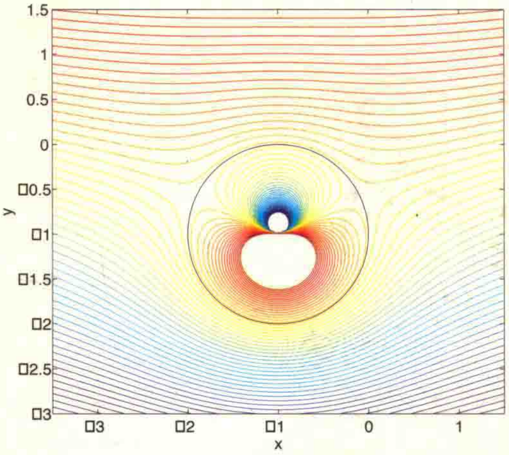
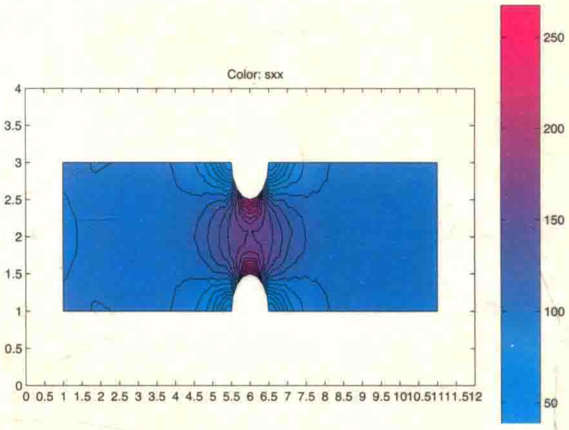
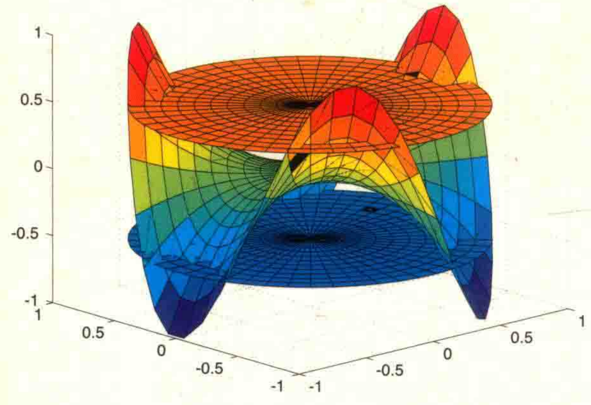
An Engineer's Guide to MATLAB®







Edward B. Magrab

Shapour Azarm

Balakumar Balachandran

James Duncan

Keith Herold

Gregory Walsh

An Engineer's Guide to MATLAB®

Edward B. Magrab

University of Maryland
Mechanical Engineering Department

Contributing Authors:

**Shapour Azarm, Balakumar Balachandran,
James H. Duncan, Keith E. Herold, Gregory C. Walsh**

University of Maryland
Mechanical Engineering Department

Prentice Hall
Upper Saddle River, NJ 07458

Library of Congress Cataloging-in-Publication Data

Magrab, Edward B..
An engineer s guide to MATLAB / by Edward B. Magrab.
p. cm.
ISBN 0-13-011335-2 (pbk.)
1. MATLAB (Computer program language) I. Title

QA76.73.P22 S43 2000
005.13'3--dc21 00-051528
CIP

Acquisitions editor: *Laura Curless*
Editorial assistant: *Laurie Friedman*
Editorial/production supervision: *Rose Kernan*
Vice-president of editorial development, ECS: *Marcia Horton*
Executive managing editor: *Vince O'Brien*
Managing editor: *David A. George*
Vice-president of production and manufacturing: *David W. Riccardi*
Cover design: *Bruce Kenselaar*
Creative director: *Jayne Conte*
Manufacturing manager: *Trudy Pisciotti*
Manufacturing buyer: *Pat Brown*
Marketing manager: *Danny Hoyt*

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

ISBN 0-13-011335-2

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Pearson Education Asia Pte Ltd
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

An Engineer's Guide to MATLAB®

For June Coleman Magrab, my muse

PREFACE

The primary goal of this book is to guide the reader in developing a strong working knowledge of MATLAB to solve engineering problems. Typically, solving these problems involves writing relatively short, one-time-use programs. Therefore, in this book we attempt to teach how to effectively develop such programs in MATLAB, ones that are compact, yet readable, are easy to debug, and execute fast.

The first seven chapters of the book are intended for use in a sophomore/junior level class that introduces programming and the use of computer languages in engineering. In the remaining seven chapters we present applications of MATLAB to a wide range of engineering topics. The emphasis of the book is on using MATLAB to obtain solutions to several classes of engineering problems, not the technical subject matter *per se*. Therefore, the technical material is presented in summary form only and no attempt has been made to present the basic material in each of these topic areas. The book also can be used in the following ways: (1) as a companion book to junior, senior and graduate level textbooks in engineering; (2) as a reference book for obtaining numerical solutions to a wide range of engineering problems, and; (3) as a source of applications of a wide variety of MATLAB solution techniques.

Engineering programming applications are typically used to do the following: (1) analyze a predictive model, such as an algebraic equation, an ordinary or partial differential equation or an approximation to these; (2) obtain statistical inferences from data; (3) visualize a model or data to enhance oneís understanding; (4) either verify or obtain an empirical model from experimental results; and (5) monitor/control/analyze external events. In this book all but the last application are addressed.

The presentation of the material in this book is made with the assumption that the reader can employ the engineering approach to problem solving; that is, one that uses approximate mathematical models to predict the response of elements, devices, and systems. This approach also requires that one have a good comprehension of the physical problem so that he or she can tell when the modelís results are correct, or at least reasonable. These qualities are an important prerequisite to creating programs that function correctly. The book also assumes that the reader is moderately fluent in calculus and engineering mathematics.

The first seven chapters are devoted to the introduction of MATLAB, where vector and matrix notation and definitions are introduced immediately and their fundamental importance to using MATLAB effectively is demonstrated. Numerous examples and detailed explanations are used to proceed through the material. The scripts and functions used to solve the example problems emphasize the employment of a relatively small number of readable MATLAB expressions to generate primarily graphical presentations of the results. This approach reinforces the importance of the vector/matrix formulations and the advantages of the resulting compactness of the code. Many of the example problems have been selected to illustrate the graphical presentation of data.

The approach used in the first seven chapters is then applied in the remaining seven chapters, each of which presents the application of MATLAB solution methods and toolboxes to classes of problems in the following areas: design of machine elements, dynamics and vibrations, controls, fluid mechanics, heat transfer, optimization, and engineering statistics. The material in these chapters is illustrated by numerous MATLAB solutions to classes of problems, and includes brief discussions of the program listings and their results. Each chapter provides exercises for which many of the solutions require annotated two or three-dimensional figures.

The MATLAB scripts and functions developed throughout the book use the most appropriate MATLAB function to obtain the numerical results and they have been verified to work correctly through version 5.3 (Release 11). We assume that the reader has access to all functions demonstrated. Furthermore, we have tried to keep to a minimum much of the detailed information that is readily available from the on-line help files. However, for a number of frequently used functions and for many functions in the toolboxes we have included a sufficient amount of information so that the reader can clearly determine what equations the functions solve and how the numerical method represented by the function can be used. In addition, several of the chapters use MATLABís controls, statistics, optimization and partial differential equation toolboxes, thereby extending the traditional range of the types of problems usually examined. Furthermore, we present solution techniques that take advantage of MATLABís numerical procedures where practical, as opposed to first obtaining algebraic solutions and then programming them.

Several preliminary versions of the book have been used over the last five semesters in the Mechanical Engineering department at the University of Maryland, both in an introductory MATLAB course and as companion material in junior and senior level courses in vibrations, controls, heat transfer, fluid mechanics, optimization, solid mechanics, and engineering statistics. The authors have found that with a good working knowledge of MATLAB they could expand the type of realistic engineering problems that could be examined in these classes, and they have frequently marveled at the relative ease with which these solutions could be obtained. We hope that the reader will also reap the rewards of applying MATLAB to determine the solutions to his or her engineering problems.

E. B. Magrab
S. Azarm
B. Balachandran
J. H. Duncan
K. E. Herold
G. C. Walsh

College Park, MD
March 2000

CONTENTS

CHAPTER 1

INTRODUCTION

Edward B. Magrab

The fundamental characteristics of the MATLAB environment and its basic syntax are introduced.

1.1 INTRODUCTION

MATLAB, which derives its name from *Mat*rix *Lab*oratory, is a computing language devoted to processing data in the form of matrices of numbers. MATLAB integrates computation and visualization into a flexible computer environment, and provides a diverse family of built-in functions that can be used in a straightforward manner to obtain numerical solutions to a wide range of engineering problems.

1.2 WAYS TO USE MATLAB

When the MATLAB program is launched, the user is placed in a window where a blinking[1] cursor appears immediately after the prompt ">>". This window, commonly called the

[1] The MATLAB window's look, management, and file management descriptions relate to a Windows™ environment. Equivalent procedures are used with other operating systems.

MATLAB command window, is a workspace that is equivalent to a blank sheet of paper. The space appearing immediately after the ">>" is called the command line. One enters numerical values for a matrix by simply typing a matrix of numbers on the command line in the prescribed format discussed in Section 2.4. To identify these numbers with a variable name, the variable name followed by an equal sign should precede them. If no variable name is used, then MATLAB assigns it the generic name *ans*. In either case the matrix can be recalled for either display to the command window or for use in another MATLAB expression by simply typing either the user-specified variable name or *ans*. However, if the same variable name is used to identify another matrix of numbers, or another set of numbers is typed at the command line without specifying a variable name, then the previously entered numbers for either the variable name or *ans*, as the case may be, will be overwritten. In addition, typing `clear` clears the workspace. This command is discussed in more detail below.

MATLAB permits one to perform arithmetic, trigonometric, and exponential operations (e.g., addition, division, cosine, and logarithm) on the variables in the same manner as with a calculator. These operations are performed by what MATLAB calls functions. In addition to these basic calculator-type functions, MATLAB has a large collection of functions that perform very sophisticated mathematical operations. MATLAB also provides the means whereby users can create their own functions, as described in Chapter 5. Another use of these functions is to allow a structured approach to managing the programming task. They differ from expressions entered at the command line in that MATLAB allots them their own private workspace and they have formally defined input-output relationships with the MATLAB environment.

When the user is required to enter many expressions or to repeatedly retype a series of expressions at the command window, the task can become tedious. To alleviate this potential problem, MATLAB has introduced script files—files that contain a list of commands, each of which will be operated on by MATLAB as if they were typed at the command line in the command window. A script file is created in either a word processor, a text editor, or the MATLAB-supplied text editor and debugger,[2] and saved as a text file with the suffix ".m". If a word processor or text editor is used, then the file is executed by typing the file name without the suffix ".m" in the MATLAB command window. If the MATLAB editor is used, then clicking on *Tools* and selecting *Run* executes the script. However, before doing this the file must first be saved.

Script files are usually employed in those cases where:

1. The program will contain more than a few lines of code.

2. The program will be used again.

3. A permanent record is desired.

4. It is expected that occasional upgrading will be required.

5. Substantial debugging is required.

6. One wants to transfer the listing to another person or organization.

[2] Clicking either the leftmost icon in the MATLAB command window (the white rectangle) or selecting from the *File* pull-down menu *New* or *M-File* accesses the MATLAB Editor/Debugger.

In addition, a script or a function typically has the following attributes:

1. *Documentation*, which, at a minimum, indicates the:

Purpose and operations performed
Programmer's Name
Date originated
Date(s) revised
Description of the input(s): number, meaning, and type
Description of the output(s): number, meaning, and type

2. *Input*, which includes numerous checks to ensure that all input values have the qualities required for the script/function to work properly.
3. *Initialization*, where the appropriate variables are assigned their numerical values.
4. *Computation*, where the major numerical evaluations are performed.
5. *Output*, where the results are presented as graphical and/or annotated numerical quantities.

1.3 CREATING VARIABLE NAMES

MATLAB permits the user to create variable names with a length of up to 31 alphanumeric characters, with the characters after the thirty-first being ignored. Each variable name must start with either an uppercase or lowercase letter, which can then be followed by any combination of uppercase and lowercase letters, numbers, and the underscore character (_). No blank spaces may appear between these characters. Variable names are case sensitive, so that a variable named *junk* is different from *junK*. There are two commonly used conventions: one that uses the underscore and one that uses capital letters. For example, if the exit pressure is a quantity that is being evaluated, then it could be defined in a MATLAB command line, script, or function as either *exit_pressure* or *ExitPressure*.

The most commonly used characters and symbols that have been set aside by MATLAB, along with their meanings, are shown in Table A.1 in the Appendix at the end of this chapter.

1.4 MANAGEMENT OF VARIABLES

During any MATLAB session—that is, during any time until the program is exited—MATLAB retains in its memory the most recently obtained values of all variables defined by each expression that has been either typed or evaluated from a script file, unless the `clear` function is invoked. The `clear` function deletes all the variables from memory. As mentioned previously, the numerical values assigned most recently to these variables are accessible anytime during the session (provided that `clear` hasn't been used) by simply typing the variable's name or using it in an expression. These variables are referred to as global variables.

Typing done in the MATLAB command window remains in the window and can be accessed by scrolling back until the scrolling memory has been exceeded, at which point the earliest entered information is lost. However, the expressions evaluated from a script file are not available, although the variable names and their numerical values are available as indicated in the preceding paragraph. This record of previously typed expressions can be removed by going to the *Edit* pull-down menu at the top of the MATLAB command window and selecting *Clear Session*, which clears the MATLAB command window, but does not delete the variables, which have to be removed by using `clear`. Also, the copy and paste icons can be used either to reproduce previously typed expressions in the current (active) line in the MATLAB command window or to paste MATLAB expressions either from the MATLAB command window into a word processor's window or vice versa.

For a listing of what variables have been created since the use of the last application of `clear`, one either types `whos` in the MATLAB command window or goes to the pull-down *File* menu and selects *Show Workspace*, which creates a window with this information. Either method reveals the names of the variables, their size, the number of bytes of storage that each variable uses, and their type: numerical, string (Section 3.1), symbolic, or an `inline` object (Section 5.3). The advantage of the latter method is that if one keeps the window active, this information is readily available, because MATLAB continually updates it.

1.5 ACCESSING SCRIPTS AND FUNCTION FILES

Script files and functions are run by typing their file name (without the ".m") in the MATLAB command window. However, MATLAB must be provided with the path to the directory in which the file resides. The path information is entered by going to the *File* pull-down menu and selecting *Set Path*. This opens the *Path Browser*. Click on *Browse* and choose the directory. Then go to *Path* and select *Add to Path* and either *Add to Front* or *Add to Back*. More than one directory may be added while in *Path Browser*. Before leaving the *Path Browser* it is suggested that you select *File* and then *Save Path*, which saves these paths for the next time MATLAB is used.

1.6 COMMAND WINDOW MANAGEMENT

To make the letters and numbers that appear in the command window more readable, MATLAB offers several options with the `format` function. Two that are particularly useful are the phrases

```
format compact
```

and

```
format long e
```

The former removes most empty (blank) lines and the latter provides a toggle from the default format of 5 digits to a format with 16 digits plus a 3-digit exponent. The `format long e` option is useful when debugging scripts that produce numbers that either change by