
# Introduction to MATLAB and SIMULINK

Lab Manual

Prepared by: Amr El-Helw

# Introduction:

## What Is MATLAB?

The name MATLAB stands for *matrix laboratory*.
MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include

- Math and computation
- Algorithm development
- Data acquisition
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

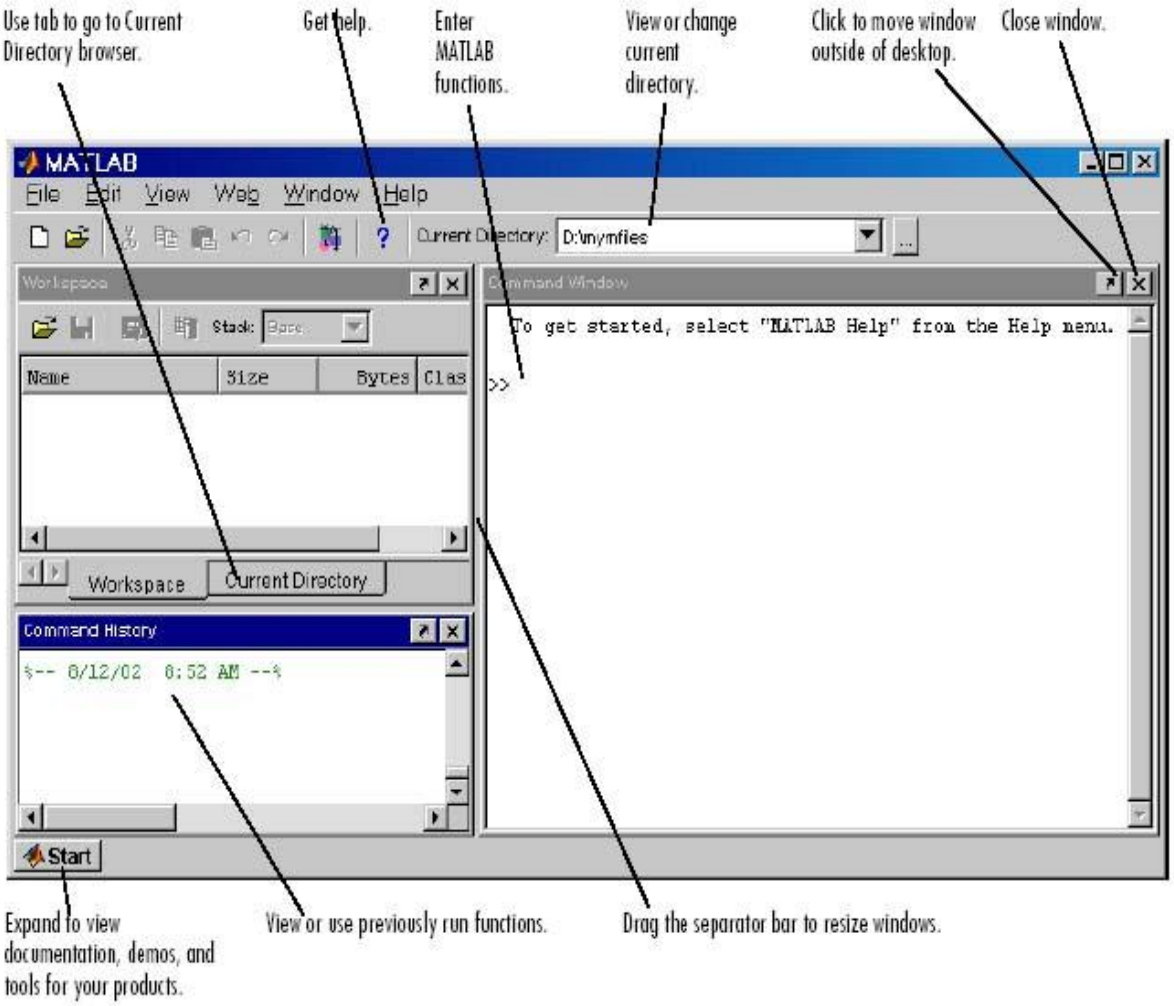
MATLAB features a family of add-on application-specific solutions called ***toolboxes***. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

## The MATLAB System

The MATLAB system consists of five main parts:
**Development Environment**. This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.
**The MATLAB Mathematical Function Library**. This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.
**The MATLAB Language**. This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.
**Graphics**. MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.
**The MATLAB Application Program Interface (API**). This is a library that allows you to write C and Fortran programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

## MATLAB Desktop

When you start MATLAB, the MATLAB desktop appears, containing tools (graphical user interfaces) for managing files, variables, and applications associated with MATLAB.
The first time MATLAB starts, the desktop appears as shown in the following illustration.

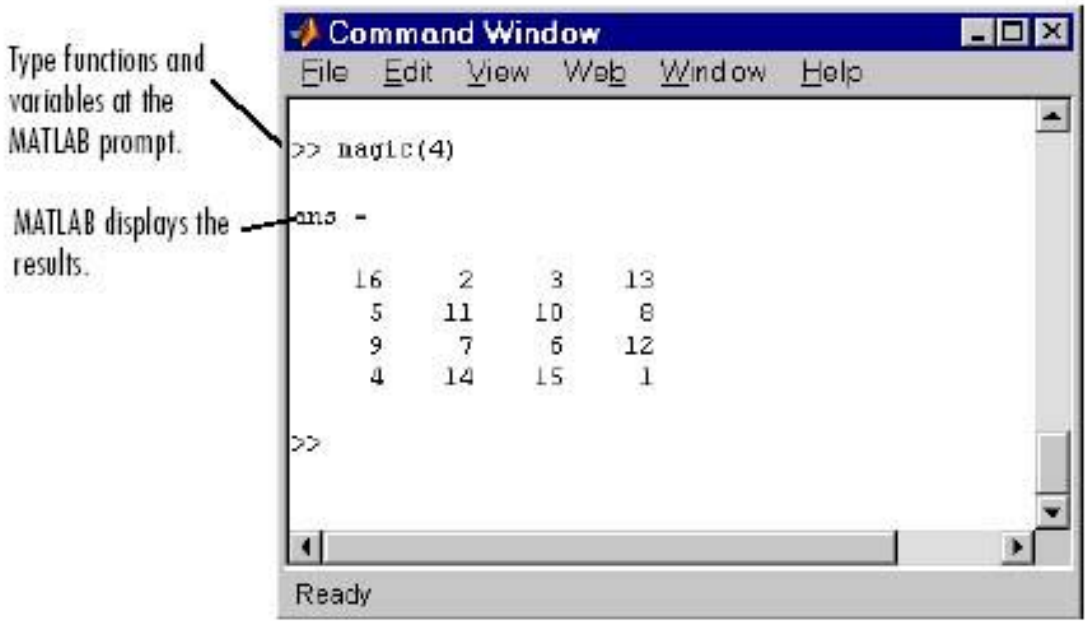



## Desktop Tools

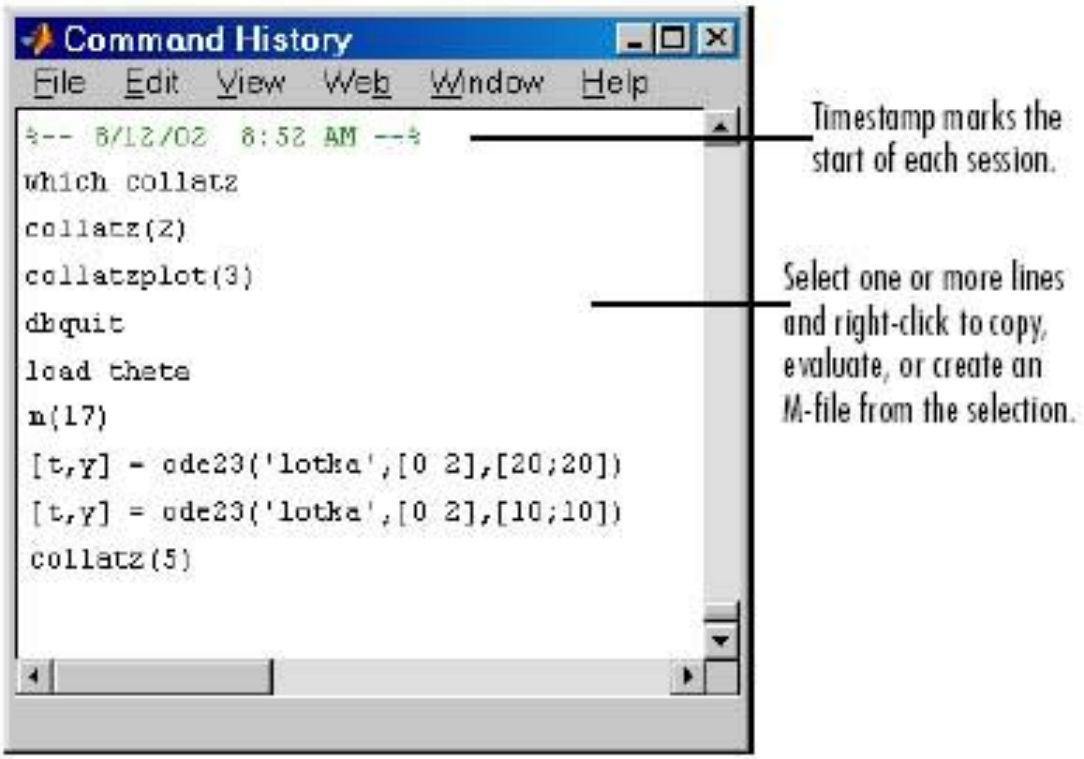
This section provides an introduction to the MATLAB desktop tools. You can also use MATLAB functions to perform most of the features found in the desktop tools. The tools are

- "Command Window"
- "Command History"
- "Start Button and Launch Pad"
- "Help Browser"
- "Current Directory Browser"
- "Workspace Browser"
- "Array Editor"
- "Editor/Debugger"
- "Profiler"

**Command Window**
Use the Command Window to enter variables and run functions and M-files.

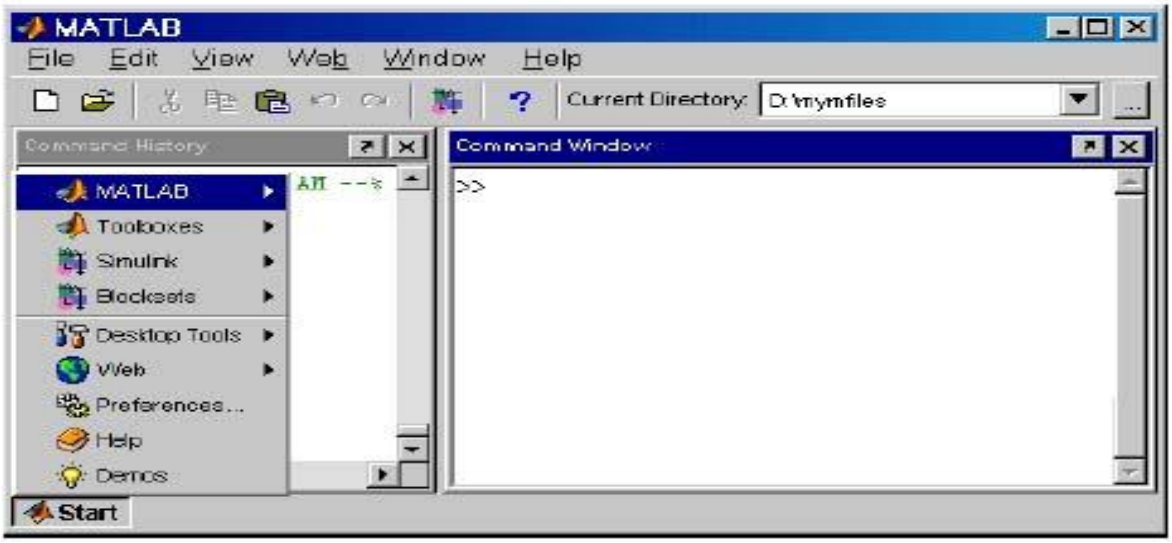



**Command History**
Statements you enter in the Command Window are logged in the Command History. In the Command History, you can view previously run statements, and copy and execute selected statements.



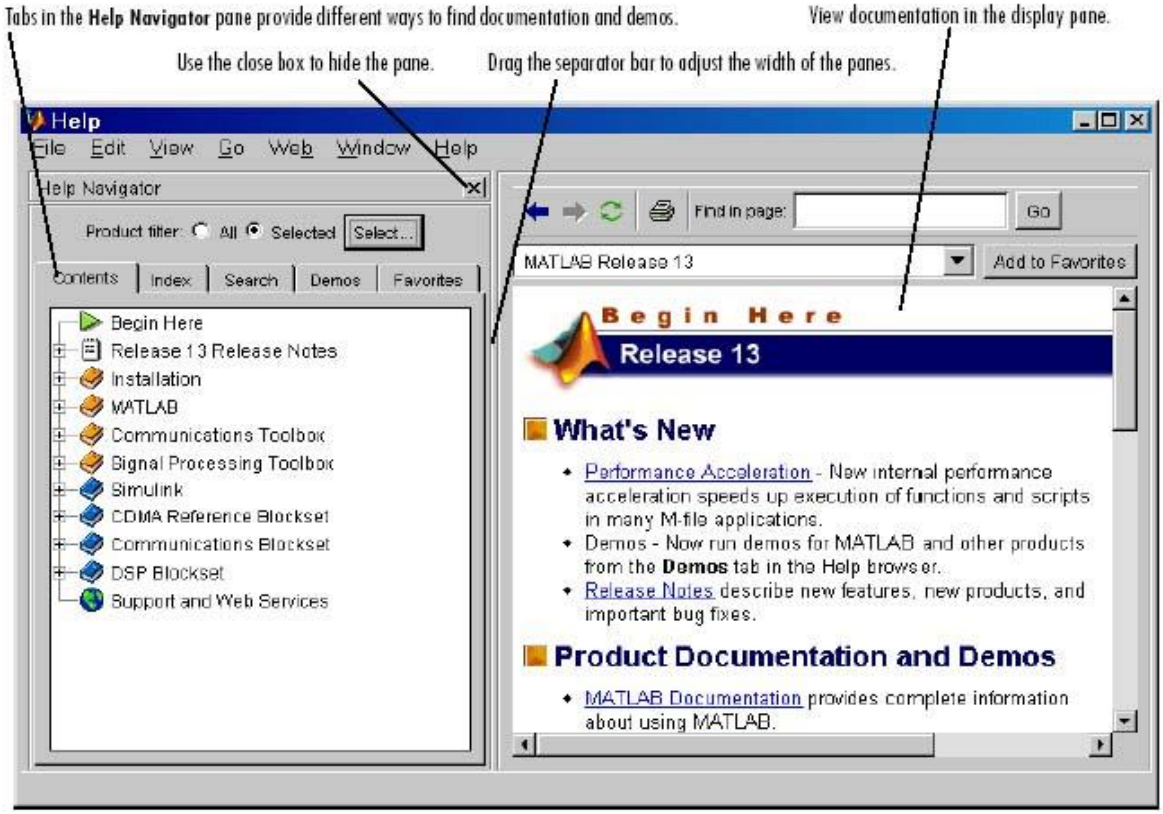
**Start Button and Launch Pad**

The MATLAB Start button provides easy access to tools, demos, and documentation. Just click the button to see the options.




**Help Browser**

Use the Help browser to search and view documentation and demos for all your MathWorks products. The Help browser is a Web browser integrated into the MATLAB desktop that displays HTML documents.To open the Help browser, click the help button in the toolbar, or type help browser in the Command Window.



**For More Help**

In addition to the Help browser, you can use help functions. To get help for a specific function, use doc.

For example, *doc format* displays documentation for the format function in the Help browser.

If you type help followed by the function name, a briefer form of the documentation appears in the Command Window.

Other means for getting help include contacting Technical Support (*http://www.mathworks.com*) and participating in the MATLAB file exchange, (*http://www.mathworks.com/matlabcentral/fileexchange/loadCategory.do*), MATLAB central (http://www.mathworks.com/matlabcentral/) and MATLAB group news (http://newsreader.mathworks.com/WebX?14@@/comp.soft-sys.matlab).

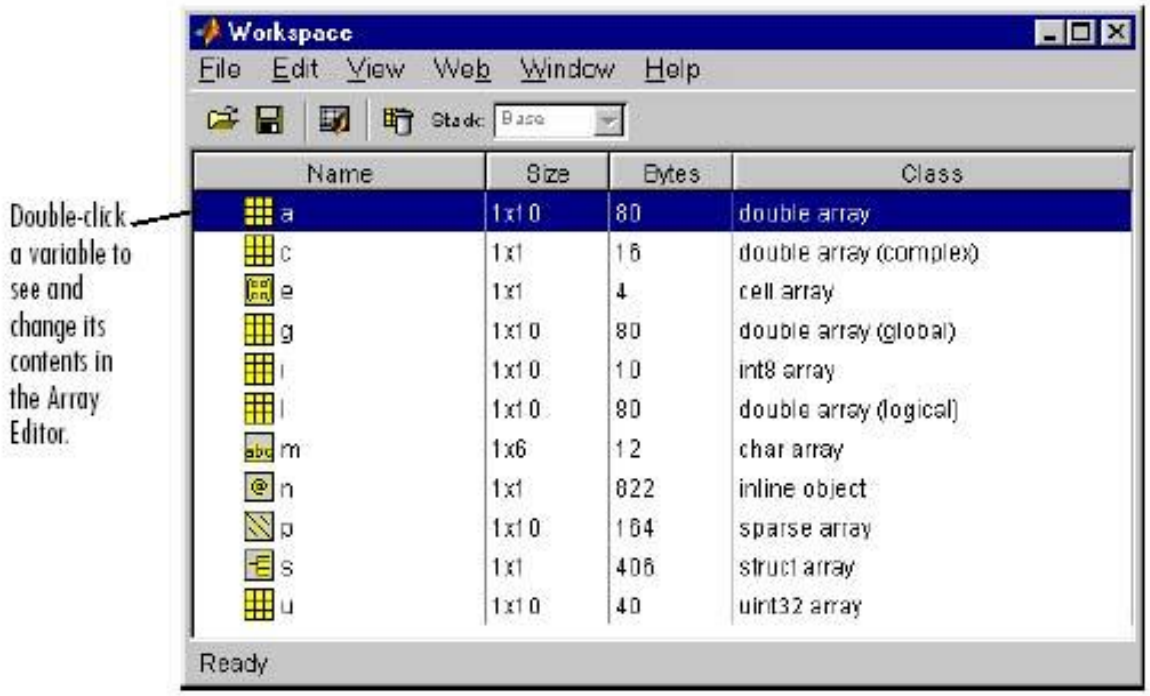
**Current Directory Browser**

MATLAB file operations use the current directory and the search path as reference points. Any file you want to run must either be in the current directory or on the search path.

A quick way to view or change the current directory is by using the Current Directory field in the desktop toolbar as shown below.


**Workspace Browser**

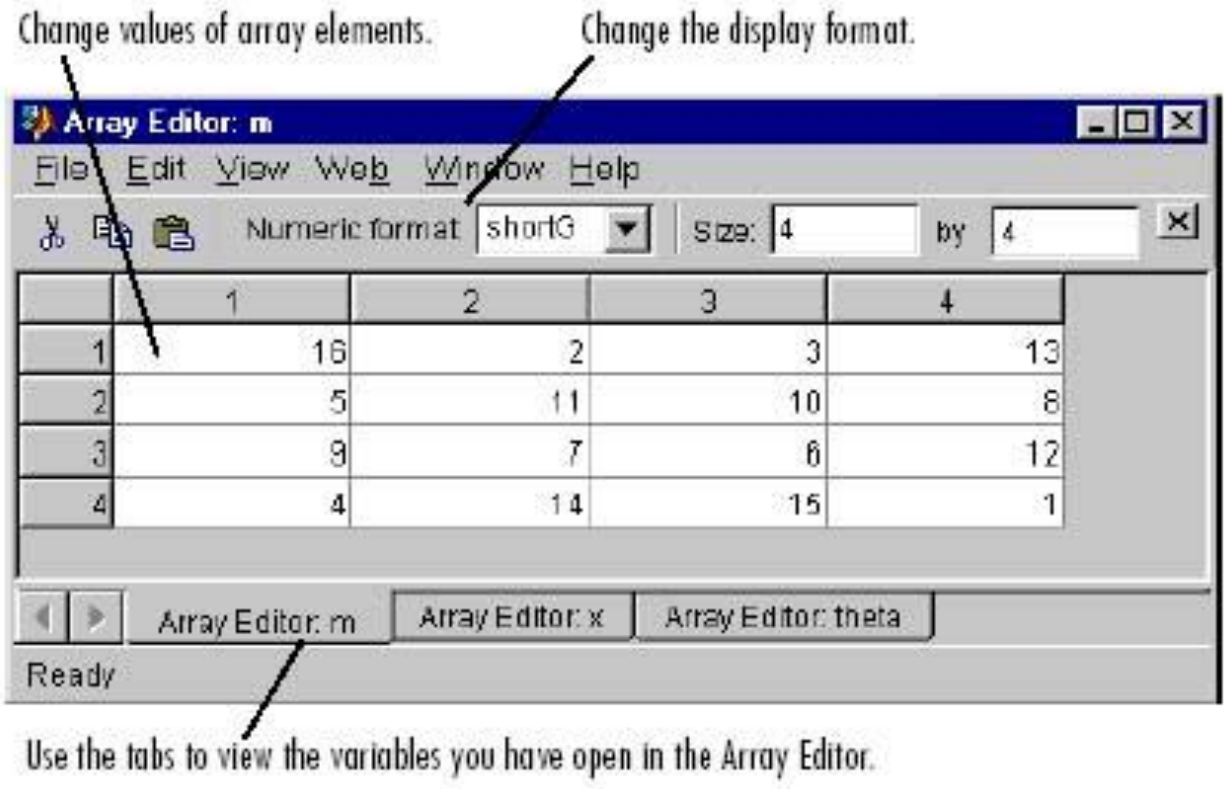
The MATLAB workspace consists of the set of variables (named arrays) built up during a MATLAB session and stored in memory. You add variables to the workspace by using functions, running M-files, and loading saved workspaces. To view the workspace and information about each variable, use the Workspace browser, or use the functions *who* and *whos*.



To delete variables from the workspace, select the variable and select Delete from the Edit menu. Alternatively, use the clear function.
The workspace is not maintained after you end the MATLAB session.
To save the workspace to a file that can be read during a later MATLAB session, select Save Workspace As from the File menu, or use the save function. This saves the workspace to a binary file called a MAT-file, which has a .mat extension.
There are options for saving to different formats. To read in a MAT-file, select Import Data from the File menu, or use the load function.

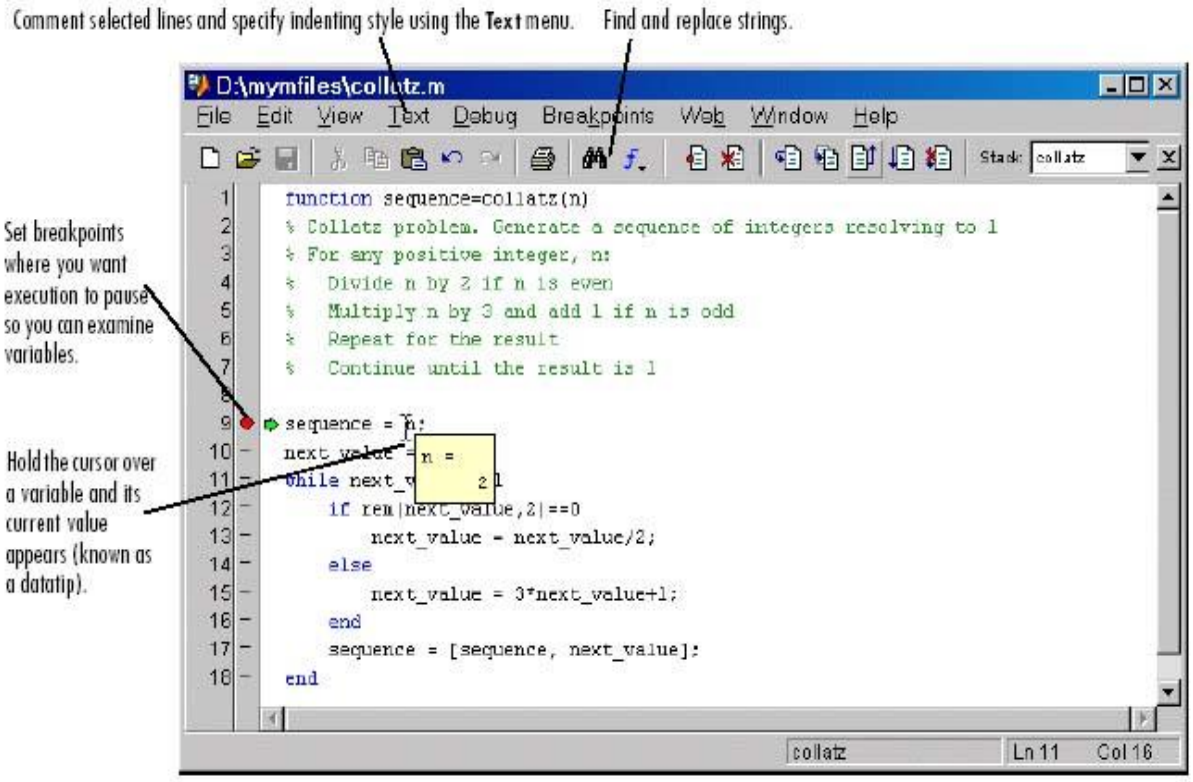
**Array Editor**

Double-click a variable in the Workspace browser to see it in the Array Editor. Use the Array Editor to view and edit a visual representation of one- or two-dimensional numeric arrays, strings, and cell arrays of strings that are in the workspace.



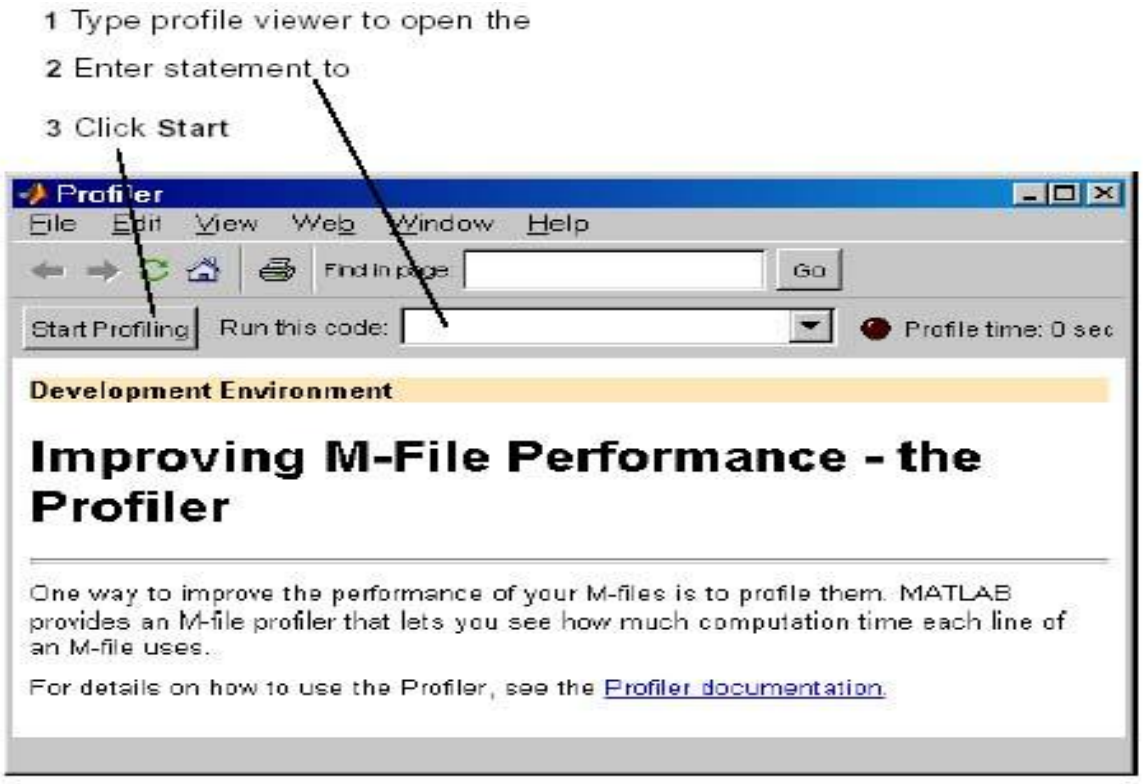

**Editor/Debugger**

Use the Editor/Debugger to create and debug M-files, which are programs you write to run MATLAB functions. The Editor/Debugger provides a graphical user interface for basic text editing, as well as for M-file debugging.
You can use any text editor to create M-files, such as Emacs, and can use preferences (accessible from the desktop File menu) to specify that editor as the default. If you use another editor, you can still use the MATLAB Editor/Debugger for debugging, or you can use debugging functions, such as dbstop, which sets a breakpoint. If you just need to view the contents of an M-file, you can display it in the Command Window by using the type function.




**Profiler**

MATLAB includes a graphical user interface, the Profiler, to help you improve the performance of your M-files.



**Other Development Environment Features**

Additional development environment features are

•Importing and Exporting Data—Techniques for bringing data created by other applications into the MATLAB workspace, including the Import Wizard, and packaging MATLAB workspace variables for use by other applications.

•Interfacing with Source Control Systems—Access your source control system from within MATLAB, Simulink®, and Stateflow®.

•Using Notebook—Access MATLAB numeric computation and visualization software from within a word processing environment (Microsoft Word).

# SIMULINK

## What Is Simulink?

Simulink is a software package for modeling, simulating and analyzing dynamical systems. It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. Simulink can easily accommodate multirate systems as well. Simulink offers a friendly, graphical environment, in which you can model, systems in the form of block diagrams, by simply clicking and dragging blocks into a model window. You can run a model at the push of a button and modify it easily. The graphical nature of Simulink models makes them easy for others to read and understand.

Simulink has a comprehensive block library of sinks, sources, subsystems (linear, nonlinear, and time-varying), connectors, and powerful conditionally-enabled blocks. You can also customize your own blocks.

You can use all the facilities of MATLAB when running Simulink. You can invoke familiar MATLAB expressions and M-file functions as temporary utilities, for instance to control display and visualization. You can even encapsulate them as blocks and place them in Simulink models.

Using Simulink, you can see data displayed in scopes as the simulation unfolds. This makes tracing and debugging a model, by quick, proof-of-concept demonstrations, much faster and easier than by working directly from the command line. Outside the simulation environment, Simulink serves as the primary link for targeting to chips, boards, and co-simulation platforms by means of automatic code generation.

Blocksets are comprehensive collections of Simulink blocks that extend the Simulink environment to solve particular classes of problems. Areas in which blocksets are available include digital signal processing, communications, embedded target for TI C6000, Xilinx and many others.

## What Is the Communications Blockset?

The Communications Blockset is a collection of Simulink® blocks for designing and simulating communication systems. With the Communications Blockset, you can design models in the form of block diagrams, using simple click-and-drag mouse operations. You can run simulations on a model at the push of a button, and change parameters while the simulation is running. The Communications Blockset contains ready-to-use blocks to model various processes within communication systems, including

- Signal generation
- Source coding
- Error-correction
- Interleaving
- Modulation/demodulation
- Transmission along a channel
- Synchronization

In addition, you can create specialized blocks, to implement your own algorithms.

All the power of MATLAB® is available to you when you use the Communications Blockset. You can run simulations from the command line and invoke MATLAB expressions and M-files.

### What Background Do I Need?

Ideally, you should know a little something about MATLAB and the easy, supportive way it leads you from elementary calculations, all the way to powerful, matrix-intensive algorithm

development and execution. But if you are a newcomer to MATLAB and Simulink, just keep reading here. This manual will carry you through the basics. Everything you need to know and do here is easy – and fun!

If you want to learn more about MATLAB and Simulink, beyond what is covered in this manual, you can take a look at Getting Started with MATLAB or Using Simulink. You can read either of these on-line by selecting the Help tab in the MATLAB Command Window. These references are also available on the MathWorks Web site at (http://www.mathworks.com/)

The topics in this manual illustrate the techniques you need to use Simulink for modeling and simulating Communication systems. If you are a Communication novice, just read on –Simulink can illuminate Communication concepts in very exciting ways.

## Getting Started

This part gets you started with the basics of Simulink. It explains how to open and run Simulink models, and how to change parameters within a model.

It also describes blocks – the basic elements that make up Simulink models – and libraries.

This parts covers the following topics:

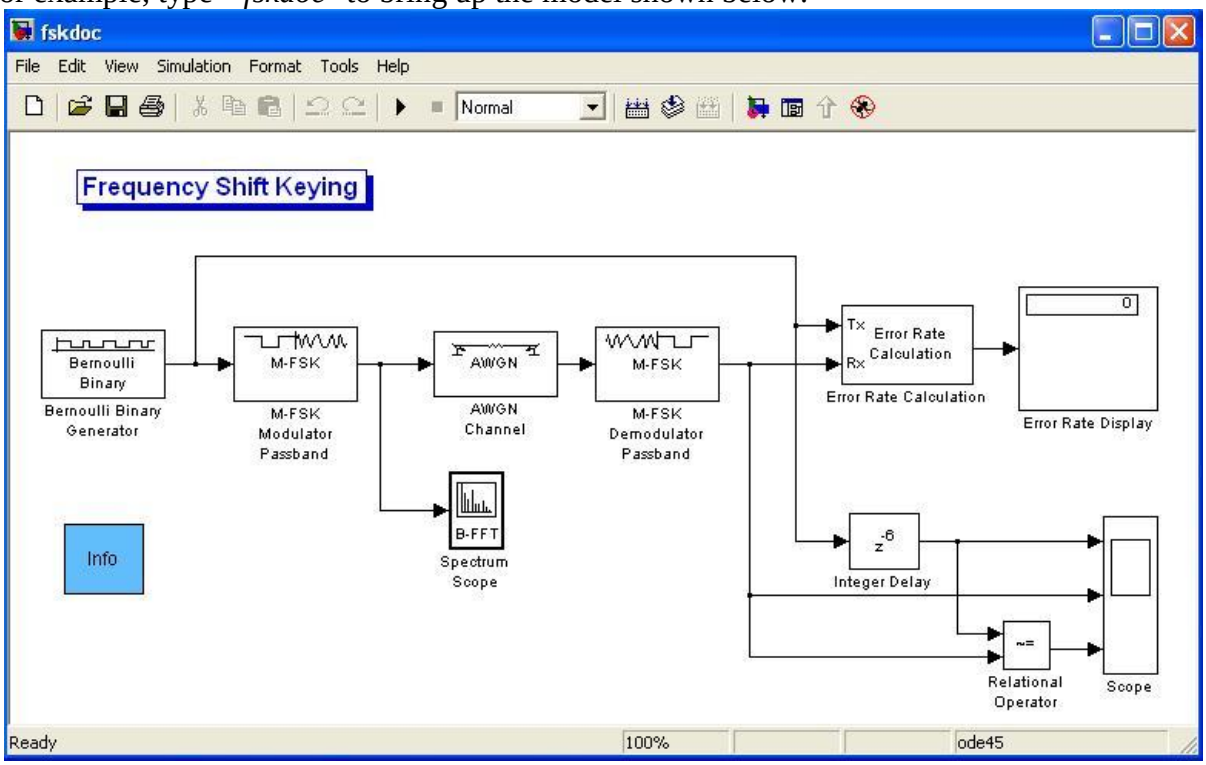
- What You Need to Use the Models in This Book
- Starting Simulink
- Running a Model
- Changing a Block's Parameters
- Changing Simulation Parameters
- Libraries

## Starting Simulink

Unlike MATLAB, there is no special command window for Simulink. It works in the background whenever you build and run models. Before using Simulink,

you must first start MATLAB. You can then load an existing Simulink model by typing its name at the MATLAB prompt.

For example, type " *fskdoc*" to bring up the model shown below.



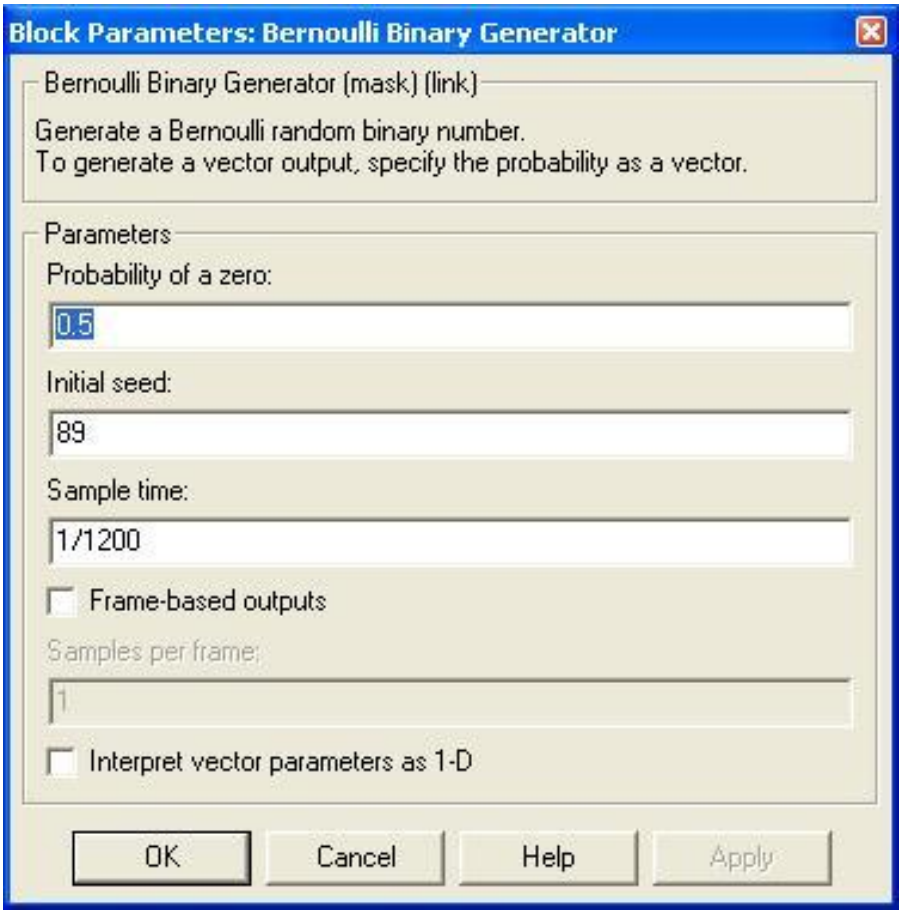
Notice that the model *fskdoc* looks like a standard block diagram. The blocks represent various processes in the model.
For example, "Bernoulli Binary Generator" block at the upper left is a source that generates a Bernoulli random binary number. The Results block is an error rate display, which displays the bit error rate. The lines between blocks represent the passage of data through the model. You can find the blocks in this model in Simulink libraries.

# Running a Model

You can run a model by selecting the Simulation menu at the top of the model window, and then selecting Start or simply by clicking on the symbol  at the top of the of the model window. The square symbol next to it stops the model.
Start the model *fskdoc* and observe the resulting burst of activity. Even if you don't understand exactly what is going on in the model, you can readily see the exciting dynamics of what Simulink can do for you. It provides incisive, visually compelling simulations that give a panoramic overview of all areas of processing. Using Simulink is more like working with laboratory equipment than computing.

## Changing a Block's Parameters

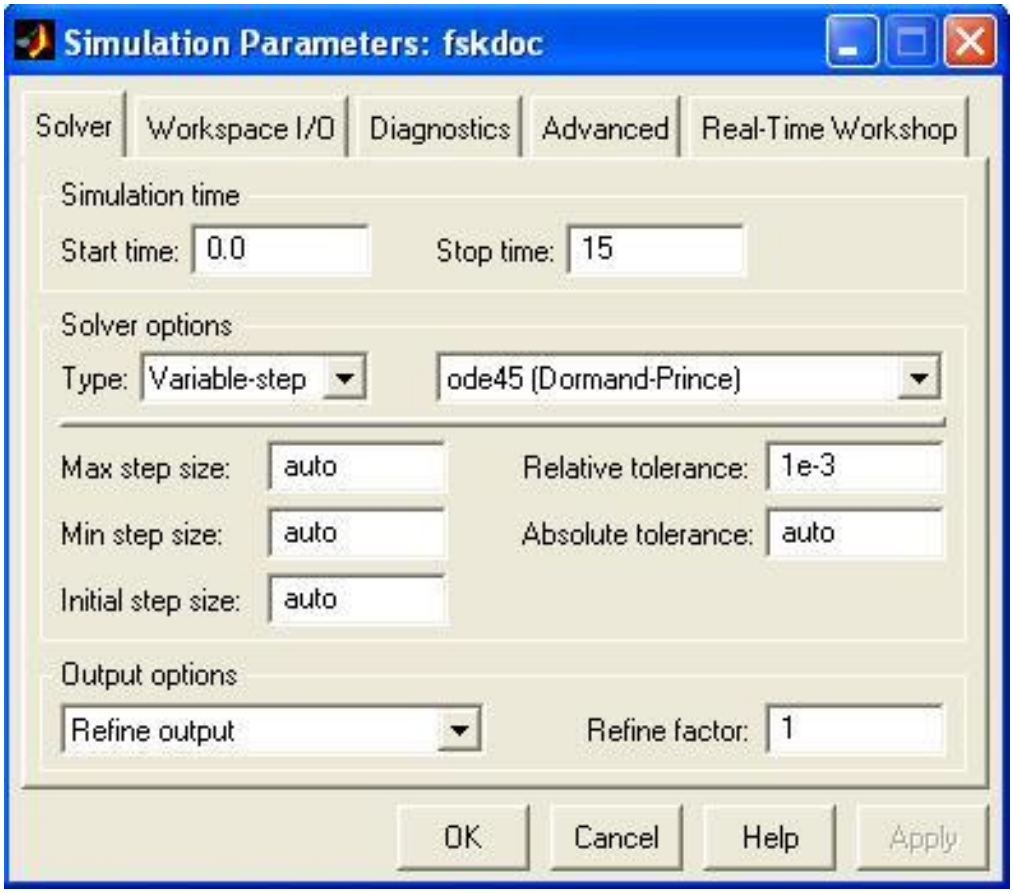
You can not change a block's parameters while a model is running. Stop the *fskdoc* model and double-click on the Bernoulli Binary Generator at the upper-left of the model window. A dialog box opens, presenting you with several parameters that control the block's operation.



Click the Initial seed parameter value, and overwrite the unity value by typing 23. Clicking the Apply button at the lower right corner of the dialog box changes the amplitude. Notice that the scope traces go off the scale.
Now, restore the Amplitude value to 89 and click OK. Then stop the demo.

## Changing Simulation Parameters

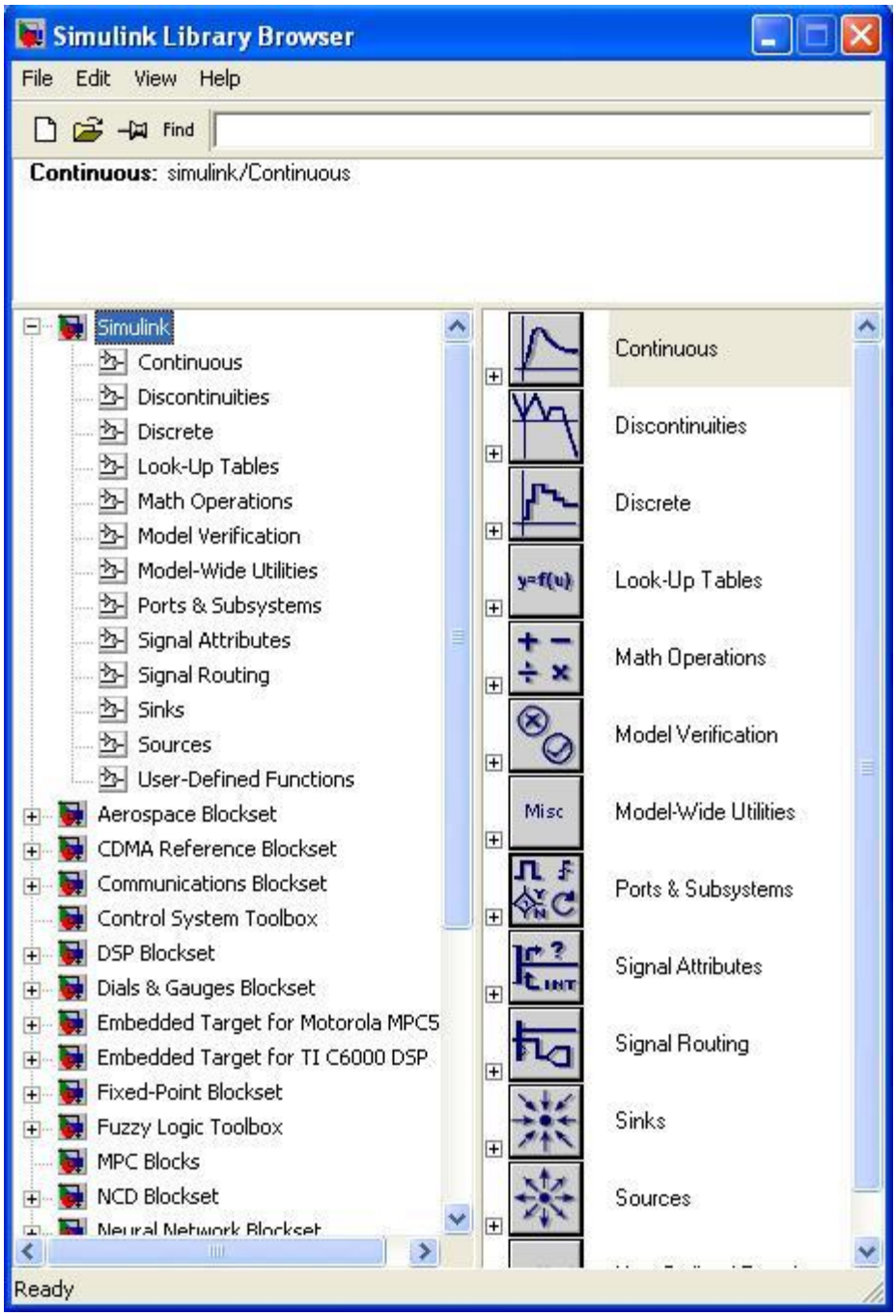
There are several parameters that control the overall simulation. You can view or change these by selecting the Simulation menu at the top of the model window, and then selecting Parameters, which opens a dialog window as in the figure below.




Change the Stop time value to 5 and select OK. Now start the model again. The simulation will stop after about five second. If you change the Stop time setting to inf (for infinite), the simulation will never terminate until you stop it manually.
Now, close the *fskdoc* model, selecting No when asked if you want to save the changes.

## Libraries

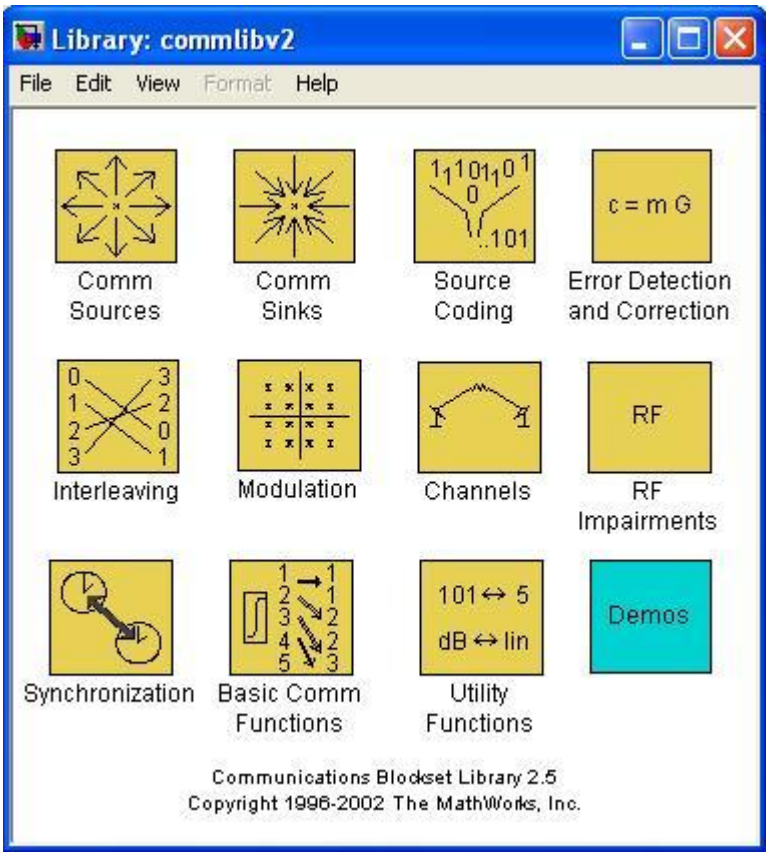
Simulink libraries are collections of Simulink blocks, which you can use to create models. To see all the available libraries, select the Simulink icon at the top of the MATLAB desktop, or type *simulink* at the MATLAB prompt. This opens the Simulink Library Browser, which lists the libraries and their sublibraries in a tree structure, as shown below. The most important libraries for this manual are the Simulink library, the DSP blockset and the Communication Blockset library.




You can open a library to see its sublibraries by clicking on the + sign next to its name. For example, clicking on the + sign next to communication Blockset opens several sublibraries. You can open a sublibrary, such as "Comm Sinks", in the same way. Furthest to the right on the tree are the individual blocks.

Clicking on the name of a library or block will display a description of it at the bottom of the Browser window.

To open a library in a window, type the name of the library at the MATLAB prompt. For example, type *commlib* to open the communication Blockset library, as seen in the top window in the figure below.




Each of the icons in this window represents a sublibrary. Click on one of the icons, such as the one labeled “Channels” and that sublibrary will appear, as seen in figure below.