Beginning MATLAB and Simulink

From Novice to Professional

—

Sulaymon Eshkabilov

Apress®

Beginning MATLAB and Simulink

From Novice to Professional

Sulaymon Eshkabilov

Apress®

Beginning MATLAB and Simulink: From Novice to Professional

Sulaymon Eshkabilov
Fargo, United States

ISBN-13 (pbk): 978-1-4842-5060-0 ISBN-13 (electronic): 978-1-4842-5061-7
https://doi.org/10.1007/978-1-4842-5061-7

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Steve Anglin
Development Editor: Matthew Moodie
Coordinating Editor: Mark Powers

Cover designed by eStudioCalamar

Cover image from rawpixel.com

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail editorial@apress.com; for reprint, paperback, or audio rights, please email bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484250600. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

To the memory of my father.
To my mother.
To my wife, Nigora, after 25 wonderful years together.

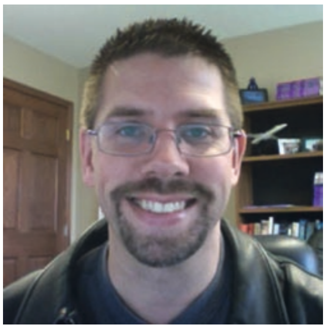
Table of Contents

About the Author

Sulaymon Eshkabilov, PhD, is currently a visiting professor at the Department of Agriculture and Biosystems, North Dakota State University, USA. He obtained an ME diploma from the Tashkent Automobile Road Institute, an MSc from Rochester Institute of Technology, NY, USA, and a PhD from Cybernetics Institute of Academy Sciences of Uzbekistan in 1994, 2001, and 2005, respectively. He was an associate professor at the Tashkent Automobile Road Institute from December 2006 until January 2017. He held visiting professor and researcher positions at Ohio University, USA, during the 2010/2011 academic year and at Johannes Kepler University, Austria from January-September, 2017. He teaches the following courses: "MATLAB/Simulink Applications for Mechanical Engineering and Numerical Analysis" and "Modeling of Engineering Systems" to undergraduate students, "Advanced MATLAB/Simulink Modeling" seminar/class, "Control Applications," "System Identification," and "Experimentation and Testing with Analog and Digital Devices" to graduate students.

His research interests are mechanical vibrations, control, mechatronics, and system dynamics. He developed simulation and data analysis models for vibrating systems, autonomous vehicle control, and studies of mechanical properties of bones. He has authored two books devoted to MATLAB/Simulink applications for mechanical engineering students and numerical analysis. He has worked as an external academic expert in the European Commission to assess academic projects in 2019.

About the Technical Reviewers



Dr. Joseph Mueller specializes in control systems and trajectory optimization. For his doctoral thesis, he developed optimal ascent trajectories for stratospheric airships. His active research interests include robust optimal control, adaptive control, applied optimization and planning for decision support systems, and intelligent systems to enable autonomous operations of robotic vehicles. Prior to joining SIFT in early 2014, Dr. Mueller worked at Princeton Satellite Systems for 13 years. In that time, he served as the principal investigator for eight small business innovative research contracts for NASA, the Air Force, the Navy, and MDA. He has developed algorithms for optimal guidance and control of both formation flying spacecraft and high altitude airships and developed a course of action planning tool for DoD communication satellites. In support of a research study for the NASA Goddard Space Flight Center in 2005, Dr. Mueller developed the Formation Flying Toolbox for MATLAB, a commercial product that is now used at NASA, ESA, and several universities and aerospace companies around the world. In 2006, Dr. Mueller developed the safe orbit guidance mode algorithms and software for the Swedish Prisma mission, which has successfully flown a two-spacecraft formation flying mission since its launch in 2010. Dr. Mueller also serves as an adjunct professor in the Aerospace Engineering & Mechanics Department at the University of Minnesota, Twin Cities campus.

Karpur Shukla is a research fellow at the Centre for Mathematical Modeling at FLAME University in Pune, India. His current research interests focus on topological quantum computation, nonequilibrium, and finite-temperature aspects of topological quantum field theories, and applications of quantum materials effects for reversible computing. He received an M.Sc. in physics from Carnegie Mellon University, with a background in theoretical analysis of materials for spintronics applications as well as Monte Carlo simulations for the renormalization group of finite-temperature spin lattice systems.

Acknowledgments

I would like to express my special gratitude to the technical reviewers of this book—Dr. Joseph Mueller and Mr. Karpur Shukla—for their very thorough work while reviewing the context and code of the book. Without their critical insights and corrections many points along the way, the book would not be as good as it is now. I would also like to express my special gratitude to Mark Powers for his timely and well planned correspondence throughout this book project.

My cordial gratitude goes to my mother, for her limitless support and love. Up until the last point of this book, she has kept checking my progress.

I would like to thank my wife, Nigora. Without her great support, I would not be able to take up the challenging task of writing this book. I have spent so much time over the weekends in my office writing and editing the book. In addition, I would like to thank our children—Anbara, Durdona, and Dovud—for being such delightful people and for being the inspiration for writing this book.

Introduction

This book is aimed at beginner-level learners of MATLAB/Simulink packages. It covers the essential, hands-on tools and functions of the MATLAB and Simulink packages, and explains them via interactive examples and case studies. The main principle of the book is “learning by doing” and it progresses from simple to complex. It contains dozens of solutions and simulation models via M/MLX files/scripts and Simulink models, which help you learn the programming and modeling essentials. Moreover, there are many recommendations for avoiding pitfalls related to programming and modeling aspects of MATLAB/Simulink packages.

“Beginning MATLAB and Simulink” explains various practical issues of programming and modeling in parallel by comparing programming tools of MATLAB and blocks of Simulink. After studying this book, you'll be proficient at using MATLAB/Simulink packages. You can apply the source code and models from the book's examples as templates to your own projects in data science, numerical analysis, modeling and simulation, or engineering.

Essential learning outcomes of the book include:

- Getting started using MATLAB and Simulink
- Performing data analysis and visualization with MATLAB
- Programming essentials of MATLAB and core modeling aspects of Simulink, and how to associate scripts and models of MATLAB and Simulink packages
- Developing GUI models and standalone applications in MATLAB
- Working with integration and numerical root-finding methods in MATLAB and Simulink
- Solving differential equations in MATLAB and Simulink
- Applying MATLAB for data analysis and data science projects

The book contains eight logically interlinked chapters.

- Chapter 1 is dedicated to introducing the MATLAB environment and MATLAB recognized data types, including numeric, cell, structure, character, logical, function handle, and table.
- Chapter 2 covers most of the essential programming tools and functions, such as `for … end` and `while … end` loop operators, `if…elseif…else…end` condition operators, symbol referencing, and most common errors and warnings. It also covers M-file debugging tools and options in MATLAB. Moreover, this chapter addresses MATLAB-specific programming tools, including function files, function handles, and display operators, such as `disp()`, `display`, `fprintf`, and `sprint`.
- Chapter 3 covers GUI development and how to write and edit GUI model callback functions.
- Chapter 4 addresses the issues of how to develop MEX files, C/C++ code, and standalone applications from the existing M-files.
- Chapter 5 is dedicated to Simulink modeling essentials. The examples in this chapter cover matrix operations, computing function values, modeling, and solving ordinary differential equations. Moreover, it covers issues around how to associate Simulink models with MATLAB scripts.
- Chapter 6 is devoted to data visualization issues, such as building 2D and 3D plots and animated plots in MATLAB.
- Chapter 7 is dedicated to matrix algebra and array operations. It addresses solving systems linear equations, eigen-value problems, matrix decompositions, matrix and vector operations, and conversions of arrays and strings via examples solved in MATLAB and Simulink in parallel.
- Chapter 8 covers some essential aspects of solving ordinary differential equations analytically and using MATLAB's built in functions and commands.

All of the source code (scripts, M/MLX/MAT files, Simulink models, SLX/MDL files, C code, MEX-files, and installation `*.exe` files) discussed in the book are available to readers via a Download Source Code button on the book's `apress.com` product page and accessible via the Download Source Code link located at `www.apress.com/9781484250600`.

Note The scripts in the book may not always be the best solutions to the given problems, but this is done intentionally in order to emphasize methods used to improve them. In some other cases, it is found to be the most appropriate solution to the author's best knowledge. Should I spot better alternative solutions to exercises, I will publish them via MathWorks' MATLAB Central User Community's file exchange, via my file exchange link there (under my name).

No matter how hard we worked to proofread this book, it is inevitable that there will be some typographical errors that might slip through and will appear in print.

September 2019 Sulaymon Eshkabilov

CHAPTER 1

Introduction to MATLAB

The MATLAB package is employed in wide ranges of engineering and scientific computing applications and is associated with the dynamic system simulation package called Simulink. The package has a few advantages and remarkable strengths, such as user-friendly and intuitive programming syntax, high-quality numerical algorithms for various numerical analyses, powerful and easy-to-use graphics, simple command syntax to perform computations, and many add-ons as toolboxes and real and complex vectors and matrices, including sparse matrices as fundamental data types.

There are diverse application areas of the package, i.e. simulation of various systems such as vehicle performances, mapping of the human genome, financial analysis in emerging economies, microbiology applications in diagnosis and treatment of small organisms, dynamic simulations of large ships in down-scaled laboratory models, simulation of the next generation network audio products, teaching computer programming to undergraduates with real-time laboratory tests and measurements, and image processing for underwater archeology and geology.

In this chapter, we discuss some essential key features of the graphical user interface (GUI) of MATLAB, how to use the help tools and library sources, how to adjust the format options and accuracy and precision settings, how to create various variables and variable structures, and how to employ the M/MLX editors to write and edit scripts/programs.

MATLAB's Menu Panel and Help

The MATLAB application can be launched from the Windows operating system by clicking the icon/shortcut from the desktop window or choosing Start ➤ All Programs ➤ MATLAB. As MATLAB loads, the user's last preserved data, files, entries, last 20 commands (by default), and menu bar and tools with the last preferences all appear.

© Sulaymon Eshkabilov 2019
S. Eshkabilov, *Beginning MATLAB and Simulink*, https://doi.org/10.1007/978-1-4842-5061-7_1

MATLAB's graphical user interface (GUI) tools and windows are customizable. Users can easily manipulate, customize, and change preferences of the package according to their needs. Figure 1-1 shows the default main window of MATLAB 2018a. It must be noted that the package's GUI menu and tools have changed over the years in an effort to make the package more user friendly and the tools more intuitive. The default window has the main menu tools, a Current Directory indicator, and Command, Workspace, and Command History windows. These windows can be docked/undocked or opened in a separate window, closed or removed from the main window, or dragged from one pane to another and maximized or minimized.

This main desktop window is shown in Figure 1-1.

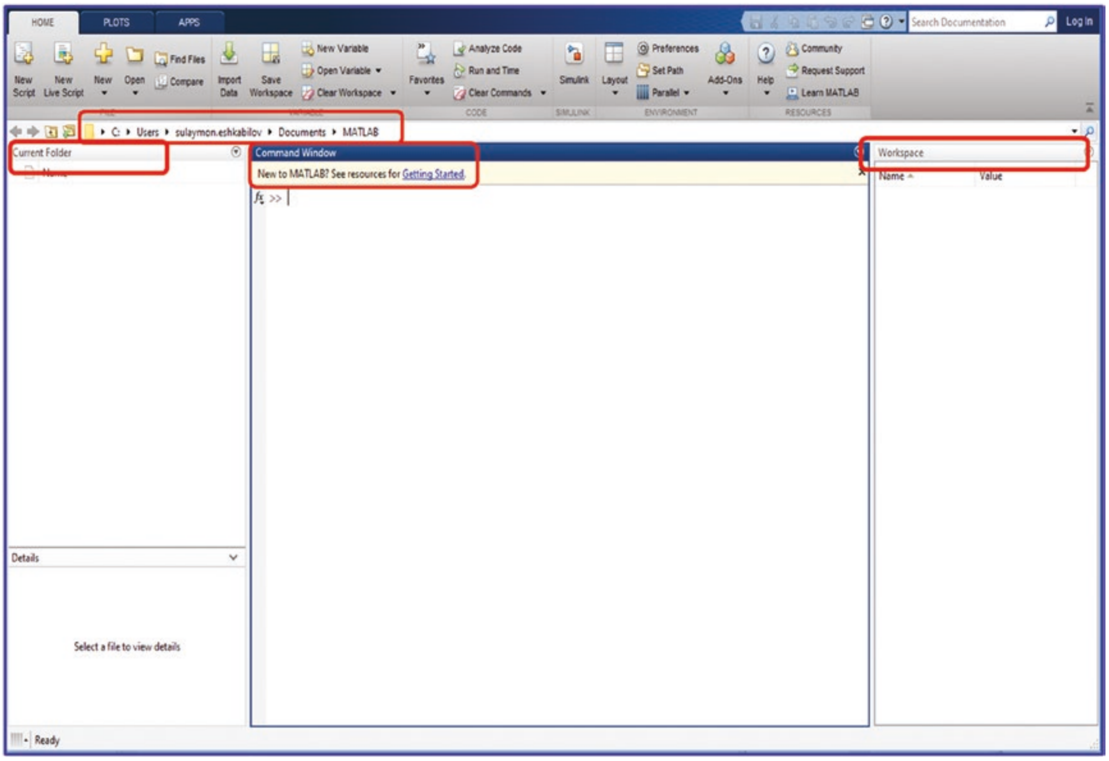


Figure 1-1. *Default MATLAB desktop window (R2018b)*

The main components of the package's graphical user interface (GUI) tools are as follows:

- Tools and toolbars are grouped into three menus—HOME, PLOTS, and APPs (see Figure 1-2). The HOME tab contains all the main tools. It's where you can create and delete new files and variables, import data, access the code analyzer, launch Simulink, change the MATLAB package desktop layout change options, set the path, and access add-ons and help options.
- The Current Directory window is in the left pane by default. This window shows all the files in the directory and the folder directory.
- The Command Window is in the central pane (by default). All commands and/or (small) scripts/codes can be entered directly after *fx* >>. By clicking on *fx* >> (see Figure 1-4) in the Command Window, you can view all built-in functions and MATLAB toolboxes. This option is only available starting from MATLAB version R2008a. All installed toolboxes can also be viewed or accessed by clicking on the APPS tab (see Figure 1-2), which is available in later versions of MATLAB, starting with the R2010a version.
- The Workspace pane of the default desktop window shows the current entries and saved variables during the session. These entries are saved temporarily until the MATLAB application is closed. All essential attributes and properties of entries/variables (variable names, values, and types) are displayed in the workspace.

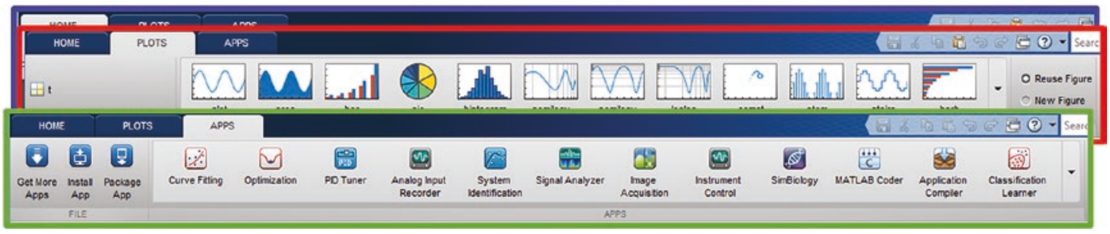
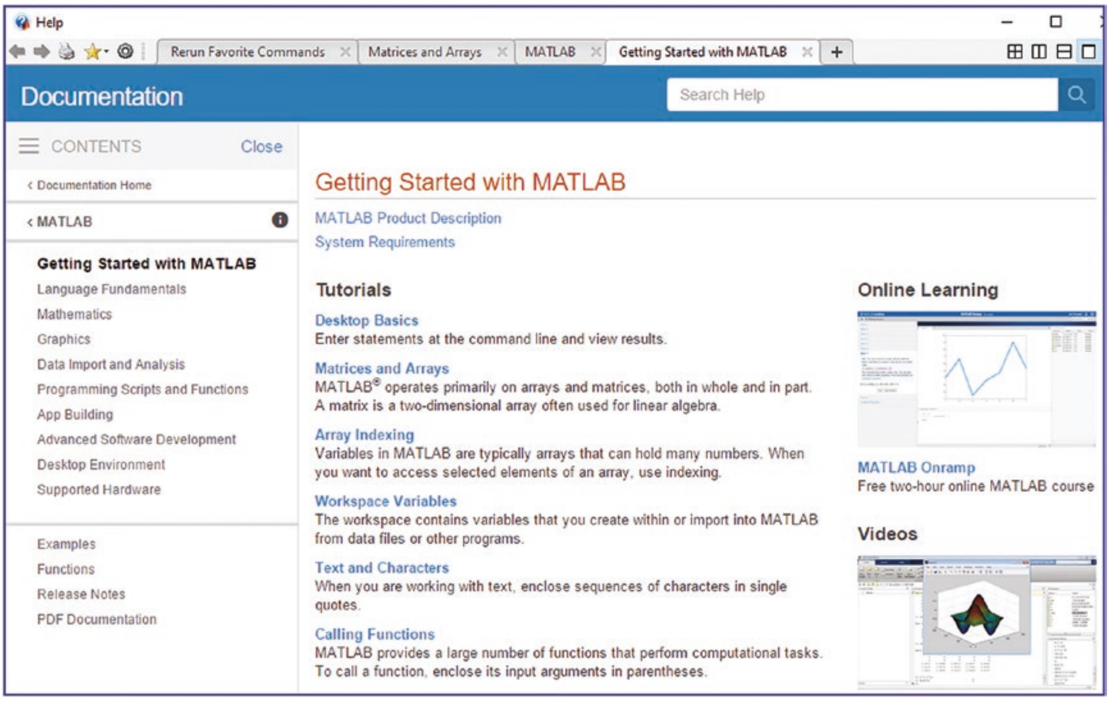


Figure 1-2. *MATLAB main menu: The HOME, PLOTS, and APPS tabs (MATLAB R2018b)*

One of the most essential strengths of this package is its help documentation, which includes help libraries and tools. They can be accessed using GUI tools and entries from the Command Window. It is a good idea to start using the package by exploring the Getting Started options, which you can access by clicking the Getting Started hyperlinked text, above the Command Window. This brings up the Help Library documentation, as shown in Figure 1-3, which contains the essential documentation and help tools, such as examples and function syntax.



Getting Started with MATLAB

MATLAB Product Description
System Requirements

Tutorials

Desktop Basics
Enter statements at the command line and view results.

Matrices and Arrays
MATLAB® operates primarily on arrays and matrices, both in whole and in part. A matrix is a two-dimensional array often used for linear algebra.

Array Indexing
Variables in MATLAB are typically arrays that can hold many numbers. When you want to access selected elements of an array, use indexing.

Workspace Variables
The workspace contains variables that you create within or import into MATLAB from data files or other programs.

Text and Characters
When you are working with text, enclose sequences of characters in single quotes.

Calling Functions
MATLAB provides a large number of functions that perform computational tasks. To call a function, enclose its input arguments in parentheses.

Online Learning

MATLAB Onramp
Free two-hour online MATLAB course

Videos

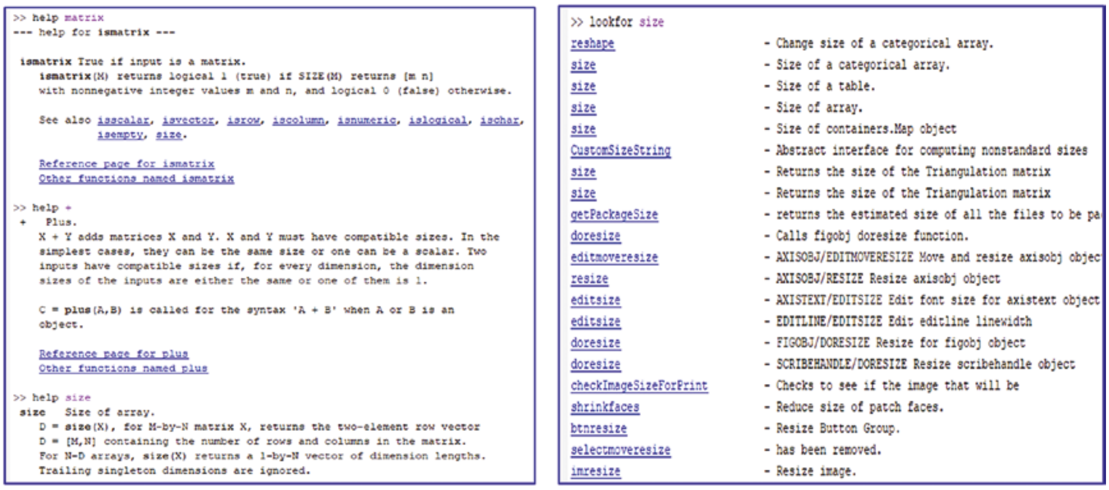
Figure 1-3. *Getting Started with MATLAB and Help Library documentation*

Note Before discussing the help options, I need to highlight one important point concerning comments. In MATLAB, users can write necessary hints and help remarks as comments within the M/MLX-files and in the Command Window. Comments need to start with an % sign. There are other options for adding comments, which I will discuss later when writing M/MLX-files.

There are a few other hands-on ways to obtain help. For example, to get quick help about MATLAB's built-in function tools and commands, including user-generated functions, users can type the following in the Command Window:

```
>> doc size;       % extended help on the command SIZE
>> helpwin size;   % help shown in a separate window on the command SIZE
>> help clear      % quick help on how to use the command CLEAR
>> help matrix;    % quick help how to use the command ISMATRIX
>> help +          % quick help on "+"
>> help size;      % quick help on SIZE
>> lookfor size    % extensive search for a list of functions and files
                     containing  the command SIZE.
```

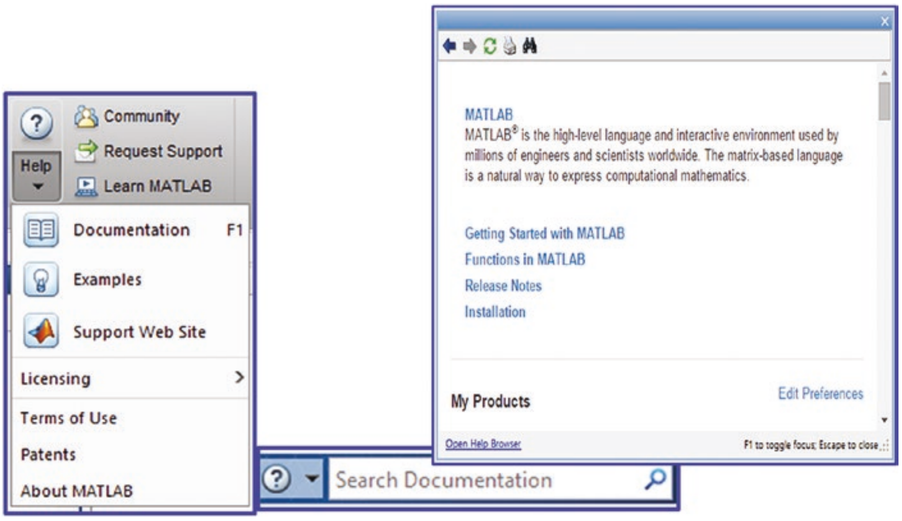
Figure 1-4 and Figure 1-5 show some of the results of quick and extensive help. In addition, the application has GUI tools and broad online library resources, product descriptions, video tutorials, and open public forums on the MathWorks website.



```
>> help matrix
--- help for ismatrix ---

 ismatrix True if input is a matrix.
    ismatrix(M) returns logical 1 (true) if SIZE(M) returns [m n]
    with nonnegative integer values m and n, and logical 0 (false) otherwise.

    See also isscalar, isvector, isrow, iscolumn, isnumeric, islogical, ischar,
             isempty, size.

    Reference page for ismatrix
    Other functions named ismatrix

>> help +
 +   Plus.
    X + Y adds matrices X and Y. X and Y must have compatible sizes. In the
    simplest cases, they can be the same size or one can be a scalar. Two
    inputs have compatible sizes if, for every dimension, the dimension
    sizes of the inputs are either the same or one of them is 1.

    C = plus(A,B) is called for the syntax 'A + B' when A or B is an
    object.

    Reference page for plus
    Other functions named plus

>> help size
 size   Size of array.
    D = size(X), for M-by-N matrix X, returns the two-element row vector
    D = [M,N] containing the number of rows and columns in the matrix.
    For N-D arrays, size(X) returns a 1-by-N vector of dimension lengths.
    Trailing singleton dimensions are ignored.
```

```
>> lookfor size
reshape                   - Change size of a categorical array.
size                      - Size of a categorical array.
size                      - Size of a table.
size                      - Size of array.
size                      - Size of containers.Map object
CustomSizeString          - Abstract interface for computing nonstandard sizes
size                      - Returns the size of the Triangulation matrix
size                      - Returns the size of the Triangulation matrix
getPackageSize            - returns the estimated size of all the files to be pa
doresize                  - Calls figobj doresize function.
editmoveresize            - AXISOBJ/EDITMOVERESIZE Move and resize axisobj objec
resize                    - AXISOBJ/RESIZE Resize axisobj object
editsize                  - AXISTEXT/EDITSIZE Edit font size for axistext object
editsize                  - EDITLINE/EDITSIZE Edit editline linewidth
doresize                  - FIGOBJ/DORESIZE Resize for figobj object
doresize                  - SCRIBEHANDLE/DORESIZE Resize scribehandle object
checkImageSizeForPrint    - Checks to see if the image that will be
shrinkfaces               - Reduce size of patch faces.
btnresize                 - Resize Button Group.
selectmoveresize          - has been removed.
imresize                  - Resize image.
```

Figure 1-4. *Getting help*



MATLAB

MATLAB® is the high-level language and interactive environment used by millions of engineers and scientists worldwide. The matrix-based language is a natural way to express computational mathematics.

Figure 1-5. *Getting help*

You can search for help in MATLAB using the Command Window and the `help`, `lookfor`, `doc`, `docsearch`, and `helpwin` commands for the `clock` function , for example:

1. Quick help can be obtained from the Command Window with the `help` command. In this case, help is displayed from MATLAB's built-in commands/functions as well as within the user's function files. This is a quick way to obtain help.

```
>> help clock
```

2. Extensive (detailed) help, with examples, is displayed in the Help Library window with the following commands, but only if such a function file (e.g., `clock`) exists.

```
>> doc  clock
>> docsearch clock
```

3. You can access an extended list of M-files containing a keyword from the Command Window by using the following help command. Note that this option is much slower than the other two search options, due to its exhaustive search for the keyword.

```
>> lookfor clock
```

4. You can view the function file explanation in the Help Library by using the following command, but only if such a function file (e.g., `clock`) exists.

   ```
   >> helpwin clock
   ```

5. All extended tips, examples, and command syntax can be viewed from the Help Library (displayed in Figure 1-3), which can be accessed by clicking on the Help menu options.
6. You can use the F1 keyboard key to open the Help Library and documentation.
7. By clicking on the Help menu from the main panel, you can access various help resources from MathWorks, such as its Help Library resources, web resources, demo examples, updates, trials, and so forth.

There are a number of hands-on help resources available online, including MathWorks website academia, the user community's published scripts and file exchanges[1], and the MATLAB answers forum [2], where learners/users/developers post their questions and seek answers, or conversely post their answers to posted questions. It also includes function files, Simulink models, online forums, tutorials of numerous universities [3], and personal web pages of professors and researchers [4], just to name a few.

The MATLAB Environment

Let's start working with the MATLAB environment by making some changes to its layout using its GUI tools, such as Layout, Preferences, and Set Path, which are located on the HOME tab. To make changes to the layout (see Figure 1-6) from the HOME tab's main menu, you have to click on the Layout drop-down menu (1). Many options and different windows are available to choose from. The Desktop window consists of Command, Command History, Current Directory, and Workspace windows if there are tick marks before those window names. You can separate or drag around any of these windows by clicking the title bar and dragging the window to the new location.