DMS
ymax = 10;
th = linspace(0, 2*pi, 40);
xx = linspace(1, ymax, 40);
bb = 0;
taper = 0.9; sweep = 0.7; dihamp = 0.02; center = complex( sqrt(r^2-g^2) \* 1, g ); [yy, th] = meshgrid( y, th ); z = r * complex( cos(tth), sin(tth) ) + center; ww = z + 1./z; xx = real(ww) .* (ymax - taper*yy) / ymax - sweep * yy; alf = 0; bet = 0; lim = zeros(3,2); figure('Color', [0, 0.5, 0.8]); while alf < 2*pi
    tw = yy .* bet / (4 * ymax);
    ct = cos(tw); st = sin(tw);
    dihed = dihamp * sin(alf);
    zz = imag(ww) .* (ymax - taper*yy) / ymax + dihed * yy^2;
    xxa = xx .* ct - zz .* st;
    zza = xx .* st + zz .* ct;
surf( xxa, yy, 2*zza, zz )
axis([-7.2037 1.1440 0 10.0000 -1 4]); axis off
alpha(0.05); shading interp; hold on
lightangle(45, 45); view(0, 15)
pause(0.03)
alf = alf + .02; bet = alf;
if bet > pi
    bet = bet - 2*pi;
elseif bet > pi/2
    bet = pi - bet;
end
if bet < -pi/2
    bet = -pi - bet;
end
alpha(0.05); shading interp; hold on
lightangle(45, 45); view(0, 15)
ENGINEERING COMPUTATION
with MATLAB®

Second Edition
The ribbed effect inside the wing’s leading edge is an interesting artifact of the overlaying process.
This book is dedicated to the

glory of Almighty God

~David M. Smith
# Contents

## Chapter 1 Introduction to Computers and Programming

1.1 Background
1.2 History of Computer Architectures
1.3 Computing Systems Today
1.4 Executing a MATLAB Program
1.5 Problem Solving

## Chapter 2 Getting Started with MATLAB

2.1 Programming Language Background
2.2 Basic Data Manipulation
2.3 The MATLAB User Interface
2.4 Scripts
2.5 Engineering Example—Spacecraft Launch

## Chapter 3 Vectors and Arrays

3.1 Concept: Using Built-in Functions
3.2 Concept: Data Collections
3.3 MATLAB Vectors
3.4 Engineering Example—Forces and Moments
3.5 MATLAB Arrays
3.6 Engineering Example—Computing Soil Volume

## Chapter 4 Execution Control

4.1 Concept: Code Blocks
4.2 Conditional Execution in General
4.3 if Statements
4.4 switch Statements
4.5 Iteration in General
4.6 for Loops
4.7 while Loops
4.8 Engineering Example—Computing Liquid Levels

## Chapter 5 Functions

5.1 Concepts: Abstraction and Encapsulation
5.2 Black Box View of a Function
5.3 MATLAB Implementation
5.4 Engineering Example—Measuring a Solid Object

## Chapter 6 Character Strings

6.1 Character String Concepts: Mapping, Casting, Tokens and Delimiting
6.2 MATLAB Implementation
Contents

6.3 Format Conversion Functions 131
6.4 Character String Operations 134
6.5 Arrays of Strings 137
6.6 Engineering Example—Encryption 138

Chapter 7 Cell Arrays and Structures 147
7.1 Concept: Collecting Dissimilar Objects 148
7.2 Cell Arrays 148
7.3 MATLAB Structures 153
7.4 Structure Arrays 156
7.5 Engineering Example—Assembling a Structure 162

Chapter 8 File Input and Output 173
8.1 Concept: Serial Input and Output (I/O) 174
8.2 MATLAB Workspace I/O 174
8.3 High-Level I/O Functions 175
8.4 Lower-Level File I/O 180
8.5 Engineering Example—Spreadsheet Data 184

Chapter 9 Recursion 191
9.1 Concept: The Activation Stack 192
9.2 Recursion Defined 193
9.3 Implementing a Recursive Function in MATLAB 194
9.4 Exceptions 196
9.5 Wrapper Functions 200
9.6 Tail Recursion 203
9.7 Mutual Recursion 205
9.8 Generative Recursion 205
9.9 Examples of Recursion 205
9.10 Engineering Example—Robot Arm Motion 211

Chapter 10 Principles of Problem Solving 219
10.1 Solving Simple Problems 220
10.2 Assembling Solution Steps 220
10.3 Summary of Operations 220
10.4 Solving Larger Problems 236
10.5 Engineering Example—Processing Geopolitical Data 238

Chapter 11 Plotting 247
11.1 Plotting in General 248
11.2 2-D Plotting 252
11.3 3-D Plotting 258
11.4 Surface Plots 261
11.5 Interacting with Plotted Data 279
11.6 Engineering Example—Visualizing Geographic Data 283

Chapter 12 Matrices 293
12.1 Concept: Behavioral Abstraction 294
12.2 Matrix Operations 294
12.3 MATLAB Implementation 297
12.4 Rotating Coordinates 300
Contents

12.5 Solving Simultaneous Linear Equations 307
12.6 Engineering Examples 311

Chapter 13 Images 319
13.1 Nature of an Image 320
13.2 Image Types 321
13.3 Reading, Displaying, and Writing Images 323
13.4 Operating on Images 323
13.5 Engineering Example—Detecting Edges 339

Chapter 14 Processing Sound 347
14.1 The Physics of Sound 348
14.2 Recording and Playback 348
14.3 MATLAB Implementation 349
14.4 Time Domain Operations 350
14.5 The Fast Fourier Transform 359
14.6 Frequency Domain Operations 364
14.7 Engineering Example—Oil Rig Structural Integrity 371

Chapter 15 Numerical Methods 379
15.1 Interpolation 380
15.2 Curve Fitting 384
15.3 Numerical Integration 390
15.4 Numerical Differentiation 394
15.5 Engineering Example—Analyzing Rocket Data 397
15.6 Engineering Example—Cleaning up Images 402

Chapter 16 Sorting 411
16.1 Measuring Algorithm Cost 412
16.2 Algorithms for Sorting Data 415
16.3 Performance Analysis 425
16.4 Applications of Sorting Algorithms 426
16.5 Engineering Example—A Selection of Countries 430

Chapter 17 Searching Graphs (online)
Chapter 18 Object-Oriented Programming (online)
Chapter 19 Linked Lists (online)
Chapter 20 Binary Trees (online)
Chapter 21 N-ary Trees and Graphs (online)
Chapter 22 The Cost of Computing (online)

Appendices
Appendix A MATLAB Special Characters, Reserved Words, and Symbols A–1
Appendix B The ASCII Character Set B–1
Appendix C Internal Number Representation C–1
Appendix D Web Reference Materials (online)
Appendix E Answers to True or False and Fill in the Blanks (online)

Index I–1
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 their 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 $42 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 MATLAB, a user-friendly language that is emerging as one of the most popular computing languages in engineering.

Since the 1st edition of this book, the release of MATLAB 7.6 (R2008a) has significant changes in the use of the product. The following changes have been incorporated into the new release:

• You can now customize the tool bars on the desktop and editor windows to reorganize your tools and de-clutter the windows by removing unused links.
• In the Editor Window, many code blocks can be collapsed to allow easier visibility into the overall structure of scripts and functions.
• The implementation of object-oriented programming (OOP) has been completely revised to be more compatible with conventional OOP languages.
• Two tools have been added to plots to enable interaction with the underlying data: Brushing allows you to select specific values on a plot and manipulate those values to change the behavior of the plot; linking allows you to link the data presented in a plot to the underlying data so that when you make changes to a plot, the underlying data are updated.
• A Variable Editor has been added to replace the old Array Editor, expanding the editing ability from arrays to include structures and objects.
Preface

The text of this book has been updated as appropriate to reflect these and other detailed changes. The examples and end-of-chapter programming projects have also been extensively revised and improved.

Pedagogical Style
Computing is not a spectator sport. Students learn computing by computing. 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.
Preface

- **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 with MATLAB* 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 a script.

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 block execution.

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

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.
Preface

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 Bucket Sort—and then compares their performance on large quantities of data.

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, Web reference materials, and answers to the True or False and Fill in the Blanks 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.

I have attempted numerous approaches to the sequence of lessons covering Chapters 1–9. The order in which these chapters appear has been the most intuitive for me as an instructor and, by and large, for the students. Chapters 10–16 are virtually independent and can be taught in any order.

Supplements

Various supplemental materials for this text are available at the book’s Companion Website: www.aw.com/smith_mat. The following are accessible to all readers (register using the code in the front of this book):

- Solutions to selected Programming Projects
- Selected full-color figures
- Source code for all MATLAB listings
Preface

• Bonus chapters including: Searching Graphs, Object-Oriented Programming, Linked Lists, Binary Trees, N-ary Trees and Graphs, and the Cost of Computing
• Bonus appendices including: Web Reference Materials and Answers to True or False and Fill in the Blanks

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 e-mail to computing@aw.com.

• Source Code
• 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 assistant professor in the Computer Information Systems department of the College of Business Administration at Georgia State University, 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
Preface

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, Stephanie Selling, Erin Davis, Heather McNally, and Jeff Holcomb. I would like to thank Greg Cantwell for creating the illustrations. I have developed a personal appreciation for three people on this team: Kathy Cantwell, my copyeditor for the first edition, who has been so talented, patient, and ready to work with material that must seem to her to be in a foreign language; Linda Knowles, who did an incredible job transforming my vague idea into a stunning cover design; and Marika Sharpe, Addison-Wesley’s representative at Georgia Tech, who finally convinced me to write this book and continues to support and encourage me. I also really appreciate the work of Dennis Free and his team at Aptara®, Inc. for the care with which they designed the second edition and the work of Suzanne Harbison for the second edition cover.

Most importantly, 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. Most of all, if there is any enduring wisdom in this book, I give all the credit to Almighty God and His Son, Jesus.