1. Use "Lab 0" document to build a simulink model to collect encoder data.

2. Use a polynomial fit to estimate velocity for some subsection of interesting data. (Use about 1 sec of data.)
   (HINT: Use both "polyfit" and "polyder".

3. Create a filter in MATLAB to calculate velocity from the encoder data (in MATLAB).

4. Calibrate data - How many radians per encoder ct?
   (a) Distance per encoder ct?

We know the downward-pointing (stable) pendulum w/ no actuation should obey this equation of motion:

(1) \( J \ddot{\theta} + B \dot{\theta} + g m \frac{L}{2} \sin(\theta) = 0 \)

Linearized (for small \( \theta \)), this is close to:

(2) \( J \ddot{\theta} + B \dot{\theta} + g m \frac{L}{2} \dot{\theta} = 0 \)

The left-hand side is the "characteristic equation" giving poles for the pendulum (linearized) dynamic system.
We can rewrite (2) in Laplace notation as:

(3) \[ s^2 + 2 \tilde{\omega}_n s + \tilde{\omega}_n^2 = 0 \]
5. From the data below, estimate:
\[ w_a = \quad, \quad S_1 = \quad, \quad w_p = \quad \]
\[ S_1 = \quad, \quad S_2 = \quad (poles) \]

6. Write the characteristic eqn. and new poles you expect for the inverted configuration of the same pendulum. \[ S_1 = \quad \]
\[ S_2 = \quad \]

7. Given eqn. 2 & your answers in 5, what would the length of the pendulum be? (Assume a solid rod...)
\[ L \approx \quad \]
8. Finally, implement a digital filter within your model to output velocity of the encoder.

- Create a single MATLAB plot that shows both a "polynomial fit" estimate of velocity (as in Problem 2), which should be smooth and have no added time delay, and the output of the "velocity" filter from your Simulink model.

*Email* some version of this figure (MATLAB "fig" file, PDF, JPG, ...) AND your Simulink "mdl" file, AND your code for the polynomial est. AND a "mat" file with encoder pos & velocity data.

katiebyl@ece.ucsb.edu

(You must also "cc" all group members you worked with, so they have copies & I know who was in what group!!)

9. **Homework** - To be done individually.

   Improve your filter (or other estimation method) for encoder velocity. **BE SURE IT WILL WORK IN REAL TIME**, although you may simulate using MATLAB & emailed encoder (raw) position data at home.
Example of a plot for part 8.

Example of filtered velocity est. vs. polynomial est.

This filter introduces a time delay of ~0.05 sec.