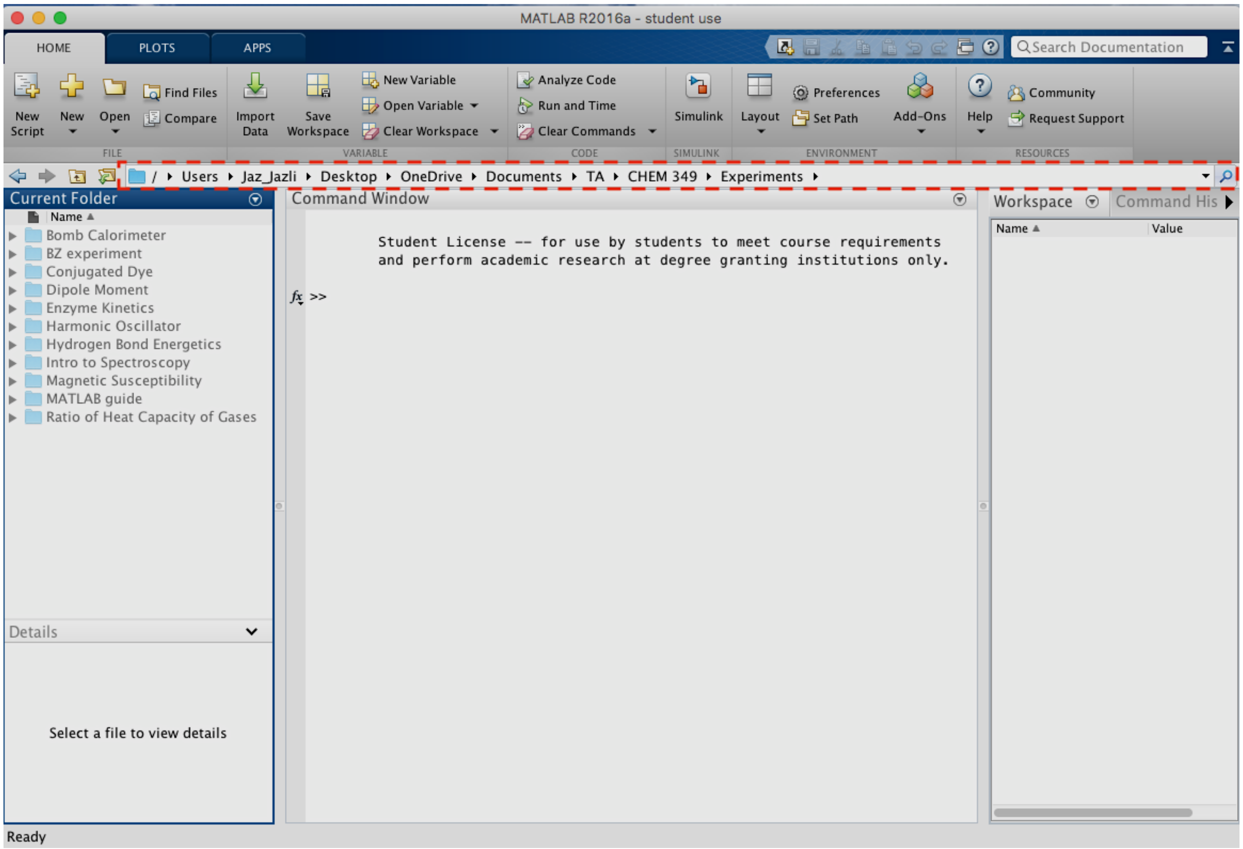
1 Basic

Keywords: import data, indexing, array operation, matrix operation, size, length

1.1 MATLAB Layout

Figure (1) shows the default layout of MATLAB. **Current Folder** window shows the contents of your *Current Folder*. All files that you save from MATLAB will be put in your *Current Folder*. The area with the dashed-line box shows you the path to your *Current Folder*. The **Command Window** is where you type the 'command' for MATLAB. Basically this is the place for you to instruct MATLAB to do something. The **Workspace** is where all of the variables for the current session will be stored temporarily. If you create a matrix or a vector, it will appear here. Any variables stored here will be deleted when the MATLAB is closed.



```
Student License -- for use by students to meet course requirements
and perform academic research at degree granting institutions only.

fx >>
```

Figure 1: Default MATLAB layout.

1.2 Variables

To create a *row vector*, type the following

$$A = [1, 2, 3, 4, 5]; \tag{1}$$

A is just the name of the vector. Use other names that aptly describes your vector, i.e, if the vector contains velocities of a wave front, then the vector can be named as *velocity.* Be careful with lowercase and uppercase because MATLAB is case-sensitive. Velocity, velocity, and VELOCITY refer to three different vectors. Alternatively, the commas in Command (1) can be replaced with a space after each element such as

$$B = [1\ 2\ 3\ 4\ 5]; \tag{2}$$

and the same result will be obtained.

To create a *column vector*, type the following

$$C = [1; 2; 3; 4; 5]; \tag{3}$$

Alternatively the apostrophe sign ($\prime$) can be used to transpose your vector.

$$D = [1\ 2\ 3\ 4\ 5]'; \tag{4}$$

Both Command (3) and (4) will give the same result (but each of the command assigned the values to different names – C and D).

To create a *matrix*, you sort of combine Command (2) and (3). For instance if you want to create a *matrix* that contains numbers from 1 to 9 and named it *numbers*, you can type the following command

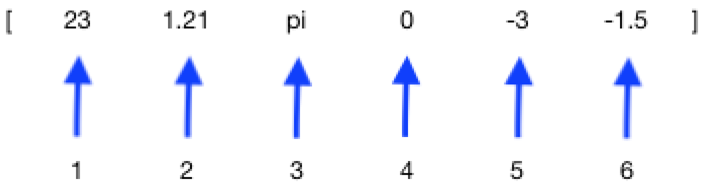
$$numbers = [1\ 2\ 3;\ \ 4\ 5\ 6;\ \ 7\ 8\ 9]; \tag{5}$$

try typing Command (5) in your **Command Window** to see the result.

Row vectors and column vectors can be thought as a matrix with a single row and single column respectively. The dimension of matrix *numbers* is [3 × 3]. The dimension of matrix C (a column vector) is [5 × 1] while the dimension of matrix A (a row vector) is [1 × 5]. The size is always in the format of [*row* × *column*].

1.3 Vector Indexing

Indexing is just the position of an element within a vector or a matrix. MATLAB indices must be real positive integers. Consider the following example



say if the name of the above vector is *NUM*, and you need to get the value of third element from vector *NUM*. You can type the following command, **NUM(3)** and MATLAB will tell you that the value of the third element in NUM is π. If you need to extract a "range of values", for example from position 2 to 5, and assign it to a new variable, you can type

$$New = NUM(2:5); \tag{6}$$

In this case, the new variable is called *New* and it contains all numbers from position 2 to 5 of vector NUM. The same method is also applied to a *column vector*.

1.4 Arithmetic Operation

Arithmetic operations can be performed on the vectors. To perform any arithmetic operation on the vectors, make sure your vector has the same size, i.e, if you are adding vector A and vector B (A + B), make sure the dimensions of A and B are the same. If the two vectors have different dimensions, MATLAB may give out error warning or it may do something that is beyond the scope of this tutorial.

For example, consider if you have two vectors **F** = [2 4 6 8 10] and **G** = [1 2 1 2 1]. If you perform addition and subtraction,

$$F + G$$
$$ans = [\ 3\ 6\ 7\ 10\ 11\ ]$$
$$F - G$$
$$ans = [\ 1\ 2\ 5\ 6\ 9\ ]$$

To do multiplication or division, use the 'element-wise' multiplication and division. To do that, add a period before the multiplication (*) and division (/). For example,

$$F. * G$$
$$ans = [\ 2\ 8\ 6\ 16\ 10\ ]$$
$$F./G$$
$$ans = [\ 2\ 2\ 6\ 4\ 10\ ]$$

There is an exceptional case when the arithmetic operations involve a vector and a single number. For example,

$$F + 2$$

$$ans = [\ 4\ 6\ 8\ 10\ 12\ ]$$

$$F - 2$$

$$ans = [\ 0\ 2\ 4\ 6\ 8\ ]$$

$$F * 2$$

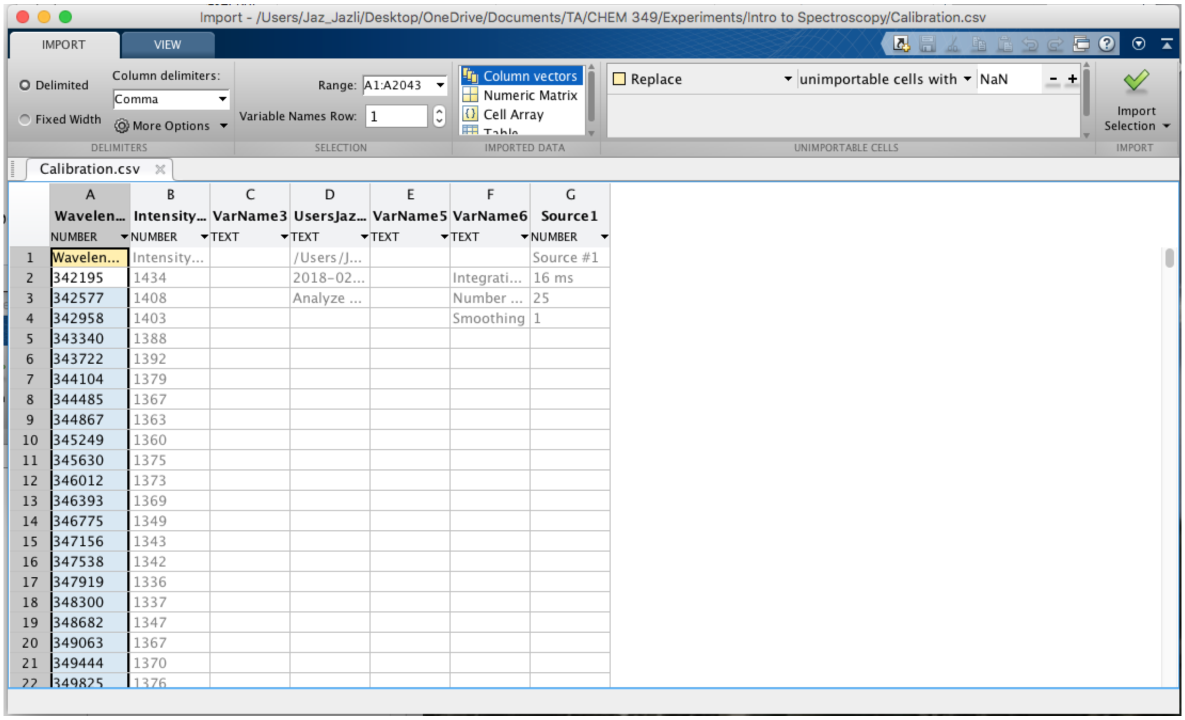
$$ans = [\ 4\ 8\ 12\ 16\ 20\ ]$$

$$F/2$$

$$ans = [\ 1\ 2\ 3\ 4\ 5\ ]$$

1.5 Import Data

MATLAB can read several types of file such as .csv and .xls (for full details go to MATLAB official website). To import data into MATLAB, make sure the file containing the data is in the *Current Folder*. Right click on the file and choose ‘Import Data...’ option. A new window such as shown in Figure (2) will pop-out.



Import - /Users/Jaz_Jazli/Desktop/OneDrive/Documents/TA/CHEM 349/Experiments/Intro to Spectroscopy/Calibration.csv

● Delimited ○ Fixed Width

Column delimiters: Comma

More Options

Range: A1:A2043

Variable Names Row: 1

Column vectors / Numeric Matrix / Cell Array / Table

□ Replace unimportable cells with NaN

Calibration.csv

	A	B	C	D	E	F	G
	Wavelen...	Intensity...	VarName3	UsersJaz...	VarName5	VarName6	Source1
	NUMBER	NUMBER	TEXT	TEXT	TEXT	TEXT	NUMBER
1	Wavelen...	Intensity...		/Users/J...			Source #1
2	342195	1434		2018-02...		Integrati...	16 ms
3	342577	1408		Analyze ...		Number ...	25
4	342958	1403				Smoothing	1
5	343340	1388					
6	343722	1392					
7	344104	1379					
8	344485	1367					
9	344867	1363					
10	345249	1360					
11	345630	1375					
12	346012	1373					
13	346393	1369					
14	346775	1349					
15	347156	1343					
16	347538	1342					
17	347919	1336					
18	348300	1337					
19	348682	1347					
20	349063	1367					
21	349444	1370					
22	349825	1376					

Figure 2: Import window for MATLAB

Select the data that need to be imported by highlighting the column that contain the data (refer to Figure (2)). Change the name of the variable if it needs to be change. This can be done double-clicking on the column header. Change the range if the range needs to be changed.

This can be done by double-clicking on the “Range” box at the top of the *Import Window*. Select the appropriate “variable” type. For this example, the variable type is *Column vectors*. Once everything has been set, click on the **Import Selection** button on the top right. If everything was done correctly, the variable should appear on the **Workspace** box.

1.6 Script and m-files

A MATLAB script is a file containing multiple sequential lines of MATLAB commands and functions. A MATLAB script file has an extension *.m*, which is usually called m-files. To open an m-file, the m-file has to be in the current directory. Then, double click on the m-files and a new ‘*Editor*’ window will pop-out. To execute the m-file, click the *run* button at the top of the *Editor* window. Alternatively, the m-files can be executed by typing the name of the m-file in the **Command Window** and hit enter.

2 Plotting

Keywords: plot, scatter, fitlm, plot tools, curve fitting tools

2.1 Scatter vs. Plot functions

There are multiple ways of making a graph in MATLAB. The two methods that will be discussed in this tutorial are **line plot** and **scatter plot**. The commands for the line plot and scatter plot are **plot(x,y)** and **scatter(x,y)** respectively. In both cases, x is the variable for your x-axis and y is the variable for your y-axis. The length of the vectors must be the same otherwise MATLAB will give error warning.

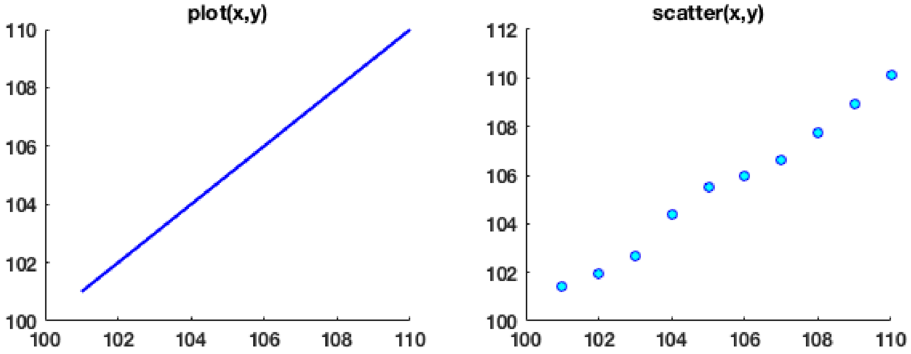


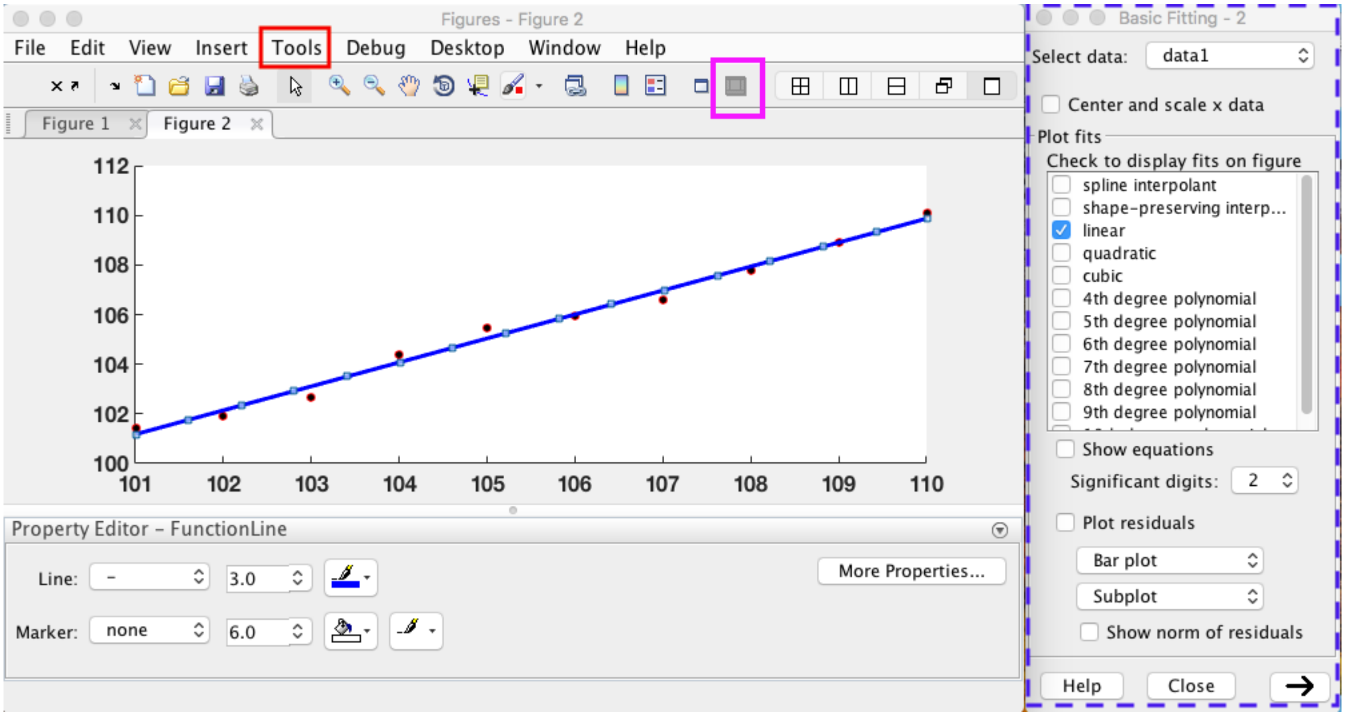
Figure 3: Graphs produced by command *plot(x,y)* (left) and *scatter(x,y)* (right).

2.2 Line Fitting

With scatter plot graph, you can fit a line to the graph. The simplest way to fit a line to your graph, is by clicking on 'Tools' > 'Basic Fitting' and a new window will pop out. This window will have a list of options for the type of line that you can fit into your graph.

2.3 Plot Customization

MATLAB also allows its users to customize the graph. To do that, just click on the 'Show Plot Tools' button (refer to Figure (4)). A new window called 'Property Editor' should pop-out, which provides you with options for customizing your graphs. The appearance of the 'Property Editor' will be different depending on the item that you are trying to customize. To customize your markers for your data points, click on one of the data points on the graph and then customize it using the 'Property Editor'. To remove the 'Property Editor', click on the 'Hide Plot Tools' button, which is located on the left of 'Show Plot Tools' button.



Line: – 3.0

Marker: none 6.0

Select data: data1

☐ Center and scale x data

☐ spline interpolant
☐ shape-preserving interp...
☑ linear
☐ quadratic
☐ cubic
☐ 4th degree polynomial
☐ 5th degree polynomial
☐ 6th degree polynomial
☐ 7th degree polynomial
☐ 8th degree polynomial
☐ 9th degree polynomial

☐ Show equations

Significant digits: 2

☐ Plot residuals

Bar plot

Subplot

☐ Show norm of residuals

Figure 4: *Figure* and *Basic Fitting* window. 'Tools' button is highlighted with solid red box while *Basic Fitting* window is highlighted with dash-blue box. The 'Plot Tools' button is highlighted with solid magenta line. The 'Plot Tools' allows users to customize the graph.

2.4 Save Plot

To save the plot, click on 'File' > 'Save as'. Give the graph a name and change the format of the graph to a format that your computer can recognize (usually, .jpg, .png, or .tif).

3 Image Processing

Keywords: imread, imshow, figure

3.1 Import Image

To import images into MATLAB, the images that you want to import have to be in the *Current Folder*. Use the function *imread* to import images.

$$pic \ = \ imread(`fileName.jpg'); \tag{7}$$

"fileName.jpg" is the name of your file and it has to be within the quotation marks (' ') as shown in Command (7). Also, make sure you have the semicolon after the function because this suppresses the output from the function. You can assign a name to the 'imported' image as shown in Command (7). In this case, the imported image is named *pic*.

3.2 View Image

To view the imported image, you can use the function *imshow* as follows,

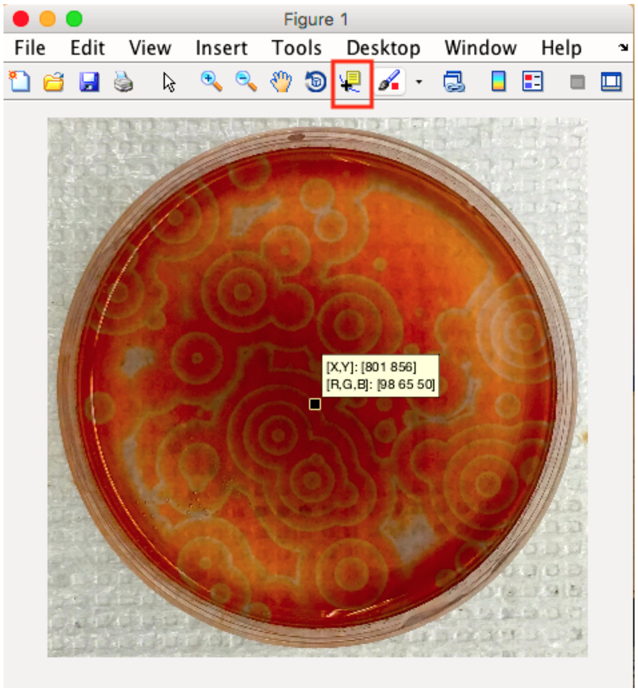
$$imshow(pic) \tag{8}$$

where *pic* is the name of the imported image variable. If there is no active *Figure* window, a new *Figure* window will pop out along with the imported image. If there is an active *Figure* window, *imshow* will replace whatever is in the current active *Figure* window with the imported image. If you want to create a new *Figure* window that has the imported image, type the command *figure;* followed by the *imshow* function in your *Command* window as follow

$$figure; \ imshow(pic); \tag{9}$$

3.3 Determine Coordinates

MATLAB allows you to determine the coordinate of each pixel. This is useful if you need to determine the distance between two pixel as this will allow you to determine the real distance between two objects provided you have the appropriate scaling system. To determine the coordinate of a pixel in a figure, click on the button that is box with solid red line in Figure (5).



[X,Y]: [801 856]
[R,G,B]: [98 65 50]

Figure 5: A *Figure* window with the imported image in the *Figure* window. The solid red box is the button for determining the coordinate of a pixel within the *Figure* window.

Final Note: For a more detail documentation on MATLAB command, you search for ‘matlab’ and the **listed keywords** using a search engine. Then, go to MATLAB website for a more thorough documentation on the commands and functions.