ENGR 4230
Power Systems

Homework 15 [10 points]

Due 04/02/19 by 4PM (on Canvas)

A common control strategy for power systems, and particularly DC motors, is to use feedback control with a PID compensator. PID control has been widely studied and is found in a variety of industrial and robotics applications. In this assignment, you will implement and tune a PID controller to control the speed of a DC motor.

Simulink Block Diagram

1. Copy the dc_motor_sim file that you created in Simulation Assignment 2 into a new folder.
2. Modify your Simulink file to match what is shown in Figure 1. In the Step block, set Step Time to 0. Both To Workspace blocks should output data as an array.

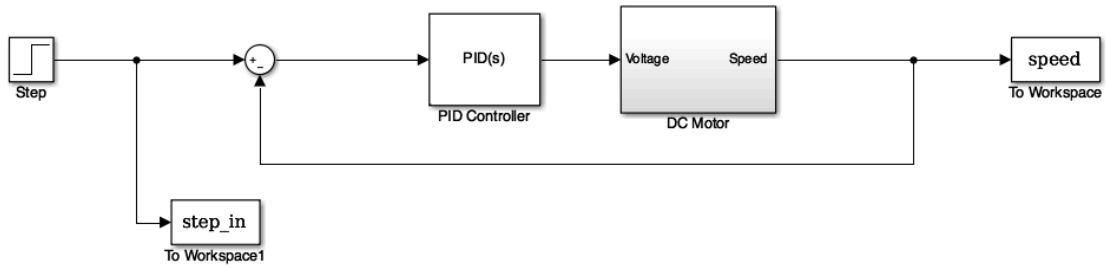

Figure 1: Simulink Model for Simulation 4.

3. Change the parameters inside the PID Controller block to match those shown in Figure 2. Save the Simulink file as dc_motor_ctl.

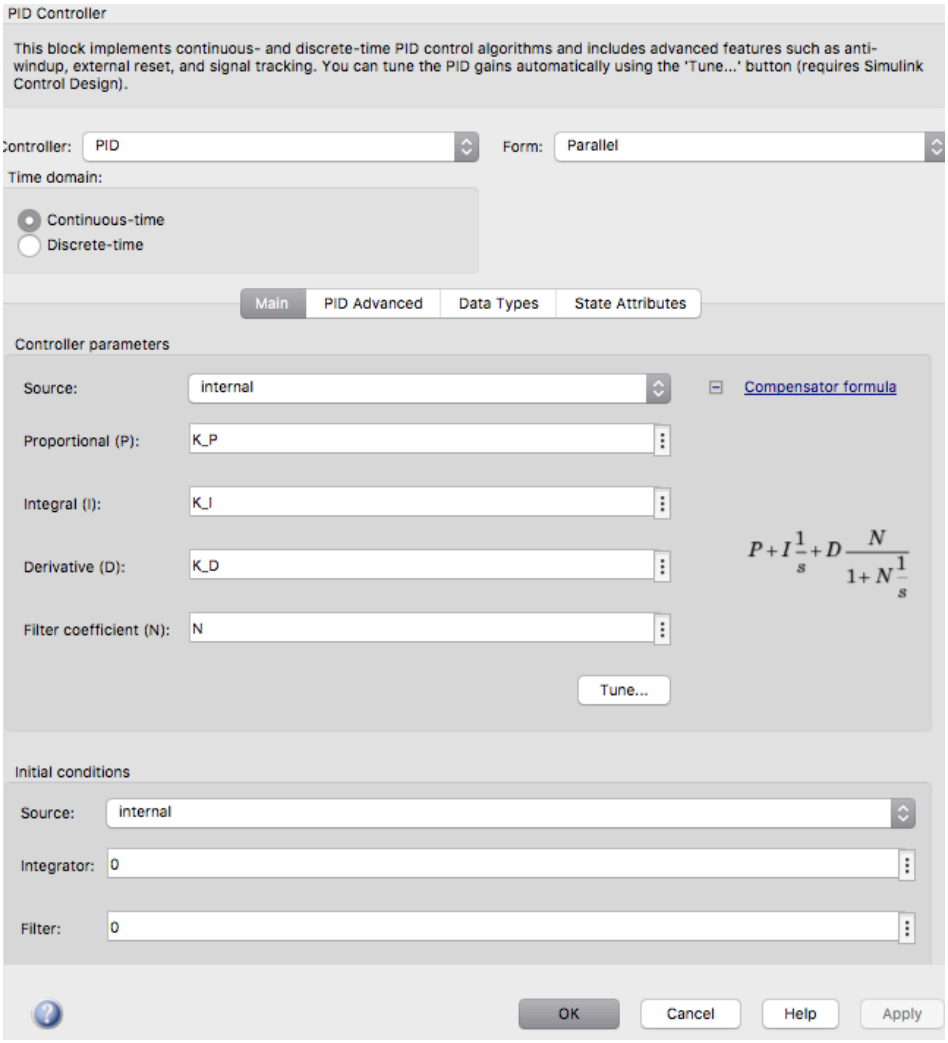

Figure 2: PID Controller parameters.

MATLAB Code

1. The motor parameters should be defined in a MATLAB m-file. They match the parameters set in Simulation 2.
2. Define control parameters K_P = 100, K_I = 0, K_D=0, and N = 100 to start.
3. Simulate the Simulink file using the command sim('dc_motor_ctl')
4. Make an appropriately labeled figure which displays both the step_in input as a function of time and the speed output as a function of time. Make sure that the two lines can be distinguished even in black and white.
5. Use the command *stepinfo* to get performance characteristics of the system. That is, in your m-file, type *stepinfo(speed,tout)*. This will provide you with performance characteristics including RiseTime, SettlingTime, Overshoot, and PeakTime.

Tuning the PID Controller

The initial response of the system displays a significant steady state error. Your goal is to tune the PID controller to meet the following performance characteristics:

- Overshoot $\leq 5\%$
- SettlingTime ≤ 1 s
- Steady State Error = 0

Table 1 shows rules for tuning PID controllers.

Table 1: Rules for tuning PID controllers. These parameter changes are achieved by increasing the controller parameters.

Parameter	Overshoot	SettlingTime	Steady State Error
K_P↑	Increases	Small Change	Decreases
K_I↑	Increases	Increases	Eliminates
K_D↑	Decreases	Decreases	No Change

1. Attempt to tune the PID controller manually. Use the information in Table 1 to try to achieve the performance requirements. Provide screenshots of a few iterations in this process, including both the figure and the results of stepinfo.
2. After about 10 iterations, navigate to the PID compensator in the Simulink file. Inside the block, as shown in Figure 2, click the button called Tune… This will open a graphical tuning window, as shown in Figure 3.

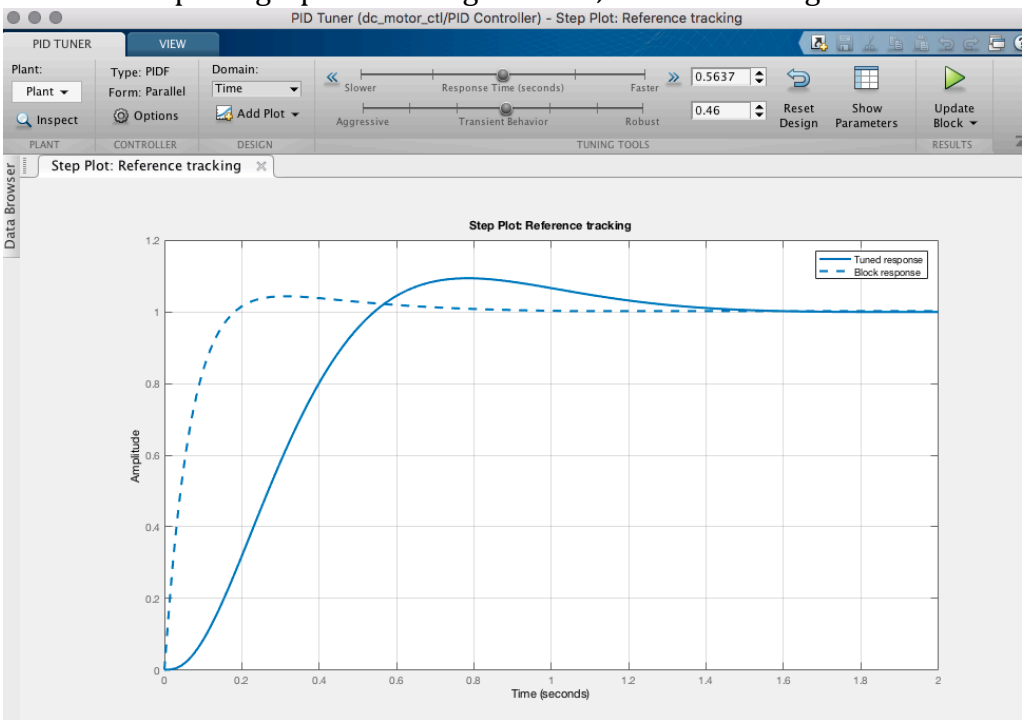

Figure 3: Tuning window.

3. Click on Show Parameters. This will allow you to see the Settling Time and Overshoot parameters you are designing for as you adjust the sliders at the top. Adjusting the sliders will allow you to achieve the design goals. This also

shows the values of K_P, K_I, K_D, and N. Take a screenshot of this window with tuned parameters.

4. Return to your MATLAB code and enter the values found using the tuning in step 3. Run the MATLAB code, and take a screenshot of the resulting figure, and the values of stepinfo, so that you can verify the control requirements.

Questions

1. Using the initial controller gains, and the fact that the step (reference) input has a value of 1, what is the initial error between the final value of the output and the input?
2. Was it easy or hard to tune by hand?
3. Which parameter made the biggest changes in the response?

Requirements

Submit in the assignments tab on Canvas, the completed MATLAB code (m-file) and Simulink file.

In addition, submit a Word document with the requested screen shots and answers to questions.