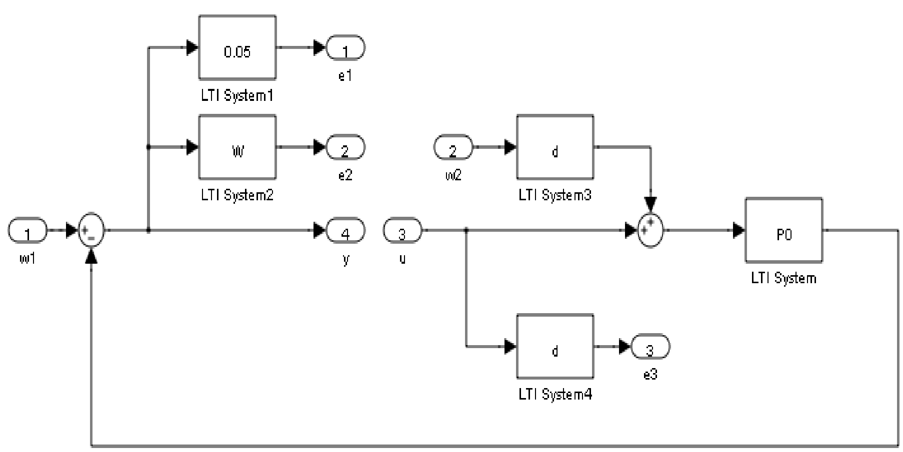

Figure 3: Open loop model for Problem 4.2

We use SIMULINK models `ps42des.mdl` (see Figure 3) and `ps42test.mdl` (see Figure 4) to define, respectively, the open and closed loop systems. It introduces two distur-

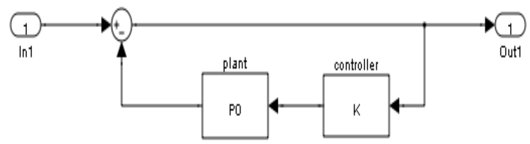

Figure 4: Closed loop model for Problem 4.2

bance inputs w_1, w_2 and three cost outputs e_1, e_2, e_3:

- w_1 is the "main" input (reference/sensor noise);
- w_2 is artificial small disturbance, to prevent sensor singularity due to the imaginary axis pole of P_0;
- e_1 enforces the closed loop sensitivity bound $|S| \leq 20$, frovided the overall closed loop L2 gain in not larger than 1;
- e_2 is defined using a low-pass filter W such that $|W(j\omega)| \geq 10$ for $|\omega| \leq b$, where b is the desired closed loop bandwidth, and therefore enforces the bandwidth constraint frovided the overall closed loop L2 gain in not larger than 1;
- e_3 is artificial small cost to prevent control singularity due to the fact that P_0 is strictly proper.

MATLAB function `ps42.m` controls design optimization and testing. The essential part of its code is shown below:

```
function [K,S]=ps42(a,b,d)
s=tf('s');                         % useful constant
P0=(s-a)/(s*(s+2*a));              % the plant
[A,B,C,D]=butter(3,b,'s');         % Butterworth filter
W=ss(A,B,C,D);
W=(10/abs(squeeze(freqresp(W,1i*b))))*W;
assignin('base','d',d)             % export variables
assignin('base','P0',P0)
assignin('base','W',W)
p=linmod('ps42des');               % extract open loop
```

```
p=ss(p.a,p.b,p.c,p.d);
[K,G]=hinfsyn(p,1,1);          % optimize controller
fprintf('success flag for w0=%f:  %f<1\n',b,norm(G,Inf))
assignin('base','K',K)         % export controller
S=linmod('ps42test');          % extract closed loop
fprintf('max(real(eig(A)))=%f\n',max(real(eig(S.a))))
S=ss(S.a,S.b,S.c,S.d);
fprintf('norm(S,Inf)=%f\n',norm(S,Inf))
ww=logspace(-2,5,10000);
Sw=abs(squeeze(freqresp(S,1i*ww)));
k=find(Sw>0.1,1);              % find actual bandwidth
w1=ww(k-1);
fprintf('actual bandwidth:  %f (vs. %f)\n',w1,b)
```

At $a = 1$, $b = 0.43$, $d = 0.001$ the code generates a stabilizing controller K of order 5 which yields closed loop bandwidth of slightly more than 0.43. Logarithmic plots of the closed loop sensitivity function and controller are provided on Figure 5.

Problem 4.3P

Continuous time scalar signal $q = q(t)$ models the (one dimensional) position of an oscillator driven by random forces in the absence of friction, according to

$$\ddot{q}(t) + \omega_0^2 q(t) = f_1(t),$$

where f_1 is a noise signal, and $\omega_0 > 0$ is a parameter. The result $g(t)$ of measuring $q(t)$ in real time is modeled according to $g(t) = q(t) + f_2(t)$, where f_2 is another noise signal. Assuming that $f = [f_1; f_2]$ is a normalized vector-valued white noise, use `h2syn.m` to find an LTI system (a "filter") which takes $g = g(t)$ as an input and outputs an estimate $\hat{q} = \hat{q}(t)$ of $q = q(t)$, minimizing the steady state value

$$J = \lim_{t \to \infty} \mathbf{E}[|e(t)|^2]$$

of the variance of the estimation error $e = q - \hat{q}$. Plot the minimal J as the function of $\omega_0 > 0$.
Hint: the system, as described, is not stabilizable (there is no provision for a feedback loop from g back to f_1). Therefore, before applying `h2syn.m`, one has to "massage" the setup into a stabilizable format. One

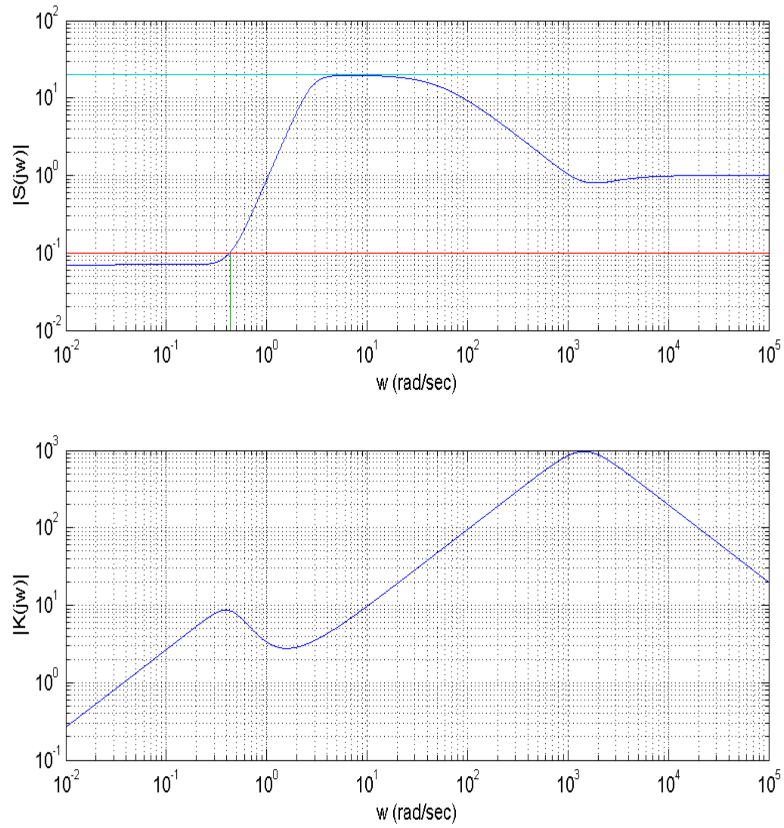

Figure 5: Closed loop Bode plots for Problem 4.2

WAY TO DO THIS IS BY STARTING WITH A PARTICULAR FILTER F_0 PRODUCING AN ESTIMATE $\hat{q}_0$ WHICH ACHIEVES $J < \infty$, AND THEN USING `h2syn.m` TO DESIGN AN LTI FILTER WHICH TAKES $g - \hat{q}_0$ AS MEASUREMENT, AND OUTPUTS THE OPTIMAL ESTIMATE $\hat{\delta}$ OF $\delta = q - \hat{q}_0$, SO THAT $\hat{q} = \hat{q}_0 + \hat{\delta}$ IS THE OPTIMAL ESTIMATE OF q.

Conclusion: the minimal mean square estimation error decreases with as ω_0 increases, as shown on Figure 6.

The design uses H2 optimization according to the setup shown on Figure 7, where

$$W(s) = \frac{1}{s^2 + \omega_0^2}, \quad L(s) = \frac{s^2 + \omega_0^2}{s^2 + \omega_0^2 + L_1 s + L_2}, \quad F0(s) = \frac{L_1 s + L_2}{s^2 + \omega_0^2 + L_1 s + L_2},$$

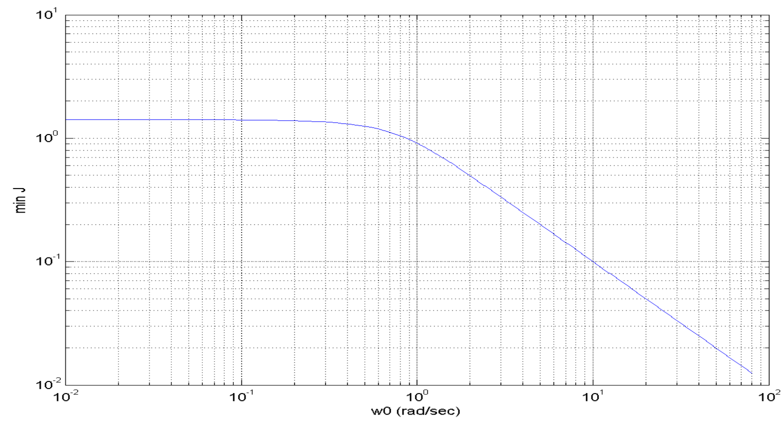

Figure 6: Minimal mean square estimation error in Problem 4.3

where $L_1 > 0$ and $L_2 + \omega_0 > 0$. Here W represents the original oscillator, F_0 is the classical output estimator for W, and L specifies the mandatory zeros of F_0.

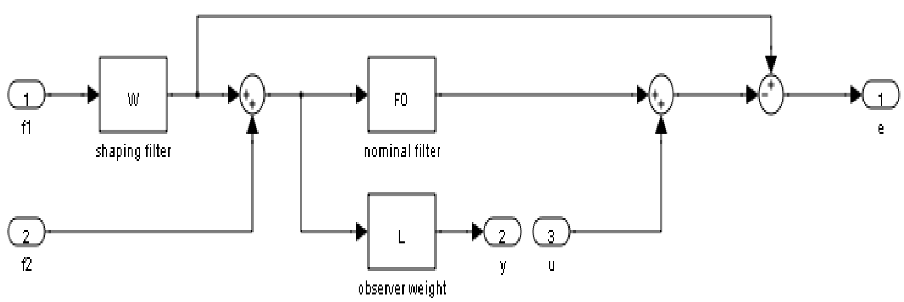

Figure 7: Design setup for Problem 4.3

Indeed, for a state space model for W, e.g.

$$
\begin{aligned}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= -\omega_0^2 x_1 + f_1, \\
y &= x_1 + f_2,
\end{aligned}
$$

the classical observer for $q = x_1$ is

$$\begin{aligned}\dot{\hat{x}}_1 &= \hat{x}_2 + L_1(y - \hat{x}_1),\\ \dot{\hat{x}}_2 &= -\omega_0^2 x_1 + L_2(y - \hat{x}_1),\\ \hat{q} &= \hat{x}_1,\end{aligned}$$

which has transfer function F_0 (from y to $\hat{q}$). Also, an estimator F achieves a finite mean square error if and only if both F and WF belong to class H_2, which is equivalent to F having representation of the form $F = F0 + \Delta L$ for some $\Delta \in H_2$, which justifies the use of the setup from Figure 7. This SIMULINK diagram, as well as the testing model

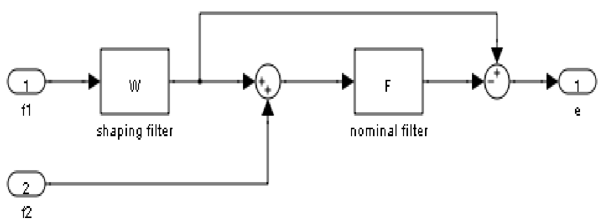

Figure 8: Test setup for Problem 4.3

shown on Figure 8, are controlled by MATLAB code `ps43.m`, the essential parts of which are shown below:

```
function [J,F]=ps43(w0,L)
s=tf('s');                         % useful constant
F0=(s*L(1)+L(2))/(s^2+w0^2+s*L(1)+L(2));
W=1/(s^2+w0^2);
L=(s^2+w0^2)/(s+w0+1)^2;
assignin('base','F0',F0)           % export variables
assignin('base','W',W)
assignin('base','L',L)
p=linmod('ps43des');               % extract open loop
p=minreal(ss(p.a,p.b,p.c,p.d)); % remove uncontrollable modes
K=h2syn(p,1,1);                    % optimize
F=minreal(F0+K*L);                 % complete filter
assignin('base','F',F)             % export variables
p=linmod('ps43test');              % extract error system
G=minreal(ss(p.a,p.b,p.c,p.d)); % remove uncontrollable modes
J=norm(G)^2;
```