

ME417 - Laboratory Exercise #2

Introduction to Simulink

This laboratory exercise is intended to provide a tutorial introduction to Simulink. Simulink is a Matlab toolbox for analysis/simulation of interconnections of dynamic systems, and it will be used heavily throughout the rest of the course/laboratory. All the exercises in this assignment can be done entirely in Matlab/Simulink.

1) Running, Plotting, Printing: In order to see a demonstration Simulink diagram type *thermo* at the Matlab prompt. Open the *scope* block, labeled “Thermo Plots” by double clicking, and then run the simulation using the buttons or pull down menus provided. Print the plot of the simulation output (scope block) and print the simulation model itself.

2) Model Building: Figure 1 shows a Simulink model which represents the motor gear system in the Controls Laboratory, with a PID controller implemented in feedback around it. Launch the simulink library browser from within Matlab by using the button or typing *simulink* (or *simulink3* for the older style which some people prefer). Then open a new model (using button or pull down menus), and build a copy of the above model. This is achieved by dragging components from the library to the model and connecting them using the mouse. Double clicking a box then allows you to edit the contents, such as entering values for the transfer function (as shown). For the PID block set the proportional gain to 0.05, and the integral and derivative gains to zero.

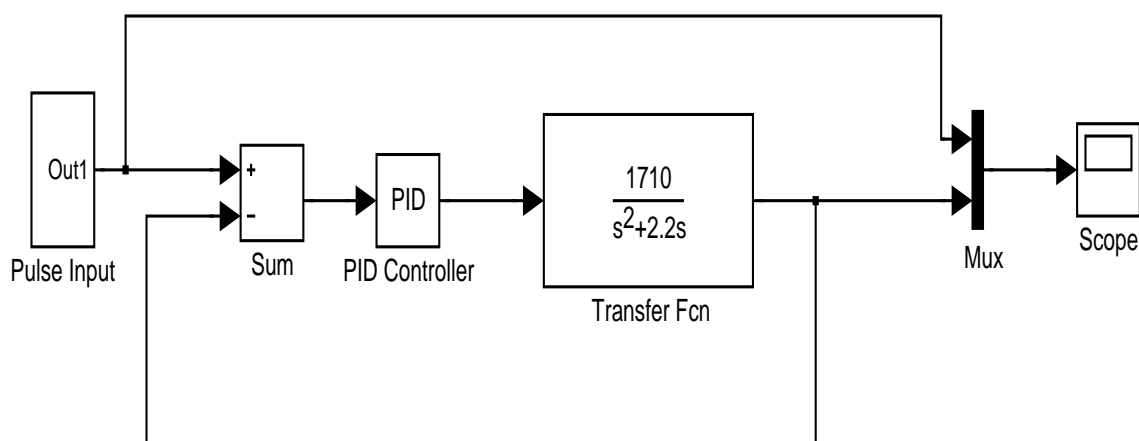


Figure 1: Simulink model of motor gear drive system

Look around at the (many) available blocks in Simulink. You will certainly need to look in *Sources*, *Sinks*, *Continuous*, *Math*, *Signals & Systems*, as well as *Additional Linear* under *Simulink Extras* for the PID controller.

Note that there is no block for “Pulse Input”, but I made that myself from basic components using the *Create Subsystem* command under *Edit*. The contents of the box are shown in figure 2. You can even use the *Mask Subsystem* command to generate your own library blocks.

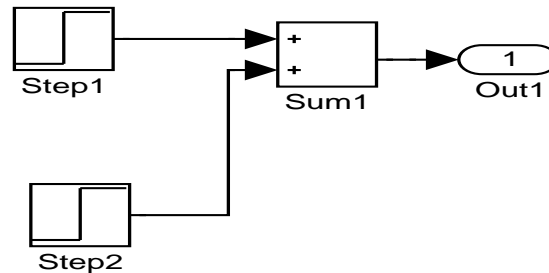
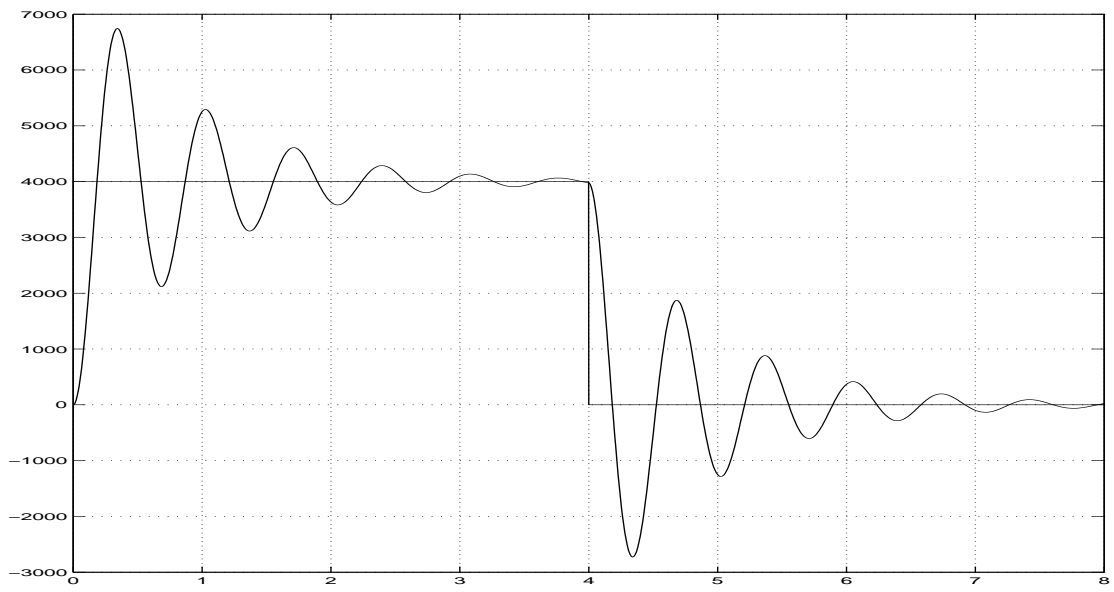


Figure 2: Simulink pulse input subsystem

When you have built a copy of the model save it with the name “gear” (it will actually be saved as gear.mdl). You can then launch this model later from Matlab simply by typing *gear* at the command line. Go under *Simulation* to *Parameters* and set the simulation time to 8 seconds. Then run the simulation and print the results from the scope block. You should get a plot like figure 3 which shows the commanded response and the actual system response (note the autoscale button on the scope). Of course note that in order to get the correct commanded response you will need to enter appropriate values for the two step input blocks that make up your pulse input.

Having completed this exercise you should have a model plot and a simulation run that essentially reproduce the figures shown here. Now try varying the parameters of the PID controller and see how they affect the closed-loop control system (note that you can enter variable names in Simulink Blocks if you like, and it will read them from the Matlab workspace). You do not need to generate large numbers of plots but plot a few of the results and discuss how the different controller parameters (Proportional, Integral, and Derivative) affect the closed loop performance. See if you can manually tune the controller to get a good step response. Later in the semester we will revisit this problem with design tools we have learned in class, and try them out both in simulation (as here) and on the actual hardware system.



Time offset: 0

Figure 3: Scope block output from simulation