ENGR101 Lab 1

Submission Requirements

All labs are due at 11:59pm the day after your lab section. See Part 2 of this lab on how to submit these files.

- Exercise 1: Save your work as lab1_1.m
- Exercise 2: Save your work as lab1_2.txt
- Project 0: Save your work as AnalyzeSoil.m

Part 1: Getting Started with MATLAB

Prior Knowledge

Prior to this lab, you should have completed Lab 0 and watched the MATLAB tutorial video. The rest of this lab is written expecting you to have learned the following. If you cannot remember or have questions on these points, please review Lab 0 or ask your instructor before getting started with the lab.

- Logging into Windows on a CAEN machine (recap in next section)
- Opening MATLAB on a CAEN machine (recap in next section)
- Navigating MATLAB
    - Command Window
    - Workspace
    - Current Folder
    - Script Editor
- Assigning variables
- Suppressing output with semicolons

Opening MATLAB on a CAEN Machine

As most of you are likely new to CAEN computers, let's take a brief minute to recap how to log into Windows and open MATLAB. If you're already familiar with this process, feel free to skip to the next section. Talk to those around you or ask the GSI if you get stuck.

1. Verify that the machine is booted into Windows
    a. If the machine isn't booted into Windows, restart the machine and select the Windows icon in the boot prompt
2. Log into Windows using your uniqname and password
    a. You should have verified a working CAEN account in Lab 0
3. Shortly after login, a browser window should appear with https://appsanywhere.engin.umich.edu/ in the address bar
4. Scroll down until you see an entry labelled: "MATLAB R2017a (64-bit) Instructional"
5. Click the green "Launch" button corresponding to this entry
    a. Sometimes this will cause the page to refresh and display that it's validating the session. If this happens, wait for validation to finish and repeat steps 4 & 5
6. You should see the application loading in the Cloudpaging Player program. Once the status is set to "Ready", select the MATLAB entry and click the "Launch" button at the top
7. MATLAB will now launch (may take a minute)

Creating a 101 Directory

Whether you are working on a CAEN computer or on your personal laptop, it is important to keep your MATLAB files organized.

1. Navigate to your desktop
2. Right click on an empty space on the desktop
3. Select New→Folder
4. Name this folder `engr101`
5. Open this folder and create a new folder called `labs`
    a. This will be the base directory for all your lab files

Creating Variables in MATLAB

From now on, when you see >>, type the following line into the command window. *Make sure to pay attention to what each individual line is doing. Note that most lines DO NOT include semicolons so that you can easily see your work.*

You have seen in the video how easy it is to create variables that show up in the Workspace. So far, we've created **scalar** variables (variables with single values).

```
>> person = 0
>> Person = 12
>> Sarah = 21
>> Person = Sarah
```

- Notice that *Sarah* doesn't actually mean the word 'Sarah', it's a variable that currently holds the value 21.

```
>> Sarah = 56
```

In this example, each variable is assigned its own value. Make sure you can identify the Workspace and you can see these variables in that window. If you can't, ask your GSI or someone around you. Notice in the Workspace that *person* and *Person* are two separate variables with two different values. This is because variable names are **case sensitive.** Additionally, *Person* first has the value of 12 but then is **reassigned** to *Sarah*'s value (21). However, these values are not linked in any way: the value 21 is copied and then stored in the variable Person. When Sarah is reassigned to 56, the value of Person remains 21.

Now try entering the following into the command window and you should see your first MATLAB error.

```
>> 21 = Sarah
```

MATLAB doesn't know how to interpret your command and thus shows an error. Remember: **=** is the assignment operator. It does not act like the equals sign in algebra. Instead, it assigns a value on its right to a variable on its left. Since 21 is a scalar value that cannot be assigned, MATLAB gets upset.

Some variables are built-in. You can use these like any other variables, but be careful: you can also overwrite these variables.

```
>> pi
>> pi = 2
```

Strings

MATLAB isn't just for numbers. You can use letters and words in your programs too. In computer terms, we like to call letters **characters** and words **strings**. (you can remember them as a 'string of characters')

```
>> a = 'a'
>> phrase = 'Hello World!'
```

Notice that we use single quotes so that MATLAB knows the difference between variable names and characters/strings.

Finally, let's go ahead and erase any indication that you did this tutorial:

```
>> clc
```

This command will clear the command window, leaving you with what looks like a blank slate to program on. However, type the following:

```
>> phrase
```

The variable you created earlier is still there! You can also see it and all other variables in the Workspace. Let's just tell MATLAB to clear that evidence.

```
>> clear
```

Voila! This clears all the variables from MATLAB – it's like restarting MATLAB.

MATLAB Scripts

Command line programming isn't a particularly feasible way to complete your projects. What if you mess up the 50th line and have to start over by typing 1-49 in again? What are you going to turn in? To avoid those problems, we use **scripts**. Scripts are just files containing lines to be executed in a given order. You should have seen how to create these in the pre-class video. If you don't know or remember, please ask someone near you!

Comments

When you're writing programs, it's important to make notes in your programs. These notes should be lines that **are not interpreted by MATLAB**. Instead, they help human readers understand what you're trying to accomplish in your program. Comments are lines that begin with the percent character (%).

Part 2: Turning in Your Assignments

Labs must be turned in using the ENGR 101 Canvas website by pointing to the menu item "Assignments", and then via the Lab 1 menu selection. Be sure to submit all files listed under Submission Requirements.

The name of your files must be exactly as specified in order to receive credit, and no late submissions will be accepted.

Exercise 1: Getting to Know MATLAB

Complete the following. If you don't know how to do any steps, go ahead and ask your neighbor, or search online. For example, you'll find below that you need to take the square root of something, so you might search for "matlab how to take square root" to find the answer.

You might notice we sometimes use operators with a "dot" symbol (e.g. `.*` instead of `*` for multiplication). We'll explain this in more detail during lecture.

You will also notice that, in lab1_1.m, semicolons are used at the end of each line. Including semicolons at the end of a line does not change the mechanics of what that line is doing, but just keeps the result of that line from appearing in your command window (when we get to large programs, you will not want the result of every line of code to appear in your command window!).

Downloading Files and Setup

1. Download the `lab1_1.m` file from Google Drive where you found this lab
2. Create a `lab1` folder in the `engr101/labs` folder
3. Place the `lab1_1.m` file in the `engr101/labs/lab1` folder
4. Navigate your Current Folder to the `engr101/labs/lab1` folder. You should see `lab1_1.m` in the list of files

Completing and Testing Your Program

5. Complete the `lab1_1.m` program by reading the comments and filling in the missing lines.
6. Run your script by typing the following into the Command Window:
   ```
   >> lab1_1
   ```
7. Test your script with different values of `Seed` and make sure you get the following values
   a. Seed = 2 → apple = 4, Banana = 7, carrot = 4, dragonFruit = 350
   b. Seed = 50 → apple = -20, Banana = -1, carrot = 0, dragonFruit = 22
8. After testing different values of `Seed`, set the final value of `Seed` to 36.97815

Understanding the Current Folder

9. Navigate your Current Folder up one directory to your `engr101` folder by clicking the following button

10. Run your script again:

```
>> lab1_1
```

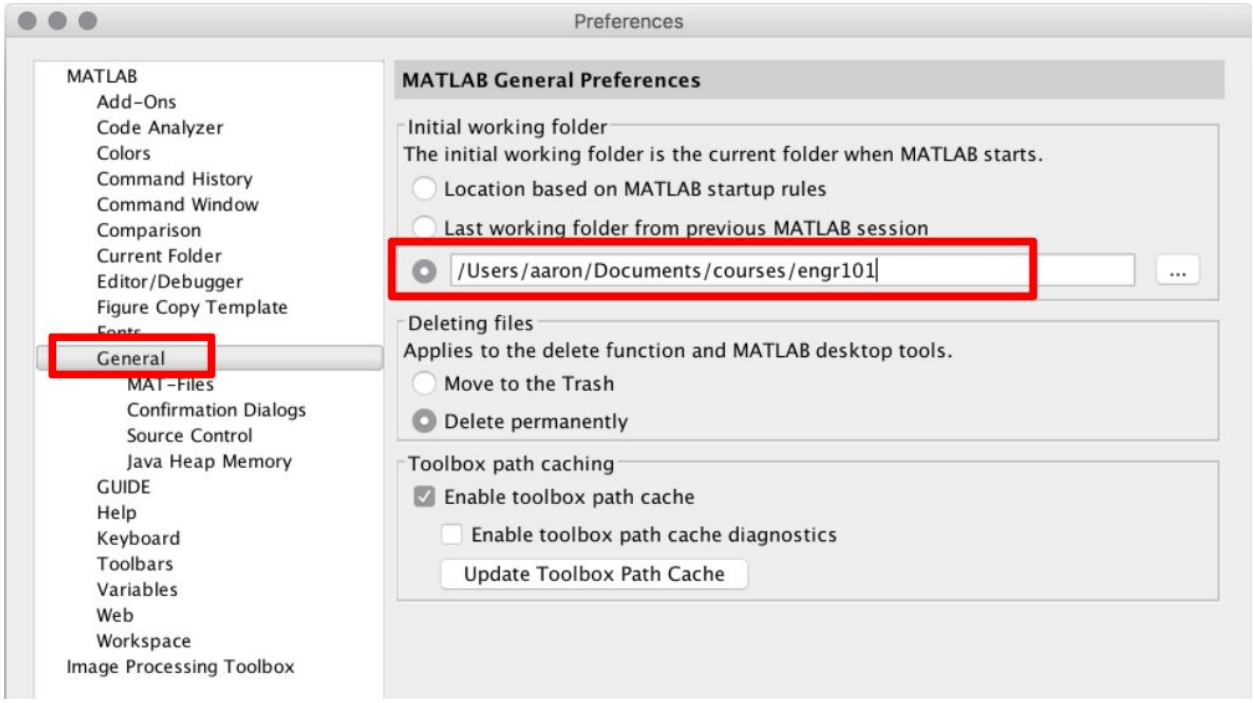
This step should result in an error. This is because the script you are trying to run is not in your Current Folder. MATLAB doesn't know what you're talking about when you say lab1_1.

11. Right click on your `lab1` folder and click *Add to Path*. Now try running the script again. This works because now MATLAB "knows" another place to look for something called lab1_1.

Setting the Default Current Folder

12. Open your *Preferences* from the *Home* tab

13. Select *General* and then change the initial working folder to your `engr101` folder



Exercise 2: Getting to Know Piazza

Piazza is our online forum. It's a useful resource for you to get help from instructors and your fellow students. We have compiled an etiquette document to help you use Piazza responsibly which you can find on Canvas under the Piazza tab.

Download lab1_2.txt and answer the questions within it. You can use any editor you'd like to edit the text file. Even the MATLAB script editor can work! Just make sure to place it in your lab1 folder to keep everything organized.

Project 0: Introduction to ENGR101 projects

Deadline: 9/14, 11:59pm (Note this is different than the rest of the lab)

Purpose

The purpose of this section is twofold: 1) to establish a theme for our ENGR101 projects, and 2) to perform some basic elements of programming in MATLAB: writing a script, creating a few variables, and writing code to perform an engineering exercise. It is meant to be a straightforward assignment that makes use of concepts reviewed in Lab 0 and Lab 1. An additional motive is to introduce you to our ENGR101 Autograder which shall be used in the remaining class projects.

Background

In August 2016, the European Southern Observatory (ESO) announced the confirmed detection of an exoplanet orbiting Proxima Centauri, the nearest star to our Sun. This alien world, designated "Proxima b", resides comfortably within the habitable zone around its parent star - its surface temperature is believed to permit water in liquid form. With this comes the alluring possibility that the planet could support life and might be the first step for humanity beyond our own Earth and into the stars.

However, we still know very little about Proxima b. The first step is to send a probe to the Proxima system[1] and land on the planet's surface. Your team is tasked with analyzing soil samples that will be collected by the probe. The probes have been designed to gather soil samples according to these recommendations by Michigan State University. The probe will then analyze the soil sample for some key measures of soil quality for farming (pH, phosphorus (P), potassium (K), calcium (Ca), magnesium (Mg), sodium (Na), organic matter, soil texture, etc.)

[1] We understand interstellar travel still lies in the realm of science fiction, but even today there are proposals for how it could be done. We'll ask that you imagine yourself as an engineer in the (perhaps not-too-distant?) future where these projects may be a reality.

According to this soil sampling schedule, not only are we supposed to take at least 20 samples across the area around proposed settlements, but the samples need to be repeated every 1-2 years so that we can build a history of how the soil changes in time. This is a LOT of samples to analyze! Let's start by analyzing one sample to make sure we have all the math worked out correctly.

Your Job

In this project, your job is to analyze one soil sample from the new planet. You will do this by calculating the percentage of sodium relative to other alkaline elements.

Your MATLAB script will calculate the *exchangeable sodium percentage (ESP)* of a sample of soil. This term is a sensitive indicator of whether or not plants can grow in the soil: too much sodium or not enough sodium and the soil cannot support plant growth. Use the following formula to calculate the exchangeable sodium percentage (ESP), the ratio of the amount of sodium in the soil relative to the sum of the amount of common alkali metals in the same sample (potassium, calcium, magnesium, and sodium)

$$ESP = \frac{Na}{K+Ca+Mg+Na}$$

Here,

- Na - amount of sodium in soil sample
- K - amount of potassium in soil sample
- Ca - amount of calcium in soil sample
- Mg - amount of magnesium in soil sample

All amounts have units of centimoles of charge per kilogram of soil ($cmol_c$ / kg).

Your task is to write a MATLAB script file (**AnalyzeSoil .m**) that:

- Creates variables for each of the alkali elements that correspond to the amount of that element in the soil sample
- Initializes the variables correctly according to the test case below
    - Make sure to suppress output from the initializations using a " ; "
- Calculates the ESP

Note
The above list can be considered the start of the *program design* for an engineering project. All engineering projects will provide detailed instructions that must be followed exactly. We'll talk more about program design throughout the term.

Required Script

There is 1 required script for this project. It must be named `AnalyzeSoil.m`. **The script must use the following variable names:**

```
sodium       % the amount of sodium in the soil sample
potassium    % the amount of potassium in the soil sample
calcium      % the amount of calcium in the soil sample
magnesium    % the amount of magnesium in the soil sample
ESP          % the exchangeable sodium percentage
```

Special Requirements
All output from MATLAB must be suppressed by using a ";"
Insert **comments**, preceded by the % character, to succinctly document your code.

Test Case

Test your code with the data in the following test case. In your script, these are the values that must appear with the appropriate variables in your submission. ***Note:* In your script, you must calculate ESP using an expression based on the formula above.** (Do not just initialize ESP to the correct value!) The correct ESP value is given to you allowing you to test (and correct) your MATLAB code.

alkali element	amount present in soil sample
Na	10.90
K	68.22
Ca	25.41
Mg	13.77
ESP	0.0921

Tips for a Successful Program

This is a simple program. It is intended as practice in coding a MATLAB script file: create some variables, perform basic calculations with the variables, add appropriate comments to your code, and submit the script file for grading. In the next project, you will start to do some more involved engineering work!

Test your code by submitting it to the Autograder at autograder.io. The autograder is an automated program that checks your code against the correct solution and returns both a grade and some interpretive text. Those of you who have previous coding experience understand that there are several possible outcomes when testing your code: 1) success (ALL DONE); 2) the program executes but does not provide the correct solution, or 3) the program does not execute and the computer provides a helpful list of errors which you can use to edit and correct your code.

The philosophy of ENGR101 is to permit multiple attempts in using the autograder to assist you in developing and improving your computer code. In other words, outcomes 2) and/or 3) are simply steps along the way to completing your task, writing a working and successful program. Please use the autograder, plus other techniques (such as a calculator) to work through the list of errors to correct your code and then resubmit. Your final Project grade will usually be related to your best autograder submission within any restrictions set in the project description. Your Lab 1 grade will be based on your submissions of Exercises 1 and 2 to CANVAS and Project 0 to the Autograder.

You must submit your final Project code to the Autograder before the deadline to receive credit for your Project. A portion of your project grade is given for short comments.

For grading purposes, consider Project 0 as an integral part of Lab 1. It is a precursor towards your efforts at successfully completing Projects 1 through 6. Individual project grades will be assigned only to Projects 1 through 6 and not to Project 0.