

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

Before start, you need

- 1. A PC installed with MATLAB.**
- 2. Control system toolbox and Simulink control design toolbox**



Section 1. Simulation using MATLAB script.

MATLAB can be used to simulate a dynamical system directly using MATLAB commands and script. This section describes step-by-step process of simulating a transfer function in MATLAB.

1. Responses of the 1st and 2nd order LTI systems

1.1 We can calculate the system step time response and get the step information using the following MATLAB commands/script in Figure 1:

```
% step response for 1st order system
k_dc = 5;
Tc = 10;
s = tf('s');
G = k_dc/(Tc*s+1) % 1st order system
step(G)
stepinfo(G)

k_dc = 1;
w_n = 10;
zeta = 0.2;

s = tf('s');
G1 = k_dc*w_n^2/(s^2 + 2*zeta*w_n*s + w_n^2); % 2nd order system
step(G1)
axis([0 3 0 2])
stepinfo(G1)
```

Figure 1. Code for Step Response of System

1.2 Using a tool - MATLAB response analyzer

MATLAB also provides a powerful graphical user interface for analyzing LTI systems that can be accessed using the syntax “**linearSystemAnalyzer('step',G)**”. If you right-click on



the step response graph and select Characteristics, you can choose to have several system metrics overlaid on the response: peak response, settling time, rise time, and steady-state.

Type *help linearSystemAnalyzer* in the command window to get helpful info.

1.3 Complete the exercises. (attach screenshots of both code and results):

1.3.1 Refer to the follow second order systems introduced in the lecture.

- A **standard form** of the second-order system

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \begin{cases} \zeta : \text{damping ratio} \\ \omega_n : \text{undamped natural frequency} \end{cases}$$

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

where the $\omega_n = 10$, $\zeta = 0.2, 1.2, 1, 0$ separately.

Task list:

- 1.) State the category of system in terms of damping ratio. (Undamped, overdamped, critical damped or underdamped)
- 2.) Use the command of '**linearSystemAnalyzer('step',G)**' and get the step response diagram of each system.
- 3.) Then fill out the chart below for the setting time, rise time, overshoot for $\zeta = 0.2$

Damping ratio	Setting Time	Rise Time	Overshoot
0.2			


Hint: An example of code is shown below:

```
% Step response for 2nd order system
w_n = 10;
zeta = 0.2; %underdamped system

s = tf('s');
G1 = w_n^2/(s^2 + 2*zeta*w_n*s + w_n^2);
linearSystemAnalyzer('step',G1)
axis([0 3 0 2])
```

Figure 2. Code Example for Exercise 1.3.1

Section 2. Simulation with Simulink

- 2.1 Single click on the icon  in the tool menu to start the Simulink. You may also bring up this window by typing '**Simulink**' in the command window. The Simulink Start Page (shown in Figure 3) will pop up and choose a Blank Model for your first trial.

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

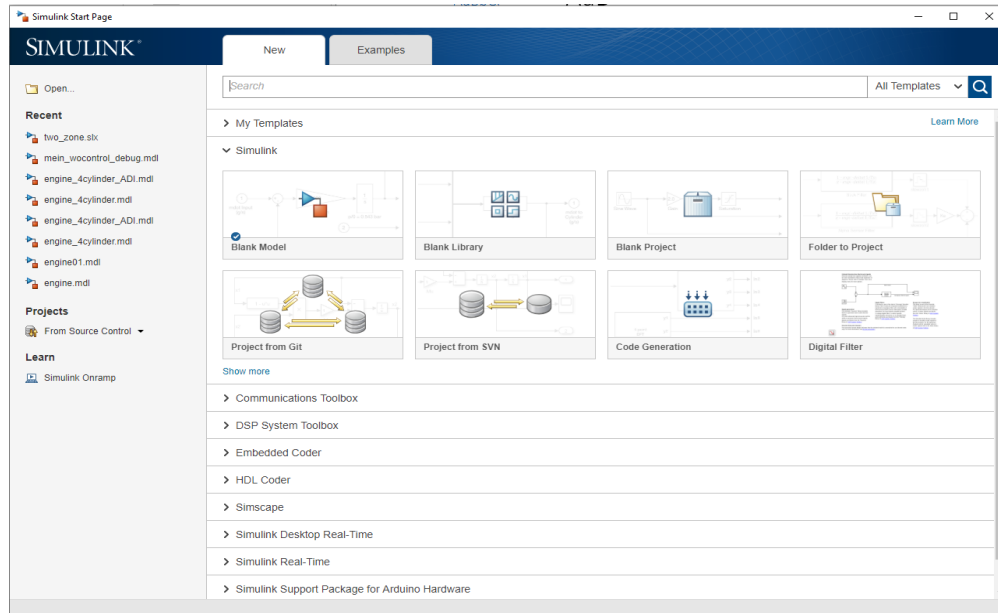



Figure 3. Simulink Start Page

2.2 Single click on the icon  to show the library. The Simulink Library Browser presents a collection of different blocks called block sets that are used to build up a model.

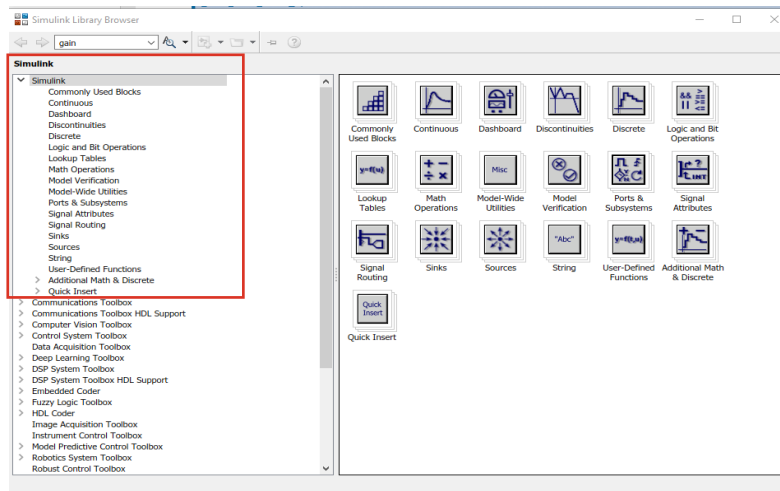


Figure 4. Simulink Library

2.3 Build the first model as shown in Figure 5. In the Library Browser, Choose the 'Step' from Source, 'Scope' from Sink, 'Transfer Fcn' from Continuous.

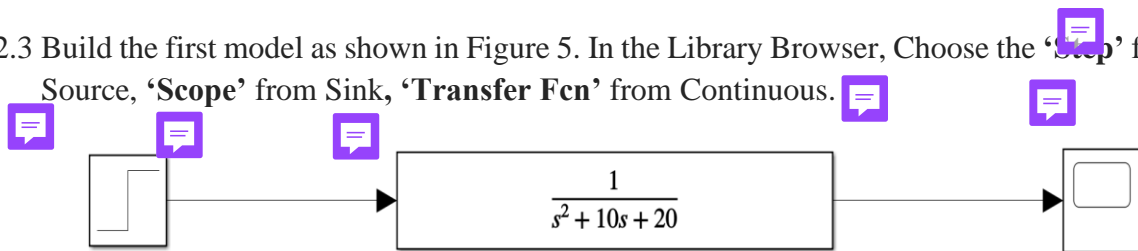


Figure 5. First Simulink Model

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

2.4 Double click on the block '**Transfer Fcn**', then the Block Parameter Window will pop up and set up the coefficients as below. The numerator and denominator coefficients are the coefficients of the polynomials of 's'.

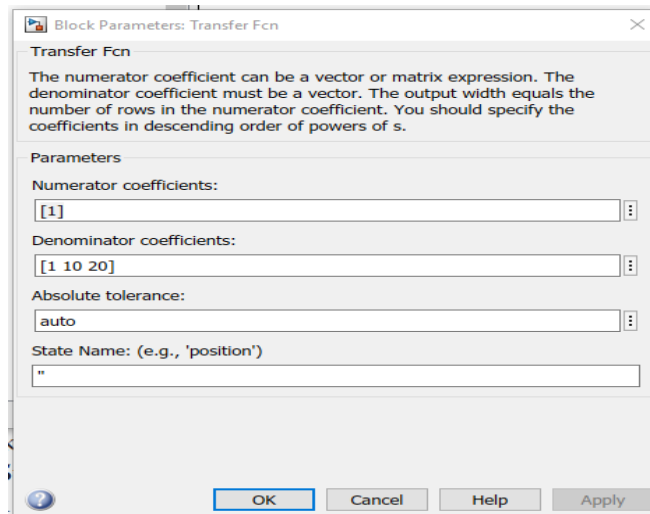


Figure 6. Parameters Configuration for Transfer Fcn Block

Note: for transfer function $\frac{s}{s^2 + 10s + 20}$, the numerator coefficient is [1, 0].

2.5 Connect these two blocks with the corresponding lines by dragging a line from its input port and drop it (release mouse button).



2.6 Single click the button to run the model. After the simulation completes, double click on the block 'Scope', you should get the same results as below.

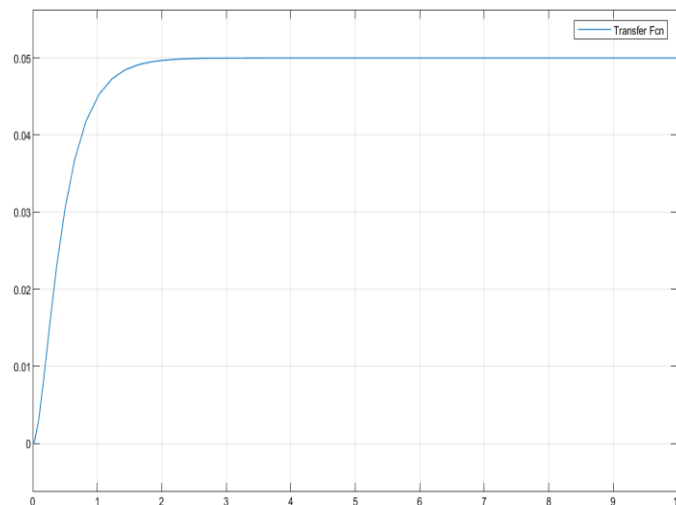


Figure 7. Result of First Run

2.7 Exercise (Show the code/model and results)

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

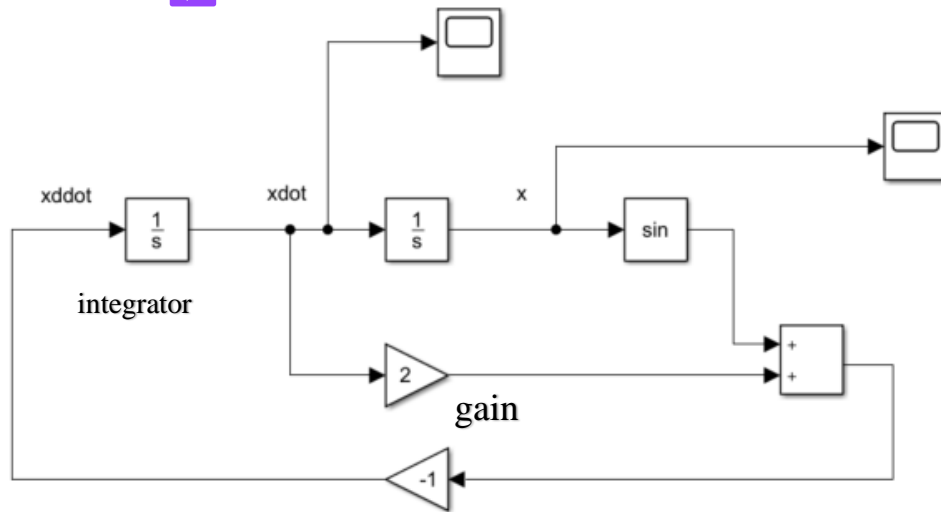
Please use $\omega_n = 10$, $\zeta = 0.2$ for the 2nd order system defined in subsection 01.3.1 and simulate the system step response.

Section 3. Simulate a nonlinear model in Simulink.

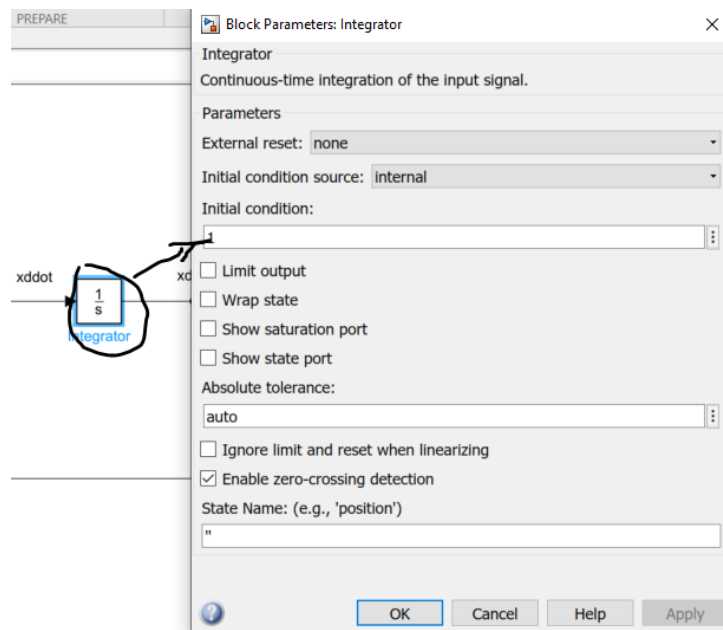
Simulink can be used to construct a nonlinear system using blocks and functions provided by Simulink.

Example: $\ddot{x} + 2\dot{x} + \sin(x) = 0$, with initial condition $x(0) = 2$, $\dot{x}(0) = 1$

We have $\ddot{x} = -(2\dot{x} + \sin(x))$, so we use the Simulink blocks, 'gain', 'scope' and 'integrator' to construct the model as



Double right click on the integrator, and set initial condition as 1 for $\dot{x}(0)$, 2 for $x(0)$.



MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

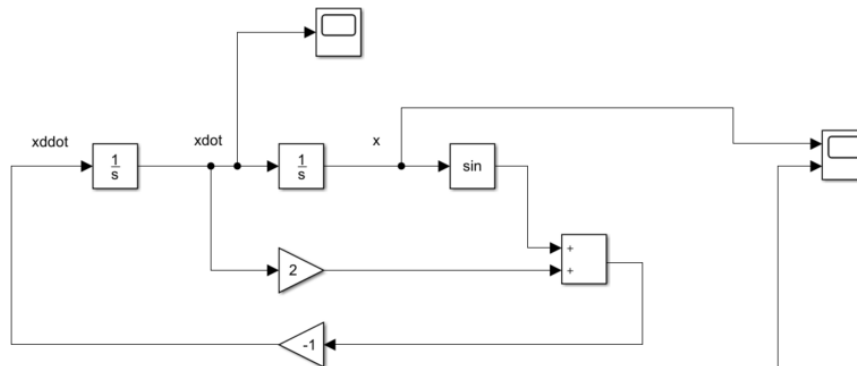


3.1 Attach the responses of x and \dot{x} from the scope. And answer the question: is this nonlinear system a stable system or unstable system ?

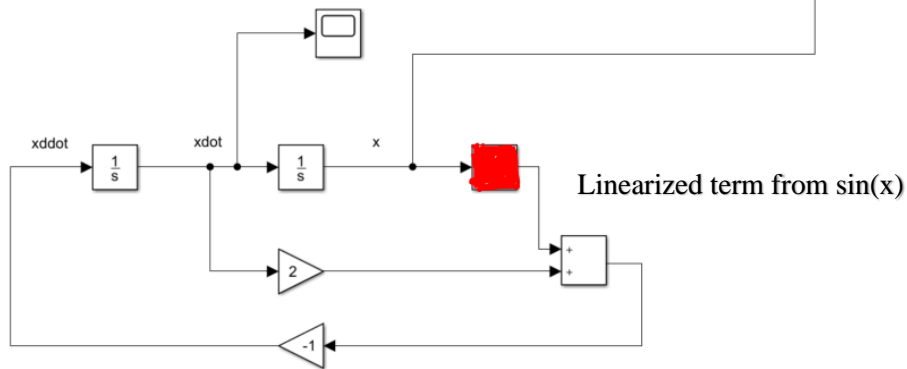
3.2 Try to linearize the nonlinear system at the operating point $x=0$, run the simulation for linear system and compare the x response with nonlinear system in the same scope.



nonlinear



linear



Section 4. PID Controller Tuning



PID Controller tuning is often conducted in an iterative way. Tune-observe-tune-observe,... until the closed-loop performance satisfies the pre-set objective.

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

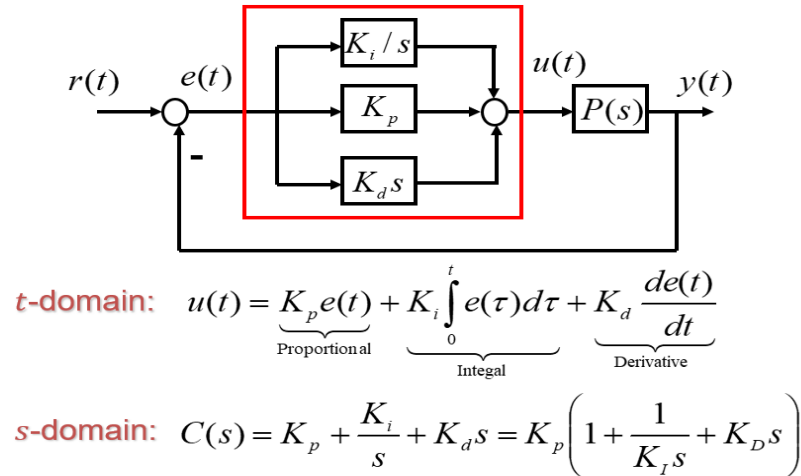


Figure 8. PID Controller



4.1 Using Matlab script pidTuner

MATLAB provides a tool for automatically optimizing (tuning) PID gains to reduce the trial-and-error effort. You can access the tuning algorithm through a nice graphical user interface (GUI) using command “**pidTuner**”. Let's explore this automated tool by first generating a proportional controller by entering the command “**pidTuner(P,'p')**”. In the shown syntax, ‘**P**’ is the previously generated plant model, and ‘**p**’ specifies that the tuner employ a proportional controller, referring to the following code.



```
s = tf('s');
P = 1/(s^2 + 10*s + 20);
pidTuner(P,'p')
```

Figure 9. Example of PIDTuner

We can now interactively tune the controller parameters and immediately see the resulting

response in the GUI window. Single click on the icon Show Parameters



in the tool



menu. The screen should be ~~same~~ as below.



MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

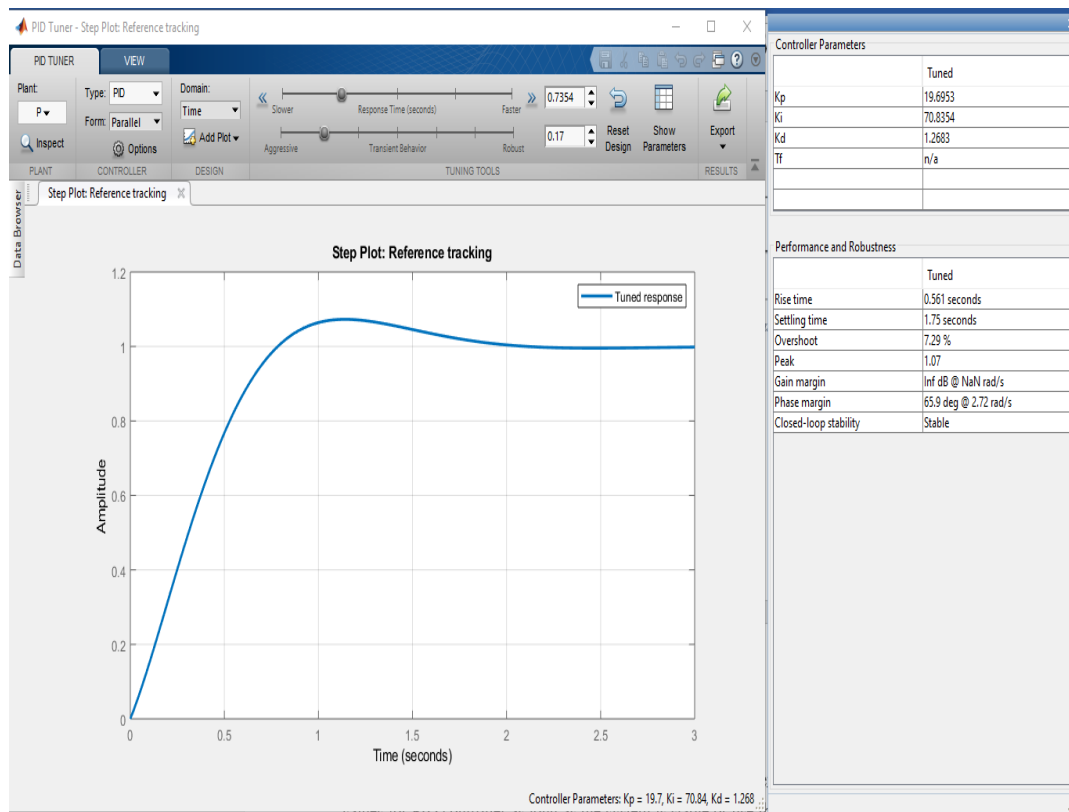


Figure 10. Example with Proportional Gain

4.2 Simulink simulation and PID tuning.

In Library Browser, Choose the block '**PID Controller**' from "Continuous" and the block '**ADD**' from "Math Operation". Double click on the add block and change the list of signs as below. Then connect each block to form the negative feedback system as shown in Figure 12.

MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

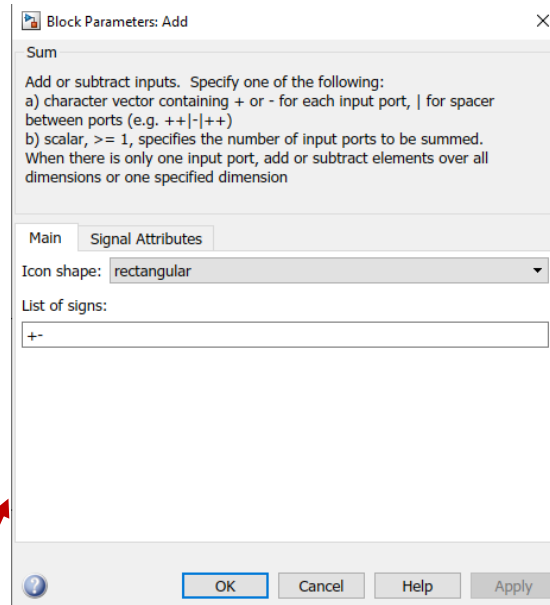


Figure 11. Block Paras for ADD

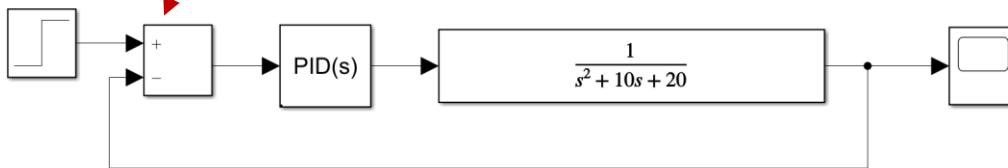
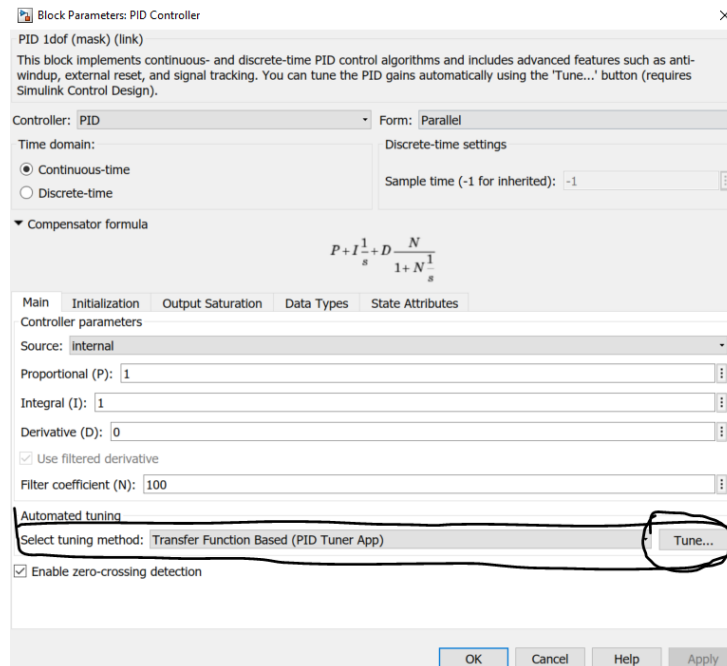


Figure 12. Simulink Model for PID Controller

Double click on PID controller block and select automated tuning 



MAE/ECE-5320 LAB 01: SYSTEM RESPONSES AND PID TUNING

The PID tuning needs Simulink® Control Design package, which can be installed in Add-on.

4.3 Exercise (Show both code/model and results)

1. build up the Simulink model for the system defined in Figure 12 and tune the PID controller such that the closed-loop system output satisfy: **rise time < 2 second, overshoot < 5%**

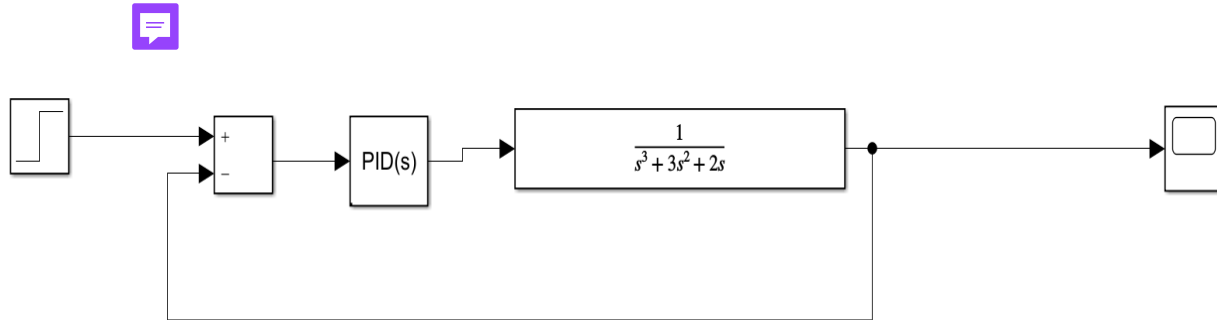


Figure 12. Simulink Model for PID Controller