MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Mechanical Engineering

2.04A Systems and Controls
Spring 2013

Problem Set #4

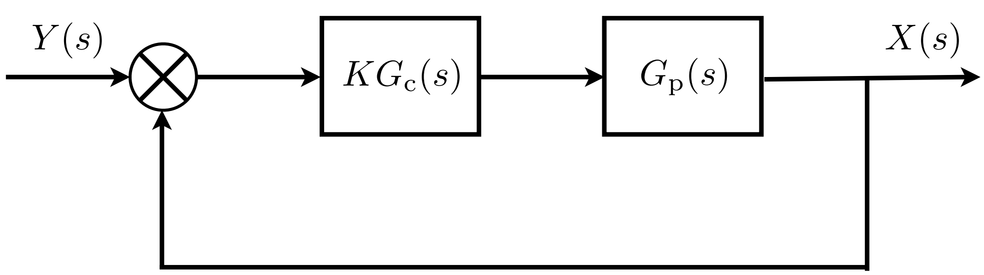
Posted: Thursday, Mar. 7, '13 **Due: Thursday, Mar. 14, '13**

---

1. *Sketch* the Root Locus for the open–loop pole–zero configurations shown in page 18 of Lecture 11. For each case, briefly justify your choices using the Root Locus properties / sketching rules. (You may use Matlab to verify your sketches, but make sure you understand the relationship between the Root Locus appearance and its properties.)

2. Consider the Root Loci shown in page 19 of Lecture 11. For each one, briefly state if they are valid, *i.e.* if they meet the Root Locus properties / sketching rules. For those that are invalid, sketch the correct Root Locus. (If you use Matlab to verify your answers, please make sure that you match Matlab's numerical answers to the Root Locus properties / sketching rules.)

3. We are given a feedback system described by the block diagram shown below.



The plant transfer function is

$$G_{\mathrm{p}}(s) = \frac{8}{(s+2)(s+4)}.$$

In this problem, we will investigate the effect of gain K and different controllers $G_{\mathrm{c}}(s)$ on system performance. You may use Matlab to obtain numerical values for the response characteristics (such as settling time, etc.) but you should make sure you verify the results with the analytical formulae from the class notes and textbook.

3.a) Before considering feedback control, calculate the values of damping ratio ζ and natural frequency ω_n for the plant transfer function $G_p(s)$. Which is the dominant pole and what is the slowest time constant that we can expect for he open–loop plant? Verify using Matlab.

3.b) Going on to feedback control for this plant, first we consider a proportional (P) controller

$$G_c = 1.$$

Using Matlab, sketch the Root Locus for the P controller and verify its appearance with the Root Locus properties / sketching rules. What is the smallest value of gain K that results in the fastest possible settling time for this controller? What is the steady–state error?

3.c) To speed up the response further, we now consider a proportional–derivative (PD) controller,

$$G_c = s + z,$$

with the zero located at $s = -6$ (it i.e., $z = 6$.) Using Matlab, sketch the Root Locus for the PD controller and verify its appearance with the Root Locus properties / sketching rules. For gain $K = 0.25$, what are the settling time and overshoot? What is the steady–state error?

3.d) To eliminate steady–state error, we first consider a pure integral (I) controller,

$$G_c = \frac{1}{s}.$$

Show that indeed the steady–state error is zero in this configuration. Using Matlab, sketch the Root Locus for the I controller and verify its appearance with the Root Locus properties / sketching rules. Point out two obvious disadvantages of the I controller in terms of the response speed and stability of the feedback loop.

3.e) To fix the problems with the pure integral controller, let us now consider a proportional–integral (PI) controller,

$$G_c = \frac{s + z_i}{s},$$

where now the zero is located very near the origin, at $s = -0.1$ (it i.e., $z = 0.1$.) Using Matlab, sketch the Root Locus for the PI controller and verify its appearance with the Root Locus properties / sketching rules. Note the similarities and differences between this Root Locus and the Root Locus of the P controller from question (b). Verify analytically that in the PI controller the steady–state error is still eliminated. Also verify using Matlab that the fastest possible setting time yielded by the PI controller is approximately the same as the fastest possible settling time of the P controller.

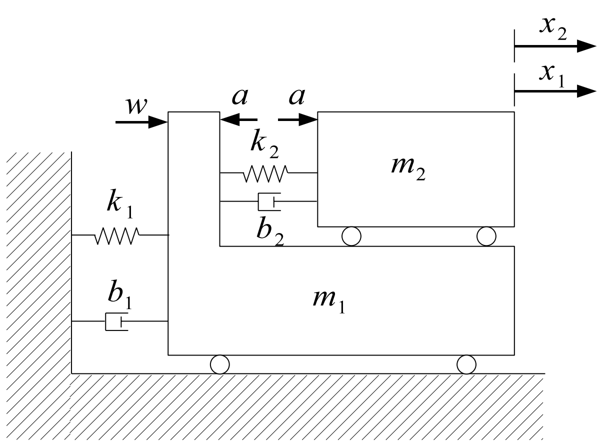
(You will have to adjust the gain K to a value of $3 \sim 4$ or higher. The PI controller is slightly slower, actually.)

3.f) To enjoy the benefits of both PD and PI control (faster settling time and no steady–state error, respectively), we finally consider the PID controller

$$G_c = (s + z)\ \frac{s + z_i}{s},$$

where the two zeros z and z_i are in the same locations as before. Using Matlab, sketch the Root Locus for the PID controller and verify its appearance with the Root Locus properties / sketching rules. Still using Matlab, adjust the gain such that the *rise* time equals approximately 0.1 sec and compute the overshoot and steady–state error for this control configuration. What do you observe?

4. In this problem, we will generate a state–space representation for the *compensated* 2.04A Tower system. The model is shown in the figure below. The wind force (disturbance) is denoted as w, and the actuator force is denoted as a. Note that the actuator exerts equal but opposite forces on the tower and slider.



4.a) Which forces are acting on the tower? Be particularly careful when you include the force due to spring k_2, since the spring extension is the *relative* displacement of the tower with respect to the slider. Similarly, be particularly careful when you include the force due to damper b_2, since damping is due to the *relative* velocity of the tower with respect to the slider. By applying force balance, obtain an equation of motion for the tower.

4.b) Which forces are acting on the slider? Be careful with the spring and damper forces for the same reasons quoted in the previous question. By applying force balance, obtain an equation of motion for the slider.

4.c) Now define the tower displacement and velocity x_1, $v_1 = \dot{x_1}$, respectively, and slider displacement and velocity x_2, $v_2 = \dot{x_2}$, respectively, as shown in the Figure. We will refer to these as the **state variables.** We also define the **state vector**

$$\mathbf{q} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{pmatrix} \equiv \begin{pmatrix} x_1 \\ v_1 \\ x_2 \\ v_2 \end{pmatrix}.$$

Rewrite the tower's and slider's equations of motion in terms of the state variables q_1, q_2, q_3, q_4.

4.d) Solve the tower's equation of motion for $\dot{q}_2$ and the slider's equation of motion for $\dot{q}_4$.

4.e) To the two equations that you obtained in the previous question append the definitions $q_2 = \dot{q}_1$, $q_4 = \dot{q}_3$. Rewrite the resulting four equations in matrix form, *i.e.* find the matrix $\mathbf{A}$ and vectors $\mathbf{B}$, $\mathbf{G}$ such that

$$\dot{\mathbf{q}}(t) = \mathbf{A}\mathbf{q}(t) + \mathbf{B}a(t) + \mathbf{G}w(t).$$

We will refer to $\mathbf{B}$, $\mathbf{G}$ as **actuation** and **disturbance** vectors, respectively. (Note that in multi–input systems $\mathbf{B}$ and $\mathbf{G}$ would actually be matrices; hence, the uppercase letter notation.)

4.f) Now substitute system parameters $m_1 = 5.11\text{kg}$, $b_1 = 0.767\text{N} \cdot \text{sec/m}$, $k_1 = 2024\text{N/m}$; $m_2 = 0.87\text{kg}$, $b_2 = 8.9\text{N} \cdot \text{sec/m}$, $k_2 = 185\text{N/m}$. Using these values into the system matrix $\mathbf{A}$ and use MATLAB to compute the eigenvectors and eigenvalues as follows: `[va,da]=eigs(a)`. This will return two matrices `va` and `da`. The *columns* of matrix `va` are the eigenvectors; the *diagonal elements* of matrix `da` are the eigenvalues corresponding to the eigenvectors column–by–column. The four eigenvalues you obtain should form two complex–conjugate pairs.

4.g) The eigenvalues of the matrix $\mathbf{A}$ are the same as the poles of the transfer function relating *any* of the state variables q_1, q_2, q_3, q_4 to either the actuation a or the disturbance w. Use this fact to obtain the respective damped oscillation frequencies and natural frequencies of the two **modes of oscillation** of this system (one mode corresponds to one complex–conjugate pair of poles≡eigenvalues.) Use the result to justify the following statement: "The 2.04A Tower has two modes of oscillation, one slow and one fast."

4.h) Use the correspondence between eigenvectors and eigenvalues to justify the following statement: "In the slow mode the tower and slider mass oscillate in phase, while in the fast mode the tower and slider mass oscillate out of phase."

4.i) In MATLAB, define the matrices `b`, `c1` such that `b` is the actuation matrix with the wind force acting as the sole input to the system and `c1` is the observation matrix with the tower displacement as the system output. Also define a scalar `d=0`. Use the command `tower1=ss(a,b,c1,d)` to obtain the state–space representation of the system. Call the LTI Viewer with the command `ltiview`, import `tower1` and select the impulse response. Which mode has been excited by the impulse input? Compare the damped frequency of oscillation of the mode that you think has been excited with the frequency of oscillation that you measure from the impulse response simulation.

4.j) Now define a new matrix `c2` such that the slider displacement is the system output, and a new state–space representation `tower2=ss(a,b,c2,d)`. Open a new LTI Viewer window (without closing the window that you generated in the previous question), import `tower2` and generate its impulse response. Is the phase relationship between the `tower1` and `tower2` responses consistent with the oscillation mode that the system is in?

4.k) Convenient as they are for observing these interesting behaviors, the tower and slider displacements are not easy to measure in a real situation of active compensation in a building. To emulate the situation better, in our experimental project we will use two independent measurements of the tower velocity v_1 and *relative* velocity $v_2 - v_1$, respectively. Define a new matrix `c3` for these output variables, and a new state–space representation `tower3=ss(a,b,c3,d)`. Generate its impulse response in the LTI viewer, and comment on how it relates to the displacement impulse responses of the previous questions.

MIT OpenCourseWare
http://ocw.mit.edu

2.04A Systems and Controls
Spring 2013

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.