MATLAB® Primer for Speech-Language Pathology and Audiology

1

Introduction to Programming With MATLAB®

Introduction to MATLAB®

What Is MATLAB®?

MATLAB (short for Matrix Laboratory) is both a programming language and computing environment developed by MathWorks, Inc. (Natick, Massachusetts), and is designed for performing calculations and data processing. It is traditionally used in engineering and other computationally intensive fields due to its ability to handle large data sets and matrices natively. Evidence-based practice, improved tools for speech and language recording, as well as a widening scope of parameters assessed in speech-language pathology practice have changed the assessment in speech pathology from subjective and general to objective and detailed. A host of quantitative measures detailing respiratory and vocal tract functions for speech as well as the neurophysiology of language and reading are now used on a regular basis in speech-pathology practice. Most of these parameters are computationally intensive and therefore a challenge to processing, and keeping them together in a record presents an even greater obstacle. In this book, we will cover the use of MATLAB not only for data processing but also for record keeping relevant to speech-language pathology and audiology.

Of note are several free alternatives to MATLAB, specifically GNU Octave and SciPy (an extension to the Python programming language). With some exceptions, code written for MATLAB is generally compatible with the Octave system, and vice versa. However, for the sake of simplicity, this book adheres to standard MATLAB syntax and assumes the presence of a MATLAB installation.

Why Use MATLAB®?

There are two main reasons why MATLAB is the work environment of choice for speech-language pathologists and audiologists: its intrinsic qualities and its pedagogical advantages. The latter consideration is important as we acknowledge programming skills do not yet figure centrally in either speech-language pathology and audiology training or practice.

MATLAB has a number of desirable features for computational work in speech-language and hearing research. It can natively handle the `.wav` format commonly used in recorded speech, and allows direct operations on audio and other waveform data, such as frequency analysis, amplitude correction, root-mean-square (RMS) calculation, and many others. It can also import large data sets in `.txt`, `.xls`, `.xlsx`, and other common data-storage formats without requiring the user to resort to the low-level file operations (e.g., manually opening and closing files, or dealing with reading binary data byte by byte) common in other languages. Furthermore, many expansion kits known as Toolboxes are available from MathWorks or are freely downloadable online, which allow the user to add functionality to the language as projects or research dictate. Of particular value is the Signal Processing Toolbox, which is capable of online computation, can produce spectrograms and other graphical representations of speech quickly and easily, and contains a number of useful functions for speech analysis.

For the purposes of teaching programming to speech-language pathologists and audiologists, the interactive nature of MATLAB is invaluable. Many other programming languages (e.g., C++ or Java) require sometimes-lengthy compile times between writing the code and seeing the result, which can make it difficult for the beginner to try new ideas or understand why a portion of code does not work as planned. MATLAB, however, executes instructions entered in its Command Window with near-immediate feedback, and thus is favorable to iterative development where a project is tested and modified extensively throughout the development cycle. Furthermore, the abstraction of frequently used code is aided by the use of `.m` files. These `.m` files can be called by any other piece of code without specific inclusion directives—they only need to be in the correct folder or search path. This is used to good effect in developing the concepts of functions and abstraction. On a more technical note, creation of variables in MATLAB is as simple as typing an assignment statement (e.g., `myVar = 12.345`). In Java or C++, variables must be explicitly declared and assigned to a particular data type before they can be used; while this practice does lead to more efficient code, it often proves a stumbling block for new programmers, or even experienced developers when prototyping a new project. MATLAB programmers can concentrate on higher-level abstractions and program semantics without bothering with memory management.

Finally, a sizable body of code already exists in MATLAB for many common speech-processing tasks and theoretical models, including the influential Klatt synthesizer (Klatt, 1980) and the DIVA model (Guenther et al., 2006). And because MATLAB works on Windows, Mac OS X, and Linux, most of this code is easily portable across platforms when appropriate conventions are used. This applies to reader-generated code as well, allowing one to, say, write code on a Mac at home and run it successfully on a PC or Linux machine in the lab or clinic the next day. Throughout the text, we will emphasize code portability when discussing use of file names, paths, and file operations.

Some Disadvantages

We would be remiss to overlook MATLAB's relatively low speed for iteration in extremely computationally intensive operations. Because MATLAB is a high-level, interpreted (not compiled) language, it runs much more slowly than languages closer to the machine's native hardware, like C++, unless proper vectorization techniques are applied. The speed issue can be alleviated through the use of `.mex` files, which are compiled code and run many times faster than the corresponding `.m` files, but which can also require translating the MATLAB code into a compiled language and running it through a separate compiler for each machine architecture (32- vs. 64-bit, e.g.). On Windows machines, precompiled DLL files can also be used with the `calllib()` function. The creation of such files, however, is beyond the scope of this introductory textbook.

The MATLAB® Environment

This is the default layout of the MATLAB window. When MATLAB is freshly installed, its main window, or Desktop, will look similar to Figure 1–1.

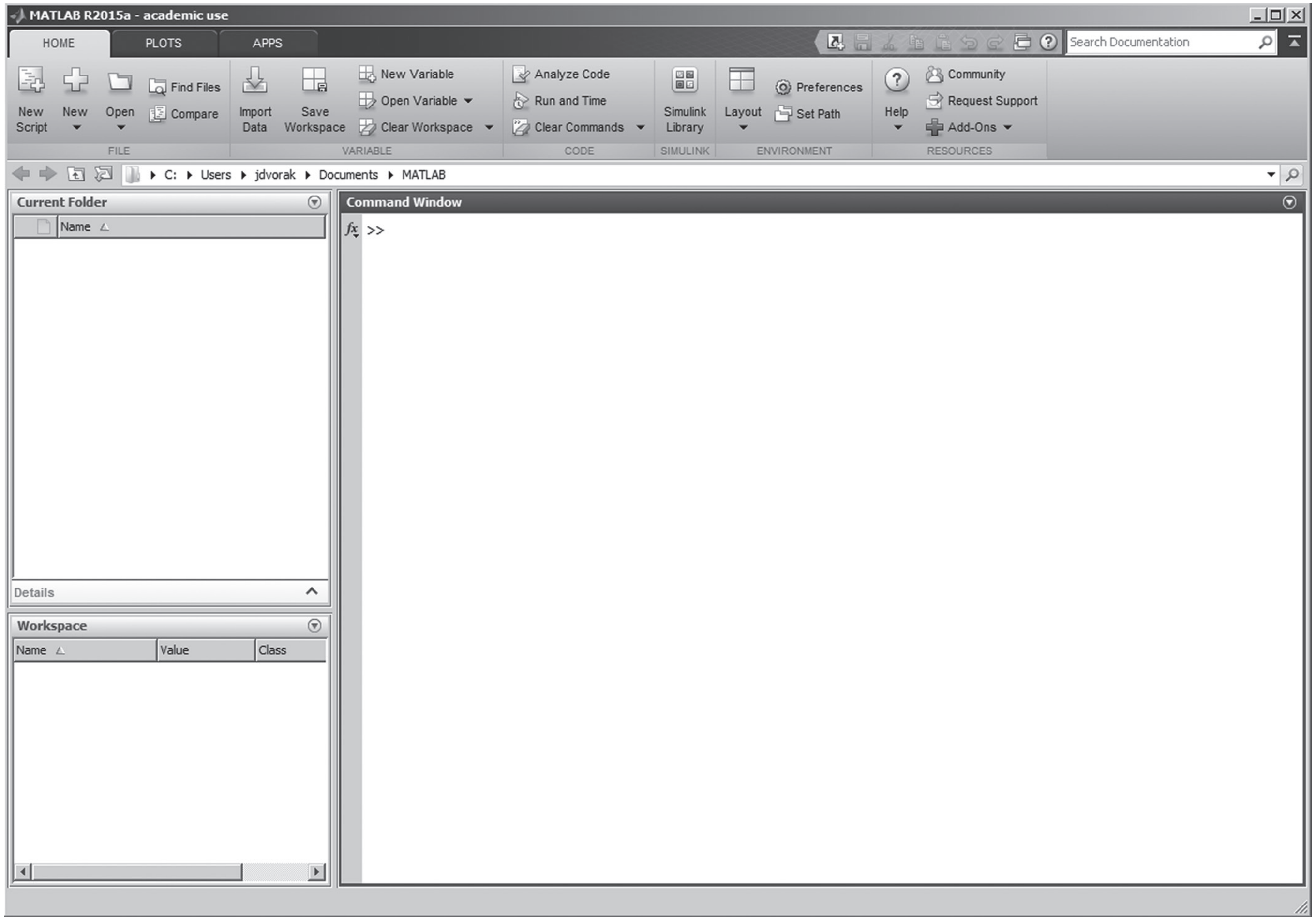


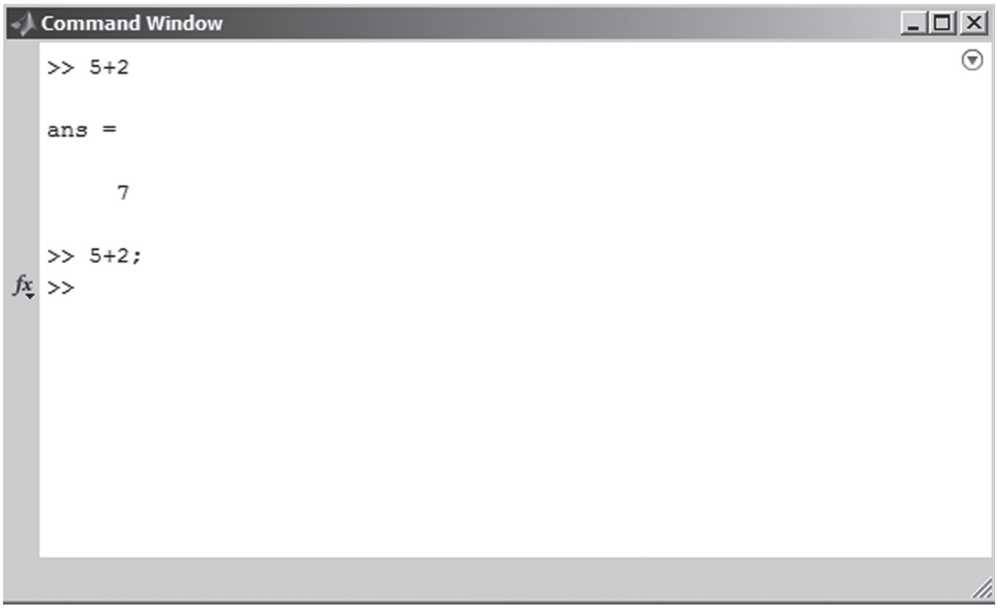
Figure 1–1. Default layout of the MATLAB window.

The topmost element of the Desktop is the Toolstrip, which is a tabbed interface to a wide variety of MATLAB functions and commands. Some functions are specific to the MATLAB environment itself, such as the "New Script" and "Open" buttons, whereas others are more applied, such as those in the "Plots" or "Apps" tabs.

Below the Toolstrip, the desktop is divided into a number of smaller windows, or panels. The most important panel is the Command Window, in which typed commands are executed, calculations are performed, and functions calls are made (Figure 1–2). Commands are entered after the prompt (>>), and submitted by pressing the ENTER or RETURN key. When a calculation is completed, results will also be displayed in the command window unless output is suppressed by terminating the statement with a semicolon (;).

A useful tool for new MATLAB programmers is the Function Browser button, which is indicated as a script *fx* to the left of the prompt in the Command Window. As the name would suggest, clicking on this button allows one to browse through all functions accessible through base MATLAB and any installed Toolboxes, insert them into scripts or the command window, or call up their documentation.

To the left of the Command Window is the Current Folder panel. Organized as a file manager, this panel displays all files and subfolders in the



```
>> 5+2

ans =

     7

>> 5+2;
>>
```

Figure 1–2. Command window.