ECE-420: Discrete-Time Control Systems Project Part D

In this part of the project you will first play around with an adaptive control system, where the controller actually knows how the plant is changing. You will then estimate the plant using your RLS block from part C of the project. You will need to download the code from the class website.

Mathematical Background: Consider a simple discrete-time transfer function with input $R(z)$ and output $Y(z)$,

$$G_p(z) = \frac{Y(z)}{U(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Cross multiplying we get

$$Y(z) + a_1 z^{-1} Y(z) + a_2 z^{-2} Y(z) = b_0 U(z) + b_1 z^{-1} U(z) + b_2 z^{-2} U(z)$$

In the time-domain this becomes

$$y(n) = -a_1 y(n-1) - a_2 y(n-2) + b_0 u(n) + b_1 u(n-1) + b_2 u(n-2)$$

We can write this as

$$y(n) = \phi^T(n)\theta$$

where

$$\phi^T(n) = \begin{bmatrix} y(n-1) & y(n-2) & u(n) & u(n-1) & u(n-2) \end{bmatrix}$$
$$\theta^T = \begin{bmatrix} -a_1 & -a_2 & b_0 & b_1 & b_2 \end{bmatrix}$$

1) Run the program **adaptive_controller_driver.m**, which runs the Simulink model **adaptive_controller.mdl.** The adaptive controller is designed to create a type one system with closed loop poles where you choose to place them. At this point, both the plant and the controller exactly know the plan and how it changes. We are not yet using the RLS algorithm to determine the plant. You need to

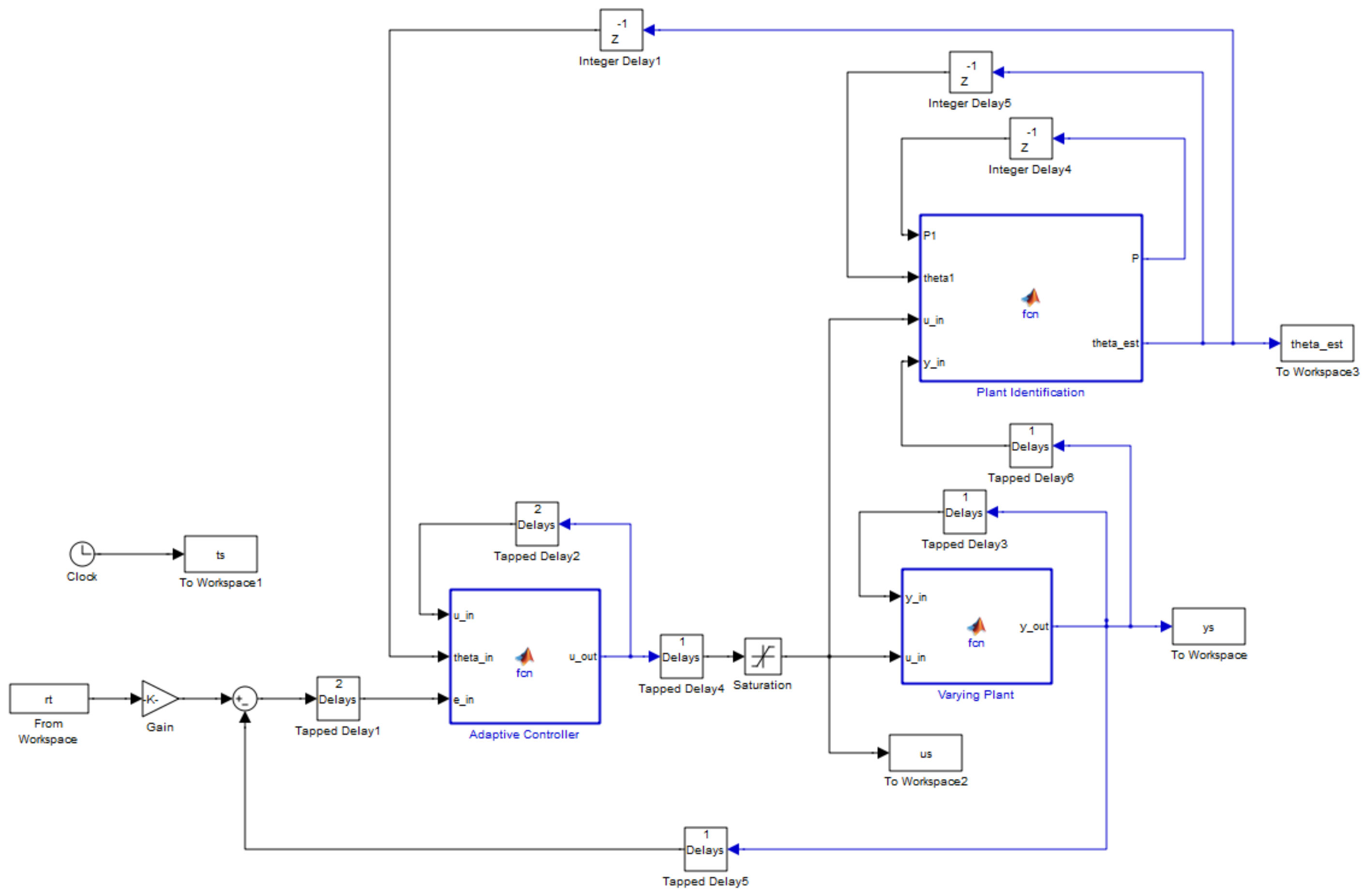
- Determine pole locations that make the system work (track the input) reasonably well
- change the saturation level to a reasonably smallest level so that the system does a reasonable job of controlling the plant.

Note that the plant changes slowly and the input is on and off, so the output will not exactly track the input (there will be spikes in it). Include the output figure in your memo. *Note that we have assumed there is a small amount of noise in our input signal, this really helps the RLS algorithm and is actually pretty realistic.*

2) We usually get better results if we sample at a higher rate, so set Ts =0.001 (sampling at 1kHz) and rerun the simulation. You can change the pole locations or saturation levels if you need to in order to get better results. Include the figure in your memo

3) Another thing we can try to do is to lowpass filter the output of the controller. Currently there are two different averaging filters at the end of the adaptive controller. Play with both of them, and include a figure with the result that you think works best (keeping Ts = 0.001) in your memo. Feel free to add your own (for extra credit as long as it is an improvement, but be sure to indicate what you did.)

4) Save **adaptive_controller.mdl** as **adaptive_controller_RLS.mdl**. Be sure to modify the Matlab code **adaptive_controller_driver.m** so it invokes this new Simulink file. Modify this new Simulink file to include the RLS (plant identification) block from Project Part C, as shown in the figure below:



The input to the RLS (plant identification) block should be the control signal and output of the plant. The estimated states should go through a unit step delay and be fed into the Adaptive Controller. You will need to modify this RLS block so it is only estimating three variables (it was estimating five). You should save **theta_est** as the estimated states (as before) and this should also be an input to the adaptive control block. At this point do not assume the controller is actually using the plant your RLS block is estimating. Uncomment the code in **adaptive_controller_driver.m** so you can see the estimated states and set Ts = 0.001.

5) Run your Matlab code, and modify the forgetting factor (lambda) until you get reasonably good estimates for the coefficients. When you get a good graph, include it in your memo.

6) Once your RLS algorithm is working, comment out the known plant in the adaptive controller, and uncomment the lines that used the estimated values. Uncomment the line at the bottom of the Adaptive Controller that starts the system off with a proportional controller (it needs some input to get started). Run your simulation and tweak the parameters to try and get a reasonable result. Include this result in your memo.

7) Change the sampling interval to Ts = 0.0001 and rerun your code. Turn in this final graph in your memo (it should work much better!)

To turn in: write a short memo including your graphs (with captions and figure numbers), and any suggestions you may have for improving this part of the project. E-mail me your memo and both your Matlab and Simulink.