MATLAB® for Engineers

Holly Moore

Fourth Edition

SOURCE

MATLAB® for Engineers

This page intentionally left blank

MATLAB® for Engineers

Fourth Edition

HOLLY MOORE
Salt Lake Community College
Salt Lake City, Utah

PEARSON

Boston • Columbus • Indianapolis • New York
San Francisco • Upper Saddle River • Amsterdam
Cape Town • Dubai • London • Madrid • Milan
Munich • Paris • Montreal • Toronto • Delhi
Mexico City • São Paulo • Sydney • Hong Kong
Seoul • Singapore • Taipei • Tokyo

VP/Editorial Director, Engineering/Computer Science: *Marcia J. Horton*
Executive Editor: *Holly Stark*
Editorial Assistant: *Carlin Heinle*
Senior Marketing Manager: *Tim Galligan*
Senior Managing Editor: *Scott Disanno*
Project Manager: *Priyadharshini Dhanagopal*
Senior Art Director: *Jayne Conte*
Cover Designer: *Bruce Kenselaar*
Full-Service Project Management: *Pavithra Jayapaul, Jouve India*
Composition: *Jouve India*

Library of Congress Cataloging–in–Publication Data

Moore, Holly.
MATLAB for engineers / Holly Moore, Salt Lake Community College, Salt Lake City, Utah. – Fourth edition.
pages cm
ISBN-13: 978-0-13-348597-4
ISBN-10: 0-13-348597-8
1. Engineering mathematics—Data processing. 2. MATLAB. I. Title.
TA345.M585 2013
620.001'51—dc23
2013034649

10 9 8 7 6 5 4 3 2 1

PEARSON

www.pearsonhighered.com

ISBN 10: 0-13-348597-8
ISBN 13: 978-0-13-348597-4

Contents

About This Book

This book grew out of my experience teaching MATLAB® and other computing languages to freshmen engineering students at Salt Lake Community College. I was frustrated by the lack of a text that "started at the beginning." Although there were many comprehensive reference books, they assumed a level of both mathematical and computer sophistication that my students did not possess. Also, because MATLAB® was originally adopted by practitioners in the fields of signal processing and electrical engineering, most of these texts provided examples primarily from those areas, an approach that didn't fit with a general engineering curriculum. This text starts with basic algebra and shows how MATLAB® can be used to solve engineering problems from a wide range of disciplines. The examples are drawn from concepts introduced in early chemistry and physics classes and freshman and sophomore engineering classes. A standard problem-solving methodology is used consistently.

The text assumes that the student has a basic understanding of college algebra and has been introduced to trigonometric concepts; students who are mathematically more advanced generally progress through the material more rapidly. Although the text is not intended to teach subjects such as statistics or matrix algebra, when the MATLAB® techniques related to these subjects are introduced, a brief background is included. In addition, sections describing MATLAB® techniques for solving problems by means of calculus and differential equations are introduced near the end of appropriate chapters. These sections can be assigned for additional study to students with a more advanced mathematics background, or they may be useful as reference material as students progress through an engineering curriculum.

The book is intended to be a "hands-on" manual. My students have been most successful when they read the book while sitting beside a computer and typing in the examples as they go. Numerous examples are embedded in the text, with more complicated numbered examples included in each chapter to reinforce the concepts introduced. Practice exercises are included in each chapter to give students an immediate opportunity to use their new skills.

The material is grouped into three sections. The first, ***An Introduction to Basic MATLAB® Skills***, gets the student started and contains the following chapters:

- Chapter 1 shows how MATLAB® is used in engineering and introduces a standard problem-solving methodology.
- Chapter 2 introduces the MATLAB® environment and the skills required to perform basic computations. It also introduces M-files, and the concept of organizing code into cells. Doing so early in the text makes it easier for students to save their work and develop a consistent programming strategy.
- Chapter 3 details the wide variety of problems that can be solved with built-in MATLAB® functions. Background material on many of the functions is provided to help the student understand how they might be used. For example, the difference between Gaussian random numbers and uniform random numbers is described, and examples of each are presented.

- Chapter 4 demonstrates the power of formulating problems by using matrices in MATLAB® and expanding on the techniques employed to define those matrices. The **meshgrid** function is introduced in this chapter and is used to solve problems with two variables. The difficult concept of meshing variables is revisited in Chapter 5 when surface plots are introduced.
- Chapter 5 describes the wide variety of both two-dimensional and three-dimensional plotting techniques available in MATLAB®. Creating plots via MATLAB® commands, either from the command window or from within an M-file, is emphasized. However, the extremely valuable techniques of interactively editing plots and creating plots directly from the workspace window are also introduced.

  MATLAB® is a powerful programming language that includes the basic constructs common to most programming languages. Because it is a scripting language, creating programs and debugging them in MATLAB® is often easier than in traditional programming languages such as C++. This makes MATLAB® a valuable tool for introductory programming classes. The second section of the text, ***Programming in MATLAB®***, introduces students to programming and consists of the following chapters:
- Chapter 6 describes how to create and use user-defined functions. It also teaches students how to create a "toolbox" of functions to use in their own programming projects.
- Chapter 7 introduces functions that interact with the program user, including user-defined input, formatted output, and graphical input techniques. The use of MATLAB®'s debugging tools is also introduced.
- Chapter 8 describes logical functions such as **find** and demonstrates how they vary from the **if** and **if/else** structures. The **switch case** structure is also introduced. The use of logical functions over control structures is emphasized, partly because students (and teachers) who have previous programming experience often overlook the advantages of using MATLAB®'s built-in matrix functionality.
- Chapter 9 introduces repetition structures, including **for** loops, **while** loops, and midpoint break loops which utilize the **break** command. Numerous examples are included because students find these concepts particularly challenging.

Chapters 1 through 9 should be taught sequentially, but the chapters in Section 3, ***Advanced MATLAB® Concepts***, do not depend upon each other. Any or all of these chapters could be used in an introductory course or could serve as reference material for self-study. Most of the material is appropriate for freshmen. A two-credit course might include Chapters 1 through 9 plus Chapter 10, while a three-credit course might include Chapters 1 through 14, but eliminate Sections 12.4, 12.5, 13.4, 13.5, and 13.6, which describe differentiation techniques, integration techniques, and solution techniques for differential equations. Chapters 15 and 16 will be interesting to more advanced students, and might be included in a course delivered to sophomore or junior students instead of to freshmen. The skills developed in these chapters will be especially useful as students become more involved in solving engineering problems:

- Chapter 10 discusses problem solving with matrix algebra, including dot products, cross products, and the solution of linear systems of equations. Although matrix algebra is widely used in all engineering fields, it finds early application in the statics and dynamics classes taken by most engineering majors.

- Chapter 11 is an introduction to the wide variety of data types available in MATLAB®. This chapter is especially useful for electrical engineering and computer engineering students.
- Chapter 12 introduces MATLAB®'s symbolic mathematics package, built on the MuPad engine. Students will find this material especially valuable in mathematics classes. My students tell me that the package is one of the most valuable sets of techniques introduced in the course. It is something they start using immediately.
- Chapter 13 presents numerical techniques used in a wide variety of applications, especially curve fitting and statistics. Students value these techniques when they take laboratory classes such as chemistry or physics or when they take the labs associated with engineering classes such as heat transfer, fluid dynamics, or strengths of materials.
- Chapter 14 examines graphical techniques used to visualize data. These techniques are especially useful for analyzing the results of numerical analysis calculations, including results from structural analysis, fluid dynamics, and heat transfer codes.
- Chapter 15 introduces MATLAB®'s graphical user interface capability, using the GUIDE application. Creating their own GUI's gives students insight into how the graphical user interfaces they use daily on other computer platforms are created.
- Chapter 16 introduces Simulink®, which is a simulation package built on top of the MATLAB® platform. Simulink® uses a graphical user interface that allows programmers to build models of dynamic systems. It has found significant acceptance in the field of Electrical Engineering but has wide application across the engineering spectrum.

Appendix A lists all of the functions and special symbols (or characters) introduced in the text. Appendix B describes strategies for scaling data, so that the resulting plots are linear. Appendix C includes the complete MATLAB® code to create the `Ready_Aim_Fire` graphical user interface described in Chapter 15. An instructor website includes the following material:

- M-files containing solutions to practice exercises. (These files are also available on the student version of the website.)
- M-files containing solutions to example problems
- M-files containing solutions to homework problems
- PowerPoint slides for each chapter
- All of the figures used in the text, suitable for inclusion in your own PowerPoint presentations
- A series of lectures (including narration) suitable for use with online classes or as reviews

Appendix E Solutions to Practice Exercises can be found at the following website:

www.pearsonhighered.com/moore

WHAT'S NEW IN THIS EDITION

New versions of MATLAB® are rolled out every six months, which makes keeping any text up-to-date a challenge. Significant changes were introduced in version 2012b, including the introduction of MATLAB® 8 which has a redesigned user-interface. The changes in this edition reflect these software updates. They include:

- All of the screen shots throughout the book were updated to reflect the 2013a release.
- Many built-in graphical user interfaces (GUIs) are now packaged in MATLAB® as "apps." Apps are discussed in Chapter 2 and the MuPad app is introduced in Chapter 12.
- The creation of user-defined symbolic functions is introduced in Chapter 12.
- The behavior of several symbolic functions has changed, which is reflected in the content of Chapter 12.
- Additional problems were added and some problems were modified, based on the feedback from both instructors and students who have used the book.
- A number of new functions are introduced throughout the book, suggested to us by adopters of the text.

Dedication and Acknowledgments

This project would not have been possible without the support of my family. Thanks to Mike, Heidi, Meagan, and David, and to my husband, Dr. Steven Purcell. I also benefited greatly from the suggestions for problems related to electricity from Lee Brinton and Gene Riggs of the SLCC Electrical Engineering Department. Their cheerful efforts to educate me on the mysteries of electricity are much appreciated. I'd also like to thank Quentin McRae, also at SLCC, who made numerous suggestions that improved the homework problems.

This book is dedicated to my father, Professor George E. Moore, who taught in the Department of Electrical Engineering at the South Dakota School of Mines and Technology for almost 20 years. Professor Moore earned his college degree at the age of 54 after a successful career as a pilot in the United States Air Force and was a living reminder that you are never too old to learn. My mother, Jean Moore, encouraged both him and her two daughters to explore outside the box. Her loving support made it possible for both my sister and I to enjoy careers in engineering—something few women attempted in the early 1970s. I hope that readers of this text will take a minute to thank those people in their lives who've helped them make their dreams come true. Thanks Mom and Dad.

This page intentionally left blank

CHAPTER

1 About MATLAB®

Objectives

After reading this chapter, you should be able to:

- Understand what MATLAB® is and why it is widely used in engineering and science
- Understand the advantages and limitations of the student edition of MATLAB®
- Formulate problems using a structured problem-solving approach

1.1 WHAT IS MATLAB®?

MATLAB® is one of a number of commercially available, sophisticated mathematical computation tools, which also include Maple, Mathematica, and MathCad. Despite what proponents may claim, no single one of these tools is "the best." Each has strengths and weaknesses. Each allows you to perform basic mathematical computations. They differ in the way they handle symbolic calculations and more complicated mathematical processes, such as matrix manipulation. For example, MATLAB® (short for **Mat**rix **Lab**oratory) excels at computations involving matrices, whereas Maple excels at symbolic calculations. At a fundamental level, you can think of these programs as sophisticated computer-based calculators. They can perform the same functions as your scientific calculator—and **many more**. If you have a computer on your desk, you may find yourself using MATLAB® instead of your calculator for even the simplest mathematical applications—for example, balancing your checkbook. In many engineering classes, the use of programs such as MATLAB® to perform computations is replacing more traditional computer programming. Although programs such as MATLAB® have become a standard tool for engineers and scientists, this doesn't mean that you shouldn't learn a high-level language such as C++, JAVA, or FORTRAN.

Because MATLAB® is so easy to use, you can perform many programming tasks with it, but it isn't always the best tool for a programming task. It excels at numerical calculations—especially matrix calculations—and graphics, but you wouldn't want to use it to write a word-processing program. For large applications, such as operating systems or design software, C++, JAVA, or FORTRAN would be the programs of choice. (In fact, MATLAB®, which *is* a large application program, was originally written in FORTRAN and later rewritten in C, a precursor of C++.) Usually, high-level programs do not offer easy access to graphing—an application at which MATLAB® excels. The primary area of overlap between MATLAB® and high-level programs is "number crunching"—repetitive calculations or the processing of large quantities of data. Both MATLAB® and high-level programs are good at processing numbers. A "number-crunching" program is generally easier to write in MATLAB®, but usually it will execute faster in C++ or FORTRAN. The one exception to this rule is calculations involving matrices. MATLAB® is optimized for matrices. Thus, if a problem can be formulated with a matrix solution, MATLAB® executes substantially faster than a similar program in a high-level language.

KEY IDEA
MATLAB® is optimized for matrix calculations

MATLAB® is available in both professional and student versions. The professional version is probably installed in your college or university computer laboratory, but you may enjoy having the student version at home. MATLAB® is updated regularly; this textbook is based on MATLAB® 8.1. If you are using earlier versions such as MATLAB® 6 or 7, you will notice a significant difference in the layout of the graphical user interface; however, the differences in coding approaches are minor. There are substantial differences in versions that predate MATLAB® 5.5.

The standard installation of the professional version of MATLAB® is capable of solving various technical problems. Additional capability is available in the form of function toolboxes. These toolboxes are purchased separately, and they may or may not be available to you. You can find a complete list of the MATLAB® product family at The MathWorks web site, www.mathworks.com.

1.2 STUDENT EDITION OF MATLAB®

KEY IDEA
MATLAB® is regularly updated

The professional and student editions of MATLAB® are very similar. Beginning students probably won't be able to tell the difference. Student editions are available for Microsoft Windows, Mac, and Linux operating systems and can be purchased from college bookstores or online from The MathWorks at www.mathworks.com.

The MathWorks packages its software in groups called *releases,* and MATLAB® 8.1 is featured, along with other products, such as Simulink, in Release R2013a. New versions are released every six months. The release number is the same for both the student and professional editions, but the student version may lag the professional version by several months. The student edition of R2013a includes the following features:

- Full MATLAB®
- Simulink, with the ability to build models with up to 1000 blocks (the professional version allows an unlimited number of blocks)
- Symbolic Math Toolbox
- Control Systems Toolbox
- Data Acquisition Toolbox
- Instrument Control Toolbox
- Simulink Control Design
- Signal Processing Toolbox
- DSP System Toolbox
- Statistics Toolbox
- Optimization Toolbox

- Image Processing Toolbox
- A single-user license, limited to students for use in their classwork (the professional version is licensed either singly or to a group)

Toolboxes other than those included with the student edition may be purchased separately. You should be aware that if you are using a professional installation of MATLAB®, all of the toolboxes available in the student edition may not be available to you.

The biggest difference you should notice between the professional and student editions is the command prompt, which is

```
>>
```

in the professional version and

```
EDU>>
```

in the student edition.

1.3 HOW IS MATLAB® USED IN INDUSTRY?

The ability to use tools such as MATLAB® is quickly becoming a requirement for many engineering positions. A recent job search on Monster.com found the following advertisement:

> . . . is looking for a System Test Engineer with Avionics experience Responsibilities include modification of MATLAB® scripts, execution of Simulink simulations, and analysis of the results data. Candidate MUST be very familiar with MATLAB®, Simulink, and C++ . . .

This advertisement isn't unusual. The same search turned up 771 different companies that specifically required MATLAB® skills for entry-level engineers. Widely used in all engineering and science fields, MATLAB® is particularly popular for electrical engineering applications. The sections that follow outline a few of the many applications currently using MATLAB®.

KEY IDEA
MATLAB® is widely used in engineering

1.3.1 Electrical Engineering

MATLAB® is used extensively in electrical engineering for a wide variety of applications. For example, Figure 1.1 includes several images created to help visualize the arrangements of electromagnetic fields in space and time. These images represent real physical situations with practical application. Cellular communications, medical diagnostics, and home computers are just a few of the technologies that exist thanks to our understanding of this beautiful phenomenon.

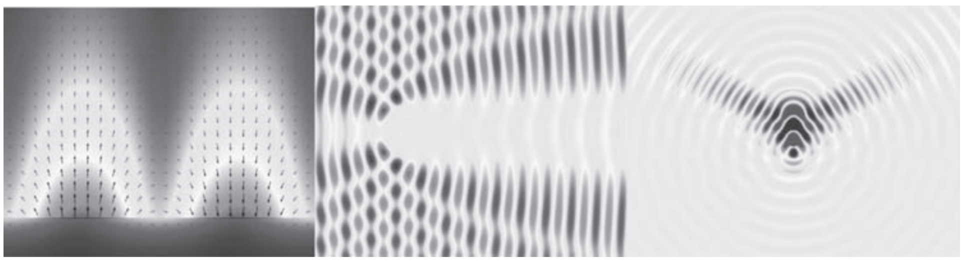


Figure 1.1
Arrangements of Electromagnetic Fields. (a) Surface Plasmon Polariton; (b) Light Scattering by a Circular Metal Cylinder (c) Beam forming by a Six-Element Dipole Array. (Used with permission of Dr. James R. Nagel, University of Utah Department of Electrical and Computer Engineering).

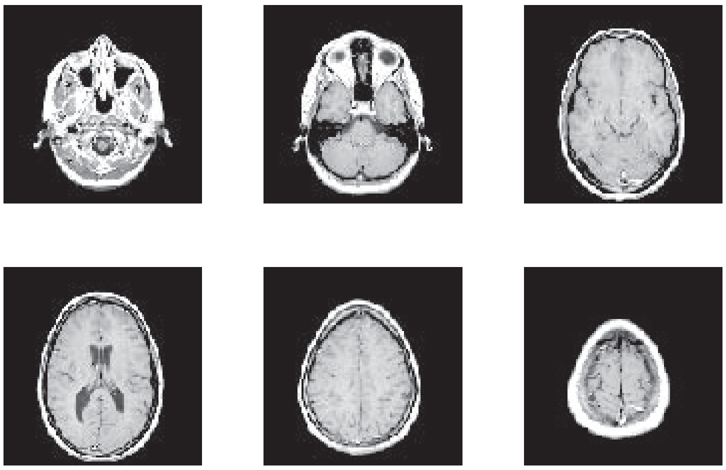


Figure 1.2
Horizontal slices through the brain, based on the sample data file included with MATLAB®.

1.3.2 Biomedical Engineering

Medical images are usually saved as dicom files (the Digital Imaging and Communications in Medicine standard). Dicom files use the file extension .dcm. The MathWorks offers an Image Processing Toolbox that can read these files, making their data available to MATLAB®. (The Image Processing Toolbox is included with the student edition and is optional with the professional edition.) The Image Processing Toolbox also includes a wide range of functions, many of them especially appropriate for medical imaging. A limited MRI data set that has already been converted to a format compatible with MATLAB® ships with the standard MATLAB® program. This data set allows you to try out some of the imaging functions available both with the standard MATLAB® installation and with the expanded imaging toolbox, if you have it installed on your computer. Figure 1.2 shows six images of horizontal slices through the brain based on the MRI data set.

The same data set can be used to construct a three-dimensional image, such as either of those shown in Figure 1.3. Detailed instructions on how to create these images are included in the MATLAB® tutorial, accessed from the help button on the MATLAB® toolbar.

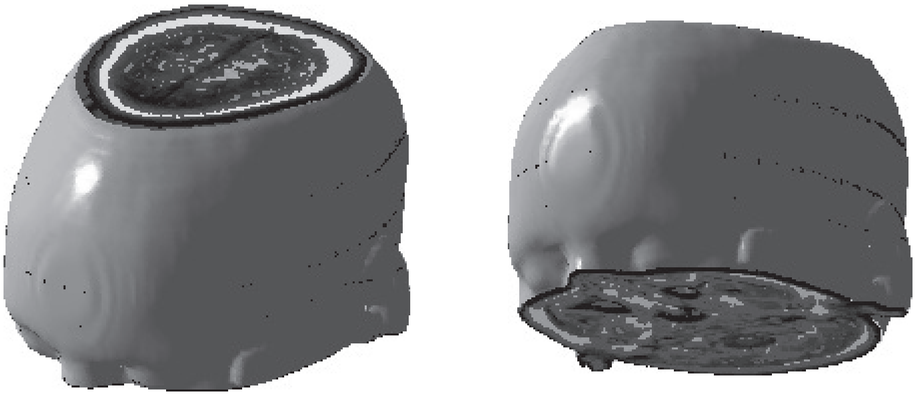
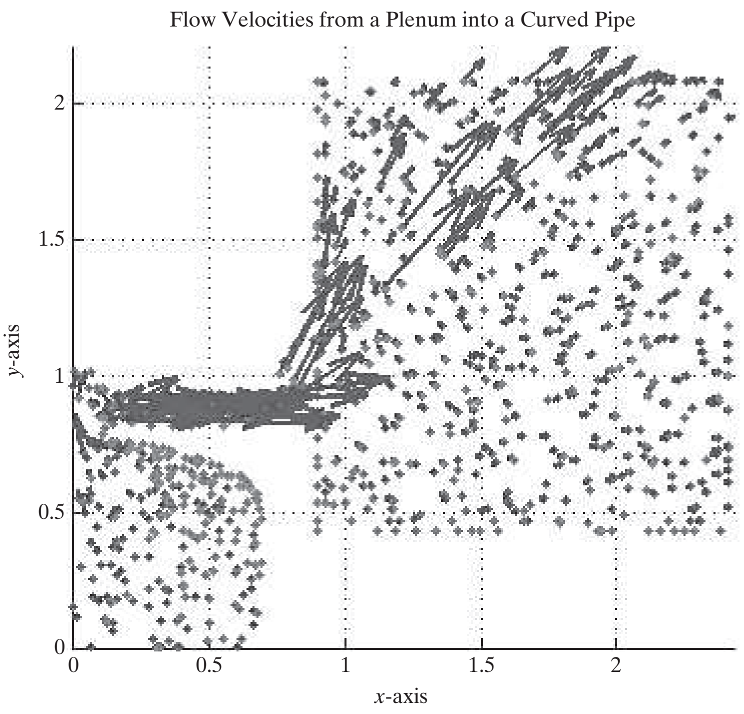


Figure 1.3
Three-dimensional visualization of MRI data, based on the sample data set included with MATLAB®.

Figure 1.4
Quiver plot of gas behavior in a thrust-vector control device.



1.3.3 Fluid Dynamics

Calculations describing fluid velocities (speeds and directions) are important in a number of different fields. Aerospace engineers in particular are interested in the behavior of gases, both outside an aircraft or space vehicle and inside the combustion chambers. Visualizing the three-dimensional behavior of fluids is tricky, but MATLAB® offers a number of tools that make it easier. Figure 1.4 represents the flow-field calculation results for a thrust-vector control device as a quiver plot. Thrust-vector control is the process of changing the direction in which a nozzle points (and hence the direction a rocket travels) by pushing on an actuator (a piston-cylinder device). The model in the figure represents a high-pressure reservoir of gas (a plenum) that eventually feeds into the piston and thus controls the length of the actuator.

1.4 PROBLEM SOLVING IN ENGINEERING AND SCIENCE

KEY IDEA
Always use a systematic problem-solving strategy

A consistent approach to solving technical problems is important throughout engineering, science, and computer programming disciplines. The approach we outline here is useful in courses as diverse as chemistry, physics, thermodynamics, and engineering design. It also applies to the social sciences, such as economics and sociology. Different authors may formulate their problem-solving schemes differently, but they all have the same basic format:

- **State the problem**.
  - Drawing a picture is often helpful in this step.
  - If you do not have a clear understanding of the problem, you are not likely to be able to solve it.

- **Describe the input** values (knowns) **and** the required **outputs** (unknowns).
  - Be careful to include units as you describe the input and output values. Sloppy handling of units often leads to wrong answers.
  - Identify constants you may need in the calculation, such as the ideal-gas constant and the acceleration due to gravity.
  - If appropriate, label a sketch with the values you have identified, or group them into a table.
- Develop an algorithm to solve the problem. In computer applications, this can often be accomplished with a **hand example**. You'll need to
  - Identify any equations relating the knowns and unknowns.
  - Work through a simplified version of the problem by hand or with a calculator.
- **Solve** the problem. In this book, this step involves creating a **MATLAB® solution**.
- **Test the solution**.
  - Do your results make sense physically?
  - Do they match your sample calculations?
  - Is your answer really what was asked for?
  - Graphs are often useful ways to check your calculations for reasonableness.

If you consistently use a structured problem-solving approach, such as the one just outlined, you'll find that "story" problems become much easier to solve. Example 1.1 illustrates this problem-solving strategy.

EXAMPLE 1.1

THE CONVERSION OF MATTER TO ENERGY

Albert Einstein (Figure 1.5) is arguably the most famous physicist of the 20th century. He was born in Germany in 1879 and attended school in both Germany and Switzerland. While working as a patent clerk in Bern, he developed his famous theory of relativity. Perhaps the best-known physics equation today is his

$$E = mc^2.$$

This astonishingly simple equation links the previously separate worlds of matter and energy and can be used to find the amount of energy released as matter is changed in form in both natural and human-made nuclear reactions.

The sun radiates 385×10^{24} J/s of energy, all of which is generated by nuclear reactions converting matter to energy. Use MATLAB® and Einstein's equation to determine how much matter must be converted to energy to produce this much radiation in one day.

1. State the Problem
   Find the amount of matter necessary to produce the amount of energy radiated by the sun every day.
2. Describe the Input and Output

   Input

Energy:	$E = 385 \times 10^{24}$ J/s, which must be converted into the total energy radiated during one day
Speed of light:	$c = 3.0 \times 10^8$ m/s

   Output

   Mass m in kg

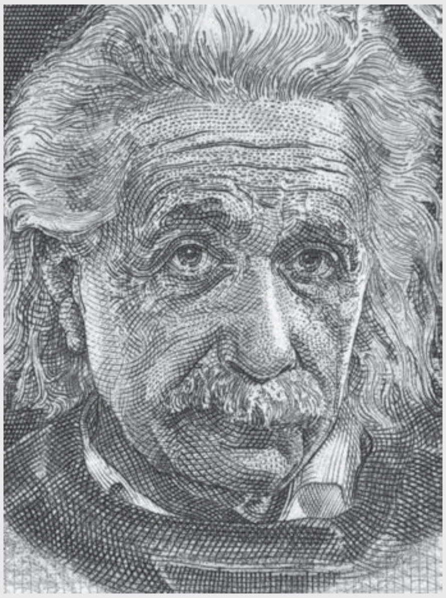


Figure 1.5
Albert Einstein.

3. Develop a Hand Example
The energy radiated in one day is

$$385 \times 10^{24}\,\mathrm{J/s} \times 3600\,\mathrm{s/h} \times 24\,\mathrm{h/day} \times 1\,\mathrm{day} = 3.33 \times 10^{31}\,\mathrm{J}$$

The equation $E = mc^2$ must be solved for m and the values for E and c substituted. Thus

$$m = \frac{E}{c^2}$$

$$m = \frac{3.33 \times 10^{31}\,\mathrm{J}}{(3.0 \times 10^{8}\,\mathrm{m/s})^2}$$

$$= 3.7 \times 10^{14}\,\mathrm{J/m^2s^2}$$

We can see from the output criteria that we want the mass in kg, so what went wrong? We need to do one more unit conversion:

$$1\,\mathrm{J} = 1\ \mathrm{kg\ m^2/s^2}$$

$$= 3.7 \times 10^{14}\frac{\mathrm{kg\ m^2/s^2}}{\mathrm{m^2/s^2}} = 3.7 \times 10^{14}\,\mathrm{kg}$$

4. Develop a MATLAB® Solution
At this point, you have not learned how to create MATLAB® code. However, you should be able to see from the following sample code that MATLAB® syntax is similar to that used in most algebraic scientific calculators. MATLAB® commands are entered at the prompt (>>), and the results are reported on the next line. The code is as follows:

```
>> E=385e24    The user types in this information
E =
   3.8500e+026    This is the computer's response
```

```
>> E=E*3600*24
E =
   3.3264e+031
>> c=3e8
c =
   300000000
>> m=E/c^2
m =
   3.6960e+014
```

From this point on, we will not show the prompt when describing interactions in the command window.

5. Test the Solution
The MATLAB® solution matches the hand calculation, but do the numbers make sense? Anything times 10^{14} is a really large number. Consider, however, that the mass of the sun is 2×10^{30} kg. We can calculate how long it would take to consume the mass of the sun completely at a rate of 3.7×10^{14} kg/day. We have

$$\text{Time} = \frac{\text{Mass of the sun}}{\text{Rate of consumption}}$$

$$\text{Time} = \frac{2 \times 10^{30}\ \text{kg}}{3.7 \times 10^{14}\ \text{kg/day}} \times \frac{\text{year}}{365\ \ \text{days}} = 1.5 \times 10^{13}\ \text{years}$$

That's 15 trillion years! We don't need to worry about the sun running out of matter to convert to energy in our lifetimes.

CHAPTER

2 MATLAB® Environment

Objectives

After reading this chapter, you should be able to:

- Start the MATLAB® program and solve simple problems in the command window
- Understand MATLAB®'s use of matrices
- Identify and use the various MATLAB® windows
- Define and use simple matrices
- Name and use variables
- Understand the order of operation used in MATLAB®
- Understand the difference between scalar, array, and matrix calculations in MATLAB®
- Express numbers in either floating-point or scientific notation
- Adjust the format used to display numbers in the command window
- Save the value of variables used in a MATLAB® session
- Save a series of commands in an M-file
- Use Cell Mode

2.1 GETTING STARTED

Using MATLAB® for the first time is easy; mastering it can take years. In this chapter, we will introduce you to the MATLAB® environment and show you how to perform basic mathematical computations. After reading this chapter, you should be able to start using MATLAB® for homework assignments or on the job. Of course, you will be able to do more things as you complete the rest of the chapters.

Because the procedure for installing MATLAB® depends upon your operating system and your computing environment, we will assume that you have already installed MATLAB® on your computer or that you are working in a computing laboratory with MATLAB® already installed. To start MATLAB® in either the Windows or Apple environment, click on the icon on the desktop, or use the start menu to find the program. In the UNIX environment, type `Matlab` at the shell prompt. No matter how you start it, once MATLAB® opens, you should see the MATLAB® prompt (`>>` or `EDU>>`), which