Published in final edited form as:

Sci Signal. ; 4(191): tr7. doi:10.1126/scisignal.2001984.

An Introduction to MATLAB

Eric A. Sobie*

Department of Pharmacology and Systems Therapeutics and Systems Biology Center New York,
Mount Sinai School of Medicine, New York, NY 10029, USA

Abstract

This two-part lecture introduces students to the scientific computing language MATLAB. Prior computer programming experience is not required. The lectures present basic concepts of computer programming logic that tend to cause difficulties for beginners in addition to concepts that relate specifically to the MATLAB language syntax. The lectures begin with a discussion of vectors, matrices, and arrays. Because many types of biological data, such as fluorescence images and DNA microarrays, are stored as two-dimensional objects, processing these data is a form of array manipulation, and MATLAB is especially adept at handling such array objects. The students are introduced to basic commands in MATLAB, as well as built-in functions that provide useful shortcuts. The second lecture focuses on the differences between MATLAB scripts and MATLAB functions and describes when one method of programming organization might be preferable to the other. The principles are illustrated through the analysis of experimental data, specifically measurements of intracellular calcium concentration in live cells obtained using confocal microscopy.

Copyright 2008 by the American Association for the Advancement of Science; all rights reserved.

*Corresponding author. eric.sobie@mssm.edu.

Educational Details

Learning Resource Type: Lecture, assignment, digital presentation

Context: Graduate

Intended Users: Teacher, learner

Intended Educational Use: Learn, plan, teach

Discipline: Biostatistics, theoretical biology

Technical Details

Format: PowerPoint (.ppt)

Size: 2.12 MB

Requirements: Microsoft PowerPoint

Format: MATLAB (.m)

Requirements: Mathworks MATLAB

Format: JPEG (.jpg)

Requirements: Mathworks MATLAB

Supplementary Materials

<http://stke.sciencemag.org/cgi/content/full/sigtrans;4/191/tr7/DC1>

Slides

Example files

Problem set files

Answer key files

Citation: E. A. Sobie, An introduction to MATLAB. *Sci. Signal*. **4**, tr7 (2011).

Keywords

Computer programming; mathematical modeling; numerical methods; simulation

Lecture Notes

Teaching computational modeling to students requires them to have a platform in which to implement their models. In the course “Systems Biology—Biomedical Modeling,” which was developed for first- and second-year graduate students in a medical school environment, students implement models in the scientific computing language MATLAB. Before students are taught specific models of biological processes, they must first learn the basics of how to program in this language. For students who have previously programmed in C++, Fortran, or Pascal (for example, engineering, mathematics, or physics majors), learning the nuances of the MATLAB syntax is not particularly difficult. However, in our experience, few students have programming experience; therefore, most students must be taught fundamentals of programming logic in addition to details that are specific to the MATLAB environment. The initial lectures emphasize basic principles that apply across all computing languages. Because of its introductory nature, students who have prior programming experience may find these lectures to be essentially a review. For students without this experience, these lectures are a prerequisite to using MATLAB.

These materials are intended for instructors who have basic familiarity with MATLAB. Additional MATLAB tutorials are available from:

the Mathworks, creators of MATLAB (http://www.mathworks.com/academia/student_center/tutorials/launchpad.html),

the University of Utah (<http://www.math.utah.edu/lab/ms/matlab/matlab.html>),

the University of Maryland (<http://terpconnect.umd.edu/~nsw/ench250/matlab.htm>),

the University of Cambridge (<http://www.eng.cam.ac.uk/help/tpl/programs/matlab.html>).

Lecture 1

One of the strengths of MATLAB as a computing tool is the fact that computations are “vectorized,” meaning that the program makes it easy to perform operations on multidimensional mathematical objects. Vectorized computing makes it easy to simultaneously manipulate, for instance, all of the values in a two-dimensional array. It also makes the program flexible in terms of how variables are defined. The first part of the lecture discusses general concepts related to vectors, arrays, and matrices, and provides examples of biological data that are stored as arrays (Slides 1 to 8). Examples include immunofluorescence images, transcriptional arrays, and results obtained through numerical integration of dynamical models.

This is a laboratory class. During the lecture, students run MATLAB and execute the basic operations described in the lecture. Students are introduced to the MATLAB operations that can: (i) define variables and perform simple arithmetic (Slide 9); (ii) create one-dimensional

vectors (Slide 10); (iii) create two-dimensional matrices using concatenation (Slides 11 and 12); and (iv) add and multiply matrices (Slide 13). This part of the lecture also describes the “whos” command, which lists the dimensions of all the currently defined variables and is useful for identifying sources of errors (Slide 14).

After the introduction to MATLAB operations, several important concepts are illustrated through a biologically relevant example—binding of a ligand to its receptor. Students receive a set of MATLAB scripts that compute and plot ligand-receptor complex as a function of ligand concentration for different values of the dissociation constant (K_D) (Slides 15 to 17 and Supplementary Materials, Example Files). The scripts (plot_LR_v1.m through plot_LR_v5.m) are arranged from the simplest to the most sophisticated, such that the progression from one to the next teaches the students increasingly more complex methods for computing in the MATLAB environment. The principles illustrated by this example include plotting in different colors, labeling plots, array-based arithmetic, for loops, and accessing sections of multidimensional variables.

Lecture 2

The second lecture begins with a review of concepts from the previous lecture (Slides 18 to 21), including accessing sections of arrays (Slide 20) and element-by-element multiplication of arrays (Slide 21). The rules underlying matrix multiplication are reviewed along with a simple example (Slide 22). The syntax of several useful builtin MATLAB functions that allow for simplified element-by-element or column-by-column computations is also described. These allow the user, for instance, to compute means and standard deviations of vectors (Slide 23), to determine values and locations of maxima (Slide 24), and to combine these functions with concatenation (Slide 25).

The lecture also includes a description of the “if/else” statement (Slide 26), a powerful computing tool because this structure can instruct MATLAB to execute certain commands only if a particular condition is met (for instance, after a maximum has been reached). The section on the basics of MATLAB programming concludes by reviewing important aspects of MATLAB syntax (Slide 27) and demonstrating how to import data and export results into formats that can be read by other programs (Slide 28).

A strength of MATLAB is that it allows for both element-by-element array multiplication and for mathematically formal matrix multiplication. To help students to differentiate between the two, an example of the latter is provided in Slides 29 to 33. Mathematical models of cardiac action potentials (1–3) typically consist of 10 to 50 ordinary differential equations and 10 to 20 parameters with obvious biological meaning, such as the density of an ion channel in the cell membrane or the maximal activity of an ion transporter (Slide 30). In order to evaluate how changes in these quantities affect important model outputs, such as action potential duration, parameter sensitivity methods based on matrix multiplication have recently been developed (4, 5). These methods are described in Slides 31 to 33 to illustrate the basic concepts of matrix multiplication.

The lecture concludes with a discussion of the MATLAB programming concept of scripts versus functions (Slides 34 to 41). A script is a file that can be edited in the MATLAB

environment, and the commands in the script can be run by typing the name of the script at the MATLAB prompt (Slides 34 and 35). If a script executes particular commands repeatedly, however, it may be more convenient to save these commands as a separate MATLAB function and to call this function from each script that requires these commands (Slides 36 and 37). It is important to note, however, that variable names within a function are generally hidden from the scripts that call these functions (Slide 38). These principles are illustrated through an example of a custom-written function that reads data files stored in a particular proprietary format and returns the raw data to the user (Slides 39 to 41). Because the details of the proprietary format are probably not of much interest to the user, it is best to “hide” these details by defining certain variables only within the function (Slide 41).

Slide 42 shows an example of data analysis performed with MATLAB that the students reproduce as part of the homework assignment.

Problem Set

Introductory Details

The homework assignment consists of two parts. The first requires the students to generate plots of ligand bound to a receptor in the presence of different concentrations of a competitive inhibitor. To generate these plots, the students use as a template a MATLAB script that plots ligand-receptor complex for different values of the dissociation constant K_D . For the second part, students write a simple routine to import, manipulate, and display experimental data (6). The manipulations required in the assignment include (i) accessing only a particular portion of the data (a defined spatial position), (ii) spatial averaging to reduce noise, (iii) normalization, and (iv) converting the data into different units.

The Supplemental Materials contain two files required to complete the homework assignments. The MATLAB script “computeLR.m” generates a plot of the concentration of ligand-receptor complex for different values of K_D . The file “flash4.jpg” contains fluorescence data obtained using a confocal microscope and a custom-built system for spatially localized flash photolysis (6). These data are analyzed by the students in Part 2.

Student Assignment: Part 1

Slides 16 and 17 illustrate plots of the concentration of ligand-receptor complex as a function of ligand concentration for different values of the dissociation constant K_D . Consider how the situation becomes more complicated when ligand binds to its receptor in the presence of a second species $[I]$ that can bind to the same site (a competitive inhibitor). This reaction is defined by

The equation describing steady-state binding of ligand to receptor is:

$$[LR] = \frac{[R]_{\text{TOTAL}} [L] K_I}{[L] K_I + K_D K_I + K_D I}$$

where $K_D = \frac{k_{-1}}{k_1}$ and $K_I = \frac{k_{-2}}{k_2}$ are the respective dissociation constants for the ligand and the competitor I . (Note: This equation is similar to that for competitive inhibition of an enzyme. Here, the situation is simplified by only considering the binding of the ligand to the receptor and not any downstream events.)

The presence of a competitive binding species decreases the apparent affinity of the primary reaction. Using the MATLAB script “computeLR.m” as a template, modify this script to include the effects of a competitive inhibitor. Plots of $[LR]$ versus $[L]$ for different values of I will show that this decrease in affinity occurs. Assume that $R_{\text{TOTAL}} = 20$ nM and that $[L]$ ranges from 0 to 200 μM . Assume that $K_D = 20$ μM and $K_I = 100$ μM .

1. Modify “computeLR.m” so that it now plots $[LR]$ versus $[L]$ for $[I] = 20$ μM , 100 μM , 250 μM , and 1 mM.
2. Generate a plot of the “apparent K_D ” versus I . Remember that, for binding without competition, K_D is defined as the concentration of $[L]$ at which 50% of the receptors are occupied. Thus, for each value of $[I]$, the apparent K_D is the concentration of $[L]$ at which 50% of the receptors are occupied. (Note: To compute the apparent K_D for the plot with the highest concentration of I , the maximum value of $[L]$ has to be increased.)

Student Assignment: Part 2

For this question, you will reproduce the data analysis shown in Slide 42 and in Fig 1. In this experiment, a rat ventricular myocyte was loaded with the Ca^{2+} -sensitive indicator fluo-3, and changes in intracellular $[\text{Ca}^{2+}]$ were recorded with a confocal microscope (6). The microscope was operated in “line scan” mode such that one dimension of the resulting “pseudo-image” is space, and the other is time. This cell was also loaded with the Ca^{2+} buffer NP-EGTA (7). This buffer has unique properties such that, when it is exposed to high-intensity ultraviolet (UV) light, it loses its ability to bind Ca^{2+} . Thus, delivering a pulse of UV light to a cell loaded with NP-EGTA provides a means to rapidly increase $[\text{Ca}^{2+}]$ by “uncaging” it from the NP-EGTA (6).

The initial, local increase in $[\text{Ca}^{2+}]$ in this image results from a spatially localized flash of UV light. The second increase results from electrical stimulation of the cell. The goal is to determine how the increase in $[\text{Ca}^{2+}]$ due to the flash affects the later increase due to the electrical stimulus. The regions of the cell not exposed to the UV light serve as internal controls.

To perform this analysis, perform the following steps:

1. Read in the file: `data = imread('flash4.jpg','jpg');`

2. Look at the data using the `imagesc` function. Orient the data so that time runs from left to right. Transpose the matrix if necessary.
3. Average over the region of the UV flash. Store this in the variable `flash`. This should have dimensions 1×634 .
4. Average over a control region that does not contain the flash. Store this in the variable `noflash`.
5. The fluorescence units are arbitrary, because the number depends on laser intensity, dye concentration, microscope detector gain, and other factors. Thus, the data are analyzed as relative changes in fluorescence (F) over the baseline value (F_0) in a resting cell. Convert from raw fluorescence to units of F/F_0 by normalizing flash and noflash to the average fluorescence in a region with no activity. (Hint: Between lines 100 and 150 is a good region).
6. To a first approximation, one can assume that the dye used in this experiment, fluo-3, only emits fluorescence when it is bound to Ca^{2+} . Thus, where K_D is the dissociation constant of the dye, and $[\text{Fluo3}]_{\text{TOTAL}}$ is the dye concentration, fluorescence is proportional to:

$$F \sim \frac{[\text{Ca}^{2+}] [\text{Fluo3}]_{\text{TOTAL}}}{[\text{Ca}^{2+}] + K_D}$$

If one makes a reasonable assumption for baseline $[\text{Ca}^{2+}]$ (example, 100 nM), the following equation can be used to convert from a ratio R (units of F/F_0) to $[\text{Ca}^{2+}]$ in units of concentration:

$$[\text{Ca}^{2+}] = \frac{RK_D}{\frac{K_D}{[\text{Ca}^{2+}]_{\text{baseline}}} - R + 1}$$

Implement this equation to convert flash and noflash from units of F/F_0 to units of $[\text{Ca}^{2+}]$ in nM. Assume $[\text{Ca}^{2+}]_{\text{baseline}} = 100$ nM and $K_D = 700$ nM. The variables flash and noflash are not just scalars (numbers) but are defined at many time points.

7. Plot both flash and noflash versus time, on the same scale, in different colors (see Fig. 1). The spacing between lines in the image shown in flash4.jpg was 1.53 ms. Thus, for the time scale to be correct, the time vector should begin at zero and consist of points spaced 1.53 ms apart (Fig. 1).

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

The author thanks Y.-S. Lee for testing the problem set.

Funding: The development of this course was supported by a Systems Biology Center grant (P50 GM071558) and a training grant in Pharmacological Sciences (T32GM062754).

References and Notes

1. Bondarenko VE, Szigeti GP, Bett GC, Kim SJ, Rasmusson RL. Computer model of action potential of mouse ventricular myocytes. *Am. J. Physiol. Heart Circ. Physiol.* 2004; 287:H1378–H1403. [PubMed: 15142845]
2. Luo CH, Rudy Y. A dynamic model of the cardiac ventricular action potential. I. Simulations of ionic currents and concentration changes. *Circ. Res.* 1994; 74:1071–1096. [PubMed: 7514509]
3. ten Tusscher KH, Noble D, Noble PJ, Panfilov AV. A model for human ventricular tissue. *Am. J. Physiol. Heart Circ. Physiol.* 2004; 286:H1573–H1589. [PubMed: 14656705]
4. Sarkar AX, Sobie EA. Regression analysis for constraining free parameters in electrophysiological models of cardiac cells. *PLOS Comput. Biol.* 2010; 6:e1000914. [PubMed: 20824123]
5. Sobie EA. Parameter sensitivity analysis in electrophysiological models using multivariable regression. *Biophys. J.* 2009; 96:1264–1274. [PubMed: 19217846]
6. Sobie EA, Kao JP, Lederer WJ. Novel approach to real-time flash photolysis and confocal $[Ca^{2+}]$ imaging. *Pflugers Arch.* 2007; 454:663–673. [PubMed: 17323075]
7. Ellis-Davies GCR, Kaplan JH. Nitrophenyl-EGTA, a photolabile chelator that selectively binds Ca^{2+} with high affinity and releases it rapidly upon photolysis. *Proc. Natl. Acad. Sci. U.S.A.* 1994; 91:187–191. [PubMed: 8278362]

Fig. 1.
Example of data analysis to be performed in the homework assignment. Traces show estimates of intracellular $[Ca^{2+}]$ recorded by fluo-3 at two locations within a heart cell.