ENG1060: COMPUTING FOR ENGINEERS

Lab 4 – Week 5

2020 OCT NOV

Welcome to lab 4. Remember that laboratories continuously build on previously learned concepts and lab tasks. Therefore, it is crucial that you complete all previous labs before attempting the current one.

Self-study:
Students are expected to attempt these questions during their own self-study time, prior to this lab session. There may be questions that require functions not covered in the workshops. Remember to use MATLAB's built-in help for documentation and examples.

Learning outcomes:

1. To revise user-defined functions and apply good programming practices
2. To differentiate between double and logical data types
3. To identify scenarios where logicals are appropriate and apply them
4. To integrate IF statements to solve condition-based problems
5. To perform input and output processes to analyse data and present results

Background:
Engineers use a variety of devices and instruments to collect data. Such large data sets are imported into software, such as MATLAB, for post-processing. The calculations may depend on certain conditions, which require branching structures such as IF statements. Additionally, the post-processed results may need to be exported to be processed by another software or for presentation purposes.

Primary workshops involved:

- Workshop 3: Functions, commenting, debugging and strings
- Workshop 4: Input, output and IF statements

Assessment:
This laboratory comprises **2.5%** of your final grade. The questions are designed to test your recollection of the workshop material and to build upon important programming skills. You will be assessed on the quality of your programming style as well as the results produced by your programs during your laboratory session by the demonstrators. Save your work in **m-files** named ***lab1t1.m***, ***lab2t2.m,*** etc. Inability to answer the demonstrator's questions will result in zero marks, at the demonstrator's discretion.

Team tasks begin at the start of the lab session so please ensure you arrive on time to form your groups. Students who arrive late will not be able to participate in the team tasks as teams will have already formed and will therefore forfeit all associated marks. These tasks will be assessed during class.

Lab submission instructions

Follow the instructions below while submitting your lab tasks.

Team tasks:
The team tasks are designed for students to test and demonstrate their understanding of the fundamental concepts specific to that lab. These tasks will occur at the start of the lab and will be assessed on the spot. Demonstrators will advise on how these will be conducted. Most team tasks do not require the use of MATLAB but MATLAB should be used for checking purposes.

Individual tasks:
The individual tasks are designed for students to apply the fundamentals covered in the team tasks in a variety of contexts. These tasks should be completed in separate m-files. There is typically one m-file per task unless the task requires an accompanying function file (lab 3 onwards). Label the files appropriately. E.g. lab6t1.m, lab6t2.m, eridium.m, etc.

Deadline:
The lab tasks are due next Friday at 9am (MYT) or 12pm (AEDT). Late submissions will not be accepted. Students will need to apply for special consideration after this time.

Submission:
Submit your lab tasks by:

1) Answering questions in Google Form, and
2) Submitting one .zip file which includes all individual tasks.

The lab .zip file submission links can be found on Moodle under the weekly sections, namely Post-class: Lab participation & submission. The submission box ("Laboratory 4") will only accept one .zip file. Zipping instructions are dependent on the OS you are using.

Your zip file should include the separate m-files for the individual tasks including function files.

It is good practice to download your own submission and check that the files you have uploaded are correct. Test run your m-files that you download. You are able to update your submission until the deadline. Any update to the submission after the deadline will be considered late.

Grade and feedback:
The team will endeavour to grade your lab files by Tuesday of the following week. Grades and feedback can be viewed through the Moodle Gradebook, which is available on the left side pane on the ENG1060 Moodle site.

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Lab 4 – Assessed questions

TASK 1

[2 MARKS – L04TE]

Note: Team tasks are designed for students to recall material that they should be familiar with through the workshops and practice of the individual questions prior to this lab session.

Students will be split into groups of 3-4 for the team tasks. Students in each group must explain aspects of the question below to receive the marks. Ensure that everyone has equal learning opportunities. Additionally, ask your table for help.

Importing and exporting:

Each group will be assigned either Set A or Set B.

Set A

Filename: A.txt

CID	Graduates	Employed
2419	2339	1976
2416	756	640
2415	856	648

Set B

`matrix = [26518 100 2; 32185 58 1; 35689 42 1; 48613 172 3];`

Employee ID	Salary (thousand, $USD)	Status
26518	100	2
32185	58	1
35689	42	1
48613	172	3

Complete the following:

1. Copy-paste the image of the table
2. [For Set A] Provide the syntax to extract the 2nd row of numerical data (as highlighted) using
   a. fopen, fgetl, fclose
   b. importdata

   [For Set B] The following code creates the data set, opens a file to export fprintf commands and closes a file. Provide the output of highlighted fprintf lines

```
matrix = [26518 100 2; 32185 58 1; 35689 42 1; 48613 172 3];
fid = fopen('export_file.txt','wt');
fprintf(fid,'%8s %8s %8s\n','ID', 'Salary', 'Status')
fprintf(fid,'%8.0f %8.0f %8.0f\n', matrix)
fclose(fid);
```

3. Discuss the task and explore any misunderstandings. Also, browse the work of other teams related to the other set and ensure that you have understood it as concepts from all sets may be required for the individual tasks.
4. Have a demonstrator assess your understanding

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Logicals:

Consider the Titanic data stored in a structure variable named *T*, such that the numerical field of the structure is accessed through T.data.

T =

Passenger ID	Survived 1=Yes, 2=No	Gender 1=Male, 2=Female	Age	Fare ($)
121	2	1	22	7.25
243	1	2	38	71.28
432	1	2	26	7.93
564	1	2	35	53.1
856	2	1	35	8.05

Complete the following – you may want to have a couple of students checking the syntax and output on MATLAB.

1. Copy-paste an image of the table above.
2. Annotate and provide the syntax to exact the:
   a. 1st column and assign it to the variable named *id*
   b. 3rd column and assign it to the variable named *gender*
   c. 4th column and assign it to a variable named *age*
3. Provide the output of:
   a. index = age>30
   b. old_age = age(index)
   c. old_age_id = id(index)
4. Provide syntax that will return a vector containing all female passenger IDs
5. Create a logical named age38 that contains a 1 (true) where the age of the passenger is 38 and a 0 (false) where the passenger age doesn't equal to 38.
6. Provide syntax to extract the 2nd and 3rd columns of information for the passenger aged 38, using the variable created in step 5
7. Discuss the task and explore any misunderstandings.
8. Have a demonstrator assess your understanding.

Remember good programming practices for all tasks even if not specifically stated. This includes, but is not limited to:

- using clc, close all, and clear all, where appropriate
- suppressing outputs where appropriate
- labelling all plots, and providing a legend where appropriate
- fprintf statements containing relevant answers

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 2

[5 MARKS]

Download the **lab4_plot_data3.txt** file from GDrive. The file contains the x (first row), y (second row), z (third row), and t (fourth row) coordinates.

Write an m-file that performs the following:

1. Read the data points from **lab4_plot_data3.txt** using fgetl(). Remember that it will read in the data as strings.
2. Create a 3-by-1 subplot.
   - [top subplot] Plot y against x using red circle markers. Also, plot $\sin(x)$ against x as a black line.
   - [middle subplot] Plot z against x using blue diamond markers. Also, plot $\cos(x)$ against x as a black line.
   - [bottom subplot] Plot t against x using green asterisk markers. Also, plot $tan(x)$ against x as a black line.

Remember to label your plots appropriately and include a legend in the southeast corner.

TASK 3

[4 MARKS]

Download the **temperatures.txt** file from Moodle/Google Drive. The file contains temperature data which has been collected **once a day** and contains noisy data.

Write a MATLAB m-file that performs the following:

A. Read the temperature values from **temperatures.txt** into a vector
B. Plot the temperature against time using red diamond markers.
C. Without using loops, remove the noisy data using the following rules:
   I. Any temperature value **above 45^{o}C** is invalid
   II. Any temperature value **below 0^{o}C** is invalid
   III. The temperature values from **days** 5, 13, 34, 40 and 42 are invalid
   Do not set the invalid temperatures to **0** (double) or **NaN**! Rather, set their logical entries to be false (0 logical).
D. After removing the invalid temperature values, plot the **valid** temperature values against the **corresponding valid times** in a new figure as blue circles.

Use either the **fgetl()** or the **importdata()** functions to import your data. You may need to use the **str2num()** command.

HINT: If temperatures are stored in T, try **valid_indices = T > 45**
HINT: What is the result of **T(valid_indices)?**

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 4

[5 MARKS]

A body of mass m falling under the action of gravity experiences drag, and so it will eventually approach a maximum speed. This maximum speed is called the terminal velocity v_T, and can be calculated as

$$v_T = \sqrt{\frac{mg}{k}}$$

where $g = 9.81$ m/s^2 is the acceleration due to gravity, and k is a parameter for the drag force ($k = 0.5c_d\rho A$, but you won't need this). The distance y_T where a fraction n of the terminal velocity occurs is given by

$$y_T = \frac{m}{2k}\ln\left(\frac{mg}{mg - k(nv_T^2)}\right)$$

where n is a fractional factor of the terminal velocity ranging from 0-1 (exclusive).

a) Write a **function** file that accepts the mass m, the drag parameter k and fractional factor n as inputs, and provides the terminal velocity v_T and the distance y_T as outputs.

b) A group of 7 people, each with equally-spaced mass values between 50kg and 100kg and a k value of 0.12kg/m jumps off a plane in a skydiving activity. Determine v_T and y_T of each person for $n = 0.9$.

c) In a 2-by-1 subplot arrangement:
   - [top subplot] plot v_T against m as a black line
   - [bottom subplot] plot y_T (for $n = 0.9$) against m as blue circles

d) Print to a text file named "skydivers.txt" using the fopen() and fprintf() functions. The file should include appropriate headers and should print the mass (kg), v_T (m/s) and y_T (m) (for $n = 0.9$). An example output is provided below.

```
mass (kg)   vt (m/s)     yt (m)
    50.00      63.93     479.71
     xx.xx      xx.xx     xx.xx
```

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 5

[4 MARKS]

Download the **ENG1060studentmarks2.txt** file from the GDrive. The file contains the following information:

- Column 1: Student ID
- Column 2—11: Lab marks (/20) worth 2.5% each (25% total of final grade)
- Column 12: Assignment mark (/10) worth 10% of the final grade
- Column 13: Other in-semester marks (/15) worth 15% of the final grade
- Column 14: Exam mark (/100) worth 50% of the final grade

A. Write a **function** that accepts a student's laboratory (as a vector), assignment, other and exam marks as inputs to determine the **final mark** to the nearest integer and the **letter grade** of the student. The inputs to this function should be for just ONE student. Your function header would be similar to:

```
function [final_mark, grade] = markscalc(lab, assignment, other, exam)
```

Note: The letter grades should be stored as a `string' and are described as follows:

final_mark ≥ 80	→ **HD**
70 ≤ final_mark < 80	→ **D**
60 ≤ final_mark < 70	→ **C**
50 ≤ final_mark < 60	→ **P**
final_mark < 50	→ **N**

B. Write an m-file that achieves the following:
   1. Prompts the user to input a student's ID number and extracts the individual assessment marks from the ENG1060studentmarks2.txt file.
   2. Use the function written in part A to calculate the student's final mark and letter grade.

Hint: You may want to use logical statements or the find() function to determine the row of the ID number which was input by the user.

```
Student with ID number 21245686 obtained a final mark of 81 corresponding to a HD grade
```

2 marks deducted for poor programming practices (missing comments, unnecessary outputs, no axis labels, inefficient coding, etc.)

END OF ASSESSED QUESTIONS

The remainder of this document contains supplementary and exam-type questions for extended learning. Use your allocated lab time wisely!

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Lab 4 – Supplementary questions

These questions are provided for your additional learning and are not assessed in any way. You may find some of these questions challenging and may need to seek and examine functions that are not taught in this unit. Remember to use the help documentation. Coded solutions will not be provided on Moodle. Ask your demonstrators or use the discussion board to discuss any issues you are encountering.

TASK 1S

The equivalent resistance R_{eq} of n number of resistors connected in **series** is given as

$$R_{eq,series} = R_1 + R_2 + R_3 + \dots R_n$$

The equivalent resistance R_{eq} of n number of resistors connected in **parallel** is given as

$$\frac{1}{R_{eq,parallel}} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_4}$$

Write an m-file that prompts the user for

A. the type of connection (series or parallel)
B. the vector containing all of the resistances

Then calculate the equivalent resistance for the configuration that the user has chosen. Finally, use an fprintf statement to provide a sentence containing the chosen configuration and the equivalent resistance value. Your output should look like the following:

```
Choose series (1) or parallel (2): 1
Input 5 resistors: [100 200 300 400 500]
Configuration is in series and equivalent resistance is 1500.000000
```

SOLUTION

```
What type of connection? Enter (1) for series OR (2) for parallel: 1
Enter resistance values AS AN ARRAY: linspace(100,1000,5)
For series connected resistance values [100   325   550   775  1000]
R_eq = 2750.0000

What type of connection? Enter (1) for series OR (2) for parallel: 2
Enter resistance values AS AN ARRAY: linspace(100,1000,5)
For parallel connected resistance values [100   325   550   775  1000]
R_eq = 58.1888
```

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 2S

Prompt the user to input a vector of integers of arbitrary length. Then determine the number of elements, the number of positive values and the number of negative values divisible by 3. Use fprintf to output the information in a message similar to the following: “The vector has XX elements. XX elements are positive values and XX elements are negative values divisible by 3”, where XX corresponds to the appropriate values.

SOLUTION

```
Enter a vector of INTEGERS any length: -10000:4:4000
The vector has 3501 elements. 1000 elements are positive values and 833 elements
are negative values divisible by 3.
```

TASK 3S

The following is a list of exam scores: 31, 70, 92, 5, 47, 88, 81, 73, 51, 76, 80, 90, 55, 23, 43, 98, 36, 87, 22, 61, 19, 69, 26, 82, 89, 99, 71, 59, 49, 64, 77, 98, 10, 15, 49, 30, 55, 71

Copy the values into a text file and save it as “exam_scores.txt”. Load this file into MATLAB using the fopen() and fgetl() functions. Then determine how many grades are between 0 and 19, between 20 and 39, between 40 and 59, between 60 and 79 and between 80 and 100. The results should be printed out using the fprintf function in messages similar to the following:

“Grades between 0 and 19: 2 students”
“Grades between 20 and 39: 4 students”
Etc.

SOLUTION

```
Grades between 0 and 19: 4 students
Grades between 20 and 39: 6 students
Grades between 40 and 59: 8 students
Grades between 60 and 79: 9 students
Grades between 80 and 100: 11 students
```

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

TASK 4S

The Body Mass Index (BMI) is a measure of obesity. In standard units, it is calculated by the formula

$$BMI = \frac{W}{H^2}$$

where W is the mass in kg and H is the height in meters. The classifications are given by:

Category	Classification
BMI ≤ 18.5	Underweight
18.5 < BMI ≤ 25	Normal
25 < BMI ≤ 30	Overweight
30 < BMI	Obese

Write an m-file to calculate the BMI of a person. You should prompt the user to enter his or her weight (kg) and height (meters). The program should display the results in a sentence that reads: "Your BMI value is XX which classifies you as XX" where XX represent the appropriate values/variables.

SOLUTION

```
Enter your height in m: 1.9
Enter your weight in kg: 100
Your BMI value is 27.70 which classifies you as a little overweight.

Enter your height in m: 1.70
Enter your weight in kg: 40
Your BMI value is 13.84 which classifies you as a little underweight.
```

TASK 5S

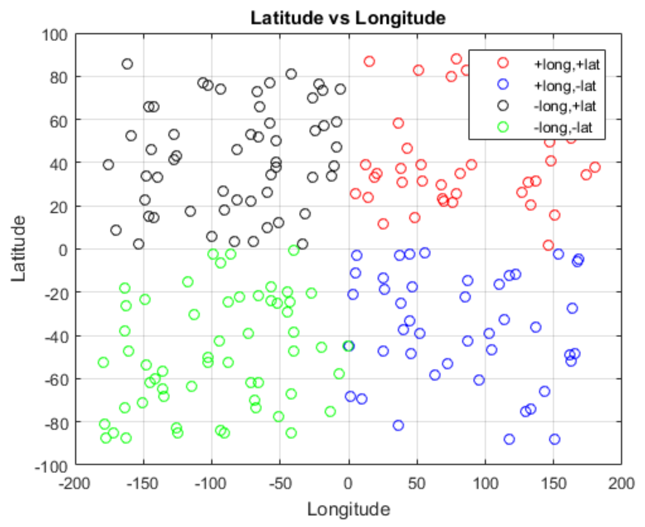
Download the file named "latlong.txt" which contains latitude and longitude data. Load the data and plot the following:

- positive latitude and positive longitude as red circles
- negative latitude and positive longitude as blue circles
- positive latitude and negative longitude as black circles
- negative latitude and negative longitude as green circles

Remember to label your plot and include a legend.

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

SOLUTION



11 **Important:** If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

Lab 4 – Exam-type questions

These questions are provided for your additional learning and are not assessed in any way. You may find these type of questions on ENG1060 exams. Solutions will not be provided on Moodle. Ask your demonstrators or use the discussion board to discuss any issues you are encountering. Additionally, you may use the exam collaboration document on Moodle (under the exam section) to share your answers.

Note: If a MATLAB statement returns an error, write down "error".

1. Consider the following matrices:

$$A = [90 \quad 70 \quad 50 \quad 30 \quad 10\,] \qquad B = \begin{bmatrix} 8 \\ -7 \\ -11 \\ 21 \\ 33 \end{bmatrix} \qquad C = [1 \quad 2 \quad 3 \quad 4 \quad 5]$$

$$D = \begin{bmatrix} 4 & 9 & 1 & 6 & 15 \\ 11 & 3 & 2 & 8 & 6 \end{bmatrix}$$

a) Provide the output of **X = ~(B < 0)**

b) Provide the output of **Y = (A >= 50) | (C <= 3)**

c) Provide the output of **Z = logical(A) + D(1, :)**

d) Provide the output of **AA = D([1 1 1 0 0; 0 0 1 1 1])**

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

2. The temperature and weight of a duck were measured over several days. **The data below is contained in a file named "duck.txt".**

Day	0	1	2	3	4	5	6
Weight (kg)	0.71	0.72	0.71	0.68	0.69	0.7	0.71
Temperature (°C)	39.4	40.1	40.4	41.2	40.8	40.6	40.2

a) Provide the syntax to import the text and numerical data into a **structure named X**.

b) Provide the syntax to extract the **1st, 2nd and 3rd rows** to variables named **D, W and T**, respectively.

c) Provide syntax to create a logical named **T40**, using logical and relational operators for **temperatures greater than 40°C**.

d) Provide syntax to create a logical named **W7**, using logical and relational operators for **weights greater than 0.7 kg.**

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

e) Provide syntax to output the days where the duck had a **temperature greater than 40°C and a weight great than 0.7 kg using the logical created in parts (c) and (d).**

f) Assume the **maximum weight of the duck is stored in maxW**, and the **day it occurred as on maxD**. Use fprintf to print a statement regarding the duck's maximum weight and the day it occurred using the maxW and maxD variables. Print to 4 decimal places for both variables.

3. Consider the following matrices:

$$A = [30 \quad 80 \quad 10 \quad 30 \quad 40\,] \qquad B = [40 \quad 80 \quad 70 \quad 50 \quad 60\,]$$

a) Describe the difference between **A = B** and **A == B**, and provide the outputs for both**.**

b) Provide syntax to create a **logical named X**, using logical and relational operators for **values greater than 30 in vector A**.

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

c) Provide syntax to create a **logical named Y**, using logical and relational operators for **values greater than 60 in vector B**.

d) Provide the output of **(A & B) < 50**

e) Provide the output of **B(B > 50)**

f) Provide the output of **~(A < 30) | (B > 50)**

g) Provide the output of **find(A == B)**

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.

4. An engineering process involves classifying products based on their weight. The following criteria is to be set:
   - Under 5kg is light
   - 5kg-10kg is moderate
   - Above 10kg is heavy

   In the box below, provide the function file that classifies if an item is light moderate or heavy. **The function should accept a weight W, and return the classification as a string named C**. Note, your function should only accept the weight of one item.

Important: If you are struggling with a task, ensure that you have performed hand-written work (e.g. hand calculations, pseudocode, flow charts) to better understand the processes involved. Do this before asking demonstrators for help and use it to assist with your illustration of the problem.