

Engineering
Computation
with **MATLAB**[®]

Third Edition

DAVID M. SMITH
Georgia Institute of Technology

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

Editorial Director: Marcia Horton
Editor in Chief: Michael Hirsch
Editorial Assistant: Emma Snider
Director of Marketing: Patrice Jones
Marketing Manager: Yez Alayan
Marketing Assistant: Kathryn Ferranti
Vice President of Production: Vince O’Brien
Managing Editor: Jeff Holcomb
Production Project Manager: Kayla Smith-Tarbox
Production Editor: Pat Brown
Senior Operations Supervisor: Alan Fischer
Manufacturing Buyer: Pat Brown

Art Director: Anthony Gemmellaro
Creative Director: Jayne Conte
Designer: Kathy Foot
Manager, Visual Research: Karen Sanatar
Manager, Rights and Permissions: Michael Joyce
Media Director: Daniel Sandin
Full-Service Project Management: Kailash Jadli/
Aptara® Inc.
Composition: Aptara®, Inc.
Printer/Binder: Courier/Westford
Cover Printer: Lehigh-Phoenix Color
Text Font: Palatino Lt Std

Photo Credits: Cover image: The cover image is of a three-dimensional model of HMS Victory developed entirely in Matlab by the Author. Page 4: Fig. 1.1: David Smith; Fig 1.2: Courtesy of the UK Government. Page 25: Fig 2.1: Screenshots copyright © 2011 by MathWorks. Reprinted with permission. Page 28: Fig 2.2, Fig 2.3: Screenshots copyright © 2011 by MathWorks. Reprinted with permission. Page 29: Fig 2.4, Fig 2.5: Screenshots copyright © 2011 by MathWorks. Reprinted with permission. Page 30: Fig 2.6: Screenshots copyright © 2011 by MathWorks. Reprinted with permission. Page 31: Fig 2.7: Screenshots copyright © 2011 by MathWorks. Reprinted with permission. Page 32: Fig 2.8: Screenshots copyright © 2011 by MathWorks. Reprinted with permission. Page 37: Fig. 2.9: AP Photo/Scaled Composites. Page 200: Fig 9.3: © PhotoDisc. Page 302: Fig 13.10: David Smith. Page 397: Fig 17.4: jorgen mcleman/Shutterstock.

Microsoft® and Windows® are registered trademarks of the Microsoft Corporation in the U.S.A. and other countries. Screen shots and icons reprinted with permission from the Microsoft Corporation. This book is not sponsored or endorsed by or affiliated with the Microsoft Corporation.

Copyright © 2013, 2010, 2008 by Pearson Education, Inc., publishing as Addison-Wesley. All rights reserved. Manufactured in the United States of America. This publication is protected by Copyright, and permission should be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission(s) to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to 201-236-3290.

Many of the designations by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed in initial caps or all caps.

Library of Congress Cataloging-in-Publication Data

Smith, David M.
Engineering and computation with MATLAB/David M. Smith.—3rd ed.
p. cm.
ISBN 978-0-13-256870-8
1. Engineering mathematics—Data processing. 2. MATLAB. I. Title.
TA345.S585 2013
620.00285’53—dc23

2012000377

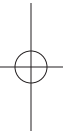
10 9 8 7 6 5 4 3 2 1

PEARSON

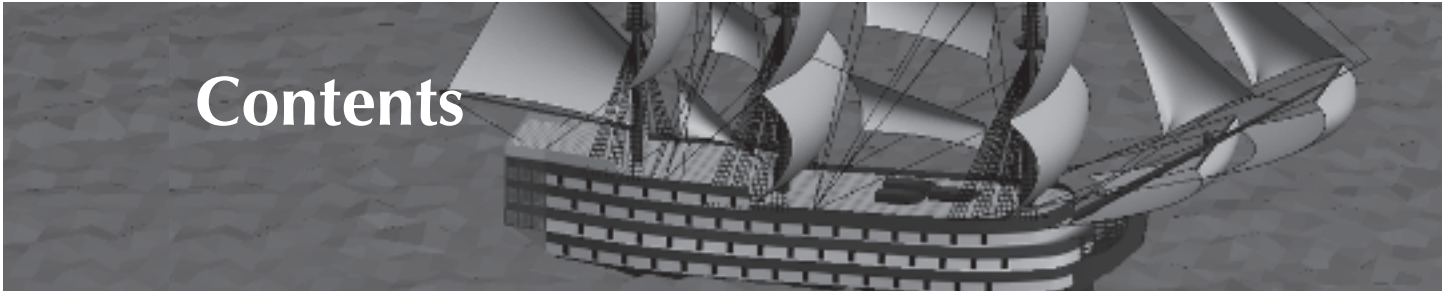
ISBN 10: 0-13-256870-5
ISBN 13: 978-0-13-256870-8

*This book is dedicated to the
glory of Almighty God*

~David M. Smith







Chapter 1	Introduction to Computers and Programming	1
1.1	Background	2
1.2	History of Computer Architectures	3
1.2.1	Babbage's Difference Engine	3
1.2.2	Colossus	3
1.2.3	The von Neumann Architecture	4
1.3	Computing Systems Today	5
1.3.1	Computer Hardware	5
1.3.2	Computer Memory	6
1.3.3	Computer Software	8
1.3.4	Running a Computer Program	11
1.4	Running an Interpreted Program	13
1.5	Anticipated Outcomes	13
1.5.1	Introduction to MATLAB	13
1.5.2	Learning Programming Concepts	14
1.5.3	Problem-Solving Skills	14
Chapter 2	Getting Started	17
2.1	Programming Language Background	18
2.1.1	Abstraction	18
2.1.2	Algorithms	19
2.1.3	Programming Paradigms	20
2.2	Basic Data Manipulation	20
2.2.1	Starting and Stopping MATLAB	20
2.2.2	Assigning Values to Variables	21
2.2.3	Data Typing	22
2.2.4	Classes and Objects	24
2.3	MATLAB User Interface	24
2.3.1	Command Window	25
2.3.2	Command History	26
2.3.3	Workspace Window	27
2.3.4	Current Directory Window	30
2.3.5	Variable Editor	31
2.3.6	Figure Window	31
2.3.7	Editor Window	32
2.4	Scripts	33
2.4.1	Text Files	33
2.4.2	Creating Scripts	33
2.4.3	The Current Directory	34
2.4.4	Running Scripts	35
2.4.5	Punctuating Scripts	35
2.4.6	Debugging Scripts	36
2.5	Engineering Example—Spacecraft Launch	36

Chapter 3	Vectors and Arrays	45
3.1	Concept: Using Built-in Functions	46
3.2	Concept: Data Collections	46
3.2.1	Data Abstraction	46
3.2.2	Homogeneous Collection	46
3.3	Vectors	46
3.3.1	Creating a Vector	47
3.3.2	Size of a Vector	48
3.3.3	Indexing a Vector	48
3.3.4	Shortening a Vector	50
3.3.5	Operating on Vectors	51
3.4	Engineering Example—Forces and Moments	58
3.5	Arrays	60
3.5.1	Properties of an Array	61
3.5.2	Creating an Array	62
3.5.3	Accessing Elements of an Array	62
3.5.4	Removing Elements of an Array	63
3.5.5	Operating on Arrays	64
3.6	Engineering Example—Computing Soil Volume	71
Chapter 4	Execution Control	81
4.1	Concept: Code Blocks	82
4.2	Conditional Execution in General	82
4.3	if Statements	83
4.3.1	General Template	84
4.3.2	MATLAB Implementation	84
4.3.3	Important Ideas	86
4.4	switch Statements	88
4.4.1	General Template	88
4.4.2	MATLAB Implementation	89
4.5	Iteration in General	90
4.6	for Loops	90
4.6.1	General for Loop Template	91
4.6.2	MATLAB Implementation	91
4.6.3	Indexing Implementation	93
4.6.4	Breaking out of a for Loop	94
4.7	while Loops	94
4.7.1	General while Template	94
4.7.2	MATLAB while Loop Implementation	95
4.7.3	Loop-and-a-Half Implementation	96
4.7.4	Breaking a while Loop	97
4.8	Engineering Example—Computing Liquid Levels	97
Chapter 5	Functions	105
5.1	Concepts: Abstraction and Encapsulation	106
5.2	Black Box View of a Function	106
5.3	MATLAB Implementation	107
5.3.1	General Template	107
5.3.2	Function Definition	107
5.3.3	Storing and Using Functions	109
5.3.4	Calling Functions	109
5.3.5	Variable Numbers of Parameters	109

	5.3.6	Returning Multiple Results	110
	5.3.7	Auxiliary Local Functions	111
	5.3.8	Encapsulation in MATLAB Functions	111
	5.3.9	Global Variables	112
	5.4	Engineering Example—Measuring a Solid Object	113
Chapter 6		Character Strings	121
	6.1	Character String Concepts: Mapping Casting, Tokens, and Delimiting	122
	6.2	MATLAB Implementation	123
	6.2.1	Slicing and Concatenating Strings	124
	6.2.2	Arithmetic and Logical Operations	125
	6.2.3	Useful Functions	125
	6.3	Format Conversion Functions	125
	6.3.1	Conversion from Numbers to Strings	125
	6.3.2	Conversion from Strings to Numbers	127
	6.4	Character String Operations	129
	6.4.1	Simple Data Output: The <code>disp(...)</code> Function	129
	6.4.2	Complex Output	129
	6.4.3	Comparing Strings	129
	6.5	Arrays of Strings	131
	6.6	Engineering Example—Encryption	132
Chapter 7		Cell Arrays and Structures	141
	7.1	Concept: Collecting Dissimilar Objects	142
	7.2	Cell Arrays	142
	7.2.1	Creating Cell Arrays	142
	7.2.2	Accessing Cell Arrays	143
	7.2.3	Using Cell Arrays	145
	7.2.4	Processing Cell Arrays	145
	7.3	Structures	146
	7.3.1	Constructing and Accessing One Structure	147
	7.3.2	Constructor Functions	148
	7.4	Structure Arrays	150
	7.4.1	Constructing Structure Arrays	150
	7.4.2	Accessing Structure Elements	152
	7.4.3	Manipulating Structures	154
	7.5	Engineering Example—Assembling a Physical Structure	156
Chapter 8		File Input and Output	167
	8.1	Concept: Serial Input and Output (I/O)	168
	8.2	Workspace I/O	168
	8.3	High-Level I/O Functions	169
	8.3.1	Exploration	169
	8.3.2	Spreadsheets	170
	8.3.3	Delimited Text Files	172
	8.4	Lower-Level File I/O	174
	8.4.1	Opening and Closing Files	174
	8.4.2	Reading Text Files	174
	8.4.3	Examples of Reading Text Files	175
	8.4.4	Writing Text Files	176
	8.5	Engineering Example—Spreadsheet Data	177

viii

Contents

Chapter 9	Recursion 185
9.1	Concept: The Activation Stack 186
9.1.1	A Stack 186
9.1.2	Activation Stack 186
9.1.3	Function Instances 187
9.2	Recursion Defined 187
9.3	Implementing a Recursive Function 188
9.4	Exceptions 190
9.4.1	Historical Approaches 191
9.4.2	Generic Exception Implementation 191
9.4.3	MATLAB Implementation 193
9.5	Wrapper Functions 195
9.6	Examples of Recursion 197
9.6.1	Detecting Palindromes 197
9.6.2	Fibonacci Series 198
9.6.3	Zeros of a Function 199
9.7	Engineering Example—Robot Arm Motion 202
Chapter 10	Principles of Problem Solving 211
10.1	Solving Simple Problems 212
10.2	Assembling Solution Steps 212
10.3	Summary of Operations 212
10.3.1	Basic Arithmetic Operations 213
10.3.2	Inserting into a Collection 213
10.3.3	Traversing a Collection 215
10.3.4	Building a Collection 216
10.3.5	Mapping a Collection 216
10.3.6	Filtering a Collection 217
10.3.7	Summarizing a Collection 218
10.3.8	Searching a Collection 219
10.3.9	Sorting a Collection 220
10.4	Solving Larger Problems 220
10.5	Engineering Example—Processing Geopolitical Data 221
Chapter 11	Plotting 231
11.1	Plotting in General 232
11.1.1	A Figure—The Plot Container 232
11.1.2	Simple Functions for Enhancing Plots 232
11.1.3	Multiple Plots on One Figure—Subplots 233
11.1.4	Manually Editing Plots 234
11.2	2-D Plotting 235
11.2.1	Simple Plots 235
11.2.2	Plot Options 237
11.2.3	Parametric Plots 237
11.2.4	Other 2-D Plot Capabilities 239
11.3	3-D Plotting 239
11.3.1	Linear 3-D Plots 239
11.3.2	Linear Parametric 3-D Plots 241
11.3.3	Other 3-D Plot Capabilities 242
11.4	Surface Plots 243
11.4.1	Basic Capabilities 243
11.4.2	Simple Exercises 243
11.4.3	3-D Parametric Surfaces 248

Contents

ix

	11.4.4	Bodies of Rotation	250
	11.4.5	Other 3-D Surface Plot Capabilities	255
	11.4.6	Assembling Compound Surfaces	256
11.5		Manipulating Plotted Data	256
11.6		Engineering Example—Visualizing Geographic Data	256
	11.6.1	Analyzing the Data	257
	11.6.2	Displaying the Data	258
Chapter 12		Matrices	267
12.1		Concept: Behavioral Abstraction	268
12.2		Matrix Operations	268
	12.2.1	Matrix Multiplication	268
	12.2.2	Matrix Division	271
	12.2.3	Matrix Exponentiation	271
12.3		Implementation	271
	12.3.1	Matrix Multiplication	272
	12.3.2	Matrix Division	273
12.4		Rotating Coordinates	274
	12.4.1	2-D Rotation	275
	12.4.2	3-D Rotation	278
12.5		Solving Simultaneous Linear Equations	281
	12.5.1	Intersecting Lines	282
12.6		Engineering Examples	283
	12.6.1	Ceramic Composition	283
	12.6.2	Analyzing an Electrical Circuit	285
Chapter 13		Images	291
13.1		Nature of an Image	292
13.2		Image Types	293
	13.2.1	True Color Images	293
	13.2.2	Gray Scale Images	293
	13.2.3	Color Mapped Images	294
	13.2.4	Preferred Image Format	294
13.3		Reading, Displaying, and Writing Images	295
13.4		Operating on Images	295
	13.4.1	Stretching or Shrinking Images	295
	13.4.2	Color Masking	296
	13.4.3	Creating a Kaleidoscope	301
	13.4.4	Images on a Surface	303
13.5		Engineering Example—Detecting Edges	306
Chapter 14		Processing Sound	315
14.1		The Physics of Sound	316
14.2		Recording and Playback	316
14.3		Implementation	317
14.4		Time Domain Operations	318
	14.4.1	Slicing and Concatenating Sound	318
	14.4.2	Musical Background	321
	14.4.3	Changing Sound Frequency	322
14.5		The Fast Fourier Transform	324
	14.5.1	Background	325
	14.5.2	Implementation	326
	14.5.3	Simple Spectral Analysis	327

Contents

14.6	Frequency Domain Operations	328
14.7	Engineering Example—Music Synthesizer	332
Chapter 15	Numerical Methods	339
15.1	Interpolation	340
15.1.1	Linear Interpolation	340
15.1.2	Cubic Spline Interpolation	343
15.1.3	Extrapolation	344
15.2	Curve Fitting	345
15.2.1	Linear Regression	345
15.2.2	Polynomial Regression	347
15.2.3	Practical Application	349
15.3	Numerical Integration	351
15.3.1	Determination of the Complete Integral	351
15.3.2	Continuous Integration Problems	353
15.4	Numerical Differentiation	356
15.4.1	Difference Expressions	356
15.5	Analytical Operations	357
15.5.1	Analytical Integration	357
15.5.2	Analytical Differentiation	357
15.6	Implementation	357
15.7	Engineering Example—Shaping the Synthesizer Notes	359
Chapter 16	Sorting	367
16.1	Measuring Algorithm Cost	368
16.1.1	Specific Big O Examples	369
16.1.2	Analyzing Complex Algorithms	370
16.2	Algorithms for Sorting Data	371
16.2.1	Insertion Sort	371
16.2.2	Bubble Sort	373
16.2.3	Quick Sort	375
16.2.4	Merge Sort	377
16.2.5	Radix Sort	379
16.3	Performance Analysis	380
16.4	Applications of Sorting Algorithms	382
16.4.1	Using sort(...)	382
16.4.2	Insertion Sort	382
16.4.3	Bubble Sort	383
16.4.4	Quick Sort	383
16.4.5	Merge Sort	383
16.4.6	Radix Sort	383
16.5	Engineering Example—A Selection of Countries	384
Chapter 17	Processing Graphs	389
17.1	Queues	390
17.1.1	The Nature of a Queue	390
17.1.2	Implementing Queues	390
17.1.3	Priority Queues	391
17.1.4	Testing Queues	393
17.2	Graphs	396
17.2.1	Graph Examples	396
17.2.2	Processing Graphs	397
17.2.3	Building Graphs	398

Contents

xi

	17.2.4	Traversing Graphs	401
	17.2.5	Searching Graphs	403
17.3		Minimum Spanning Trees	404
17.4		Finding Paths through a Graph	406
	17.4.1	Exact Algorithms	407
	17.4.2	Breadth-First Search (BFS)	407
	17.4.3	Dijkstra’s Algorithm	408
	17.4.4	Approximation Algorithm	411
	17.4.5	Testing Graph Search Algorithms	413
17.5		Engineering Applications	415
	17.5.1	Simple Applications	415
	17.5.2	Complex Extensions	415

Appendices

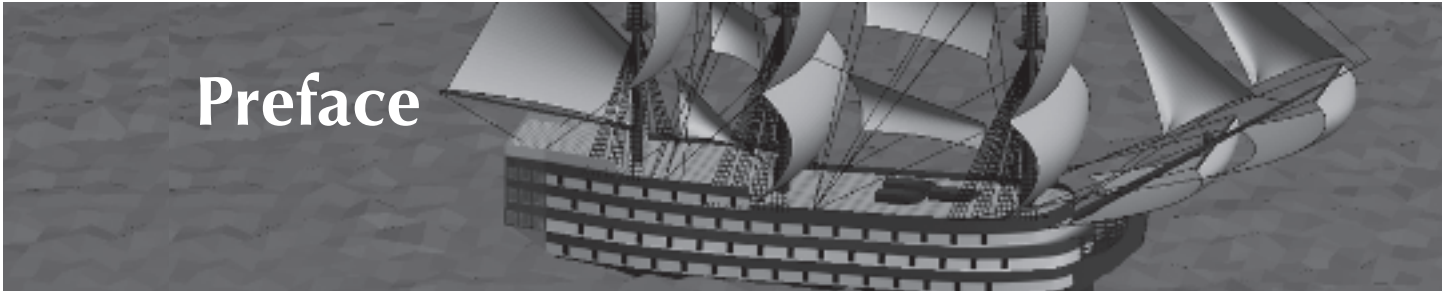
Appendix A	MATLAB Special Characters, Reserved Words, and Functions	A–I
Appendix B	The ASCII Character Set	B–I
Appendix C	Internal Number Representation	C–I
Appendix D	Answers to True or False and Fill in the Blanks	D–I

Index	I–I
-------	-----

About the Author

David Smith has been teaching introductory computer science classes for engineers at the Georgia Institute of Technology since 1997 when he retired from industry. Previously, he worked 31 years for Lockheed-Martin at its Marietta, Georgia, facility as a systems and software specialist with a focus on intelligent systems. He was active in designing and developing software for the C-130J, C-27J, F-22, and C-5 aircraft and was the technical leader of the Pilot’s Associate program, a \$64 million research project sponsored by the Defense Advanced Research Projects Agency.

Mr. Smith has a bachelor’s degree in aeronautical engineering from Southampton University and a master’s degree in control systems from Imperial College, London.



Preface

“That of all the several ways of beginning a book which are now in practice throughout the known world, I am confident my own way of doing it is the best—I’m sure it is the most religious—for I begin with writing the first sentence, and trusting to Almighty God for the second.”

Laurence Sterne (1713–1768), British author, clergyman

This book introduces the power, satisfaction, and joy of computing to beginning engineering students who have little or no previous computing experience. It began as a snapshot of the content of a Georgia Tech course that introduces engineers to computing. However, it has been extensively enhanced to meet the needs of a wider audience of students and educators who want to understand programming for other reasons. In this book, to understand computing, we use the basic syntax and capabilities of MATLAB, a user-friendly language that is emerging as one of the most popular computing languages in engineering.

New to the Third Edition

Many engineering disciplines use the concept of graphs to represent specific ideas. We have added a chapter that deals with the fundamentals of graph manipulation from an engineering standpoint—specifically, how to find a minimum spanning tree, and both exact and approximate methods for finding the best path from one point to another. We also try to note those new features of MATLAB that are relevant to students in an introductory programming class. For examples, features were added recently allowing a user to manipulate plotted data by adjusting and saving values. Although interesting, one can achieve the same result with more traceability and repeatability by editing to the source data and repeating the plots.

One interesting observation emerged when refreshing the analysis of sorting algorithms in Chapter 16. In older versions of MATLAB (prior to R2008), our crude recursive implementations of Merge Sort and Quick Sort did not achieve the expected performance. The reason we deduced was that when data are passed into and out of a function, they must be copied between the workspaces of the calling and called functions. With R2011, however, the same code works splendidly, suggesting that the earlier inefficient parameter passing mechanisms have been significantly improved.

Pedagogical Style

Computing is not a spectator sport. Students learn computing by using a computing system to solve problems. This text not only presents computing concepts and their MATLAB implementation, but also offers students extensive hands-on exercises. The text illustrates the ideas with examples from the world of engineering, provides style points, and presents sample problems that students might encounter. Each chapter includes topics that go a step beyond the basic content of an introductory class. This gives professors the choice to progress slowly, and more thoroughly, through the material in two semesters. It also offers advanced students enrichment materials for their personal study.

The overall philosophy of this text approaches programming tools in the following manner:

1. Explain a computing concept in general
2. Discuss its implementation in MATLAB
3. Provide exercises to master the concept

To help facilitate students’ understanding of the concept and its implementation, the text uses two features: general templates and MATLAB listings. The general templates provide a foundation for students to understand concepts in general and can be applied to any language. The MATLAB listings show students how to implement concepts in MATLAB and are followed by detailed explanations of the code.

Features of the Text

- **Exercises:** Allow students a “Do It Yourself” approach to master concepts by trying what they just learned. Exercises follow each new topic.
- **Style Points:** Advise students about writing quality code that is easy to understand, debug, and reuse.
- **Hints:** Enrich students’ understanding of a topic. Hints are interspersed through the book at points where students may benefit from a little extra “aside.”
- **Engineering Examples:** Provide robust models and apply to real-world issues that will motivate students. Examples from different engineering disciplines are presented at the end of each chapter.
- **Special Characters, Reserved Words, and Functions:** Provides a quick reference for the key MATLAB principles discussed in each chapter.
- **Self Test:** Helps students to check their understanding of the material in each chapter.
- **Programming Projects:** Offer a variety of large-scale projects that students can work on to solidify their skills.

Chapter Overview

Chapter 1: *Introduction to Computers and Programming* discusses the history of computer architectures as they apply to computing systems today. The chapter provides an overview of computer hardware and software and how programs execute.

Chapter 2: *Getting Started* discusses some basic concepts of computing and then introduces the basic operation of the MATLAB user interface. The chapter also describes how to capture simple MATLAB programs in the form of scripts.

Chapter 3: *Vectors and Arrays* introduces the fundamental machinery that sets MATLAB apart from other languages—its ability to perform mathematical and logical operations on homogeneous collections of numbers.

Chapter 4: *Execution Control* describes the common techniques used to control the execution of code blocks—conditional operation and iteration.

Chapter 5: *Functions* describes how to implement procedural abstraction by defining reusable code blocks.

Chapter 6: *Character Strings* discusses how MATLAB operates on variables containing text.

Chapter 7: *Cell Arrays and Structures* discusses two kinds of heterogeneous data collections accessed by index and by name.

Chapter 8: *File Input and Output* describes three levels of ability provided in MATLAB for transferring data to and from data files—saving workspaces, specific tools that read and write specific data files, and general-purpose tools for processing any kind of file.

Chapter 9: *Recursion* discusses and illustrates a widely used alternative approach to repetitive code execution.

Chapter 10: *Principles of Problem Solving* introduces ideas that help students design solutions to new problems and avoid the “blank sheet of paper” syndrome—how to start a program.

Chapter 11: *Plotting* takes the student from basic plotting in two dimensions to the advanced tools that draw representations of three-dimensional objects with smooth shading and even multiple light effects.

Chapter 12: *Matrices* describes specific MATLAB capabilities that implement matrix algebra.

Chapter 13: *Images* discusses how to use vector and array algebra to manipulate color pictures.

Chapter 14: *Processing Sound* shows how to analyze, synthesize, and operate on sound files.

Chapter 15: *Numerical Methods* introduces numerical techniques that commonly occur in engineering: interpolation, curve fitting, integration, and differentiation.

Chapter 16: *Sorting* presents five algorithms for ordering data, each of which has applicability under certain circumstances—Insertion Sort, Bubble Sort, Quick Sort, Merge Sort, and Radix Sort—and then compares their performance on large quantities of data.

Chapter 17: *Processing Graphs* discusses how to represent graphs in general and then how to solve two important engineering problems—finding a minimal spanning tree and finding an optimal path between two nodes of the graph.

Appendices provide a summary of the MATLAB special characters, reserved words, and functions used throughout the text, the ASCII character set, the internal number representation inside the computer, and answers to the True or False and Fill in the Blank questions.

Paths through the Book

Not all courses that cover programming and MATLAB follow the same syllabus. *Engineering Computation with MATLAB* is designed to facilitate teaching the material with different styles and at different speeds. For example, Chapters 3, 4, and 5 cover MATLAB array manipulation, iteration, and writing your own functions. There are three schools of thought about the appropriate way to introduce these concepts. One would introduce array constructs first and follow up with the more “traditional” concept of iteration; another would teach iteration first and deal with the MATLAB-specific array operations later; and the third would treat functions first. I chose to order the book according to the arrays-first approach, to suit a particular teaching style. However, should you prefer iteration or functions first, Chapters 3, 4, and 5 can be used in any order you wish. In practice, over the years, our course has shifted to a functions-first approach so that we can use function interfaces to isolate students’ code for automated code grading. Chapters 6–9 should be taught in sequence—there are dependencies between chapters that would make it awkward change the order. Chapter 10 is an important chapter that is difficult to place on a class schedule. Where it stands in the book appears to be a logical position. However, at that point in the semester, beginning students are still not ready to think about larger problems. I have usually covered this material (if at all) at the end of each semester by way of review. Chapter 11 provides basic plotting capability and is necessary for the remaining chapters. After that, Chapters 12–17 are virtually independent and can be taught in any order, but should follow Chapters 2–9 and 11.

Supplements

Various supplemental materials for this text are available at the book’s Companion Web site: www.pearsonhighered.com/smith. The following are accessible to all readers:

- Solutions to selected Programming Projects
- Selected full-color figures
- Source code for all MATLAB listings
- Bonus chapters including: Object-Oriented Programming, Linked Lists, N-ary Trees and Graphs, and the Cost of Computing

In addition, the following supplements are available to qualified instructors at Addison-Wesley’s Instructor Resource Center. Please visit www.pearsonhighered.com/irc, or send an e-mail to computing@aw.com.

- Solutions to all of the Programming Projects
- PowerPoint lecture slides

Acknowledgments

The underlying philosophy of this book and the material that forms its skeleton originated in the work of Professor Russell Shackelford around 1996. Dr. Melody Moore, currently an Associate Professor in the Interactive Computing department of the College of Computing at Georgia Tech, was instrumental in creating many of the teaching materials (then as overhead transparencies) from which this class was first taught. I am deeply indebted to Professor James Craig from the Aerospace Engineering department at Georgia Tech, who joined me in co-teaching the first engineering version of CS1, taught me much about MATLAB, and pioneered this class from the original 35 students to its current size of over 1,000 engineering students per semester. This engineering class became a vessel for introducing the students to the MATLAB language.

I would like to thank the following reviewers for their insight and wisdom during the process of manuscript development:

- Kenneth Rouse, *Auburn University*
- Suparna Datta, *Northeastern University*
- Gerardine G. Botte, *Ohio University*
- Mica Grujicic, *Clemson University*
- Kuldip S. Rattan, *Wright State University*
- Y.J. Lin, *The University of Akron*
- Mark Nagurka, *Marquette University*
- Michael Peshkin, *Northwestern University*
- Howard Silver, *Fairleigh Dickinson University*
- Steve Swinnea, *The University of Texas at Austin*

The material has benefited from the efforts of every Georgia Tech teaching assistant (TA), graduate student, instructor, and professor who has taught CS1, a list too long to enumerate. In particular, those wonderfully creative TAs who developed the ideas for examples used in this text have enriched it immeasurably. I wish to credit Professor Aaron Bobick with an important contribution made in the course of one short conversation. That conversation was responsible for pulling the class back from the brink of being merely a MATLAB programming class to one with roots in CS concepts. Professor Bobick taught CS1 with me in the fall of 2004. Early in the semester he made a very simple request: he said it would be easier for him to teach the class if we explicitly expressed the computing concepts inherent in each lesson, rather than leaving him—and the students—to tease the concepts out of the teaching materials.

I cannot adequately express my appreciation for the team at Addison-Wesley that helped bring this book to fruition. Many of them have done, and I am sure continue to do, their work “behind the scenes”: Michael Hirsch, Emma Snider, Yez Alayan, Kayla Smith-Tarbox, Pat Brown, and Jeff Holcomb. I also really appreciate the work of Kailash Jadli and his team at Aptara[®], Inc. for the care with which they designed the third edition.

Most important, I would like to acknowledge the personal contributions of those people without whom this book would not exist. My wife and best friend, Julie, has been an unwavering source of strength and encouragement during the process of writing this text. Bill Leahy was a student in the first CS1 class I taught in the spring of 1998. In spite of this beginning, he continued to a master’s in computer science from Tech and is now an instructor in the College of Computing. Beyond his uncountable technical contributions to the material in this book, I want to acknowledge his friendship, encouragement, and wise judgment, all of which have been an inspiration to me during the process of developing this text.