CMPSC 497 LAB #1 (Individual)

Instructor: R. Avanzato

Lab #1 MATLAB (v1.5) Statistics

Katherine Banis

**Objective:** Explore use of MATLAB to solve engineering problems.

**Materials:**

* MATLAB software (installed in computer lab)
* MATLAB tutorial (Canvas document or equivalent)

**Part #1:** **Data Analysis**

Give a list of 15 voltage measurements [21.2, 19.5, 20.1, 18.3, 17.7, 15.0, 21.9, 24.7, 23.1, 20.2, 16.3, 22.8, 18.4, 23.5, 21.1], create a MA[LAB script (m-file), called analysis.m, to calculate and display the **1) min, 2) max, 3) average, 4) standard deviation, 6) median, 7) number of values above average (use find() function), 8) display each value above average, 9) plot of raw data, 10) distribution plot (histogram), 11) display all data in sorted order (lower to higher value; one value on each line).** Assume all measurements are in volts. Include axis labels and a title for each of the 2 plots. Include a single script (with header and comments), all output results and the two plots with proper labels.

Include a fprintf() statement for each output value. Example :

**fprintf**('The average voltage is **%.2f** volts \n', avg) % Note: %.2f is a placeholder   
 % Note: \n is line break

No user input prompt is required for this step. That is, you can hardcode the input data into Matlab. For example: voltages = [21.2, 19.5, 20.1, …]) Use MATLAB tutorial on Canvas for background information.

* See part one code and results on next page

**Code for part 1:**

% analysis.m

voltages = [21.2, 19.5, 20.1, 18.3, 17.7, 15.0, 21.9, 24.7, 23.1, 20.2, 16.3, 22.8, 18.4, 23.5, 21.1];

% Basic statistics

minimum = min(voltages);

maximum = max(voltages);

average = mean(voltages);

standard\_deviation = std(voltages);

med = median(voltages);

% Values above average

index = find(voltages > average);

above\_average = voltages(index);

% Plot data

figure;

plot(voltages, 'o-');

ylabel('Voltages (Volts)');

title('Raw Voltage Data');

% distribution plot (histogram)

figure;

histogram(voltages, 'BinWidth', 1);

xlabel('Voltage (Volts)');

ylabel('Frequency');

title('Histogram of Voltage Distribution');

% Sort voltages

sorted = sort(voltages);

% Print results

fprintf('Minimum voltage is %.2f volts \n', minimum);

fprintf('Maximum voltage is %.2f volts \n', maximum);

fprintf('Average voltage is %.2f volts \n', average);

fprintf('Standard deviation is %.2f volts \n', standard\_deviation);

fprintf('The median voltage is %.2f volts \n', med);

fprintf('Index of values above average:\n');

disp(index);

fprintf('Values above average:\n');

disp(above\_average);

fprintf('Sorted voltages:\n');

disp(sorted);

**Results for part 1:**

Minimum voltage is 15.00 volts

Maximum voltage is 24.70 volts

Average voltage is 20.25 volts

Standard deviation is 2.76 volts

The median voltage is 20.20 volts

Index of values above average:

1 7 8 9 12 14 15

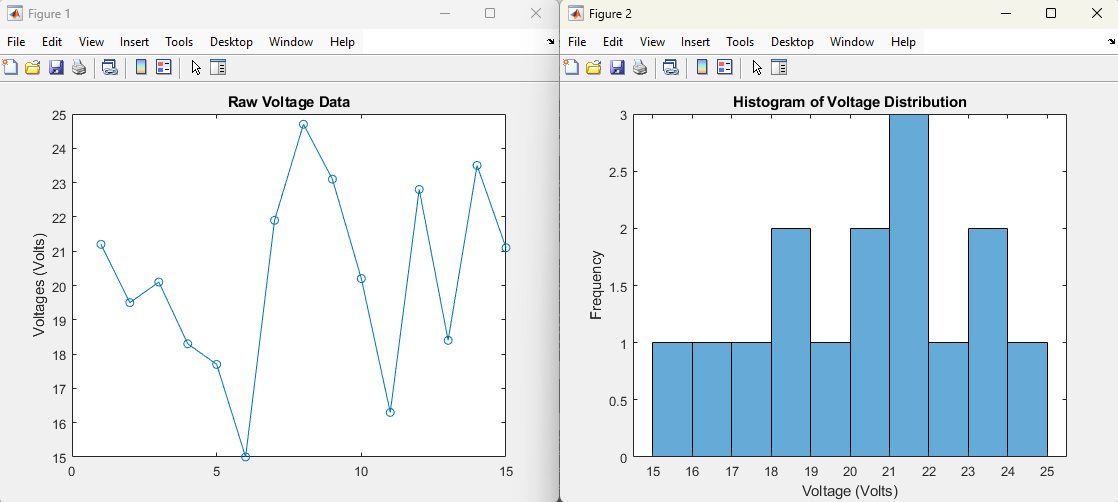
Values above average:

21.2000 21.9000 24.7000 23.1000 22.8000 23.5000 21.1000

Sorted voltages:

15.0000 16.3000 17.7000 18.3000 18.4000 19.5000 20.1000 20.2000 21.1000 21.2000

21.9000 22.8000 23.1000 23.5000 24.7000



**Part#2**: **Resistor Calculations**

Create and test two, separate MATLAB scripts (m-files) to calculate the total resistance of n resistors in 1) series, and 2) parallel. Each MATLAB script should prompt the user to enter the number of individual resistors, n, and then prompt the user for each resistor value. One script should be called series.m and the other script should be called parallel.m. Then the program should display the total resistance in a user-friendly, formatted style. Include the scripts (with header and comments) and sample user input and output in the formal report. **Include 3 test cases for each script.**

**NOTE:** 1) for series resistors, Rtotal = R1 + R2 + R3 + …

2) for parallel resistors, Rtotal = 1/(1/R1 + 1/R2 + 1/R3 + …)

* See part two code and results on next page

**Code for part 2 – series:**

% series.m

% calculates total resistance of n resistors connected in series

% ask user for number of resistors in series

n = input('Enter number of resistors in series: ');

total\_resistance = 0;

for i = 1 : n

r = input('Enter resistance (ohms): ');

total\_resistance = total\_resistance + r;

end

% display total resistance

fprintf('Total resistance for series is %.2f ohms\n', total\_resistance');

**Test Cases for part 2 – series:**

Enter number of resistors in series: 3

Enter resistance (ohms): 100

Enter resistance (ohms): 150

Enter resistance (ohms): 250

Total resistance for series is 500.00 ohms

Enter number of resistors in series: 2

Enter resistance (ohms): 50

Enter resistance (ohms): 50

Total resistance for series is 100.00 ohms

Enter number of resistors in series: 5

Enter resistance (ohms): 10

Enter resistance (ohms): 20

Enter resistance (ohms): 30

Enter resistance (ohms): 40

Enter resistance (ohms): 50

Total resistance for series is 150.00 ohms

**Code for part 2 – parallel:**

% parallel.m

% calculates total resistance of n resistors connected in parallel

% asks user for number of resistors parallel

n = input('Enter number of resistors in parallel: ');

sum\_inverse = 0; % initialize sum of inverse of resistances

% accumulate inverse of each resistor's value

for i = 1 : n

r = input('Enter resistance (ohms): ');

sum\_inverse = sum\_inverse + 1/r;

end

% calculate total resistance

total\_resistance = 1 / sum\_inverse;

% display total resistance

fprintf('Total resistance for parallel is %.2f ohms\n', total\_resistance');

**Test Cases for part 2 – parallel:**

Enter number of resistors in parallel: 3

Enter resistance (ohms): 100

Enter resistance (ohms): 200

Enter resistance (ohms): 300

Total resistance for parallel is 54.55 ohms

Enter number of resistors in parallel: 2

Enter resistance (ohms): 500

Enter resistance (ohms): 500

Total resistance for parallel is 250.00 ohms

Enter number of resistors in parallel: 4

Enter resistance (ohms): 100

Enter resistance (ohms): 100

Enter resistance (ohms): 100

Enter resistance (ohms): 100

Total resistance for parallel is 25.00 ohms

**Questions:**

1. What is MATLAB? (3 to 5 sentences)
   1. MATLAB is short for Matrix Library. It is a high-level programming and numeric computing platform, often used by engineers and scientists. It has many libraries for solving many different mathematical problems. MATLAB is also used for data visualization like creating plots.
2. Is MATLAB compiled or interpreted? Explain.
   1. MATLAB is an interested language, meaning it is read and executed line-by-line by the interpreter rather than being compiled into machine language beforehand.
3. Is MATLAB case sensitive? Give an example.
   1. Yes, MATLAB is case sensitive. For example, if you set ‘sum = 5’, then type ‘Sum’, MATLAB will not recognize ‘Sum’.
4. What are the advantages and disadvantages of using MATLAB instead of languages such as C++/C#/Java/Python?
   1. MATLAB is strongly typed so it can be easier to catch type-related errors before the program is executed. It is also designed to be a high-level programming language with built-in support for matrix and array operations. However, MATLAB is proprietary and not open source. So while it has many standard libraries, Python, for example, has much more open source libraries for almost any programming need.
5. Is MATLAB a “dynamically typed” language? Explain. Give one example.
   1. Yes, MATLAB is a “dynamically typed" language, meaning you don’t need to declare the type of variable before using it. For example, if I set ‘name = John’, MATLAB will automatically assume that ‘name’ is a string type variable.
6. Explain the colon “:” operator in MATLAB. Give an example.
   1. The colon ‘:’ operator in MATLAB is used to create row vectors, specify ranges, and select portions of data. For example, a colon can specify ranges. So typing   
      ‘1 : 5’, will create the vector ‘[1 2 3 4 5]’.
7. What is Simulink? (Research this on Mathworks website)
   1. Simulink is a block diagram environment for Model-Based design. It allows the user to design systems with multidomain models, simulate before moving to hardware, and deploy without writing code.

This lab should be completed individually.

Appendix: MATLAB Video Tutorials.

* <http://www.mathworks.com/videos/getting-started-with-matlab-68985.html>   
  (Getting Started with MATLAB)
* <http://www.mathworks.com/videos/writing-a-matlab-program-69023.html>   
  (Writing a MATLAB Program)
* <http://www.mathworks.com/videos/using-basic-plotting-functions-69018.html>   
  (Basic Plotting with MATLAB)
* <http://www.mathworks.com/videos/working-with-arrays-in-matlab-69022.html>   
  (Working with Arrays in MATLAB)
* <http://www.mathworks.com/academia/student-competitions/auvsi/>  
  (autonomous robot boat competition using MATLAB)