Lab 0

MATLAB Structure and Use

This first laboratory session is an introduction to the MATLAB programming suite. Basic interface, operations, functions, and programming techniques are covered. Students will complete tasks that demonstrate the advantages of using MATLAB in engineering modeling, testing, and analysis.

0.1 Pre-Lab Assignment

Many of the key features—some would say *quirks*—of MATLAB are described in detail in Appendix B. Students should first consult this appendix to ensure that they are familiar with MATLAB to complete this experiment.

0.2 Warm-Up Assessment

Take about ten minutes to complete the following assignment. The first time you need help, ask you neighboring students for their recommendation. The second time you need help, call the instructor over for assistance.

Write a script that takes a large vector (at least 100 elements) of your creation and performs the following tasks:

1. Display the length of the vector and its minimum and maximum values.
2. Display the statistical mean and mode. If the mode is not uniquely defined, display a message that indicates this.
3. Find the sum of all elements that are less than 1.
4. Sort the elements of your vector from low to high and plot them against a time scale of the same length as your vector. Create appropriate labels on the plot.

5. Take the Fourier transform of your sorted vector (`fft` is fine). Create two new plots: one for the magnitude and another for the phase. Label all axes.

0.3 Laboratory Procedure

The procedure of this experiment consists of the student reproducing the examples in the tutorial (Appendix B) to gain familiarity and confidence with MATLAB. Later, the students are asked to demonstrate their understanding with some exercises.

0.3.1 Starting MATLAB Session

1. Log on and start MATLAB. Change the display format with `format compact`.
2. Clear the Command Window using `clc`
3. In the Command Window, write your name, the date and the lab session as comments.

0.3.2 MATLAB Statements

1. Reproduce the examples from Section B.1.1.
2. Enter some data and create some variables of your own using the concepts in Section B.1.1.
3. Print the Command Window and show it to the instructor.

The remainder of the experiment should be completed using m-files. If it is not already, the Editor may be opened with the command `edit`. Using the debugging mode of the Editor will help in writing your scripts. One can set break points, indicated by a red dot, to run the code from the beginning to a specific line. After finishing with break points, click the "Exit Debug Mode" menu button.

0.3.3 Numeric Format and Variables

1. Create and enter some complex numbers of your own using the concepts in Section B.1.5.
2. Reproduce the examples of matrices and vectors in the section on arrays in B.1.5.
3. Use the MATLAB command `roots` to find the roots of the polynomial $2x^2+4x+10$. Manually verify your answer by using the quadratic formula.

4. Reproduce the character string example from Section B.1.5.
5. Create the character string variable

```
c = 'All your base are belong to us'
```

   Now print the statement

```
All your songs belong to us
```

   by only selecting the needed letters and spaces from the variable `c`.

0.3.4 Arithmetic Operations

1. Reproduce the examples from Section B.2.1.
2. Find the matrix resulting from $A - BC^2 + 2D'$ where:

$$A = \begin{bmatrix} 1.5 & 3.3 \\ 6 & -4.5 \\ -2.5 & 0.7 \end{bmatrix}, \qquad B = \begin{bmatrix} 0.5 & 0.3 \\ -0.1 & 0.2 \\ 0.4 & -0.3 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}, \qquad D = \begin{bmatrix} 3.1 & 1.4 & -0.3 \\ -0.5 & 1.6 & 0.1 \end{bmatrix}$$

3. We define the element of matrix a in row i and column j as a_{ij}. Use array operations to find $b_{ij} - c_{ij}d_{ij}^4$ for all i and j for your choice of three 2x3 arrays b, c, and d. Manually verify the result for $i = 1$ and $j = 2$.

0.3.5 Function m-Files

The input-output characteristics of a script m-file make it very impractical to use as a subroutine within a program. Function m-files are useful for this purpose since they can be created with input and/or output variables. No variables remain in the Workspace after a function m-file has completed. As such, data must be passed back out or saved with a `save` statement within the function if it is to be used again.

1. Read (but not necessarily reproduce) Section B.1.3.
2. Create a function called `sumsin` that sums two sinusoids. The inputs should be `t`, `f1`, `f2`. The outputs should be $s_1 = \sin(2\pi f_1 t)$, $s_2 = \sin(2\pi f_2 t)$, and $s_3 = s_1 + s_2$.

To test your function you may wish to call it in the Command Window. To do this, define two frequencies and an appropriate time span and issue the command

```
[s1 s2 s3] = sumsin(t,f1,f2);
```

If there are no errors, you may assume that your function works correctly.

0.3.6 Logical Operators

1. Reproduce the logical operator examples in Section B.2.2.

2. In a script m-file, use logical and array operations to produce b from a where

$$a = \begin{bmatrix} 1.2 & -3.2 & 24 \\ 0.6 & -0.3 & -0.5 \\ -2.3 & 1.6 & 20 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} 1.2 & 0 & 0 \\ 0 & -0.6 & -1 \\ 0 & 1.6 & 0 \end{bmatrix}$$

0.3.7 Mathematical Operations

1. Read and reproduce some of the MATLAB math functions in Section B.2.3.

2. Try the three functions `round`, `floor`, and `ceil` on

$$x = \begin{bmatrix} -3.6 & -2.5 & -1.4 & -1 & 0 & 1.4 & 2.5 & 3.6 \end{bmatrix}$$

   to observe their characteristics.

3. Find the value of x when

$$x = \frac{\ln\left[2 + \sin^2(3)\right] + e^{-0.2}}{\sqrt{2^{1.6} + 3^{-0.5}}}$$

4. If t varies from -1.2 to 1.2 in steps of 0.4, find:

   (a) $w(t) = 3t^3 + 2t^2 - t + \sin t$

   (b) $x(t) = \begin{cases} 0, & t < 0 \\ 2, & t \geq 0 \end{cases}$

0.3.8 Flow Control Statements

1. Reproduce the examples in Section B.3.

2. Use `for` statements to find the values of $x(t) = 3\cos(2\pi ft + 0.1)$ for $t = 0, 0.1, 0.2$, and 0.4 seconds when $f = 10, 15$, and 20 Hz. Use one set of statements to compute the values for all three frequencies and store the results in a two-dimensional array. (Hint: use nested loops.) What is the value when $x(0.3)$ when $f = 15$ Hz?

3. Use `while` statements to find the largest value of positive t for which $e^{1.2}\cos(\omega t)$ and t^3 are both less than 10. Make the computation for $\omega \in \{35, 40, 45\}$ and find the solution to the nearest 0.01.

4. Evaluate $x(t) = e^{-|t|}$ for $-1 \leq t \leq 1$ in steps of 0.2 using `for` and `if` statements. Your logical test will likely be something like `if t < 0` then do "this," `else` do "that". Repeat the computation using logical operations and 0-1 (logical) arrays. Here you will want to produce a vector with ones in the same places that the time vector is negative and zeros where the time vector is positive. Now reduce the step size to 0.0002. Do not print the values of $x(t)$ to the Command Window as there are 10,001 of them. Be sure to initialize the array first. Do you notice a difference in computation times between the two methods? How does the step size affect the computation time?

0.3.9 Numerical and Data Analysis

1. Use some of the commands in Table B.5 (Section B.2.4) on the array

$$a = \begin{bmatrix} 1 & 0 & 2 & 3 & 0 & 4 \\ 4 & 0 & 3 & 2 & 0 & 1 \\ 1 & 2 & 3 & 4 & 0 & 0 \end{bmatrix}$$

   Be sure to practice the functions on both dimensions.

2. Create an 11-element vector for the values of $x(t) = 4\cos(2\pi t + 0.2) + 3\sin(\pi^2 t)$ at equally spaced times in the interval $0 \leq t \leq 1$. *Without printing the vector*, find:

   (a) The maximum element value

   (b) The minimum element value

   (c) The average value of the elements

   (d) The indices for which the element magnitude is greater than 4

   Check your results manually by printing the vector.

3. Given the array:

$$A = \begin{bmatrix} 1 & 4 & 3 & 2 \\ 4 & 1 & 2 & 5 \\ 3 & 3 & 5 & 1 \end{bmatrix}$$

   use MATLAB statements to find:

   (a) The number of rows and columns in A

   (b) The maximum and minimum element values in A

   (c) The maximum and minimum element values in each row of A

   (d) The average value in each column

   (e) The average of all element values

0.3.10 Plotting Functions

1. Create a script that uses the sum of sinusoids function created in Section 0.3.5 and plot the output signals. Suggested values are $t \in [0, 10]$ and 0.2 and 0.425 Hz. Plot all three signals on the same axis. Create appropriate axis labels, title, and legend.

2. In a new figure window, plot all three sinusoids on separate axes but in the same window. Create appropriate labels.

0.3.11 Signal Processing

1. Peruse Section B.5 and study the example using the fast Fourier transform.

2. Using your `sumsin` function, add a 1 Hz sinusoid to a 1000 Hz sinusoid over a one second interval. Use any appropriate discretization (or sampling rate) you see fit. Now pass the summed signal through an AWGN (zero-mean, unit variance) channel. Use `fft` to analyze the power spectra of the clean and noisy signals. Plot the spectra with appropriate labels.

0.4 Take-Home Assignment

Write a function m-file that accomplishes the tasks outlined below. Save your function as your last name followed by your first initial and `Lab0`. For example, if your name is Kane Warrior, then your function should be called `WarriorKLab0.m`.

The function should take as its input a vector of arbitrary length and a time step value. If this vector represents discrete values of a function $x(t)$, the function should return the derivative and integral as functions of time: $\dot{x}(t)$ and $\int_0^t x(s)ds$. Also, in the same figure, the function should output graphs of the function, its derivative, and its integral on separate axes. These plots should be labeled nicely.

Kane Warrior's first line in his script would look like this:

```
function    [xdot integ] = WarriorKLab0(x,deltat)
```

Of course Kane would be encouraged to write a short introduction or help section for his function.