EAS 199B MATLAB Lesson 01
Introduction to MATLAB

Learning Objectives

- Be able run MATLAB in the MCECS computer labs
- Be able to perform simple interactive calculations
- Be able to open and view an m-file downloaded from the class web site
- Be able to execute simple recipes for plotting data from files
- Be able to describe the difference between plain text files, m-files, CSV files, XLSX, and DOCX files. Be able to use a programming editor in the MCECS labs. We need to recommend one!
- Be able to reuse a program (m-file) on new (potentially large) data sets.

Outline of this lesson

1. What is MATLAB?
2. Tutorial: Plot sin(x)
3. Saving Commands in an m-file
4. Re-using an m-file
5. A note on file types and plain text files
6. Practice
7. Homework

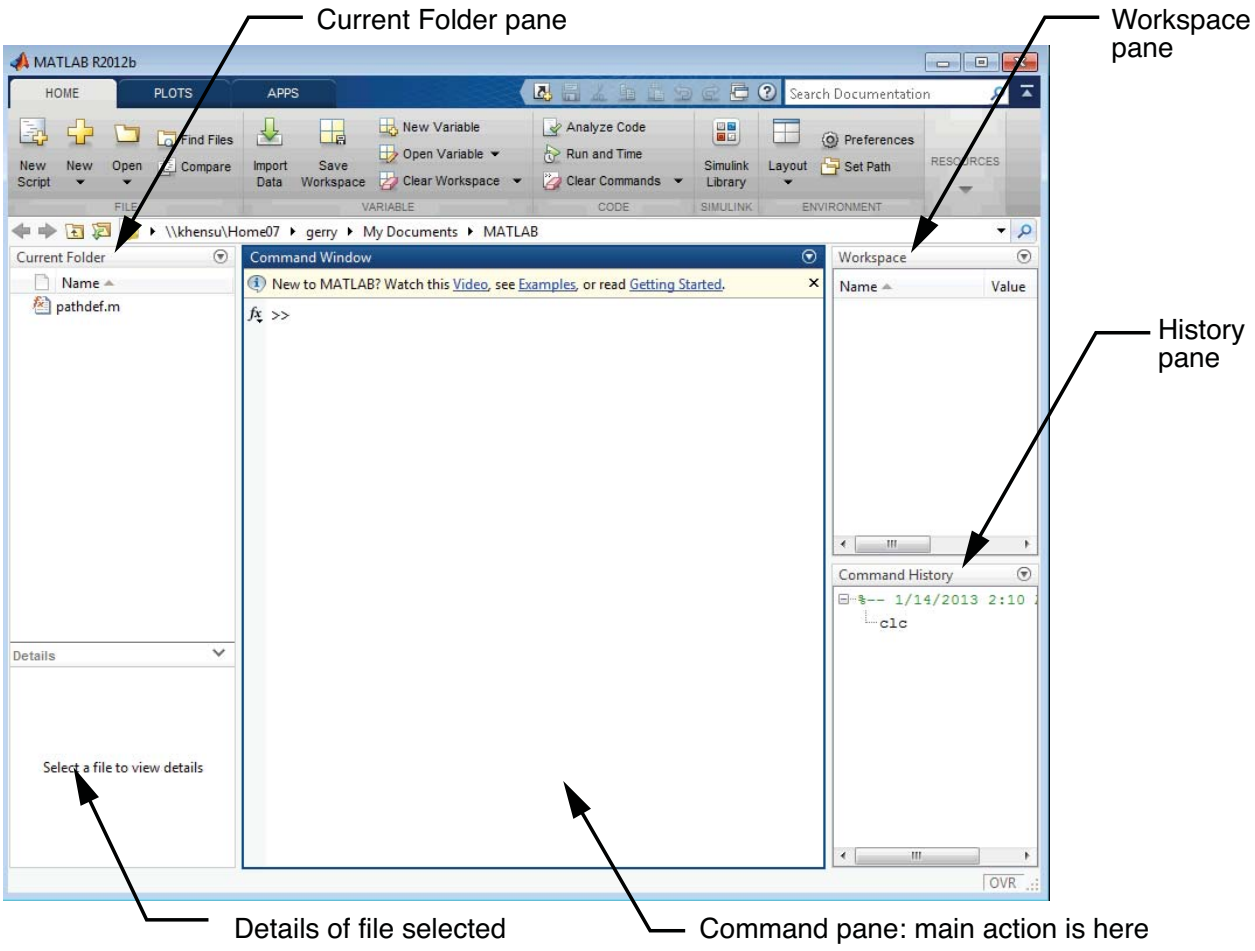
What is MATLAB?

- MATLAB is a commercial product. Open-source and commercial competitors exist.
- MATLAB was created as an interface to pre-existing libraries (LINPACK) for solving systems of equations (linear algebra).
- MATLAB is a powerful tool for control systems simulation, signal processing, analysis of experimental data and physical modeling. Its technical capabilities far exceed Excel. It makes more professional-looking plots (by default) than Excel. It is widely, but not universally used by engineers.
- MATLAB is controlled (mostly) by typing commands that execute mathematical expressions. Sequences of commands can be stored in script or function files that can be reused. Complex programs can be written and libraries can be developed using a

high level programming language that borrows ideas from Fortran and C. Arduino programmers should adapt quickly.

A Quick Tutorial

Launch MATLAB from the Windows Start Menu: *MATLAB R2012b 64-bit*. A window appears that should be similar to that shown below. The key area is the *Command Pane*. Much of the work of using MATLAB involves typing commands into the command pane. The *Workspace pane* (upper right) shows the names of variables in the current session. The History pane (lower right) shows all of the commands entered into the Command pane during the current session. The Current folder pane shows the names of any files in the folder (or directory) where MATLAB is currently working. Understanding the location and use of this directory is very important in working with MATLAB.



Make a simple plot

- Enter the following commands at the prompt (>>) in the Command pane. Do not type the prompt characters

```
>> x = linspace(0,2*pi);
>> y = sin(x);
>> plot(x,y)
```

- Move the plot window to the side where you can see both it and the command window. Enter the following command in the command window

```
>> plot(x,y,'o')
```

  In other words, repeat the plot command, but add a 'o' after the y

- Look for more information on the plot command

```
>> help plot
```

  or

```
>> doc plot
```

- Note the path in the "Current Folder" window. Navigate to a new folder (like "EAS199B" or "MATLAB"

Saving Commands in an m-file

- Open the editor and type in the following commands. Note that there are no prompt characters in the editor window. Don't add them! Just type the following lines exactly as shown here. Include a semicolon at the end of each line except for the first line.

```
function myplot

x = linspace(0,2*pi);
y = sin(x);
plot(x,y);
```

- Save the file as "myplot.m" – no spaces in the file name, and the file name must end with ".m"
- Quit MATLAB

Reusing an m-file

- Restart MATLAB
- Type "myplot" in the command window
- Change directories to the location where myplot.m is stored and re-enter the command, "myplot"
- Open myplot.m in the m-file editor and make the following changes
  (a) Add "(xmax)" to the end of the first line

(b) Replace “2*pi” with “xmax” in the line where x is created with linspace. (Do not include quotes.)

(c) Save the modified file.

The finished function should look like this:

```
function myplot(xmax)

x = linspace(0,xmax);
y = sin(x);
plot(x,y)
```

- Run the new function by entering the following command

```
>> myplot
```

This gives an error message

```
??? Input argument "xmax" is undefined.

Error in ==> myplot at 3
x = linspace(0,xmax);
```

The myplot function now expects you to supply the input argument. The correct usage is to supply a value inside parenthesis after the command name.

```
>> myplot(12)
```

The power of function is that the input argument changes what the function does. In this simple example, it only changes the upper limit of the x values used in the plot. Experiment by supplying different values of xmax

```
>> myplot(2*pi)
>> myplot(25*pi)
>> myplot(-3)
```

The last example is somewhat surprising because –3 is the minimum value. The linspace command created a list of numbers between 0 and –3.

A Note on File Types

MATLAB m-files and Arduino programs are stored as *plain text*, meaning that the file only contains the text (characters) that you see when you look at the program. MATLAB and the Arduino IDE do not use formatting instructions or any information other than the text of the program. Therefore, when working with MATLAB m-files or Arduino programs, it is important to keep the files in plain text format. Do not open and then save m-files with Word, or other word-processors. Refer to the separate document, *A Note on File Types*, for a more detailed discussion.

Practice

1. Change the formula being plotted
   - Replace sin(x) with cos(x), tan(x)
   - Replace sin(x) with sqrt(x), exp(x), exp(-x)

2. Use the .^ operator to raise all elements of x to a power
    - plot x^2, 2*x – 3*x^2
3. Use the ./ operator when dividing lists of numbers
    - plot x^2, 2*x – 3*x^2

Homework