

## 3

## Quick Start

In this chapter, we will discuss the basics of building and executing Simulink models. We will start with a simple first-order system. Next, we will build a more complex model that includes feedback and illustrates several important procedures in Simulink programming. Finally, we will discuss the Simulink Help system.

## 3.1 INTRODUCTION

Simulink is a powerful programming language, and a big part of that power is its ease of use. In this chapter, we will introduce Simulink programming by building and executing two simple models. Our purpose here is to cover the basics of model building and execution. We will discuss these topics in more detail in later chapters.

The examples shown in this chapter (and the rest of the book) were produced using Simulink 5.0 and MATLAB 6.5 in the Microsoft Windows XP environment. There are very few differences between using Simulink in the Windows XP environment and using it in the X-Windows environment; we will point out those differences where necessary.

## 3.1.1 Typographical Conventions

Before we build the first model, we need to establish some typographical conventions:

**Computer Type.** All computer input, output, variable names, and command names are shown in monospaced sans-serif font.

**Using Menus.** The Simulink user interface employs pull-down menus located on a menu bar at the top of the Simulink model window. To facilitate discussion of the various menu choices, we will use the convention: **Menu bar choice:Pull-down menu choice**. For example, choosing “File” followed by “Save As” (as shown in Figure 3.1) will be written **File:Save As**.

**Dialog Box Fields.** Simulink makes extensive use of dialog boxes to set simulation parameters and to configure blocks. Dialog box fields are indicated in **bold type**. For example, choosing **File:Save As** opens the dialog box shown in Figure 3.2. The fields in this dialog box are **Save in**, **File name**, and **Save as type**.

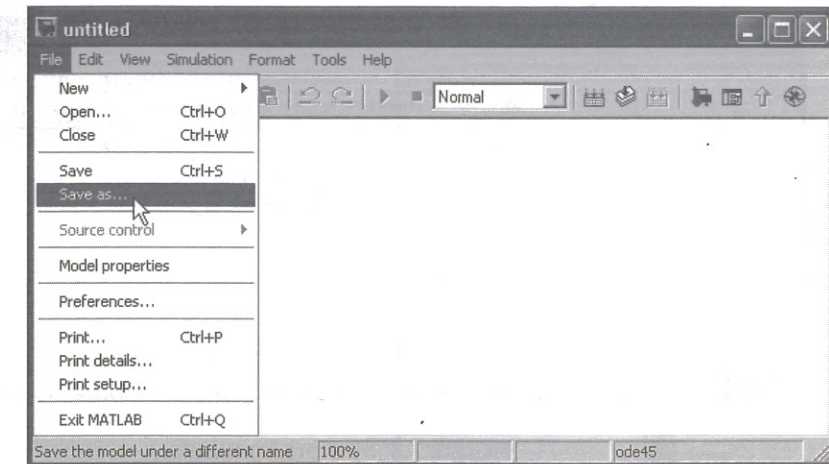


FIGURE 3.1: Menu selection

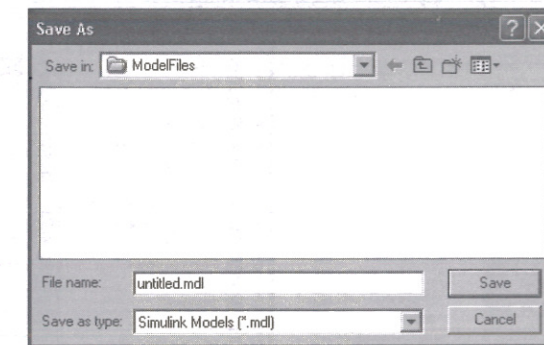


FIGURE 3.2: File:Save As dialog box

## 3.2 BUILDING A SIMPLE MODEL

Let's start by building a Simulink model that solves the differential equation

$$\dot{x} = \sin(t) \quad (3.1)$$

with initial condition  $x(0) = 0$ .

Simulink is an extension of MATLAB, and it must be invoked from within MATLAB. Start Simulink by clicking the Simulink icon on the MATLAB toolbar (Microsoft Windows), as shown in Figure 3.3, or by entering the command `Simulink` at the MATLAB prompt. On an X-Windows system, enter the command `Simulink` at the MATLAB prompt.

The Simulink library browser, shown in Figure 3.4, will open. Click the New Window icon, opening an empty model window, as shown in Figure 3.5. It is this empty model window, initially named `untitled`, in which you will build the Simulink model.



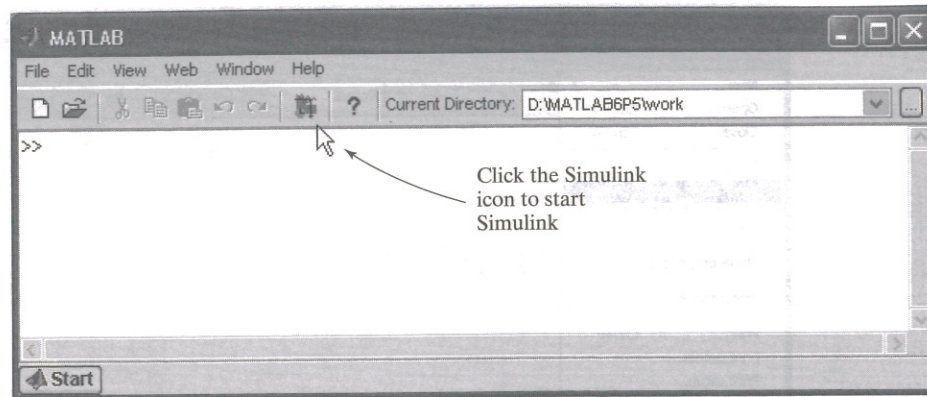


FIGURE 3.3: Starting Simulink

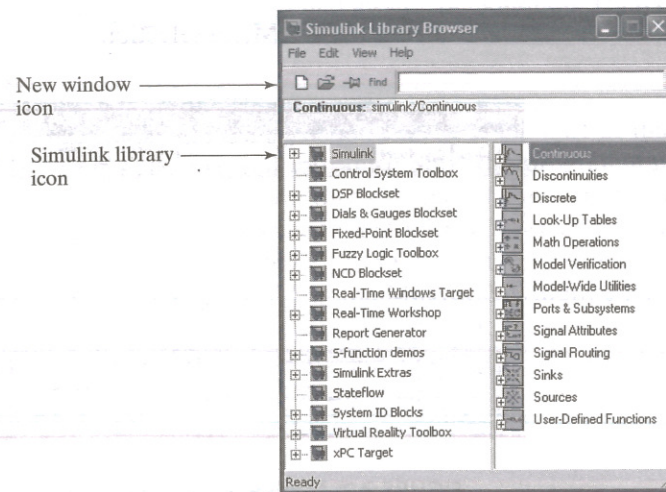


FIGURE 3.4: Simulink Library Browser window

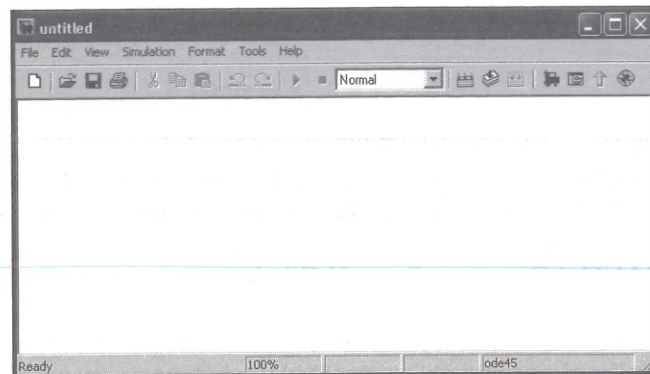
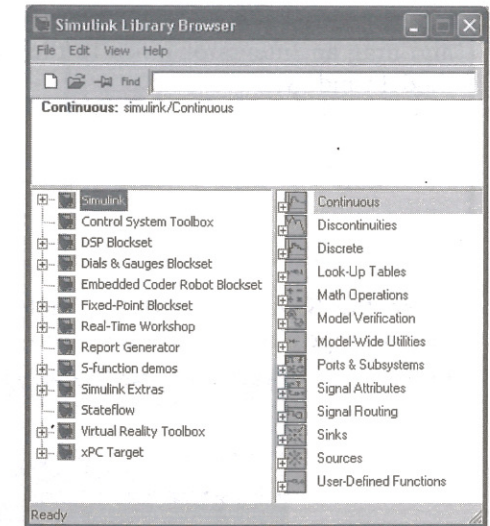
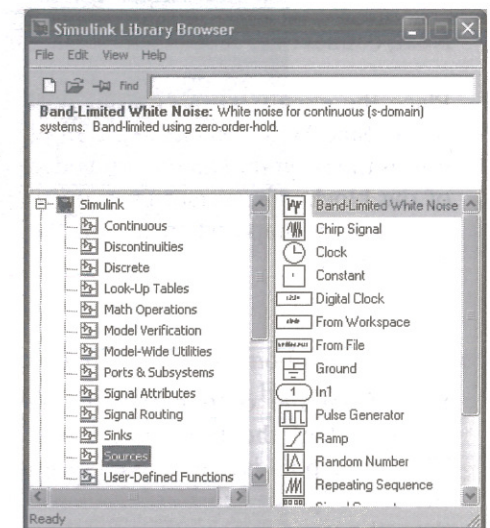


FIGURE 3.5: Empty model window

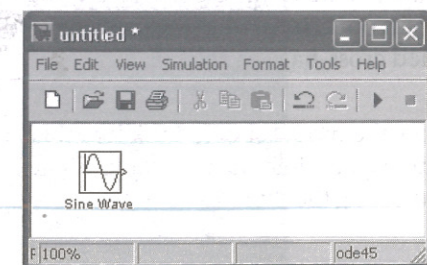
Double-click the Simulink library icon, or click the +, which expands the Simulink library as shown. Notice that the right pane contains a list of sublibraries.



Select the Sources library to open the library in the right pane.

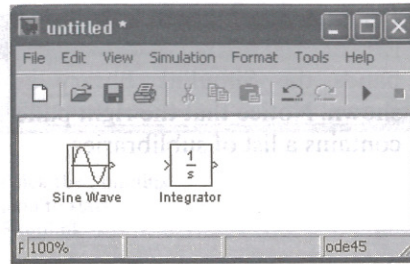


Drag the Sine Wave block from the Sources block library to the model window, positioning it as shown. A copy of the block is placed in the model window. Note that, in the figure, we resized the model window to save space.

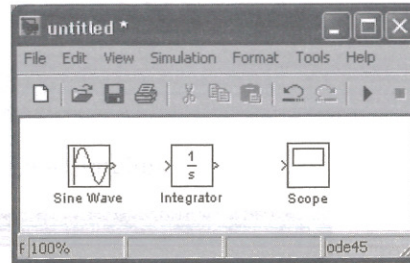




Open the Continuous block library, and drag an Integrator block to the model window.

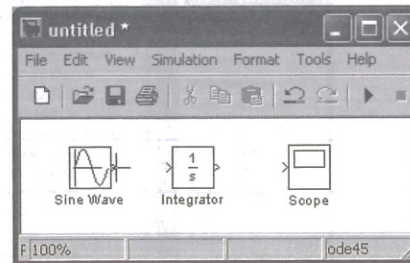


Open the Sinks block library, and drag a Scope block to the model window.

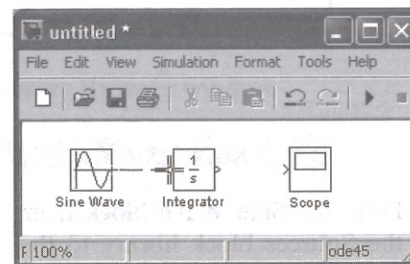


Next, connect the blocks with signal lines to complete the model.

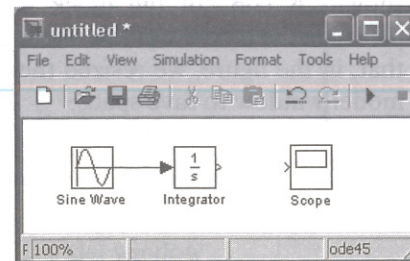
Place the cursor on the output port of the Sine Wave block. The output port is the > symbol on the right edge of the block. The cursor changes to a cross-hair shape when it's on the output port.



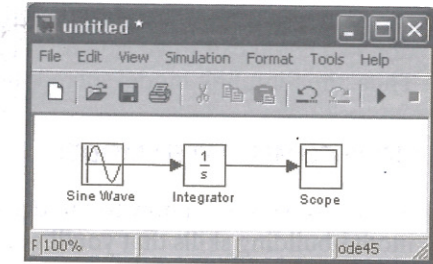
Drag from the output port to the input port of the Integrator block. The input port is the > symbol on the left edge of the block. As you drag, the cursor retains the cross-hair shape. When the cursor is on the input port, it changes to a double-lined cross-hair, as shown.



Now the model should look like this. The signal line has an arrowhead indicating the direction of signal flow.



Draw another signal line from the Integrator output port to the Scope input port, completing the model.



Double-click the Scope block, opening a Scope window, as shown in Figure 3.6.

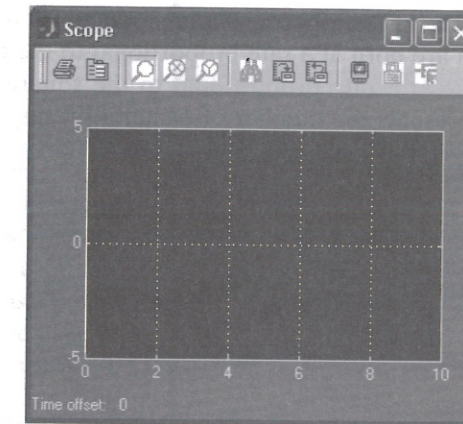


FIGURE 3.6: Scope window

Choose **Simulation:Start** from the menu bar in the model window. The simulation will execute, resulting in the scope display shown in Figure 3.7.

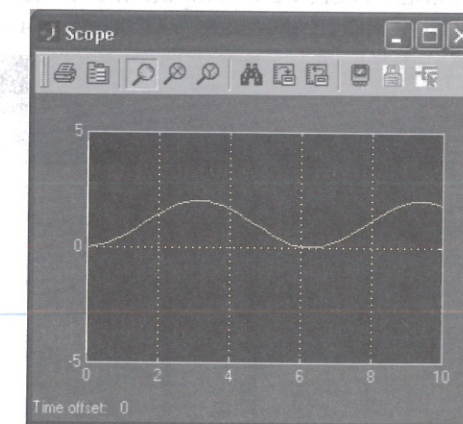


FIGURE 3.7: Scope display after executing the model



To verify that the plot shown in Figure 3.7 represents the solution to Equation (3.1), you can solve Equation (3.1) analytically, with the result  $x(t) = 1 - \cos t$ . Thus, the Simulink model solved the differential equation correctly.

### 3.3 A MORE COMPLICATED MODEL

So far, we've shown how to build a simple model. There are a number of additional model-building skills that you'll need to acquire. In this section, we use a model of a biological process to illustrate several additional skills: branching from signal lines, routing signal lines in segments, flipping blocks, configuring blocks, and configuring the simulation parameters.

Scheinerman [1] described a simple model of bacteria growth in a jar. Assume that the bacteria are born at a rate proportional to the number of bacteria present, and that they die at a rate proportional to the square of the number of bacteria present. If  $x$  represents the number of bacteria present, then the bacteria are born at the rate

$$\text{birth rate} = bx$$

and die at the rate

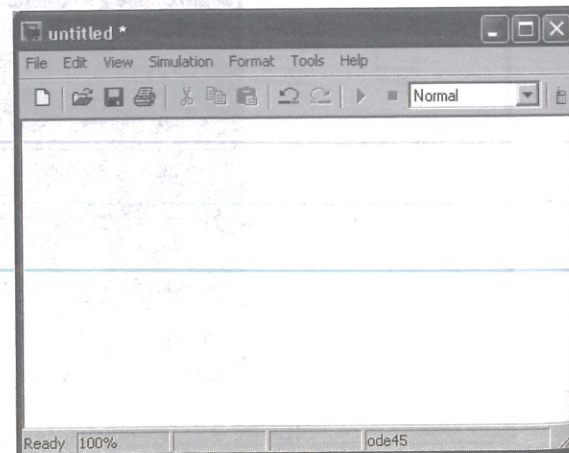
$$\text{death rate} = px^2$$

The total rate of change of bacteria population is the difference between birth rate and death rate. This system can therefore be described with the differential equation

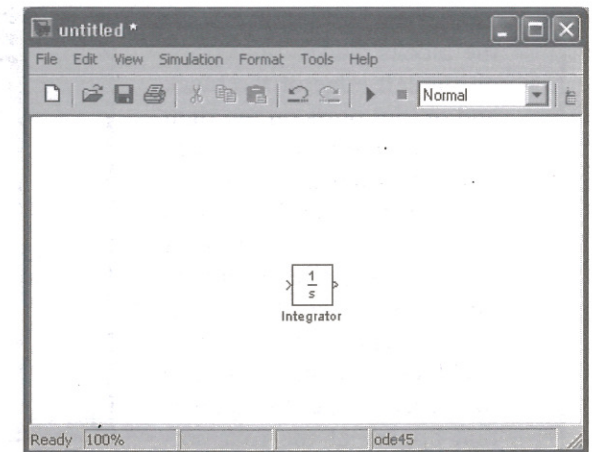
$$\dot{x} = bx - px^2 \tag{3.2}$$

Let's build a model of this dynamical system assuming that  $b = 1/\text{hour}$  and  $p = 0.5/\text{bacteria-hour}$ . Then, we'll compute the number of bacteria in the jar after 1 hour, assuming that initially there are 100 bacteria present.

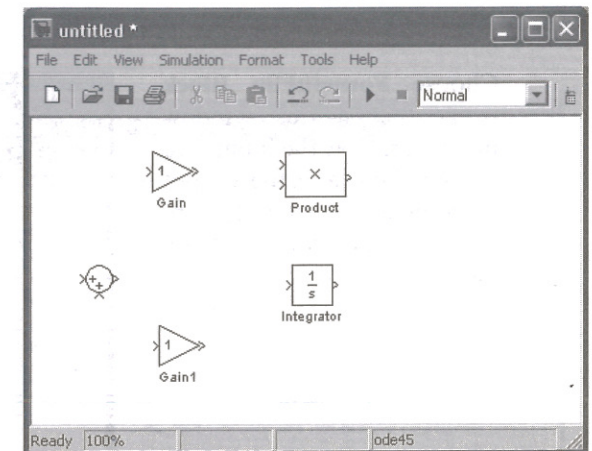
Open a new model window as discussed earlier.



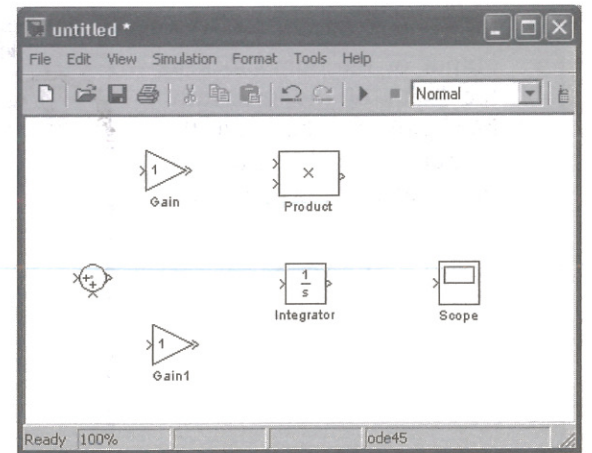
This is a first-order system, so one integrator is required to solve the differential equation. The input to the integrator is  $\dot{x}$ , and the output is  $x$ . Open the Continuous block library and drag the integrator block to the position shown.



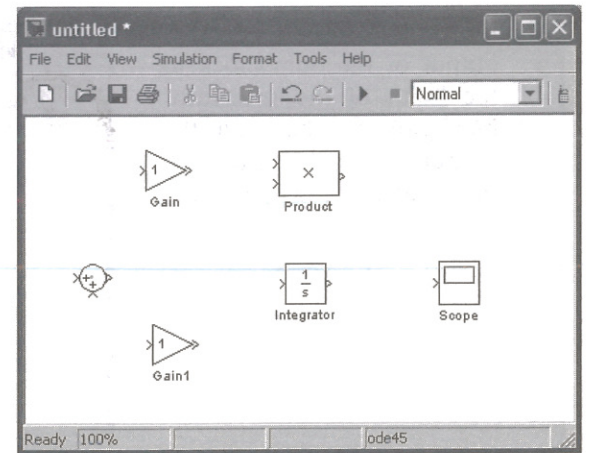
Drag two Gain blocks from the Math block library, and place them as shown. Notice that the name of the second Gain block is Gain1. Simulink requires each block to have a unique name. We'll discuss changing the name in Chapter 4.



Drag a Sum block and a Product block from the Math block library, and position them as shown. We will use the Product block to compute  $x^2$ .

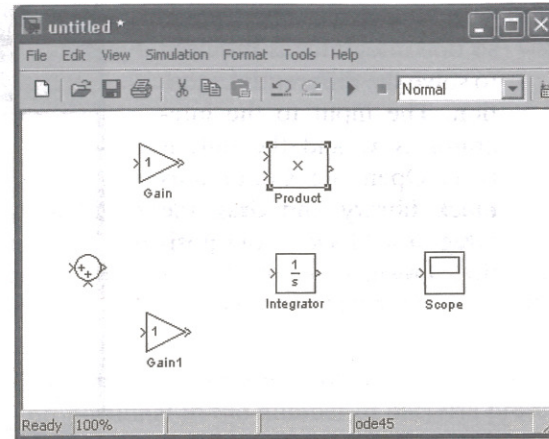


Open the Sinks block library and drag a Scope block to the model window, as shown.

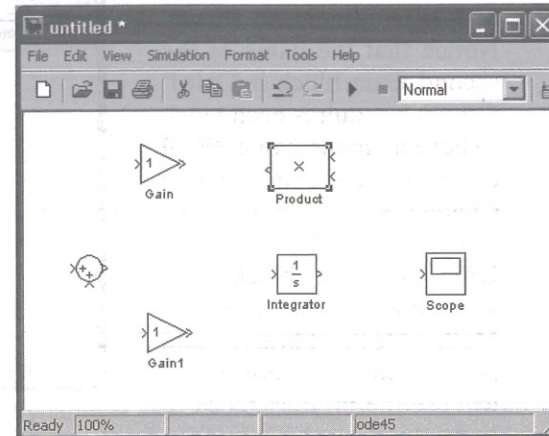




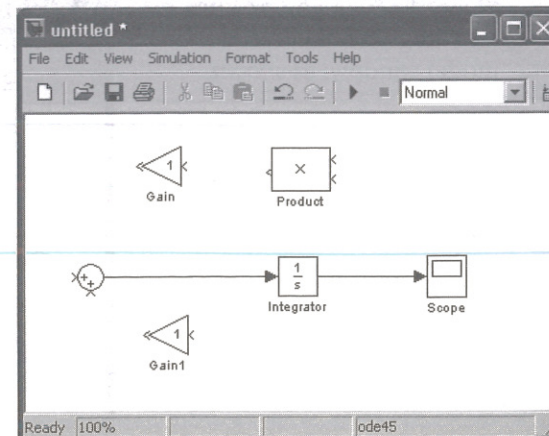
The default orientation of all the blocks places input ports on the left edge of the block and output ports on the right edge. The model will be much easier to read if we flip the Product block and the Gain blocks so that the input ports are on the right edge, and the output ports are on the left edge. Starting with the Product block, click the block once to select it. Notice the handles that appear at the four corners of the block, indicating that it is selected.



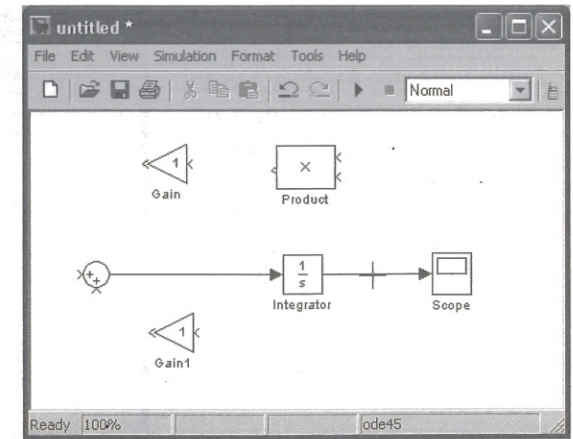
Choose **Format:Flip Block** from the model window menu bar. Now the inputs are on the right, and the output is on the left. Repeat the flipping operation for each Gain block.



Draw a signal line from the output of the Sum block to the input of the Integrator block, and another from the output of the Integrator to the input of the Scope block.

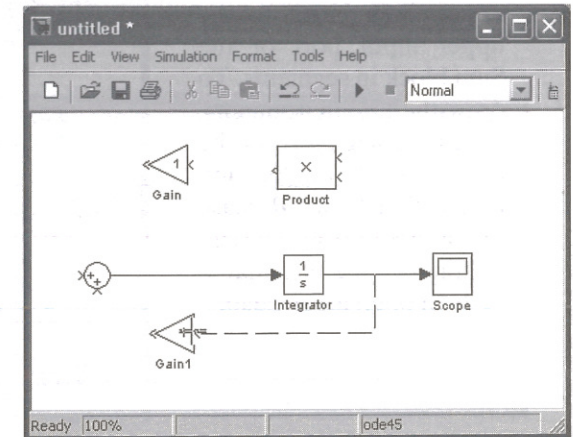


Next, we need to branch from the signal line connecting the Integrator and Scope blocks to feed the value of  $x$  to the lower Gain block. Press and hold the Control key and click the signal line. The cursor will change to a cross-hair shape. Continue to depress the mouse button and release the key.

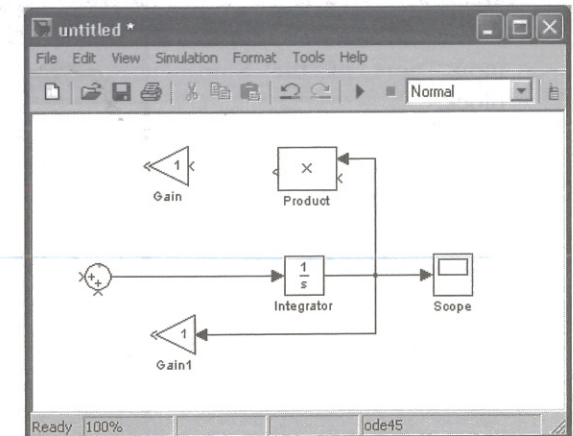


If the mouse has two or three buttons, clicking and dragging using the right mouse button is an alternative to branching using the Control key.

Drag directly to the input port of the Gain block. Notice that the signal line is dashed, and the cursor changes to a double cross-hair when it is on the input port of the gain block. Simulink automatically routes the signal line using 90-degree bends.

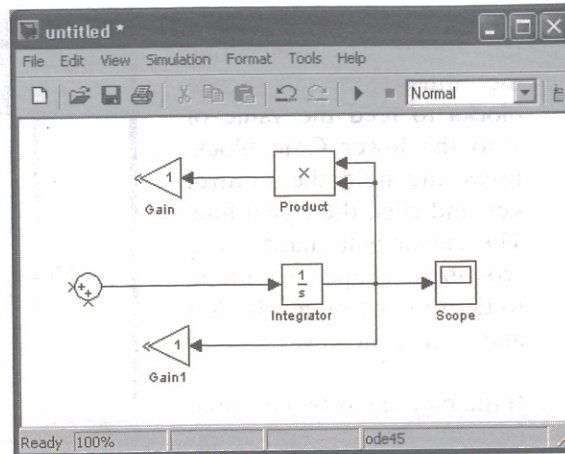


In a similar manner, branch from the signal line connecting the Integrator and Scope blocks to the top input port of the Product block.

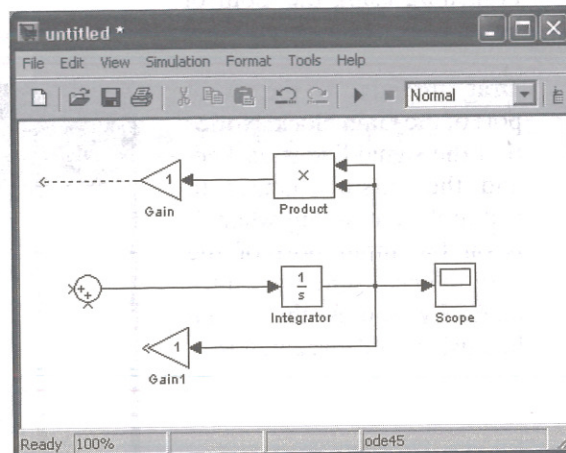




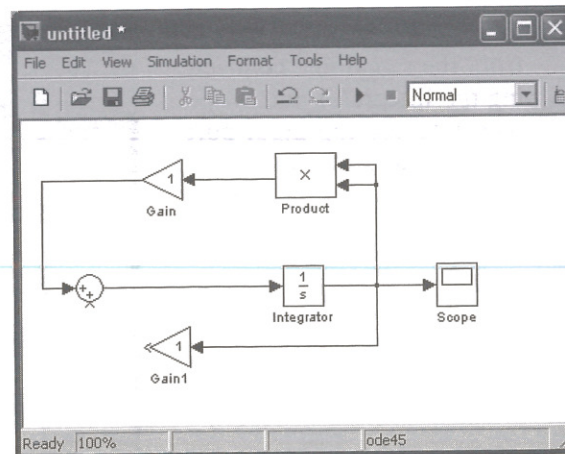
Branch from the signal line entering the upper input port of the Product block to the lower input port of the Product block. Thus, the output of the Product block is  $x^2$ . Connect the output of the Product block to the input of the upper Gain block.



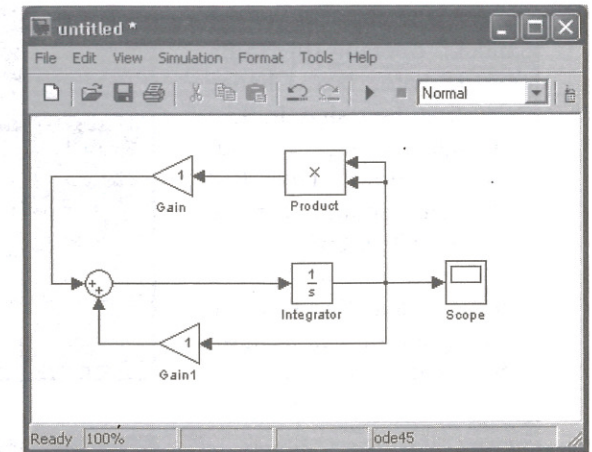
Draw the signal line from the output port of the upper Gain block to the upper input port of the Sum block in segments. To draw the line in segments, start by dragging from the output port to the location of the first bend. Release the mouse button. The signal line will be terminated with an open arrowhead.



Next, drag from the open arrowhead to the upper input port of the Sum block.

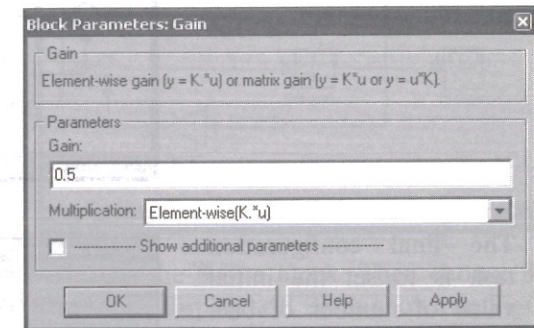


In a similar manner, draw the signal line from the output of the lower Gain block to the lower input of the Sum block.

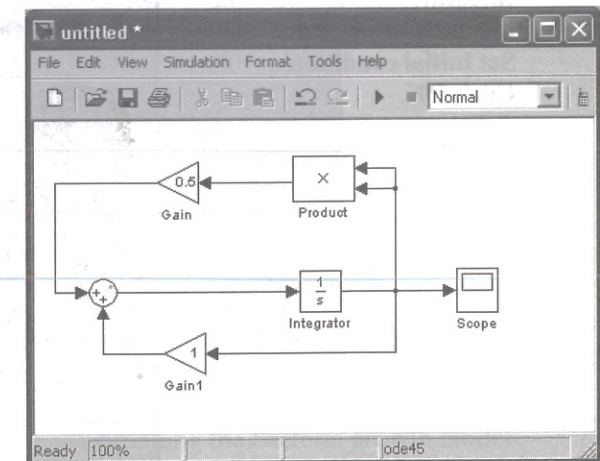


The model is now complete, but several of the blocks must be configured. Currently, the value of gain for both Gain blocks is the default value of 1.0. The Sum block adds its two inputs instead of computing the difference. Finally, the initial value of the integrator output [the initial number of bacteria,  $x$ ] must be set, as it defaults to 0. Start with the Gain blocks.

Double-click the upper Gain block, which corresponds to coefficient  $p$ . The Gain block dialog box will be displayed. Change the default value in field **Gain** to 0.5. Click **OK**.



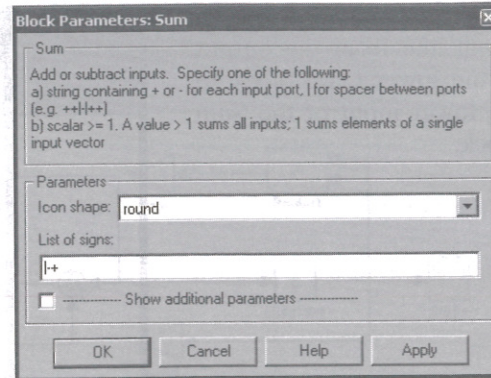
Notice that the value of gain on the block icon is now 0.5.



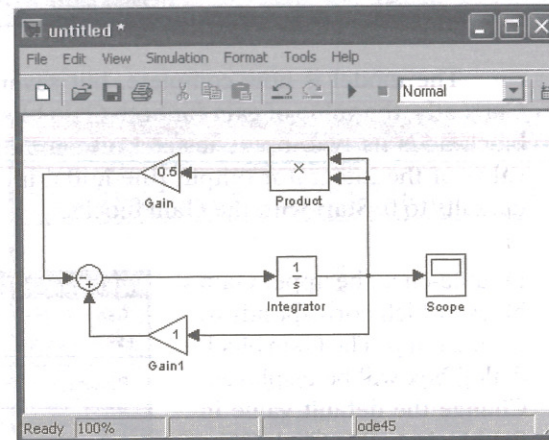


Double-click the Sum block, opening the Sum block dialog box. Change the first (+) sign in List of signs to (-) (a minus sign) as shown. Thus, List of signs should contain (|-+). Click **OK**.

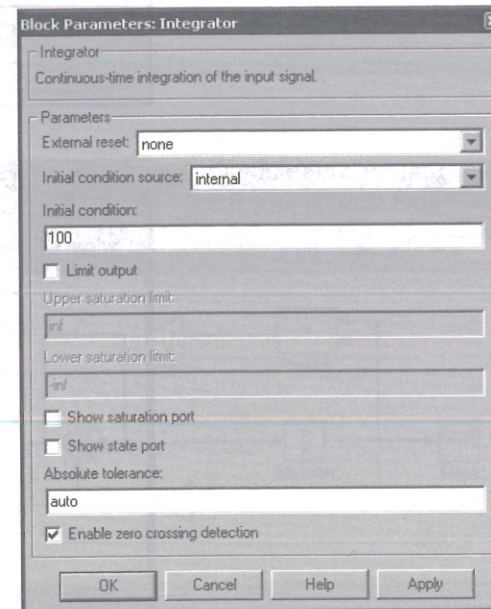
Notice that the sign on the higher Sum block input port changed to (-).



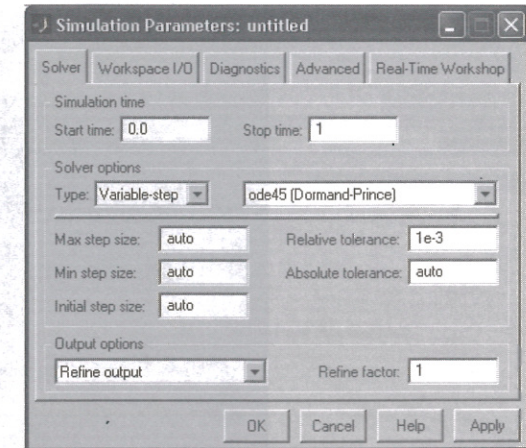
Now the Sum block is configured to compute the value of  $\dot{x}$  according to Equation (3.2), after substituting in the values of  $b$  and  $p$ .




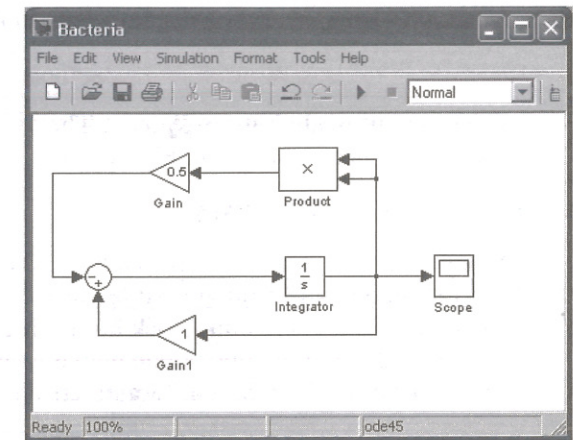
The final configuration task is to set the initial value of number of bacteria ( $x$ ). Double-click the Integrator block, opening the Integrator dialog box. Set **Initial condition** to 100. Click **OK**.



The simulation start time defaults to 0, and the stop time defaults to 10.0. To change the stop time to 1, open the Simulation parameters dialog box by choosing **Simulation:Parameters** from the model window menu bar. Set **Stop time** to 1, then click **OK**.



The model is now complete and ready to run. It's always a good idea to save a model before running it. To save the model, choose **File:Save** from the model window menu bar (or click the  icon), and enter a file name, say **Bacteria**, without an extension. Simulink will save the model with the extension **.mdl** and change the model window name from **untitled** to the name you entered.



Open the Scope by double-clicking the Scope block. Then choose **Simulation:Start** to run the simulation. The scope display will be as shown in Figure 3.8.

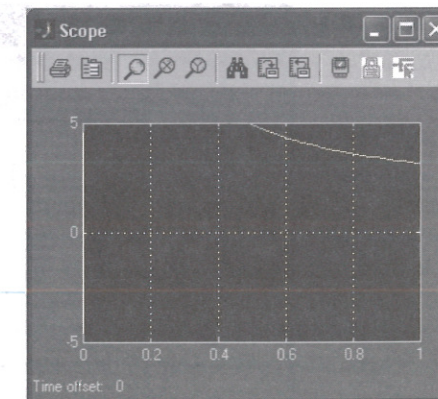


FIGURE 3.8: Scope display after running the bacteria growth model



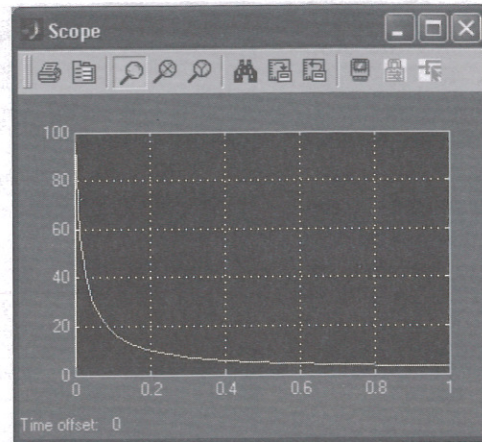



FIGURE 3.9: Scope display after running the bacteria growth model

Click on autoscale button  icon. The scope will resize the scale to fit the entire range of values, as shown in Figure 3.9.

### 3.4 THE Simulink HELP SYSTEM

Simulink includes an extensive Help system in the form of an online HTML format User's Guide and an integrated browser. Detailed online documentation for all of the blocks in the Simulink block library is available through the browser, shown in Figure 3.10. Additionally, online help is available by clicking the **Help** button in the dialog boxes for **Simulation:Parameters** and **Edit:Block Properties**.

#### 3.4.1 Opening the User's Guide Browser

The procedure for opening the user's guide browser for a particular block is as follows:

Double click a block,

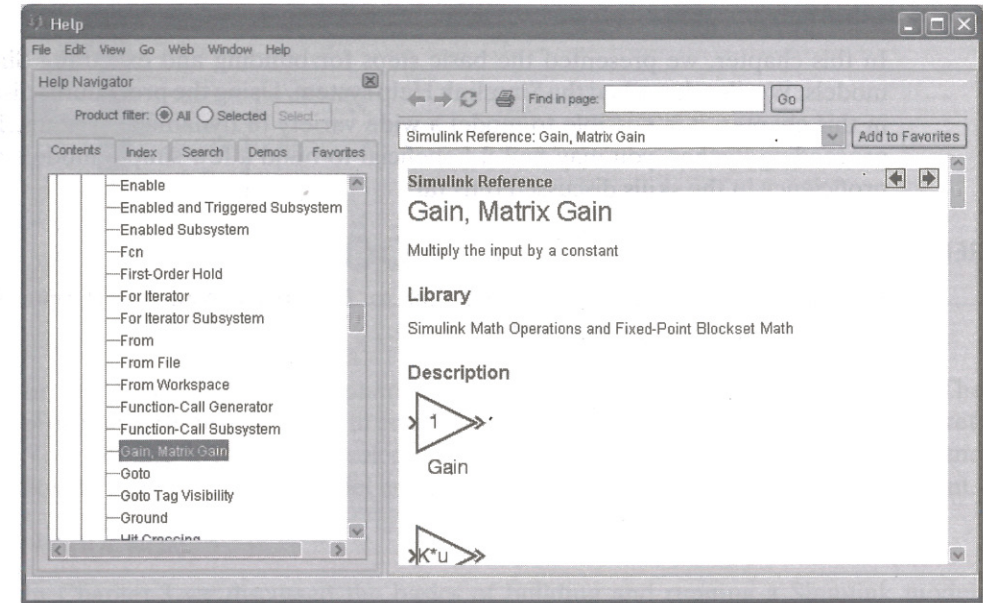
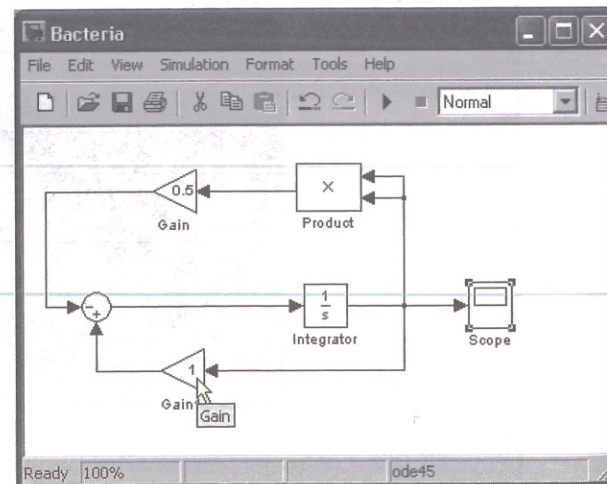
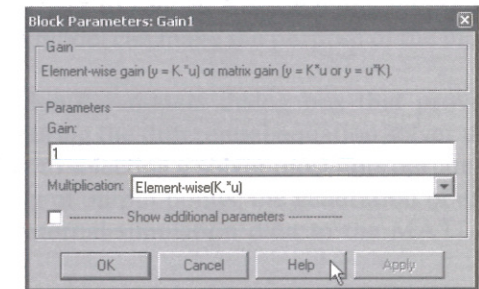


FIGURE 3.10: Simulink user's guide browser

opening the block dialog box. Click the **Help** button.



The browser (Figure 3.10) will open to the page for the block you selected.

#### 3.4.2 User's Guide Browser Window

There are two frames to the user's guide browser window. The left frame contains navigation information, and the right frame contains the selected text and graphics. The four tabs on the navigation frame allow you to display a table of contents, an alphabetical index, a text search window, and a window containing a customizable list of favorite links. The contents tab is shown in Figure 3.10. Each of the tabbed lists operates in a manner similar to the Simulink Library Browser. You can add any page to the list of favorites by choosing the **Add to Favorites** button while the page is displayed in the right frame. The browser window is actually a special-purpose html browser, and it can also display Web pages (such as [www.mathworks.com](http://www.mathworks.com)) if the computer is connected to the Internet. In fact, the default entry in the favorites list is for the MathWorks technical support Web page.



### 3.5 SUMMARY

In this chapter, we presented the basic steps for building and executing Simulink models. We also described the Simulink Help system. Using the procedures discussed in this chapter, it is possible to model a wide variety of dynamical systems. Before proceeding further, you may find it beneficial to build a few simple models to gain proficiency in the skills discussed thus far.

### REFERENCE

- [1] Scheinerman, Edward C., *Invitation to Dynamical Systems*. (Upper Saddle River, N.J.: Prentice Hall, 1996), 22–24.

# 4

## Model Building

In this chapter, we will explain the mechanics of model building in detail. The procedures discussed here will enable you to build models that are easy to interpret. You will also learn how to select and configure a differential equation solver and how to print a Simulink model and embed a model in a word-processing document.

### 4.1 INTRODUCTION

In Chapter 3 we discussed the basics of building and running a Simulink model. Using the procedures discussed there and on the online Help system, you can build extremely complex models. However, as models get more complex, they become more difficult to interpret. The procedures discussed in this chapter will enable you to make your models easier to understand. First, careful arrangement of blocks and signal lines can make the relationships easier to follow. Next, naming blocks and signal lines and adding annotations to the model can make the purpose of the model elements easier to understand.

We will also describe the **Simulation:Parameters** dialog box that provides extensive options for selecting and configuring the differential equation solver used to perform model simulation. In addition to selecting a solver most appropriate for your problem, you can control the spacing of output points, the generation of error and warning messages, and even send internal simulation data to the MATLAB workspace.

Finally, we will explain how to print your models. You can print directly to a printer or embed an image of the model in a word-processing document.

Our purpose in this chapter is to explain the mechanics of model building: manipulating blocks, drawing and editing signal lines, annotating the model, and so on. Once you master these mechanics, you will be ready to begin building models using the procedures covered in the subsequent chapters.

#### 4.1.1 Elements of a Model

A Simulink model consists of three types of elements: *sources*, the *system* being modeled, and *sinks*. Figure 4.1 illustrates the relationship among the three elements. The central element, the system, is the Simulink representation of a block diagram of the dynamical system being modeled. The sources are the inputs to the dynamical system. Sources include constants, function generators such as sine waves and step functions, and custom signals you create in MATLAB. Source blocks are found in the Sources block library. The output of the system is received by sinks. Examples