

## Simulink Tutorial

### Introduction

This document is designed to act as a tutorial for an individual who has had no prior experience with Simulink. It is assumed that the reader has already read through the Beginner and Intermediate Matlab Tutorials. For any questions or concerns, please contact

Christopher Lum  
lum@u.washington.edu

### Starting the Program

1. Start Matlab.
2. Simulink is an extra toolbox that runs on top of Matlab. To start this, type “simulink” in the Command Window or click on the Simulink icon.

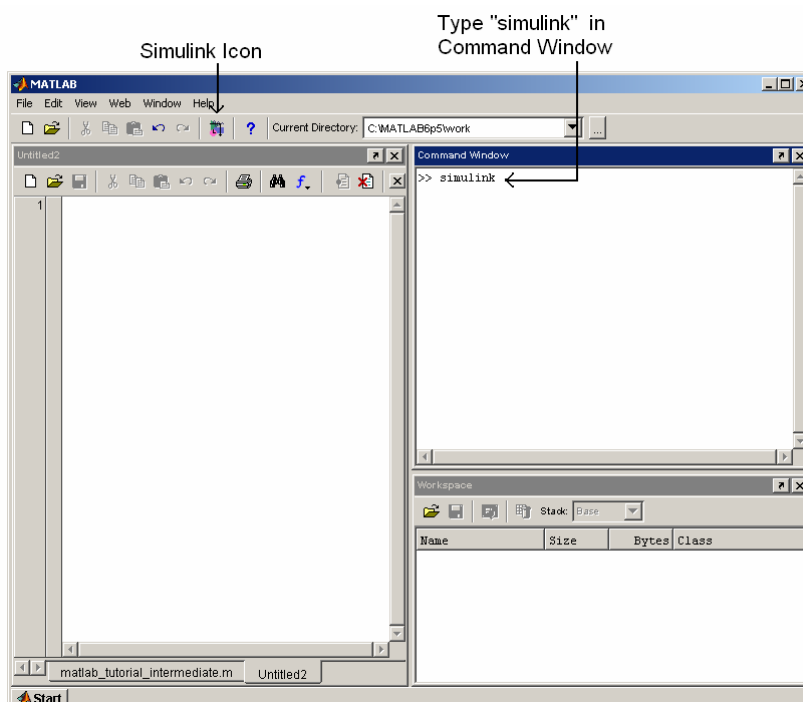


Figure 1: Starting Simulink using icon or Command Window

3. The simulink interface should now appear as shown below in Figure 2.

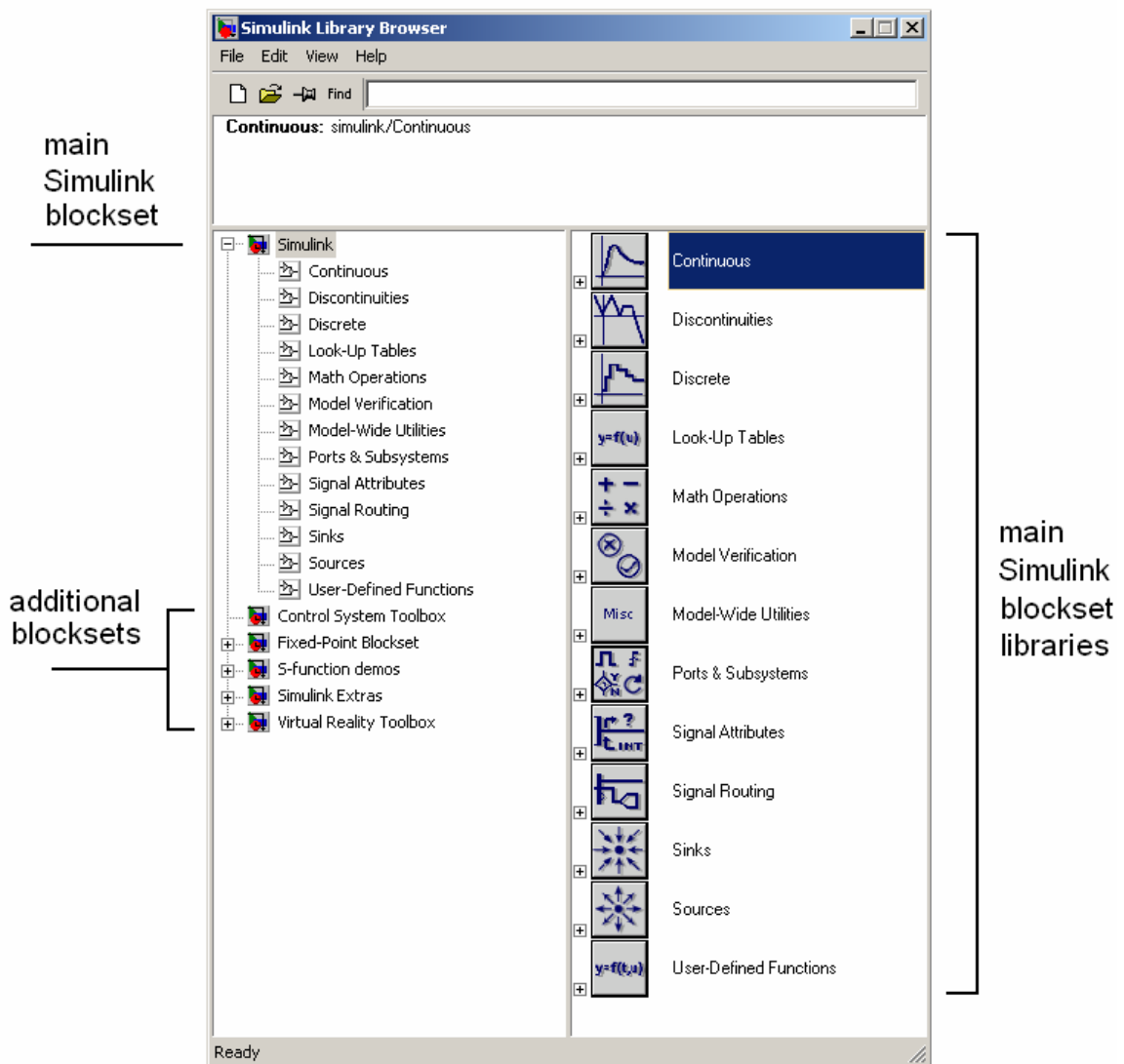


Figure 2: Simulink interface

4. Start a new Simulink model using File > New > Model
5. Let's use Simulink to simulate the response of the Mass/Spring/Damper system described in Intermediate Matlab Tutorial document. Recall that the state space representation of this system was given by

$$\begin{aligned}\dot{\bar{x}} &= A\bar{x} + Bu \\ y &= C\bar{x} + Du\end{aligned}\quad \text{Equation 1}$$

where  $A = \begin{pmatrix} 0 & 1 \\ -k/m & -c/m \end{pmatrix}$   $B = \begin{pmatrix} 0 \\ 1/m \end{pmatrix}$   $C = (1 \ 0)$   $D = 0$

For this problem, let  $k = 2$ ,  $m = 1$ ,  $c = 0.5$ .

Double click on the “Continuous” library from the main Simulink Blockset. This opens library.

6. Click on the “State-Space” block and drag this into your blank model. Your model should now look like Figure 3.

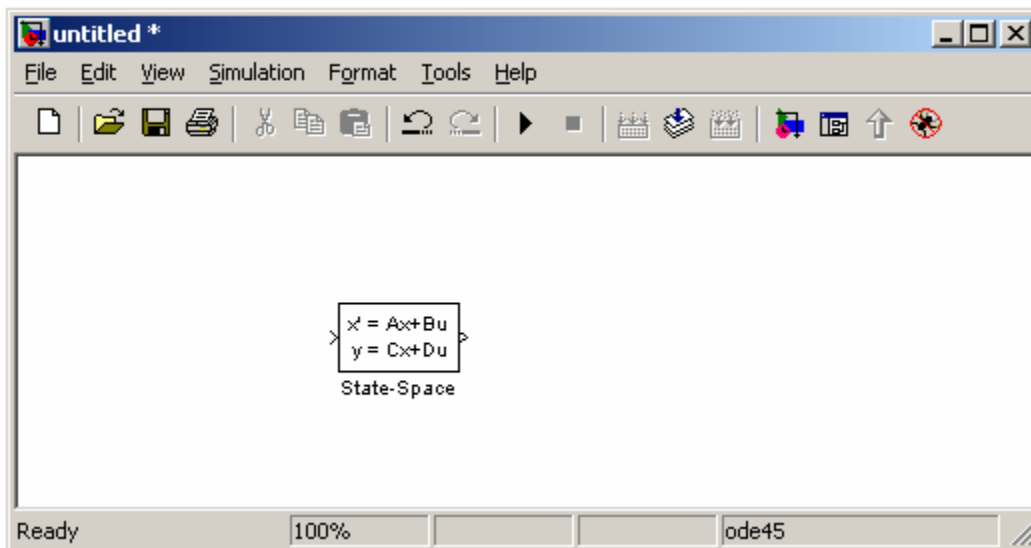


Figure 3: Simulink model with just state space block added

7. We now need to define the parameters of this block. Double click on the block to enter the parameters.

Enter in the A, B, C, and D matrices. Leave the initial conditions as 0. (Note that this is the initial state vector,  $\bar{x}(0)$  and since there are 2 states, 0 actually implies  $\bar{x}(0) = (0 \ 0)^T$ )

8. Now, let’s subject this system to a unit step input which occurs at  $t = 1$  second. Click on “Sources” in the Simulink interface and find the “Step” block. Drag this into the model and connect the output of the step to the input of the state space model (this can be done by clicking on the Step then holding Ctrl and then clicking on the state-space block).
9. We would like to be able to view to output of the system so Click on “Sinks” in the Simulink interface and find the “Scope” block. Drag this into the model and

connect the output of the state-space block to the input of the sink. Your simulink model should now look like

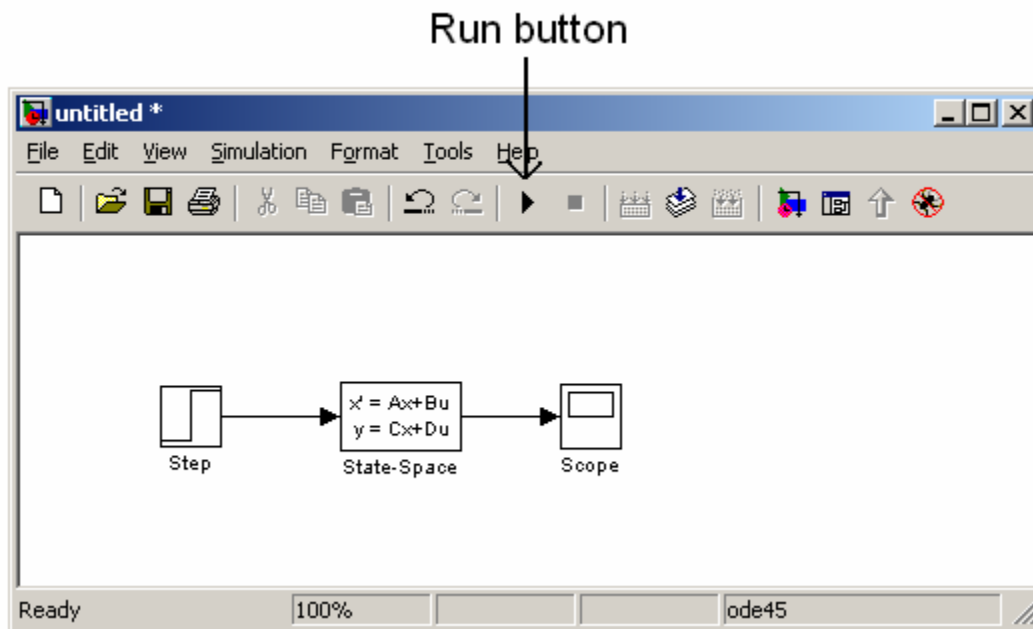


Figure 4: Simulink model with source and sink

10. Save the model as “mass\_spring\_damper\_model.mdl”
11. We can now simulate the system. This can be done many ways as listed below
  - a. Click on the “Run” button
  - b. Go Simulation > Start
  - c. Using the “sim” command from the Command window.

For now, simply use option a or b, we will visit using option c later.

12. We would like to look at the response of the system using the scope. Double click on the scope block to open it up. Autoscale the plot so that you can see the response (the autoscale button looks like a pair of binoculars). You should see something similar to Figure 5.

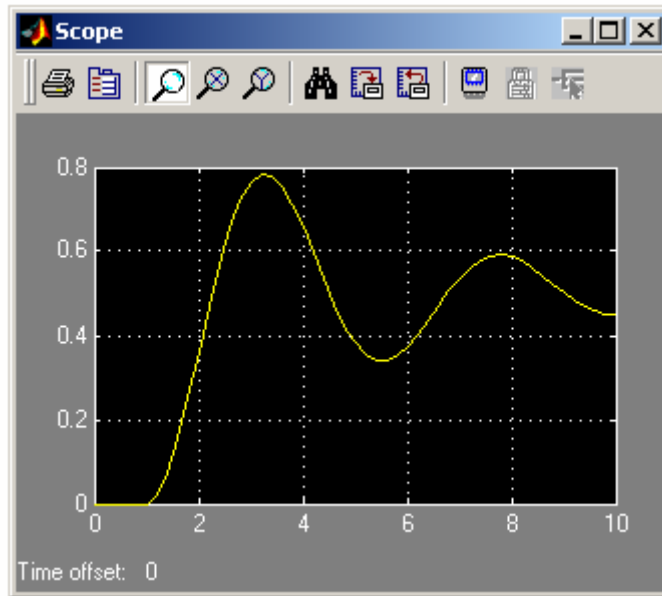


Figure 5: Simulated response of system from scope block

13. Although this is nice for simple analysis, we would like to interface this with Matlab so we can analyze the data using Matlab functions.

Let's analyze how the system response changes if we use different damping coefficients of  $c = 0.5, 1$  and  $1.5$ .

This would be very tedious if we had to change the A matrix each time by hand and then simulate the system and then look at the plot. Therefore, we will use the m-file to write a script which will do this for us.

14. Start a new m-file.
15. Let's first analyze the system response when  $c = 0.5$ . Define the A, B, C, and D matrices in the m-file. A sample code is shown below

```

%-----Part 15-----
%Define the constants
k = 2;
m = 1;
c = 0.5;

%Enter the A, B, C, and D matrices
A = [0 1;
     -k/m -c/m];

B = [0;
     1/m];

C = [1 0];

D = [0];

```

16. We can actually use variables in all the simulink blocks provided that they are defined in the Workspace before the model is run. Now change the parameters of the State-space block to match the matrices that you defined in the m-file. The state space block should look similar to

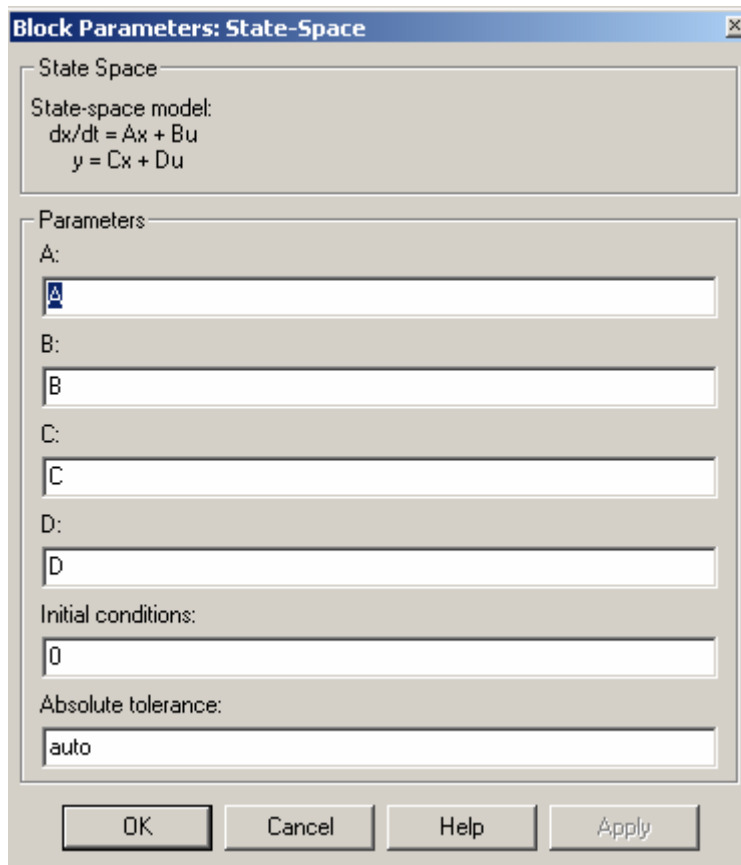


Figure 6: State space block parameters using variables as parameters

17. We need to export the data from simulink to Matlab so that we can plot it. Namely, we would like to see both the input and output of the system. To do this, we use the “To Workspace” block which can be found in the Sinks library. Drag 2 of these blocks into your model and connect them to the input and output (Note: to make a branch of a signal, right click on the signal and then drag to the second connection)
18. We need to modify the parameters of these two blocks slightly. The appropriate parameters are shown below

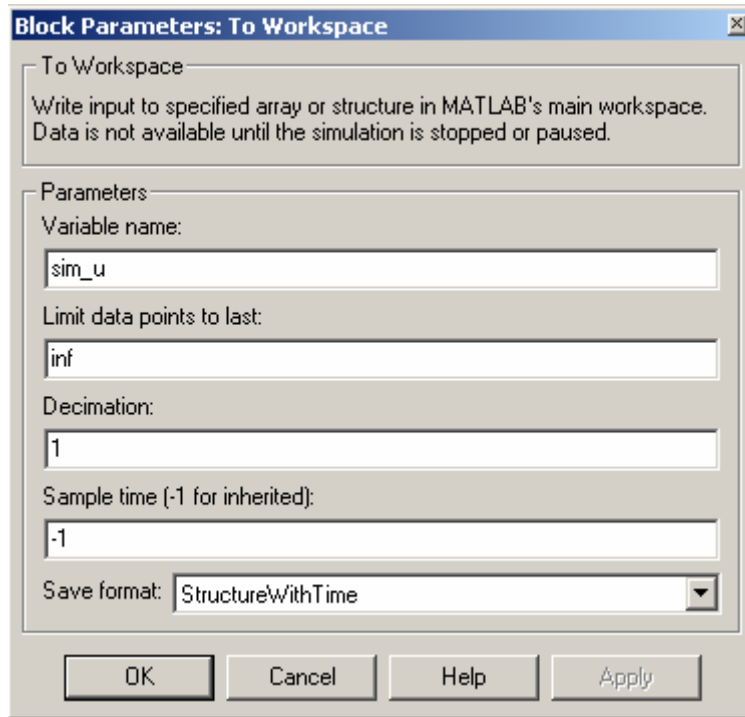


Figure 7: Parameters for the “To Workspace” block

Your simulink model should now look like

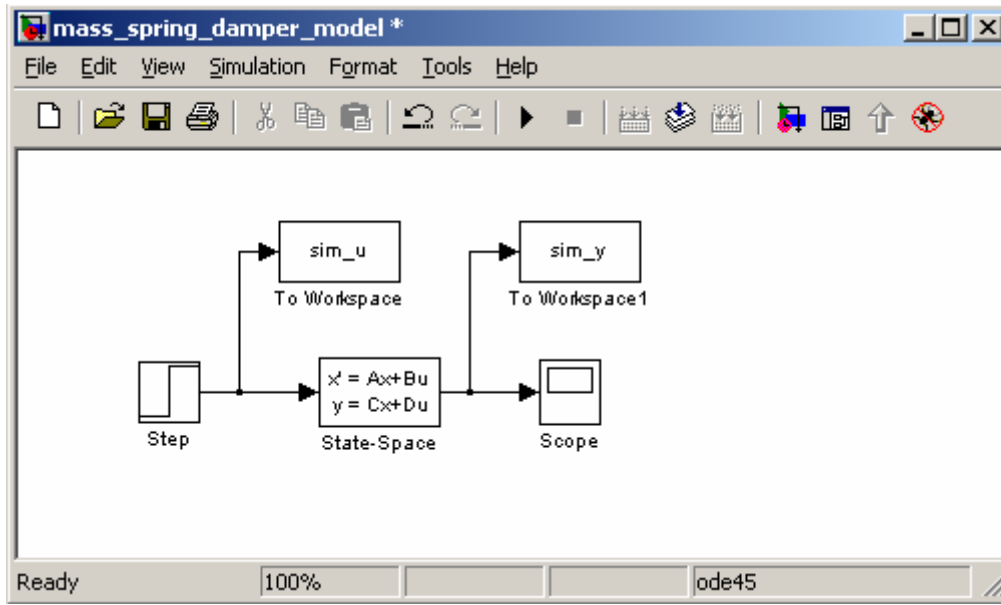


Figure 8: Simulink model with “To Workspace” blocks

19. The simulink model is now setup to export the data to the workspace. When the model is run, three variables will be created in the workspace. They are

Variable Name	Class	Comments
sim_u	struct array	Created by 1 <sup>st</sup> “To Workspace” block
sim_y	struct array	Created by 2 <sup>nd</sup> “To Workspace” block
tout	double array	Automatically created by Simulink when you run a model

20. Write code to run the model from the m-file using the “sim” command. Also, write code to extract the data (namely the input and output of the model). A sample code is shown below.

<i>Functions:</i>	<i>sim</i>
<i>Note:</i>	<p><i>For our purposes, use sim with only 1 argument, the name of the model which you are trying to run</i></p> <pre> %-----Part 20----- %Let's run the model from the m-file using the "sim" command sim('mass_spring_damper_model');  %Now extract the data t = sim_u.time; u = sim_u.signals.values; y = sim_y.signals.values; </pre>

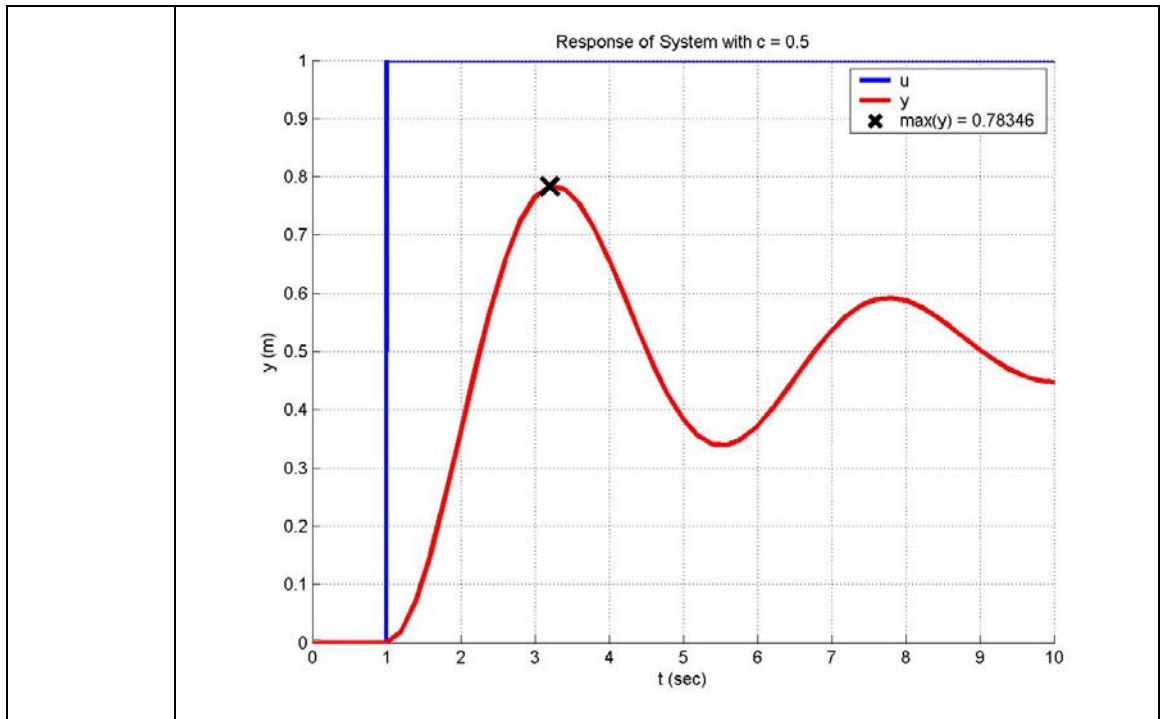


21. Find the maximum response of the system. That is, find  $\max(y)$ . A sample code is shown below

<i>Functions:</i>	<i>max, find,</i>
<i>Note:</i>	<p><i>Here is an example of how to use these functions</i></p> <pre> %-----Part 21----- %Let's find the maximum response of the system max_y = max(y);  %Now that we know max(y), we want to find out where in the vector this %occurs. This means that we want to find the index. max_index = find(y==max_y); </pre>

22. Plot both the input and output of the system on the same graph. Plot the input as a thick blue line and plot the output as a thick red line. Mark the maximum response of the system with a large, thick, black x. Label the plot appropriately. Add a grid and a legend. In the legend, report what the maximum value of  $y(t)$  is. A sample code and the output (after being exported as a .jpg) is shown below

<i>Functions:</i>	<i>figure, hold on, plot, grid, xlabel, ylabel, legend, num2str</i>
<i>Note:</i>	<p><i>The num2str function is one of the most useful plotting functions. Be sure that you understand how to use it.</i></p> <pre> %-----Part 22----- %Now let's plot the data to verify that we have it correct figure hold on plot(t,u,'b-','LineWidth',3) plot(t,y,'r-','LineWidth',3) plot(t(max_index),y(max_index),'kx','MarkerSize',15,'LineWidth',3) grid title('Response of System with c = 0.5') xlabel('t (sec)') ylabel('y (m)') legend('u','y',['max(y) = ',num2str(max_y)]) hold off </pre> <p><i>The output of this code should appear as</i></p>



23. Now let's see how the response of the system changes with different damping coefficients. Simulate and plot the response of the system with three different damping coefficients of  $c = 0.5, 1.0,$  and  $1.5$ . We can use a for loop to make the coding easier. A sample code is shown below.

<i>Functions:</i>	<i>for, length, ...</i>
<i>Note:</i>	<i>The "... " symbol mean continue typing on the next line. This allows a very long line of code be broken up into two separate lines.</i>

```

%-----Part 22-----
%Now let's repeat this process for three different values of c. We would
%like to use a for loop to make this easier. First, we define a vector
%which contains the values of c which we would like to use
c_range = [0.5 1 1.5];

%Start a new figure to plot all the results in (we do this outside the for
%loop because we only want 1 figure, not 3). Also turn the hold on
figure
hold on

%Define a color map which basically defines what different colors we use to
%plot the different responses
color_map = jet(length(c_range));

for n = 1:length(c_range)
    %Obtain the current value of c
    c = c_range(n);

    %Redefine the A, B, C, and D matrices with the new value of c
    A = [0 1;
         -k/m -c/m];

    B = [0;
         1/m];

    C = [1 0];

    D = [0];

    %Run the Simulink model with the new values of A, B, C, and D
    sim('mass_spring_damper_model');

    %Extract the data
    t = sim_u.time;
    u = sim_u.signals.values;
    y = sim_y.signals.values;

    %Find the max of y and the index where this occurs

```

```

max_y = max(y);
max_index = find(y==max_y);

%Also we want to store this data so we can use it in the legend later
max_y_data(n) = max_y;

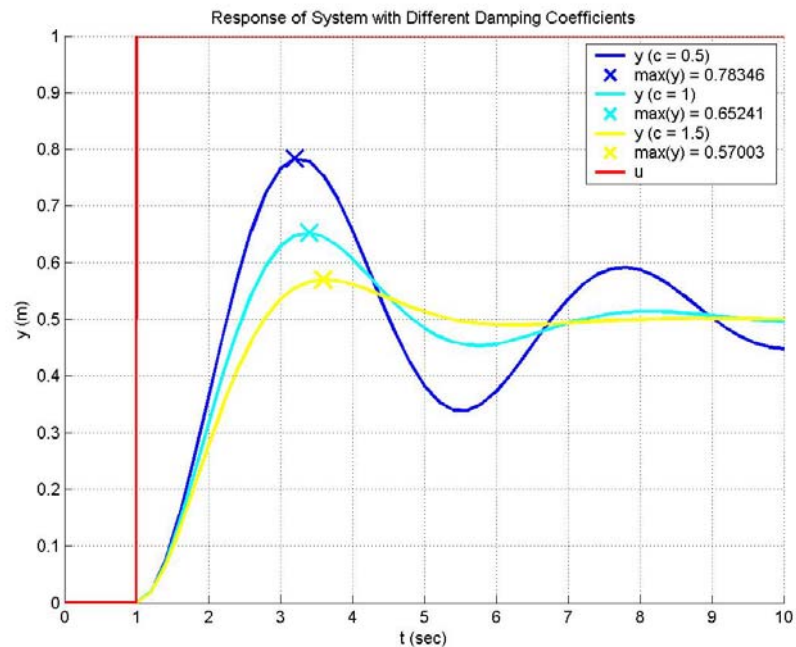
%Plot the new data
plot(t,y,'LineWidth',2,'color',color_map(n,:))
plot(t(max_index),y(max_index),'x','color',color_map(n,:),...
      'MarkerSize',15,'LineWidth',2)
end

%Plot the input only once
plot(t,u,'r-','LineWidth',2)

%Label the figure
title('Response of System with Different Damping Coefficients')
xlabel('t (sec)')
ylabel('y (m)')
legend(...
      ['y (c = ',num2str(c_range(1)),')'], ['max(y) = ',num2str(max_y_data(1))],...
      ['y (c = ',num2str(c_range(2)),')'], ['max(y) = ',num2str(max_y_data(2))],...
      ['y (c = ',num2str(c_range(3)),')'], ['max(y) = ',num2str(max_y_data(3))],...
      'u')
grid
hold off

```

*The output should look like*



---

Version History: 09/28/04: Created:  
11/23/05: Updated: made this format match other to-do documents  
and removed references to AA547.  
12/01/05: Updated: changed header to match how-to template  
12/09/05: Updated: Made changes to layout and added footer.  
10/31/06: Updated: Fixed typo of "Tout" to "tout"  
11/01/06: Updated: Touched up some of the graphics.