International Graduate School on Control: Lesson 4

1. **Hard Disk Drives: Uncertainty Modeling**

A typical hard disk drive (HDD) works by spinning a magnetic disk while a magnetic head reads/writes data from/to circular tracks on the disk. To keep up with the increasing performance requirements, most modern HDDs utilize a dual-stage actuator composed of a voice coil motor (VCM) and a micro actuator (MA) to control the position of the magnetic head (Figure 1). The VCM is the primary actuator that provides a full range of motion for the magnetic head across all disk tracks. The MA provides a greater tracking accuracy than the voice coil motor but has a limited range of motion.

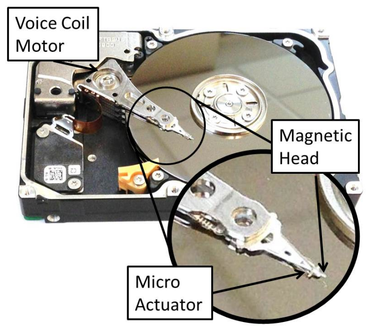


Figure 1: Typical Internal structure of a HDD with a dual stage actuator.

In this problem you will investigate the uncertainty associated with the VCM dynamics. The file `HDDdata.mat` posted with this homework contains the following data for the VCM:

- `wexp`: An N_w-by-1 vector of frequencies (rad/sec) for response data
- `RespData`: An N_w-by-N_{exp} matrix of experimental frequency response data. The i^{th} column `RespData(:,i)` provides the experimental frequency response data (at the frequencies in `wexp`) for the i^{th} system.
- `[A0,B0,C0,D0]`: State space data for a nominal design fit.

(a) Let G_0 denote the nominal design model and $\{G_i\}_{i=1}^{50}$ denote fifty experimental frequency responses. Create a single Bode plot (magnitude and phase) with both the experimental responses and the nominal model. Select colors and or line widths so that the nominal model can be distinquished from the experimental responses. Note that the nominal model is accurate at low frequencies but becomes less accurate at high frequencies. What frequency (roughly) does the model accuracy begin to degrade?

(b) Define the relative error between the nominal model and each frequency response:

$$E_i(\omega) := \frac{|G_0(j\omega) - G_i(j\omega)|}{|G_0(j\omega)|}$$

$E_i(\omega)$ can be interpreted as a frequency dependent relative error. For example, if $E_i(\omega) = 0.05$ then the model error is 5% of the nominal gain. Compute the relative error E_i for each response. Plot all relative errors $\{E_i\}_{i=1}^{50}$ on a single plot. Up to what frequency (roughly) does the relative error remain below 0.1 (=10%)?

(c) We will approximate the model mismatch using linear time-invariant (LTI) uncertainty. In particular, we'll define the following set of models:

$$\mathcal{M} := \{G_0(1 + \Delta W_u) \ : \ \Delta \text{ is LTI } \ \& \ \|\Delta\|_\infty \leq 1\} \text{ where } W_u(s) := \left(\frac{s+1}{0.5s+3}\right)^4.$$

The transfer function W_u was selected to roughly approximate the relative error as a function of frequency. For example, if $|W_u(j\omega)| = 0.05$ then the the models in $\mathcal{M}$ can vary by 5% from the nominal model G_0. Create a Bode magnitude plot for W_u and confirm that it roughly approximates the relative errors E_i. [Note: W_u only approximately captures the behavior of the relative errors E_i. This was selected by hand but a more precise transfer function fit W_u can be computed, e.g. using tools like `ucover` in Matlab.]

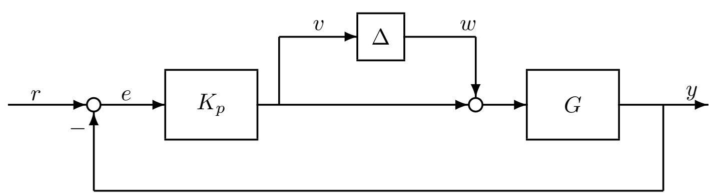
2. **Nonlinear Dynamic Uncertainty**

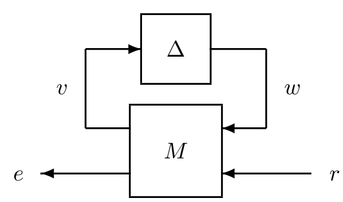
(a) Consider the classical feedback system shown below with plant dynamics $G(s) = \frac{4}{s^2+8s+7}$ and proportional control $K_p = 20$. Assume Δ is a nonlinear, dynamic system with a norm bound $\|\Delta\| \le 0.2$. The nominal feedback system is given by $\Delta = 0$. Construct the nominal sensitivity function S_{nom} from r to e and compute the gain $\|S_{nom}\|_\infty$.

(b) Construct an LTI system M so that the uncertain feedback system is given in LFT form by $F_u(M, \Delta)$. You may construct M by hand or by using functions in Matlab (e.g. `udyn` and `lftdata`). Verify your construction by comparing the Bode plots of $F_u(M, 0)$ with the S_{nom} constructed in part (a).

(c) Use the condition derived in class to compute an upper bound on the largest possible gain from r to e over all possible nonlinear, norm-bounded, dynamic uncertainties.

(d) Finally, let β denote the bound on the uncertainty, i.e. $\|\Delta\| \le \beta$. In the previous part you computed the gain from r to e for $\beta = 0.2$. In this part, compute the bound on the gain from r to e for several values of β. Draw a plot of this gain vs. the uncertainty level β.

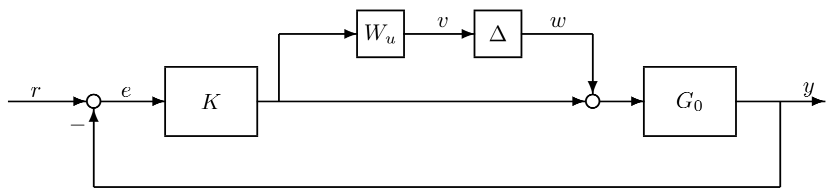




3. **Hard Disk Drives: Robustness Analysis**

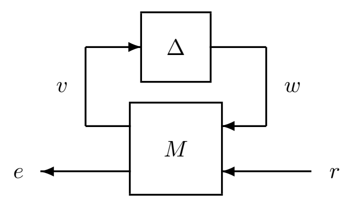
Consider the classical feedback system shown below with the nominal HDD model G_0 and uncertainty weight W_u from the previous problem. In addition, consider the following controller:

$$K(s) = \frac{0.03101s + 0.004135}{s + 1.2}$$



(a) The nominal feedback system is given by $\Delta = 0$. Construct the nominal sensitivity function S_{nom} from r to e and compute the gain (H_∞ norm) of S_{nom}.

(b) Construct an LTI system N so that the uncertain feedback system is given in LFT form by $F_u(M, \Delta)$. You may construct M by hand or by using functions in Matlab (e.g. `udyn` and `lftdata`). Verify your construction by comparing the Bode plots of $F_u(M, 0)$ with the S_{nom} constructed in part (a).



(c) Use the condition derived in class to compute an upper bound on the largest possible gain from r to e over all possible <u>**nonlinear**</u>, norm-bounded, dynamic uncertainties ($\|\Delta\| \leq 1$). [Note: We constructed the uncertain model in problem 1 assuming Δ is LTI. Hence the gain computed in this part will be an upper bound on the true worst-case gain.]

(d) Use the condition derived in class to compute an upper bound on the largest possible gain from r to e over all possible <u>**LTI**</u>, norm-bounded, dynamic uncertainties ($\|\Delta\|_\infty \leq 1$). The example file `LTIUncertainty.m` from class can be used to solve this problem. You should notice that this condition yields a significantly smaller upper bound on the induced gain.

Note: This is written in Matlab and uses CVX. It requires some functionality to easily connect systems and I'm not sure if are similar functions in Python. The posted code also generates results using the function `wcgain`. This solves for the largest gain for systems with LTI uncertainty. It uses a specialized numerical implementation to solve this problem but the theory is similar to that used to develop our SDP upper bound condition.