

# **E105: Simulink Tutorial**

Andrew C. Smith

07/23/2007

# 1 Introduction

## 1.1 General description

<sup>1</sup> Simulink is a platform for multidomain simulation and Model-Based Design of dynamic systems. It provides an interactive graphical environment and a customizable set of block libraries that let you accurately design, simulate, implement, and test control, signal processing, communications, and other time-varying systems.

Add-on products extend the Simulink environment with tools for specific modeling and design tasks and for code generation, algorithm implementation, test, and verification.

Simulink is integrated with MATLAB, providing immediate access to an extensive range of tools for algorithm development, data visualization, data analysis and access, and numerical computation.

### Key Features

- Extensive and expandable libraries of predefined blocks
- Interactive graphical editor for assembling and managing intuitive block diagrams
- Ability to manage complex designs by segmenting models into hierarchies of design components
- Model Explorer to navigate, create, configure, and search all signals, parameters, and properties of your model
- Ability to interface with other simulation programs and incorporate hand-written code, including MATLAB algorithms
- Option to run fixed- or variable-step simulations of time-varying systems interactively or through batch simulation
- Functions for interactively defining inputs and viewing outputs to evaluate model behavior
- Graphical debugger to examine simulation results and diagnose unexpected behavior in your design

---

<sup>1</sup>The MathWorks 2007

- Full access to MATLAB for analyzing and visualizing data , developing graphical user interfaces, and creating model data and parameters
- Model analysis and diagnostics tools to ensure model consistency and identify modeling errors

## 1.2 Why are WE using it?

Simulink allows us to rapidly develop and simulate both linear and nonlinear systems. Control logic can be tested and evaluated without the need to do time consuming hardware experiments. For the purposes of this class we are giving you an introduction into the use of Simulink and hopefully convey the practicality and usefulness of this essential tool. This tutorial will show you the basics required for creating a new model and doing some analysis. By the end of the tutorial you will have created the model required for homework 4.

## 2 Getting started

This section goes through opening Simulink and familiarizing yourself with the Simulink environment and tools. There are many elements of Simulink that we won't cover in this tutorial, please browse around and discover all the cool tools.

### 2.1 Opening Simulink

Simulink requires that MATLAB is open. So to begin:

1. Open MATLAB
2. Click **File**, **New**, then **Model**

You should now have a Simulink model window open, see Figure 1. This is where all the work is done. Blocks are dragged into this window and connections made. Take a few minutes to go through the menus and hover over some of the buttons to see what they do. A couple of worthy items to note are the **Simulation stop time** and the **Start/Stop**

**simulation** buttons. The simulation stop time indicates how long (in seconds) the simulation will be run. The start/stop simulation buttons do what you'd think. Make sure you locate these items.

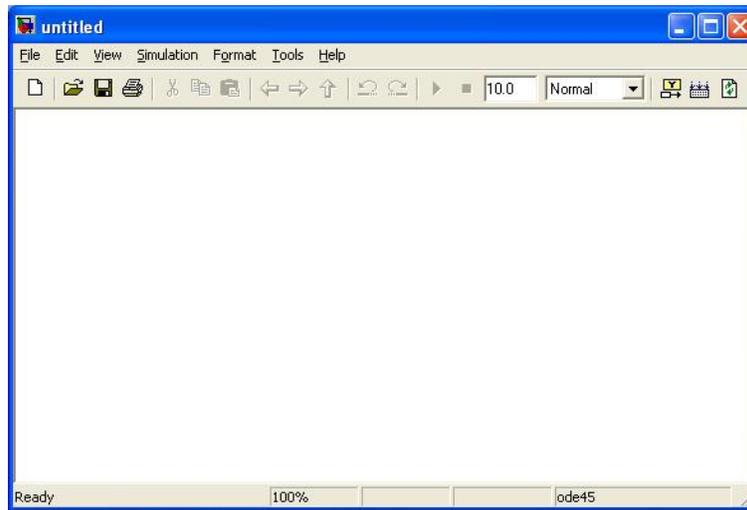


Figure 1: New Simulink model

Now let's open up the **Simulink Library Browser**. This window contains all the blocks that are used in Simulink. To put a block in your Simulink model you need to drag it from this library.

1. Click **View**
2. Select **Library Browser**

The **Simulink Library Browser** should now be visible in your screen, see Figure 2. If you scroll down the left frame you'll see several expandable nodes, each representing different packages and blocksets that have been purchased on your version of MATLAB. Explore these different sets and see what is available. For this tutorial and this homework we will confine ourselves to the top node, the **Simulink** node, shown in Figure 2. Have a look through all the elements in this set. As you click on an item in the left frame, scroll around in the right plane to see all the blocks that are contained in this set. Pay particular attention to the **Continuous**, **Math Operations**, **Sinks** and **Sources**. Try and locate the block for an integrator. Where do you think this type of function

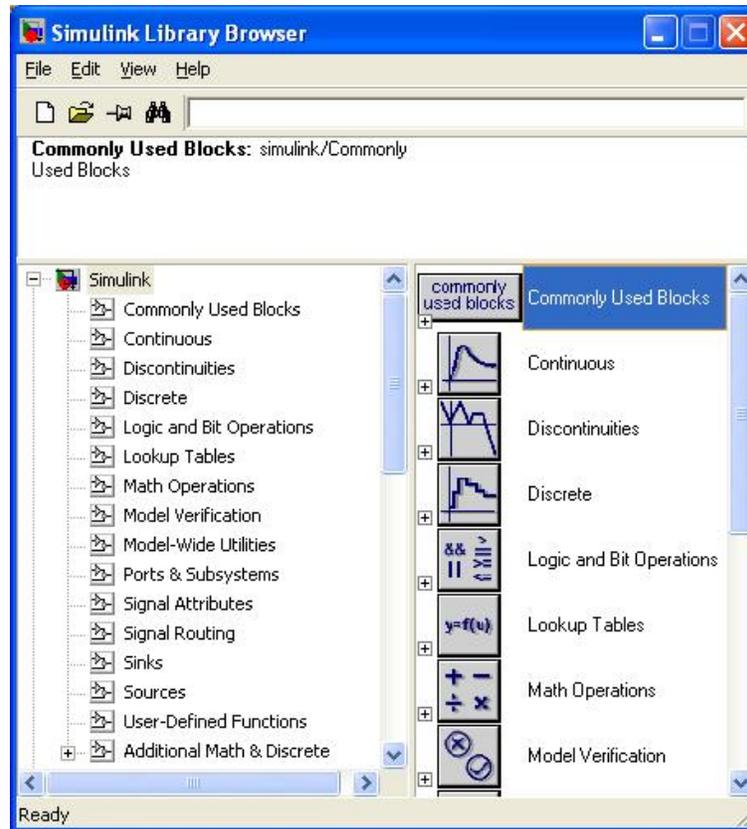


Figure 2: Simulink Library Browser

would live? Hint, this is a **Continuous** function. Next try and find the block for a step. If you recall, a unit step is one of our **Sources** for input. Almost there, locate the block for a gain. I don't think I need to remind you that a gain is a **Math Operation**. Finally, find the scope block, this is usually where you want your outputs to **Sink** to. Pictures of these three blocks are shown in Figure 3.

## 2.2 Building your first model

Since you have found these blocks, put one of each into your model window. This can be done using one of the following methods:

- Highlight the block and click **Edit, Add to the current model**

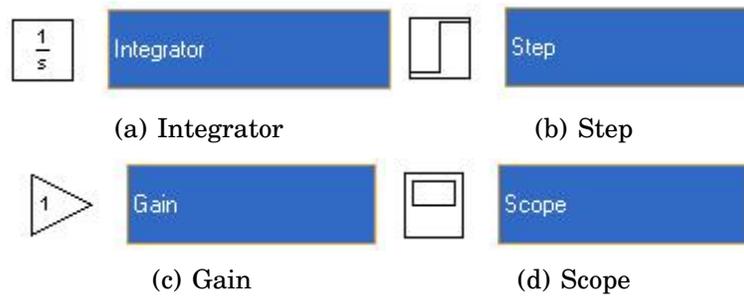


Figure 3: Common Simulink blocks

- Right click on the block and click **Add to untitled**, note that untitled will change to whatever name you save it to later.
- Highlight the block and click, hold, drag the block onto your model window and drop.

Once you've done this for the **Step**, **Integrator**, **Gain** and **Scope**, rearrange the placement of your blocks so your window looks like Figure 4. Be sure to have the gain block placed well below the other blocks, this

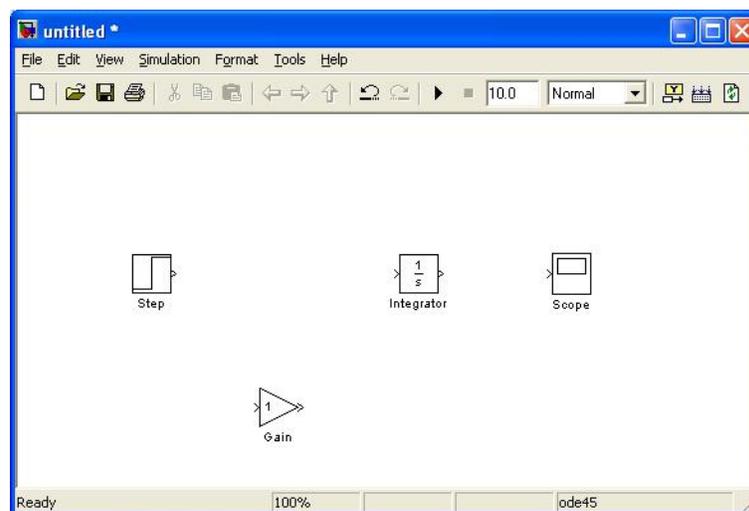


Figure 4: Model window with some basic blocks

is needed to show a trick with connections.

We now have a bunch of blocks in our model but blocks by themselves won't do us any good, we need to connect them to indicate the flow of information. Blocks can be connected in three ways:

1. Moving the cursor overtop of the input/output of a block. The cursor will change from a pointer to a crosshair. Click and drag a line from the input/output of one block to the output/input of another block. When you've reached the other connection point the crosshair will change to a double crosshair, release the mouse button.
2. Click on one block, then while that block is highlighted, hold down **CTRL** and click on the block you would like to connect it to.
3. If a connection already exists between two blocks, you can drag a new block in between them to make a connection to both blocks from the new one.

Let's give this a try on our model. Use method 1 to connect your **Step** and **Integrator** blocks. Remember not to click and drag until you see the crosshairs and don't release the mouse until you see a double crosshair. If you mess up, don't worry you can always click on the line and hit delete to remove it. If you let go of the mouse button too early and the line didn't make it to the next block, you can click on the open end of the line and continue dragging that to your desired point.

Now you've got one connection, good stuff. Let's try connecting our **Integrator** with our **Scope**. To do this we'll use method 2. This is a really quick way to make connections. Click on the **Integrator** block so it's selected, you don't need to click and hold. Now press down and hold the **CTRL** button. With the **CTRL** button depressed, click on the **Scope** block. You should see a line now connecting your **Integrator** and **Scope** blocks. Your model should now look similar to Figure 5.

Ok, so we've got one final connection to make, we need to place the **Gain** block in between our **Integrator** and **Scope**. You could just delete the current connection and make two new connections. Instead, let's try connection method 3 from above. To do this, click, hold and drag the **Gain** block over top of the present connection between the **Integrator** and **Scope**. Once you are over top of the connection, release the mouse button and your **Gain** block should be inline between the **Integrator** and **Scope**.

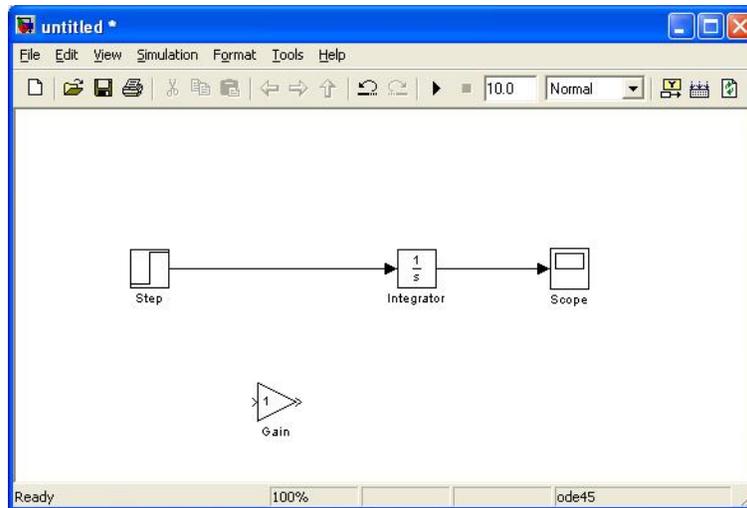


Figure 5: Simple model with 4 blocks and 2 connections

Very cool, we now have a full model. The next step is to alter our model blocks to suit our simulation. The following steps will change our model to match a particular dynamical system.

1. Double click on the **Step** block. A new dialog box will popup. In the field marked **Final value**, change the value from 1 to **2**. Also, have a look at the other parameters of the step block.
2. Double click on the **Gain** block. Change the field marked **Gain** from 1 to **2**.

### 2.3 Simulating the system

Now that we have our block all ready to go, it's time we simulated it and see what the output is. Follow these steps to run this system and plot the output:

1. Confirm that your simulation is set to run for a duration of **10.0** seconds. This duration is set in the **Simulation stop time**, found near the top of the window.
2. Click on the **Start simulation** button, located to the left of the **Simulation stop time**. Alternatively, you could use the menu,

**Simulation then Start.**

3. Your simulation is complete, double click on the **Scope** block. This will bring up a plot of the output. Right click on the the plot background and select **Autoscale** so you can see the entire plot. You should see the plot given in Figure 6.

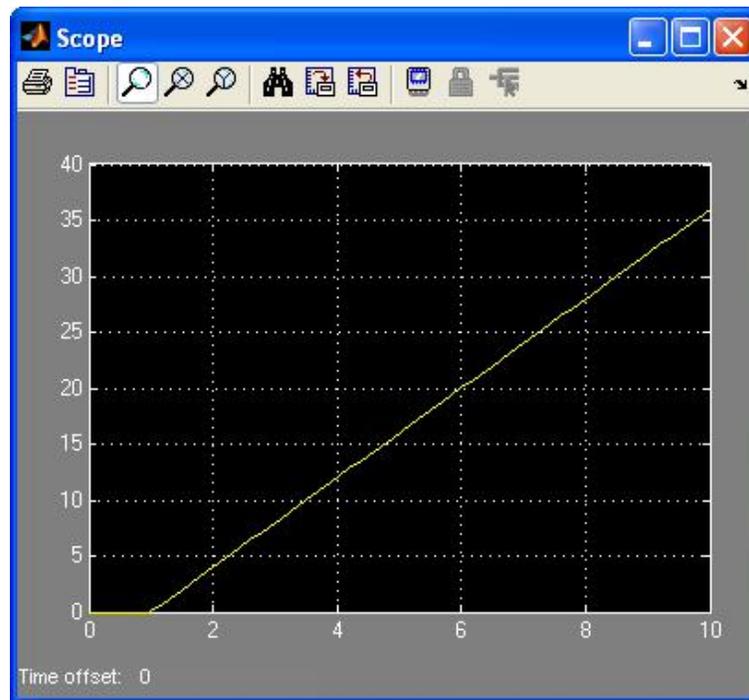


Figure 6: Scope plot of our initial system

This is pretty cool, since we are just applying an integrator and a gain to a step input, it makes sense that our output is a ramp. Now we're done with the basics of Simulink. The next section will go through a few more elements that you'll need to know and along the way we will have constructed the model needed for the next homework.

### 3 Homework model

Now that you are completely familiar with the Simulink model window and the Simulink Library Browser, it's time we built our first control

system. We are going to construct the model required for your next homework. We have several parts from it already and now just need to add a few details. We are asked to simulate the system

$$G(s) = \frac{\text{Plant}}{s} = \frac{1}{s}$$
$$D(s) = \frac{\text{Nominal PI controller}}{s} = \frac{k_p s + k_I}{s}$$

and the system is in unity feedback. As well, to simulate the saturation of the integral controller, we'll need to add a saturation block. So to complete your model, follow these steps

1. Remove all of your existing connections, so we're back with just our four blocks.
2. Space all your blocks out a bit, so we've got some room to put stuff in between them. To do this you can just click and drag the blocks around the window.
3. Double click on your **Step** block, change the final value back to 1.
4. In most systems we would need to add a **Transfer Fcn** block for the plant dynamics. This block can be found in the **Continuous** group of blocks and is shown in Figure 7(a). However, since our plant is just a  $\frac{1}{s}$  plant, we can just use our integrator block. Make a copy of your **Integrator** block by holding down the **CTRL** button, clicking and dragging the **Integrator** block, then release. This should place a copy of the integrator in your model. Place this integrator block to the right of the other one and to the left of the scope.
5. Find the **Saturation** block, located in the **Discontinuities** set and shown in Figure 7(b). Put this block between the two **Integrator** blocks.
6. Double click on the **Saturation** block. Change the **Upper limit** and **Lower limit** fields to **1** and **-1**, respectively. Click **OK**.

7. As you'll notice we have already implemented part of our controller, namely  $\frac{k_I}{s}$  using the **Gain** and **Integrator** blocks. Change the value of the gain block to **4**. Now we need to add the proportional part. Make a copy of the gain block (either by dragging in a new one, CTRL dragging the original one or copy and pasting), place it above the other one. Change the value of the gain to **2**.

At this point you should have a model looking like Figure 8.

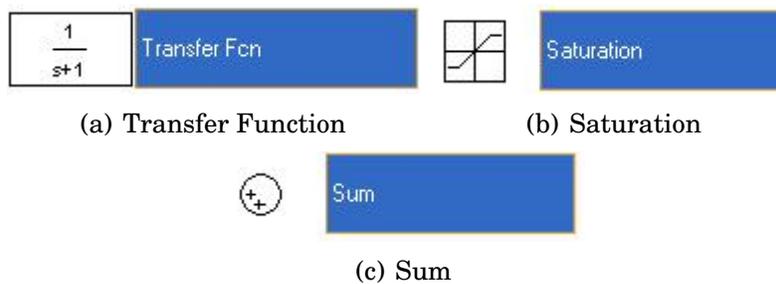


Figure 7: Homework Simulink blocks

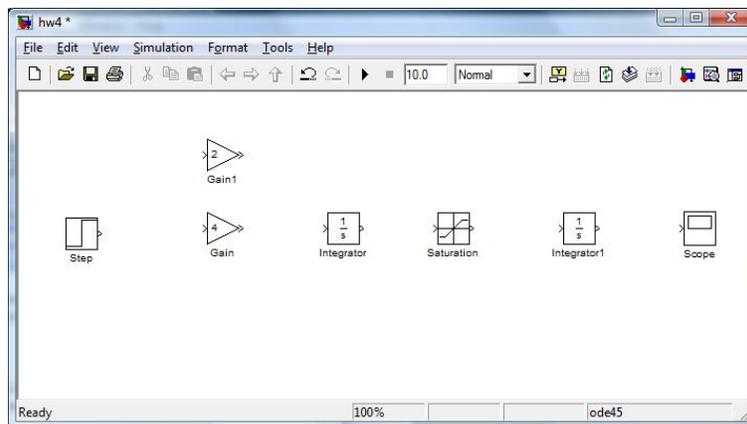


Figure 8: Homework model blocks

Now all we have left to add is the connections and the antiwindup part.

1. As mentioned above, this system is in unity feedback with an integral control. Therefore, to construct the feedback we'll need to

add a **Sum** block, which is located in the **Math Operations** set and shown in Figure 7(c). Drag this block in between your **Step** and **Gain** block.

2. As you'll notice the **Sum** block has positive signs on both inputs. We would like to have a negative sign on the feedback. To do this, double click on the **Sum** block, change the field **List of signs** from  $|++$  to  $|+-$ . After you hit **OK**, you should see a negative sign on the bottom of the **Sum** block.
3. We need to add three more of these. Make three copies of the **Sum** block, drag them to the appropriate places shown in Figure 9.22 in FPE.
4. Double click on the **Sum** block after the first integrator, change the **List of signs** to  $++|$ . This will put the + signs and the output in the appropriate places.
5. Double click on the bottom **Sum** block. Again change the **List of signs** to  $+ - |$ . You'll notice that the block looks almost right, except it's backwards. Right click on the block, scroll down to **Format** on the menu that pops up, then click **Flip Block** on the next menu that comes out. Now you should see that your block is facing the right way.
6. Now let's add the antiwindup gain. Copy one of the other gain blocks and place it the the left of the bottom sum block. Change the gain value to 10. Flip the block using the same procedure outlined above.
7. Let's setup our scopes. Change the name of the existing scope by clicking on the word **Scope**, change it to **Output**. Copy this scope to a location above it and change the name of it to **Control**.
8. To finish the model off, connect all the items together in a row, using whatever connection method you like.
9. To connect the  $k_p$  gain block, you'll need to start a connection on the  $k_p$  gain block, click and drag to a location between the two sum blocks, release when the single crosshair goes to a double

crosshair. Connect the output of the  $k_p$  block to the positive summer. Having ample amount of space between your sum blocks helps here.

10. Do similar steps to connect the sum and  $K_a$  blocks.
11. Connect the negative terminal on the first **Sum** block to the output of the **Transfer Fcn**. Do this using the connection method 1 described above and reviewed again here.
  - (a) Hover over the negative terminal of the **Sum** block. You will see your cursor change to a crosshair. Draw a line straight down from the negative terminal, doesn't matter how long it is and release the mouse button.
  - (b) You will now have a dashed line going into the **Sum** block. Hover over the free end and again you'll see a crosshair. Click and drag a line up to a point between the last **Integrator** and **Output (Scope)** blocks. You'll notice that this line is not drawn straight, but rather as two perpendicular lines, this looks much better. While still dragging, hover over the connection between the **Integrator** and **Output** blocks. You will see your cursor change to a double crosshair. Release your mouse button and you will see that this has become a pick off point for your feedback.
12. Finally, connect the **Control (Scope)** block to a point between the **Saturation** and last **Integrator** block.

Your initial model is now complete. Select **File, Save** and save with an appropriate file name. If you followed the directions properly, you should be left with a model that looks like Figure 9. Give the simulation a test run, open up the **Output** and you should see a plot like Figure 10.

When it comes time for comparing results, there are several ways to represent and plot the data. One very common way to plot several outputs at the same time is to multiplex (merge) the signals. This is done with the **Mux** block, shown in Figure 11(a), found in the **Signal Routing** set. Place this block in front of the **Scope** block and add in as many signals as you'd like, all of which will be plotted in the **Scope**. Alternatively, you can output signals to the workspace using

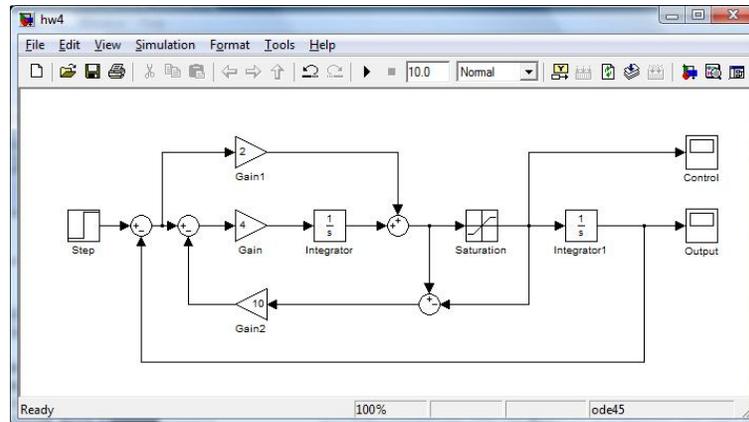


Figure 9: Initial homework Simulink model

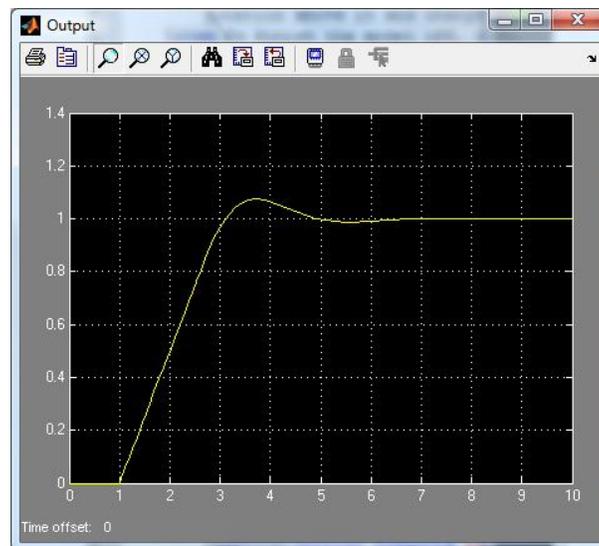
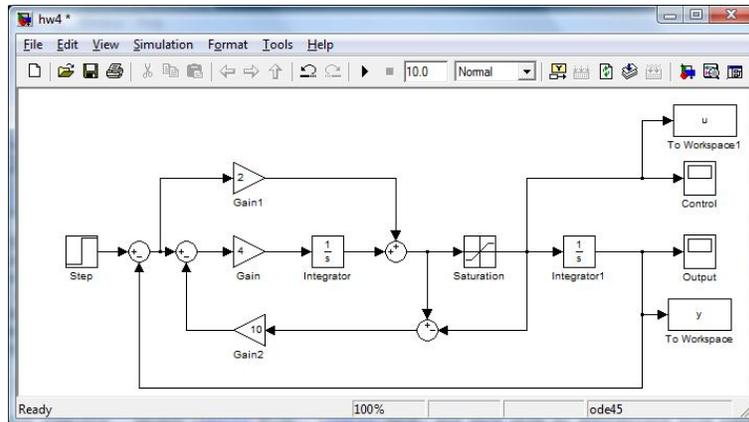


Figure 10: Initial homework Simulink model plot

the **To Workspace** block, see Figure 11(b), found in the **Sinks** set. Connect one of these to each of the outputs you'd like to send to the workspace. For ease of use, it's best to change the **Save format** in the **To Workspace** block to **Array**. This will make it easier to work with in the workspace. As well, it's best if you change the **Variable name** to something more appropriate. An example of using of the **To**

**Workspace** block is shown in Figure 11(c).



(c) To Workspace example

Figure 11: Helpful homework Simulink blocks

Simulink is a very powerful tool, the more exposure and practice you have with it will help you along in your controls career. We encourage you to play around, explore and add another controls tool to your belt, good luck.