ECE-521 Control Systems II
Laboratory 3
System Modelling, Controller Design, and the Real World

<u>Preview</u> In this Lab you will first obtain a second order model of your spring/mass/damper system, design a controller for it, implement the controller on the real system, and then compare the predicted response with the actual response. You will probably make at least two observations during this lab:

- models are not perfect, they are just guides

- real motors have real limits, which can make designing more difficult

You will go through the following steps:

1. Estimate an initial second order system model using time domain analysis (using either the **log_dec** or **fit** programs).

2. Measure the frequency response (make one measurement at 1Hz, 2Hz, ..., 7 Hz, and at least 4 points near the resonant peak)

3. Use the **fit_bode** command to estimate the gain of the system.

4. Use the **opt_fit_bode** command to fine tune the system model.

5. Determine the closed loop system gain k_{clg} (to be described below)

6. Design P, I, PI, PD, and PID controllers using the model developed above and Matlab's *sisotool*

7. Implement the controller design on the system and examining the step response. Export the step response of the real system and edit the data so Matlab can read it. Use the program **closed_loop_response** to plot both the predicted closed loop response (based on the system model) and the true closed loop response (the real system).

8. Tweak the parameters on your controllers to see if they improve the performance on the real system.

9. Examine the response of your controllers to other types of inputs.

Since we are going to be implementing real controllers, it is very likely the motors are going to become unhappy at some point. You will know they are unhappy if you hear a buzzing. You need to be prepared to shut off the system quickly when this happens. Also, if your systems appear to unstable (usually identifiable as oscillations growing in magnitude), shut off the system.

1) Print out this lab and **read** it.

In what follows, assume we have the unity feedback system shown below. $G_c(s)$ is the controller that you will be implementing, ω_n and ζ are estimates obtained using either time-domain or frequency domain methods, and k_{clg} is the system closed loop gain. Note that we are modelling the motor as contributing only a gain to the system, and we are lumping both the plant's gain and the motor gain together into one parameter.
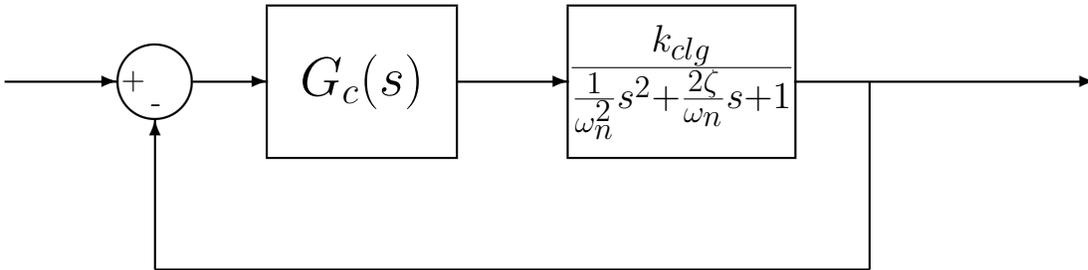
2) Show that for

$$G_c(s) = k_p$$

the steady state output y_{ss} due to a step input of amplitude A is given by

$$y_{ss} = \frac{A k_p k_{clg}}{1 + k_p k_{clg}}$$

which can be rewritten as

$$k_{clg} = \frac{y_{ss}}{k_p} \frac{1}{A - y_{ss}}$$

This is the expression we will use to estimate the closed loop system gain k_{clg}.

3) Show that for a PD controller

$$G_c(s) = k(s + z)$$

the position error is given by

$$e_p = \frac{1}{1 + k k_{clg} z}$$

4) The standard form for a PID controller is

$$G_c(s) \;=\; k_p + k_i \frac{1}{s} + k_d s$$

show that

a) for a PID controller in the form $G_c(s) = k(s+z_1)(s+z_2)/s$, $k_p = k(z_1+z_2)$, $k_i = kz_1z_2$, $k_d = k$

b) for a PI controller in the form $G_c(s) = k(s + z)/s$, $k_p = k$, $k_i = kz$, $k_d = 0$

c) for a PD controller in the form $G_c(s) = k(s + z)$, $k_p = kz$, $k_i = 0$, $k_d = k$

We generally need to keep the k_p, k_i and k_d less than about 0.1. (Sometimes I have used a k_i of about 9) You need to use the above relationships when choosing where the poles and zeros of your controllers should be.

Estimating the Closed Loop Gain k_{clg}

0. Set the units

Click **Setup → User Units** and set the units to **cm**.

1. Setting up the controller

Click **Setup → Control Algorithm**. Be sure the system is set for *Continuous Time*. Select **PID** under **Control Algorithm**. Click on **Setup Algorithm**. Be sure **Feedback** is from **Encoder 1**. Set k_p to a small number (less than or equal to 0.05) and be sure $k_d = 0$ and $k_i = 0$. Then click **OK**. Next Click **Implement Algorithm**. The click **OK**.

2. Setting up the closed loop trajectory

Click **Command → Trajectory**. Select **Step** and click on **Setup**. Select **Closed Loop Step** and set **Step Size** to 1 to 2 cm. Be sure to record this step size (we'll refer to the amplitude as A below). Set the **Dwell Time** to something like 2000 ms, this is the time the system will be recording data. Finally click **OK**, then **OK** and you should be back to the main menu.

3. Executing the closed loop step

Click **Command Execute**. A menu box will come up with a number of options, and a big green **Run** button. Click on the **Run** button. When the system has finished collecting data, a box will appear indicating the how many sample points of data have been collected. (If you have hit a stop, the system stops recording data. This usually means you're input amplitude was too large or k_p was too large.) Click on **OK** to get back to the main menu.

4. Determining the steady state value

Click **Plotting → Setup Plot**, or just **Plotting Data → Plot Data**. Look at the steady state value (y_{ss}). You may need to change the dwell time if your system has not reached steady state.

5. Estimating the closed loop gain

Estimate the closed loop gain k_{clg} using the formula derived in the prelab: $k_{clg} = \frac{y_{ss}}{k_p} \frac{1}{A - y_{ss}}$.
You need to go through this procedure at least three times for each configuration. You must use at least two different values of k_p and two different values of input amplitude A. If none of the steady state values is larger than 0.4 cm, increase either k_p or A. Average the three results to get your k_{clg} (they should be similar). For the trials I've run, I've got k_{clg} between 10 and 20. Your's may be outside this range though.

$\boxed{\text{6. Estimating the transfer function}}$

The final plant transfer function we will use is then

$$G(s) \quad = \quad \frac{k_{clg}}{\frac{1}{\omega_n^2}s^2 + \frac{2\zeta}{\omega_n}s + 1}$$

where ω_n and ζ were found previously. This is the transfer function to be used in both Mathlab's *sisotool* and in the **closed_loop_response** program.

Designing and Implementing the Controllers

For each of the controller types, you should attempt to

- produce a position error of less than 0.15

- reach steady state within 5 seconds (the faster the better)

- have as little overshoot as you can manage

I represent management, which means I expect things which may not be possible. If you cannot meet these vague criteria, do the best you can with each controller type to meet the criteria.

Here are some general ideas:

1. Use Matlab's *sisotool* program to design the controllers.

2. Try and keep all k_p and k_d gains less than about 0.1 (it's OK if they are larger, just be prepared for the motor to buzz and the controller to not work) Look at the prelab for the relationships between the controller poles and zeros and k_p, k_i and k_d.

3. Be aware that you enter the control parameters in the order k_p, k_d, then k_i

4. Be sure to zero out the gains you are not using (i.e., if you are using a PI controller, be sure k_d is set to zero)

5. You may need to reset the controller often, such as every time you want to implement a new controller. Click **Utility** → **Reset Controller**. Only do this **before** you have implemented a controller.

6. You may need to rephase the motor. Click **Utility** → **Rephase Motor**

7. Be sure to **Implement** the controller you have designed.

8. Try and track a step with an input amplitude of 1 cm.

9. Try and track at least two **different** inputs (ie., a sinusoid, a ramp, a parabola .. not another step of a different amplitude) once your system is designed. Do not worry about whether your controller works for these other inputs.

Memo

Your memo should compare (briefly) the response of the model and the response of the real system for the different types of controllers. You should also indicate if any of the controllers could not be made to work, and why. Finally you should have some description of the configuration of the system you were trying to control.

You should include the following items as attachments. Most of these are figures which should have reasonable captions.

- The step response of the time-domain model.

- The initial frequency response of the system.

- The optimized frequency response of the system.

- The data used to determine the closed loop gain.

- The final model of the system.

- The predicted and actual response of the system to each of the different controllers.

- The response of the system to different inputs.