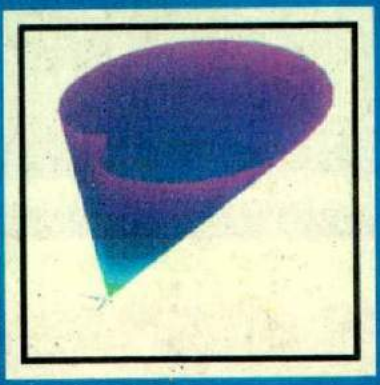
THOMSON

INDIA EDITION



MATLAB® Programming for Engineers

Third Edition

Stephen J. Chapman

Bookware Companion Series

MATLAB®
Programming for Engineers

·d Edition

ephen J. Chapman
YSTEMS Australia

THOMSON
LEARNING

Australia · Canada · Mexico · Singapore · Spain · United Kingdom · United State

THOMSON
LEARNING

MATLAB® Programming for Engineers, Third Edition
by Stephen J. Chapman

Associate Vice-President
and Editorial Director:
Evelyn Veitch

Publisher:
Bill Stenquist

Sales and Marketing Manager:
John More

Developmental Editor:
Kamilah Reid Burrell

Production Service:
RPK Editorial Services

Copy Editor/Proofing:
Harlan James

Indexer:
Shelly Gerger-Knechtl

Production Manager:
Renate McCloy

Creative Director:
Angela Cluer

Cover Design:
Peter Papayanakis

Compositor:
PV & M Publishing Solutions

First Reprint 2004 by Thomson Asia Pte Ltd., Singapore

Second Reprint 2005

ISBN: 981-254-893-9

Printed and bound at Eastern Press (Bangalore) Pvt. Ltd., 100% EOU,
110B, Bommasandra Industrial Area, Hosur Road, Anekal Taluk,
Bangalore 560 099 (INDIA).

2 3 4 5 08 07 06 05

Library of Congress Control
Number: 2004095122

ISBN: 0-534-42417-1

For sale in the Indian Subcontinent only.

This book is dedicated to my wife Rosa,
the mother of our eight children,
after almost 30 wonderful years together.

Preface

MATLAB (short for MATrix LABoratory) is a special-purpose computer program optimized to perform engineering and scientific calculations. It started life as a program designed to perform matrix mathematics, but over the years it has grown into a flexible computing system capable of solving essentially any technical problem.

The MATLAB program implements the MATLAB language and provides a very extensive library of predefined functions to make technical programming tasks easier and more efficient. This extremely wide variety of functions makes it much easier to solve technical problems in MATLAB than in other languages such as Fortran or C. This book introduces the MATLAB language and shows how to use it to solve typical technical problems.

This book teaches MATLAB as a technical programming language, showing students how to write clean, efficient, and documented programs. It makes no pretense at being a complete description of all of MATLAB's hundreds of functions. Instead, it teaches the student how to use MATLAB as a language, and how to locate any desired function with MATLAB's extensive on-line help facilities.

The first six chapters of the text are designed to serve as the text for an "Introduction to Programming/Problem Solving" course for freshman engineering students. This material should fit comfortably into a 9-week, 3-hour course. The remaining chapters cover advanced topics such as I/O and graphical user interfaces. These chapters may be covered in a longer course or used as a reference by engineering students or practicing engineers who use MATLAB as a part of their coursework or employment.

Changes in the Third Edition

The third edition of this book is specifically devoted to MATLAB 7. MATLAB 7 contains many language and tool changes, and this book had to be revised extensively for the new version. Some of the major changes include:

- Case-sensitive function and directory names on all platforms.
- Function handles.
- The use of an `end` statement at the end of functions.
- Nested functions.
- Math operations with single and integer data types.
- Major revision of the GUI code, including the addition of panels, button groups, and toolbars. The code auto-generated by `guide` has been totally changed. Frames have been deprecated.
- Major revisions to programming tools, such as the addition of conditional breakpoints and the `mlint` tool to check for poor programming practices within an M-file.

The changes in MATLAB are so major that many MATLAB 7 programs (especially GUIs) may not be backward compatible with earlier versions of MATLAB. If you are working with an earlier version of MATLAB, do not use this book—get the Second Edition instead. Both the Second Edition and the Third Edition of this book will remain available until the transition to MATLAB 7 is essentially complete.

The Advantages of MATLAB for Technical Programming

MATLAB has many advantages compared to conventional computer languages for technical problem solving. These include:

1. **Ease of Use**
   MATLAB is an interpreted language, like many versions of Basic. Like Basic, it is very easy to use. The program can be used as a scratch pad to evaluate expressions typed at the command line, or it can be used to execute large prewritten programs. Programs may be easily written and modified with the built-in integrated development environment and debugged with the MATLAB debugger. Because the language is so easy to use, it is ideal for educational use and for the rapid prototyping of new programs.

   Many program development tools are provided to make the program easy to use. They include an integrated editor/debugger, on-line documentation and manuals, a workspace browser, and extensive demos.

2. **Platform Independence**
   MATLAB is supported on many different computer systems, providing a large measure of platform independence. At the time of this writing, the language is supported on Windows NT/2000/XP, Linux, Unix, and the

Macintosh. Programs written on any platform will run on all of the other platforms, and data files written on any platform may be read transparently on any other platform. As a result, programs written in MATLAB can migrate to new platforms when the needs of the user change.

3. **Pre-Defined Functions**
MATLAB comes complete with an extensive library of predefined functions that provide tested and prepackaged solutions to many basic technical tasks. For example, suppose that you are writing a program that must calculate the statistics associated with an input data set. In most languages, you would need to write your own subroutines or functions to implement calculations such as the arithmetic mean, standard deviation, median, amonh others. These and hundreds of other functions are built right into the MATLAB language, making your job much easier.

   In addition to the large library of functions built into the basic MATLAB language, there are many special-purpose toolboxes available to help solve complex problems in specific areas. For example, a user can buy standard toolboxes to solve problems in signal processing, control systems, communications, image processing, and neural networks, among many others.

4. **Device-Independent Plotting**
Unlike other computer languages, MATLAB has many integral plotting and imaging commands. The plots and images can be displayed on any graphical output device supported by the computer on which MATLAB is running. This capability makes MATLAB an outstanding tool for visualizing technical data.

5. **Graphical User Interface**
MATLAB includes tools that allow a program to interactively construct a graphical user interface (GUI) for his or her program. With this capability, the programmer can design sophisticated data analysis programs that can be operated by relatively-inexperienced users.

6. **MATLAB Compiler**
MATLAB's flexibility and platform independence are achieved by compiling MATLAB programs into a device-independent p-code and then interpreting the p-code instructions at run-time. This approach is similar to that used by Microsoft's Visual Basic language. Unfortunately, the resulting programs can sometimes execute slowly because the MATLAB code is interpreted rather than compiled. We will point out features that tend to slow program execution when we encounter them.

   A separate MATLAB compiler is available. This compiler can compile a MATLAB program into a true executable that runs faster than the interpreted code. It is a great way to convert a prototype MATLAB program into an executable suitable for sale and distribution to users.

Features of This Book

Many features of this book are designed to emphasize the proper way to write reliable MATLAB programs. These features should serve a student well as he or she is first learning MATLAB and should also be useful to the practitioner on the job. They include:

1. **Emphasis on Top-Down Design Methodology**
   The book introduces a top-down design methodology in Chapter 3 and uses it consistently throughout the rest of the book. This methodology encourages a student to think about the proper design of a program *before* beginning to code. It emphasizes the importance of clearly defining the problem to be solved and the required inputs and outputs before any other work is begun. Once the problem is properly defined, it teaches the student to employ stepwise refinement to break the task down into successively smaller subtasks and to implement the subtasks as separate subroutines or functions. Finally, it teaches the importance of testing at all stages of the process, both unit testing of the component routines and exhaustive testing of the final product.

   The formal design process taught by the book may be summarized as follows:

   1. *Clearly state the problem that you are trying to solve.*
   2. *Define the inputs required by the program and the outputs to be produced by the program.*
   3. *Describe the algorithm that you intend to implement in the program.* This step involves top-down design and stepwise decomposition, using pseudocode or flow charts.
   4. *Turn the algorithm into MATLAB statements.*
   3. *Test the MATLAB program.* This step includes unit testing of specific functions, and also exhaustive testing of the final program with many different data sets.

2. **Emphasis on Functions**
   The book emphasizes the use of functions to logically decompose tasks into smaller subtasks. It teaches the advantages of functions for data hiding. It also emphasizes the importance of unit testing functions before they are combined into the final program. In addition, the book teaches about the common mistakes made with functions, and how to avoid them.

3. **Emphasis on MATLAB Tools**
   The book teaches the proper use of MATLAB's built-in tools to make programming and debugging easier. The tools covered include the Editor/Debugger, Workspace Browser, Help Browser, and GUI design tools.

4. **Good Programming Practice Boxes**
   These boxes highlight good programming practices when they are introduced for the convenience of the student. In addition, the good programming practices introduced in a chapter are summarized at the end of the chapter. An example Good Programming Practice Box is shown below.

Good Programming Practice

Always indent the body of an `if` construct by two or more spaces to improve the readability of the code.

5. **Programming Pitfalls Boxes**
   These boxes highlight common errors so that they can be avoided. An example Programming Pitfalls Box is shown below.

Programming Pitfalls

Make sure that your variable names are unique in the first 63 characters. Otherwise, MATLAB will not be able to tell the difference between them.

6. **Emphasis on Data Structures**
   Chapter 7 contains a detailed discussion of MATLAB data structures, including sparse arrays, cell arrays, and structure arrays. The proper use of these data structures is illustrated in the chapters on handle graphics and graphical user interfaces.

Pedagogical Features

The first six chapters of this book are specifically designed to be used in a freshman "Introduction to Program/Problem Solving" course. It should be possible to cover this material comfortably in a 9-week, 3 hour-per-week course. If there is insufficient time to cover all of the material in a particular engineering program, Chapter 6 may be deleted, the remaining material will still teach the fundamentals of programming and using MATLAB to solve problems. This feature should appeal to harassed engineering educators trying to cram ever more material into a finite curriculum.

The remaining chapters cover advanced material that will be useful to engineers and engineering students as they progress in their careers. This material includes advanced I/O and the design of graphical user interfaces for programs.

The book includes several features designed to aid student comprehension. A total of 15 quizzes appear scattered throughout the chapters, with answers to all questions included in Appendix B. These quizzes can serve as a useful self-test of comprehension. In addition, there are approximately 160 end-of-chapter exercises. Answers to all exercises are included in the Instructor's Manual. Good programming practices are highlighted in all chapters with special Good Programming Practice boxes, and common errors are highlighted in Programming Pitfalls boxes. End-of-chapter materials include Summaries of Good Programming Practice and Summaries of MATLAB Commands and Functions.

The book is accompanied by an Instructor's Manual, containing the solutions to all end-of-chapter exercises. The source code for all examples in the book is available from the book's Web site, and the source code for all solutions in the Instructor's Manual is available separately to instructors.

A Final Note to the User

No matter how hard I try to proofread a document like this book, it is inevitable that some typographical errors will slip through and appear in print. If you should spot any such errors, please drop me a note via the publisher, and I will do my best to get them eliminated from subsequent printings and editions. Thank you very much for your help in this matter.

I will maintain a complete list of errata and corrections at the book's World Wide Web site, which is http://www.engineering.thomsonlearning.com. Please check that site for any updates and/or corrections.

Acknowledgments

I would like to thank Bill Stenquist and the crew at Thomson for the support they have given me in getting this book to market. It has been gratifying to see the user response to the first and second editions, which were the result of our joint efforts.

In addition, I would like to thank my wife Rosa and our children—Avi, David, Rachel, Aaron, Sarah, Naomi, Shira, and Devorah—for being such delightful people and the inspiration for my efforts.

STEPHEN J. CHAPMAN
Melbourne, Australia

Contents

Chapter 2 MATLAB Basics 21

Chapter 3 Branching Statements and Program Design 85

Chapter 4 Loops 147