**Colorado State University**
**Department of Mechanical Engineering**

# MECH 417 - Laboratory Exercise #1

## Introduction to MATLAB

# Contents

## *Purpose*

This laboratory exercise is intended to provide a tutorial introduction to Matlab, which will be used heavily throughout the remainder of the course. All of the problems in this Lab should be solved entirely in Matlab. Detailed documentation requirements are listed at the end of the Lab. You might want to refer to these before beginning.

## *General hints*

- The Live Editor or the section/cell (%%) and publish features of Matlab can be useful for generating complete documentation of your work and results; otherwise, you can just copy and paste code, results, and plots into a Word file (for later formatting and documentation). The *mypublish.m* file on the course website can be a useful example if you want to use the publish approach (which is recommended).

- To have multiple plots appear from a single M-file or script section, precede each *plot* command with a *figure* command (see *help figure* for more info). You can use a *pause* and *close all* at the end of your file to give yourself time to view the figures and to close all of the windows (after hitting Enter in the command window).

- **Do not use spaces in Script names! Also, always name a function file name the same as the function** (e.g., *myfunc.m* for *myfunc()*)**!**

## *Summary of Documentation requirements*

Section 1 – Show Matlab code and answers for i - v
Section 2 – Show code and answers for i – vi
Section 3 – Show code and plots with labels for i and ii
Section 4 – For i and ii:
- i) Present the script with comments, the commands entered into Matlab to run it, and three clearly-labeled plots with different values for the number of harmonics.
- ii) Show the script containing the function definition and the results of calling the functions for the three matrices a, b, and c.

Be sure to number and label everything in your final document so your presented work is easy to follow.

## *1) Vectors and Matrices*

Data entry in Matlab is achieved by separating columns of matrices by spaces, and rows by semi colons. For example, to enter the matrix:

$$a = \begin{bmatrix} 1 & 3 & 7 \\ 2 & 5 & 6 \end{bmatrix}$$

one would type:

```
a = [1 3 7; 2 5 6]
```

After defining matrices, one can easily perform computations with them in Matlab. Matlab provides a wealth of matrix functions, each of which has online help to describe its operation. Type the following commands to learn about some functions that will be helpful in the exercises below:

*help arith*

*help inv*

You can also browse for help by typing *doc* (or hitting *F1* or by clicking on the *Help* question mark in the *Resources* tab) and entering the MATLAB section. Explore the *Getting Started* module (in the top tab) and *MATLAB Functions* module (in the bottom tab) to familiarize yourself with what is available in Matlab. You should also get help on the following topics: *ops, elmat, elfun,* and *matfun*. To do this, type *help* (text-based), *helpwin* (text-based in separate window), or *doc* (rich hyperlinked in separate window) at the command-window prompt, followed by each of the topic names (one at a time). We will not be using all of the available tools, but we will use some frequently, and you will build your understanding of Matlab as the course progresses.

Enter the following matrices (note that the imaginary number j = $\sqrt{-1}$ is recognized in Matlab):

$$a = \begin{bmatrix} 1 & 3 & 7 \\ 2 & 5 & 6 \end{bmatrix}$$

$$b = \begin{bmatrix} 4 & 7 & 9 \\ 8 & 1 & 3 \\ 1 & 1 & 1 \end{bmatrix}$$

$$c = \begin{bmatrix} 1-j & 1+j & 3 \\ 2-4j & 8+3j & 6j \\ 0 & 0 & 7 \end{bmatrix}$$

and compute the following quantities:

i) bc
ii) $b^{-1}c$
iii) ab
iv) b+c
v) $ba^T$

**Hint**: $a^T$ denotes the transpose of a. The transpose operator is the apostrophe (‘)

## *2) Polynomials*

Matlab has many functions related to polynomial calculations. A polynomial is stored in Matlab as a coefficient vector. For example, the polynomial:

$$s^3 + 3s^2 - 7s + 8$$

is stored as the vector:

p = [1 3 -7 8]

Evaluating the polynomial and calculating the roots is then relatively straightforward. Type the following commands to see what functions are available and how they work:

*help polyval*

*help roots*

*help polyfun*

Then carry out the following exercises:

i) Evaluate $s^3 + 2s^2 + 4s - 8 \;\; at \;\; s = 1$

ii) Evaluate $s^3 + 2s^2 + 4s - 8 \;\; at \;\; s = 2 - 4j$

iii) Evaluate $s^3 + 4s \;\; at \;\; s = 2j$

iv) Compute the roots of $2s^3 - 3s^2 + 6s + 7$

v) Compute the roots of $s^3 - 12s^2$

vi) Compute all cube roots of 1

**Hint**: Think about how a cube root is defined ($x = 1^{1/3}$), and try to convert this definition into a polynomial for which you can find roots.

## *3) Plotting and Printing*

Matlab has an array of commands for plotting, labeling, and editing plots. Type the following command:

*doc plot*

Note that "See also" commands are suggested at the end of every help menu. Sometimes these are helpful.

As a simple example of how to plot, the following series of commands generates a plot of a cosine wave:

```
t = 0:0.01:2*pi;
y = cos(t);
plot(t,y);
title('Plot of a cosine wave');
```

Note that the first line is used to automatically generate a time vector (see also *linspace* and *logspace*). The semicolons keep the commands silent (i.e., prevents output display to the command window). Generate plots of the following functions. Please include titles and axis labels.

$i)\, \sin^2(t) - \cos(3t) \;\; for \; 0 \le t \le 2\pi$

$ii)\, t^3 - 3t^2 - 2t \qquad for \;\; 0 \le t \le 5$

**Hint**: Use the element-by-element dot (.) operator where appropriate.

## *4) Scripts and Functions*

Any collection of Matlab commands can be gathered together as a script. For instance, save the following commands in a new script called plotit.m

```
numhar = 5;
t = linspace(0,2*pi,300);
y = zeros(1,300);
for ii = 1:numhar
y = y + ((-1)^(ii+1))*(1/(2*ii-1))*cos((2*ii-1)*t);
end
plot(t,y)
title('Harmonic decomposition of a square wave')
```

**Note -** The for loop above is evaluating the following Fourier Series at all values in vector t:

$$y(t) = \sum_{i=1}^{n} \frac{(-1)^{i+1}}{2i-1} \cos[(2i-1)t]$$

You can then run the script to build a square wave for a given number of harmonics and plot it. Note that all variables defined in the script will exist in the workspace after it is run. You can also write functions, which use local variables that don't persist in the workspace. Type *doc function* to see the syntax for doing this. The *myfunc.m* example on the course website can also be useful. Carry out the following programming tasks:

i) Write a script to generate and plot a triangular wave for a given number of harmonics. Use a variable name for the number of harmonics so it can be altered as desired in the workspace before running the script. The Fourier Series for a triangular wave is:

$$y(t) = \frac{8}{\pi^2} \sum_{i=1}^{n} \left[ \frac{\sin\left(\frac{i\pi}{2}\right) \sin(it)}{i^2} \right]$$

Plot the wave for different numbers of harmonics to see how the shape improves with the number of terms.

ii) Write a function script that calculates the sum of the squares of the elements of a matrix:

$$f = \sum_{i=1}^{n} \sum_{j=1}^{m} [a_{ij}]^2$$

Only the matrix name should be passed to the function as an argument. The function should work for a matrix of any size. Test your routine on the matrices *a, b, c* from Section 1.

See the hint below.

**Hint**: a double sum can be created with nest *for* loops:

```
for i …
   for j …
     ….
    end
end
```

**Another Hint**: get help on the *size* command.

## *Additional Resources*

In addition to the above you find out more about Matlab by typing *demo*. You can also try the following Websites:

http://www.mathworks.com/
http://www.engin.umich.edu/group/ctm/basic/basic.html

The Mathworks Website is a general reference site, and the second is specifically designed as a tutorial introduction to Matlab.