Intrduction to Matlab

Professor Crodelle[*]

Spring 2022

1 Introduction

What is MATLAB?

Matlab is a computational software program used widely in several areas of research in mathematics and science. It works efficiently with large data and is a powerful numerical solver. It allows users to quickly produce, analyze, and edit data and create quality graphics. It also has symbolic computation capabilities.

1.1 Starting up MATLAB

Open Matlab by clicking on the icon. Matlab automatically creates a folder in Documents called Matlab in which it will look for and save all files. I advise you to create a folder in your Google Drive for Matlab code, so that you may have access to it from anywhere.

You might need to reset the directory in Matlab every time. When downloading Matlab files for this course, always save them to the same directory.

1.2 Getting Help

Matlab provides help resources from within the program. You can access them by doing any of the following:

- Click on the 'Help' icon in the 'Resources' section in the 'Home' tab to access the help documentation file, or
- type the command >> `doc` into the command line and hit enter. (Try this!)
- Get help for a specific command by typing >> `help commandName` directly into your command line and hitting enter. This command will output a description of how to use the `commandName` and a linked list of related commands. Try typing >> `help linspace` to see this feature.

Matlab is very well documented, and many troubleshooting questions have been asked and answered on the Mathworks website (http://www.mathworks.com) and by doing an internet search.

Finally, do not forget to utilize each other and your professors as resources. Working together with peers to learn a new software language is a great idea. You are welcome and encouraged to ask Matlab questions during office hours.

[*]Most of the content was prepared by Professor. Kubacki

2 Command Practice

2.1 Arithmetic, variables, and your workspace

Arithmetic operations are very intuitive in MATLAB. The arithmetic operations $+, -, \times, \div$ can be executed with the commands `+,-,*,/`. Use parentheses, `()`, to preserve order of operations, but please note that adjacent parenthesis, `()()`, and variables, `5x` or `ab`, do not translate to multiplication in MATLAB. Unlike written mathematics, you must indicate multiplication with `*`, meaning (5)(10) or ab in MATLAB must be typed as `5*10` or `a*b`.

Exercise 1. *Execute the following commands in your own command prompt (hitting enter after each). Verify that you generate the correct output.*

1. $2^3 - (3-6)/2 + 7(.5)$ *(Output: `13`)*
2. $\frac{3^2-2(3)+4}{2+7}$ *(Output: `0.777777777777778`)*
3. $\frac{3+4.5}{7^2-2(9)} - \frac{2(1.5)-1}{3-6.2}$ *(Output: `0.866935483870968`)*
4. $\left(\frac{\sqrt{7}-\sqrt{6}}{4}\right)^2$ *(Output: `0.002407412699017`)*

Tip: MATLAB remembers your previous commands. To recall a previous command, hit the up-arrow key while your cursor is in the command prompt.

Exercise 2. *You can assign variable names to calculations. Repeat the commands in (1)-(4) in the previous exercise assigning the variable names `a,b,c`, and `d` to each calculation in 1-4 above. To do this, type `a =` before the calculation, then hit enter to execute. Once you have created the variables, you will see them in your workspace (usually found on the left-hand side, under the contents of your directory). Now, type, >> `(a+b)/c+d^2` in your command line. The output should be `15.892512255584224`.*

Tip: You can clear variables using the command `clear`. To do this, type `clear a` and hit enter and watch `a` disappear from your workspace. Type `clear all` to clear all variables from your workspace.

MATLAB has a few reserved variables, given in Table 1. Avoid accidentally redefining any of these variables. ***Take special note that the number `e` is not among the reserved variables.*** Type >> `help variableName` into your command prompt to learn more about each reserved variable.

2.2 Working with arrays

The power of MATLAB lies in its ability to work efficiently with arrays (vectors, matrices) when performing calculations. Thus, whenever possible, we try to define our problems using arrays. Instructions for constructing arrays in MATLAB are given in the table below.

variable	**meaning**
`ans`	result of the previous calculation
`computer`	type of computer you are on
`eps`	machine epsilon
`i, j`	$\sqrt{-1}$
`inf`	infinity; this is also the result of dividing 1 by 0
`NaN`	'not a number,' results from $0/0$, ∞/∞, $0 \cdot \infty$
`pi`	π
`realmax, realmin`	largest and smallest real numbers represented

Table 1: Reserved Variables in Matlab

Arrray	**Math**	**Matlab**
row vector	$\begin{bmatrix} a_1, & a_2, & a_3, & a_4 \end{bmatrix}$	`[a1 a2 a3 a4]` or `[a1, a2, a3, a4]`
column vector	$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$	`[a1; a2; a3; a4]`
matrix	$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$	`[a11 a12; a21 a22]`

Table 2: Arrays in Matlab

Exercise 3. *In Table 2 you see how different array variables are created. Use it to create the following variables. Examine the variables in your workspace. Try double clicking each one.*

$$\texttt{rowVec} = [1.01, 17, \pi, 1/2] \qquad (\textit{use } \texttt{pi} \textit{ for } \pi), \qquad \texttt{colVec} = \begin{bmatrix} 3/4 \\ 15.8 \\ 3 \\ -5 \end{bmatrix}, \qquad \texttt{matrix} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Exercise 4. *When performing arithmetic with arrays, you need to obey the dimension rules you learned in linear algebra. Try the following operations with the variables you just created. Notice how when you violate dimension rules you generate an error* [1].

1. `matrix^2` *Answer or Error?*
2. `rowVec + colVec` *Answer or Error?*
3. `matrix * colVec` *Answer or Error?*
4. `rowVec * matrix` *Answer or Error?*
5. `rowVec * colVec` *Answer or Error?*
6. `matrix + rowVec` *Answer or Error?*

[1] Note that in some cases, Matlab does **not** throw an error, but instead calculates something completely different. Always make sure that Matlab is computing what you expect it to compute.

2.3 Functions in Matlab

Not surprisingly, elementary functions, such as e^x and $\sin(x)$ are built-in to the MATLAB library (see Table 3 for examples).

function	MATLAB function
e^x or $\exp(x)$	`exp(x)` *** not `e^x` ***
$\ln(x)$	`log(x)` *** not `ln(x)` ***
$\cos(x)$, $\sin(x)$, $\tan(x)$	`cos(x)`, `sin(x)`, `tan(x)`
$\arctan(x)$	`atan(x)`
$\sqrt{x}$	`sqrt(x)`

Table 3: Built-in MATLAB functions

Exercise 5. *Use* MATLAB *to calculate the following (expressions).*

1. *Compare* $\sin(\pi)$ *to* $\sqrt{1-\cos^2(\pi)}$. *Is the answer what you would expect? Why or why not?*
2. *Compute* $\arctan(1/\sqrt{3}) - \arctan(\sqrt{3}/3)$. *Is the answer what you would expect? Why or why not?*

It is important to remember that in many cases, MATLAB performs calculations numerically unless otherwise told. This means that calculations are performed using finite arithmetic and numerical algorithms (as needed). In many cases, calculations are accurate to with about 16 decimal places (although there are exceptions). If you desire MATLAB to perform a symbolic computation you must use either *symbolic variables* or the command `sym`.

Exercise 6. *Repeat the calculations from the previous exercise. This time preface each calculate with the command* `sym`. *(type* `sym(calculation)`*). What changes?*

Sometimes it may be convenient to define your own function. You can do so by creating an **anonymous function handle**. For example, if you wanted to define the function $f(x) = x^2 - \ln x$, in MATLAB you would type the command: `>> f = @(x)x^2 + log(x)`.

Exercise 7. *Define the anonymous function* $f(x) = x^2 - \ln x$ *as described above. Then use it to make the following calculations* $f(1), f(0)$, $f(e)$, $f(\pi)$. *Record their values out to 4 decimal places.*

Suppose we wanted to evaluate the function $f(x)$ on a list of x-values, $x = 1, 2, 3, \ldots, 100$. The efficient way to find these values is to *vectorize* the function f by redefining it with **componentwise-arithmetic** so that it can act *componentwise* on arrays. In other words, we want f to act on each component of the input vector (or matrix):

$$f([x_1, x_2])) = [f(x_1), f(x_2)].$$

By default, addition (+), substraction ($-$) and scalar multiplcation/division act componentwise on arrays. However, multiplication and division do not. To invoke componentwise arithmetic, include a period, `.` , before any multiplication or division operator (e.g. `.*`, `./`, `.^`), then MATLAB will perform that calculation componentwise on arrays.

Exercise 8. *Redefine* f *from the previous exercise using componentwise arithmetic. Define the variable* `X=1:100;` *(this command produces the vector* `X = [1 2 3...100]`*, and* `;` *suppresses the output). Now execute* $f(X)$.

Tip: Suppress any unnecessary output with `;` at the end of the command line. This is essential to remember since we will often do calculations large arrays. If you forget to suppress output and want to clear your command prompt, type >> `clc`.

2.4 Plotting

The default command for plotting in MATLAB is `plot`. This command acts on numerical input (think: vectors). The basic syntax is `plot(Xdata, Ydata)`, where `Xdata` is a vector containing the x-values of the data you would like to plot, and `Ydata` the corresponding y-values (both vectors must be of the same length). MATLAB then plots `Xdata` and `Ydata` by pairwise plotting the points (`Xdata`(i), `Ydata`(i)) and connecting subsequent points with a straight line.

Exercise 9. *Using the same f as in the previous exercise, define `Xdata = 1:100;` and `Ydata=f(X);`. Plot (`Xdata`, `Ydata`) using the `plot` command. Using >> `help plot`, figure out how to change the color of your plot and marking data points with o's.*

You can make multiple plots using a single `plot` command. To do so, you continue listing your x and y-data pairwise: `plot(Xdata1, Ydata1, Xdata2, Ydata2, ...)`. MATLAB will automatically assign a different color to each plot.

Tip: When you execute the command `plot`, MATLAB creates a separate figure window with your plot. Sometimes this window may be hidden behind the main window of MATLAB. Typing the command `close` into your command line and hitting enter will close your most recently created figure window. The command `close all` will close all figure windows.

Exercise 10. *Create a plot of the graphs of $\sin(x)$ and $\cos(x)$ over the interval $[0, 2\pi]$. Use enough x-data points to make your plot appear smooth to the eye.*

3 Creating, using, and saving Matlab files

You can make and save pdfs of your commands/code and output by creating a live script. This is how we will complete homework problems that require MATLAB. To do this, follow the steps below.

1. Click the 'New Live Script' button on the top left. This opens a new document in which you can type both text and MATLAB code.

2. Select the editor tab. Save your file by clicking on the button 'Save' and selecting 'Save as'. Notice you are automatically saving the file in the designated folder you created and selected earlier. The extension will be '.mlx' for a live script.

3.1 Titles, sections, and comments

You should use the Text option to create a title for your document by choosing 'Heading' under text options. To write MATLAB code, click the code button on the toolbar. You should notice that the font is different when typing code and that line numbers will appear on the left-hand side. You can use `%%`-sign to separate sections of code. A single `%` followed by text is a comment

(`% comment text`). Comments appear in green. You can usually use `%% titles` as breaks between sections of code; however, in the live script editor, you must click the 'Section Break' button to separate chunks of code.

3.2 Exporting to PDF

To hand in your work, make sure to export your live script as a pdf. To do this, click on the arrow below 'Save' to pull up a menu. Click on 'Export to PDF' and click save. **Make sure to always save your mlx-file and export the final output to PDF.

Tip: For more help on live scripts, explore the following page: `https://www.mathworks.com/help/matlab/matlab_prog/create-live-scripts.html`

$$\frac{1}{2}$$