ECE-320: Linear Control Systems
Homework 8

Due: Tuesday October 27, 2009 at the beginning of class

Exam #2, Thursday October 29

1) For the plant

$$G_p(s) = \frac{1}{s(s+2)}$$

show that the first order controller that put the closed loop poles at $-1 \pm j$ and -3 is $G_c(s) = 2$.

2) For the plant

$$G_p(s) = \frac{1}{s+1}$$

a) Determine a controller so the closed loop pole is at -10, and then determine a prefilter so the steady state error for a unit step is zero.

b) Determine a controller $G_c(s)$ so the system is a type 1 system and the closed loop poles are at $-2 \pm j$. Show that the resulting closed loop transfer function is

$$G_o(s) = \frac{3s+5}{s^2+4s+5}$$

and determine the steady state error for a unit ramp.

3) For the plant

$$G_p(s) = \frac{s-1}{s+1}$$

a) Determine a **strictly proper** controller $G_c(s)$ so the closed loop poles are at $-2 \pm j$, determine the closed loop transfer function, and find a constant prefilter so the steady state error for a unit step is zero. Show that the resulting closed loop transfer function (without the prefilter) is

$$G_o(s) = \frac{-s+1}{s^2+4s+5}$$

b) Determine a **strictly proper** controller $G_c(s)$ so the closed loop poles are at $-2 \pm j$ and -5 ($D_o(s) = s^3 + 9s^2 + 25s + 25$), and the system is a type one system. Show that the resulting closed loop transfer function is

$$G_o(s) = \frac{(-25-21s)(s-1)}{s^3+9s^2+25s+25}$$

4) Consider the plant

$$G_p(s) = \frac{\alpha_0}{s+\alpha_1} = \frac{3}{s+0.5}$$

where 3 is the nominal value of α_0 and 0.5 is the nominal value of α_1. In this problem we will investigate the sensitivity of closed loop systems with various types of controllers to these two parameters. We will assume we want the settling time of our system to be 0.5 seconds and the steady state error for a unit step input to be less than 0.1.

a) (*ITAE Model Matching*) Since this is a first order system, we will use the first order ITAE model,

$$G_o(s) = \frac{\omega_o}{s+\omega_o}$$

i) For what value of ω_o will we meet the settling time requirements and the steady state error requirements?

ii) Determine the corresponding controller $G_c(s)$.

iii) Show that the closed loop transfer function (using the parameterized form of $G_p(s)$ and the controller from part ii) is

$$G_o(s) = \frac{\frac{8}{3}\alpha_0(s+0.5)}{s(s+\alpha_1)+\frac{8}{3}\alpha_0(s+0.5)}$$

iv) Show that the sensitivity of $G_o(s)$ to variations in α_0 is given by

$$S_{\alpha_0}^{G_0} = \frac{s}{s+8}$$

v) Show that the sensitivity of $G_o(s)$ to variations in α_1 is given by

$$S_{\alpha_1}^{G_o} = \frac{-0.5s}{s^2+8.5s+4}$$

b) (*Quadratic Optimal*) For $q = 7.083$, show that the quadratic optimal closed loop transfer function is approximately

$$G_o(s) = \frac{8}{s+8}$$

Hence the results of both of our model matching methods will be very nearly the same.

c) (*Proportional Control*) Consider a proportional controller, with $k_p = 2.5$.

i) Show that the closed loop transfer function is

$$G_o(s) = \frac{2.5\alpha_0}{s + \alpha_1 + 2.5\alpha_0}$$

ii) Show that the sensitivity of $G_o(s)$ to variations in α_0 is given by

$$S_{\alpha_0}^{G_0} = \frac{s + 0.5}{s + 8}$$

iii) Show that the sensitivity of $G_o(s)$ to variations in α_1 is given by

$$S_{\alpha_1}^{G_o} = \frac{-0.5}{s + 8}$$

d) (*Proportional+Integral Control*) Consider a PI controller with $k_p = 4$ and $k_i = 40$.

i) Show that the closed loop transfer function is

$$G_o(s) = \frac{4\alpha_0(s + 10)}{s(s + \alpha_1) + 4\alpha_0(s + 10)}$$

ii) Show that the sensitivity of $G_o(s)$ to variations in α_0 is given by

$$S_{\alpha_0}^{G_0} = \frac{s(s + 0.5)}{s^2 + 12.5s + 120}$$

iii) Show that the sensitivity of $G_o(s)$ to variations in α_1 is given by

$$S_{\alpha_1}^{G_o} = \frac{-0.5s}{s^2 + 12.5s + 120}$$

e) (*Diophantine Equations*) Consider a controller designed using the Diophantine equations. Assume we want a type 1 system and closed loop poles both at -8.

i) Show that the controller is given by

$$G_c(s) = \frac{21.333 + 5.1667s}{s}$$

ii) Show that the sensitivity of $G_o(s)$ to variations in α_0 is given by

$$S_{\alpha_0}^{G_0} = \frac{s(s+0.5)}{(s+8)^2}$$

iii) Show that the sensitivity of $G_o(s)$ to variations in α_1 is given by

$$S_{\alpha_1}^{G_o} = \frac{-0.5s}{(s+8)^2}$$

f) Using Matlab, simulate the unit step response of each type of controller (except the quadratic optimal). Plot all responses on one graph. Use different line types and a legend. Turn in your plot and code.

g) Using Matlab and subplot, plot the sensitivity to α_0 for each type of controller (except the quadratic optimal) on ***one graph*** at the top of the page, and the sensitivity to α_1 on one graph on the bottom of the page. Be sure to use different line types and a legend. Turn in your plot and code. Only plot up to about 8 Hz (50 rad/sec) using a semilog scale with the sensitivity in dB (see below). ***Do not*** *make separate graphs for each system!*

In particular, these results should show you that the model matching methods, which essentially try and cancel the plant, are generally more sensitive to getting the plant parameters correct that either the PI or the Diophantine methods for low frequencies. However, for higher frequencies the methods are all about the same. The P controller, the simplest of the four, is always the worst.

Hint: If $T(s) = \frac{2s}{s^2+2s+10}$, *plot the magnitude of the frequency response using:*

```
T = tf([2 0],[1 2 10]);
w = logspace(-1,1.7,1000);
[M,P]= bode(T,w);
Mdb = 20*log10(M(:));
semilogx(w,Mdb); grid;
xlabel('Frequency (rad/sec)');
ylabel('dB');
```

Preparation for Lab 8

5) In this problem we are going to be adding a diophantine equation solver to your **closedloop_driver.m ,** then reproducing the results from problems 1 and 2 (This solver won't work for problem 3, since there are some unusual requirements in that problem.)

a) Download the function **solve_diophantine.m** from the class web site.

b) Make the plant transfer function the same as in problem 1 and set the input amplitude to 0.5.

c) Comment out all of the other controllers, and add the lines

```
p = [-1+j -1-j -3];      % desired closed loop pole locations
m = 1;                      % the order of the controller

Gc = solve_diophantine(Gp,p,m);
if (isempty(Gc))
  return;
end;
```

d) Set the **saturation_level** to something like 100, *temporarily*.

e) Run the code to check your answers for problem 1. Turn in your plot. Be sure to run the simulation long enough to reach steady state, and be sure your **constant** prefilter is set correctly to give you a steady state error of 0.

6) Modify the code to duplicate the results from problem 2. Run the code and check your answers for both parts of problem 2. Turn in your plots. Again use an input amplitude of 0.5. Be sure to run the simulation long enough to reach steady state, and be sure your **constant** prefilter is set correctly to give you a steady state error of 0.

7*) You will be using this code and the following designs in Lab 8, so come prepared!*

a) Get the state variable model files for one rectilinear (model 210) or one rotational (model 205) system you modeled in lab (only one of the 1 dof systems). *Since you will be implementing these controllers during lab 8, if you have any clue at all you and your lab partner will do a different system!*

You will need to have ***closedloop_driver.m*** *load the correct state model into the system!*

b) Design a type 0 controller for your *rectilinear* (Model 210) or *torsional* (Model 205) system using the Diophantine equation method. You need to set the **saturation_level** to the correct level. You will need to choose the closed loop pole locations (This is a guess and check sort of thing. The biggest problem is making sure the control effort is not too large.) Your resulting design must have a settling time of 0.6 seconds or less and must have a percent overshoot of 25% or less. You should use a 1 cm step or a 15 degree step input (be sure to convert to radians). Your design should not saturate the system (control effort), your controller must be stable, and you should use a **constant prefilter**. *Note that your system may do strange things, like starting off in the negative direction. That's ok as long as you meet the constraints.*

c) Run your simulation for 2.0 seconds. Plot both the system output (from 0 to 2 second) and the control effort (from 0 to 0.2 seconds). Plot the control effort only out to 0.2 seconds since the control effort is usually largest near the initial time. If your control effort reaches its limits, you need to go back and modify your design. Turn in your plot and your controller (you can just write this on your plot).

d) Modify your design from part b to implement a type 1 controller, with all other conditions the same (you may have to modify the closed loop pole locations). The prefilter should be a **constant** and the controller should be stable. Turn in your plot.

e) (*Potentially Tricky!*) Modify your design from part d to use a **dynamic** prefilter, as you did in the last homework. Here the denominator of the prefilter is the numerator of the closed loop transfer function, and the numerator is set for a steady state error of zero (Think!) Turn in your plot, your controller, and your prefilter (you can write the controller and prefilter on the graph). ***The prefilter must be stable***, hence the numerator must have all of its zeros in the LHP. You may have to modify your pole locations from part d. For my systems, I had some complex conjugate poles with imaginary parts larger than the real parts, but this may not help you. You will probably need to iterate here.

Turn in your code and your plots.