ENGR 151

Fall 2023

Lab 4: MATLAB Writing and Reading Files & Data Analysis

Introduction

Why do we care about data analysis using MATLAB? Well, there are several reasons, and we're going to give one example. Imagine you're doing experiments and recording them in spreadsheets, and you need to create graphs and obtain descriptive statistics such as mean, mode, and standard deviation. You can always do these by hand on Excel. In fact, Excel does a pretty good job helping you do so! However, with MATLAB, you can just write a function to do this data analysis and plotting once, and all you have to do is change the name of the file being read in, saving you time and energy!

Of course, there's multiple paths to the same destination, but depending on what you'll be working on in the future, using MATLAB to perform data analysis may very well make the most sense out of all the other options!

MATLAB Documentation Links

We ***strongly*** encourage you to look at the documentation to address any obstacles you may encounter while completing this lab.

https://www.mathworks.com/help/matlab/ref/readmatrix.html
https://www.mathworks.com/help/matlab/ref/writematrix.html
https://www.mathworks.com/help/matlab/ref/categorical.html
https://www.mathworks.com/help/matlab/ref/bar.html
https://www.mathworks.com/help/matlab/ref/find.html

Exercise 1: Game Stats (Plotting and File Input/Output)

You will be analyzing the scores history of 6 different Big Ten teams. The data is provided to you in the spreadsheet “stats.xlsx”. In this spreadsheet there are 4 different worksheets: “football_wins”, “football_total”, “basketball_wins”, and “basketball_total”.

Each worksheet looks similar to the table below:

	A	B	C	D	E	F	G
1		Michigan	Ohio State	Michigan State	Notre Dame	Northwestern	Penn State
2	2000	4	11	13	11	3	7
3	2001	5	11	10	10	7	3
4	2002	10	7	10	10	3	2
5	2003	8	6	12	9	8	3
6	2004	4	8	13	9	6	1
7	2005	8	12	8	6	6	6
8	2006	8	15	8	11	2	2
9	2007	5	10	12	14	1	7
10	2008	9	10	15	8	8	10
11	2009	7	14	14	10	7	3
12	2010	9	16	9	14	7	9
13	2011	13	13	13	13	8	4
14	2012	12	13	13	11	4	2
15	2013	15	10	12	6	6	6
16	2014	8	11	12	14	6	4
17	2015	10	11	13	11	8	7
18	2016	10	7	10	12	10	6
19	2017	13	15	16	8	6	9

where the top row is the name of the school, the leftmost column is the year, and the data shows:

- “football _wins”: football games won by that school that year
- “football _total”: football games played by that school that year
- “basketball _wins”: basketball games won by that school that year
- “basketball _total”: basketball games played by that school that year

Your job is to calculate the teams’ **average Win Ratio** and write this data into the spreadsheet. Write your code in a file called **stats.m**.

Note: MATLAB recommends that, after 2019, users should refrain from using xlsread() and xlswrite(). Therefore, this lab uses the recommended `readmatrix()` and `writematrix()` functions. These functions are also usable on MacOS Catalina, while `xlswrite()` may have issues writing to xlsx files on Catalina.

Data File Input: Brief Overview

This is a brief overview of the functions you may be using for the lab.
You will use the function `readmatrix()` to read data from the spreadsheet. The simplest form of `readmatrix()` takes in an input of the filename, reads the values

from the first sheet, and returns an array of those numbers. (Example form: `data_arr = readmatrix('data.xlsx')`).

But, for this lab, you'll want more functionality. Specifically, you want to utilize the additional input arguments such as sheet number and cell range. An example would be (`data_arr = readmatrix('data.xlsx', 'Sheet', 1, 'Range', 'A1:A2');`). This grabs the data in cells A1 and A2 in Sheet 1 and outputs a 2 x 1 array and assigns it to `data_arr`.

The `readmatrix()` function can also return a string array, which comes in handy if you want to extract the column/row names of a dataset! Reference MATLAB's documentation to find a way to extract the column names from the stats.xlsx file, which in our case are the school names. (Hint: you need the input argument 'string').

We encourage you to **use the command window** to view the data you are reading to make sure it is what you want. Before you make any plots, try simply reading in the data.

Data Analysis and File Output: Your Task

You will be calculating **Average Win Ratio** over all the years for each school and writing this data back into the `stats.xlsx` file on a new sheet called 'ratios' using `writematrix()`. This function works similar to the `readmatrix()` function, but use `help writematrix` or visit the documentation linked above to learn what else you may need to know.

To find the **Average Win Ratio** value of Michigan basketball, you find the ratio of **"wins" to "total games played"** for each year **and then average those values**.

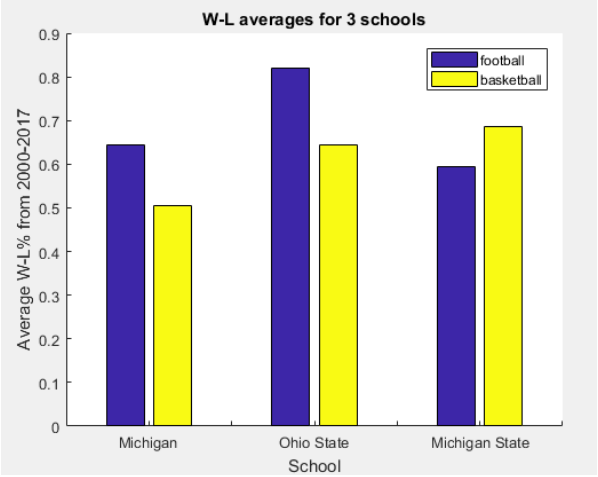
Write the data into a new worksheet within stats.xlsx called 'ratios'. You will write 3 rows. The top row contains the school names. The second row contains the football averages. And the third row contains the basketball averages. For example, the first 3 columns should look something like this:

	A	B	C
1	Michigan	Ohio State	Michigan State
2	0.651147099	0.826884519	0.591754736
3	0.518640351	0.631432749	0.691959064
4			

Plotting

You will be plotting the **Average Win Ratio** over all the years for each school, for both sports. For example, to find the value of Michigan basketball, you would find the ratio of "wins" to "total games played" for each year and then average those values.

Plot each sport in a different color as shown below. (The example shows 3 schools, but you must include all)



For this you will need the function `bar()`. Note that if you call the bar function on a 2D array, it will **group each row into the same bin**. For example, in the picture above, each row would have 2 elements

```
[ MI_fb, MI_bb;
  OSU_fb, OSU_bb;
  MSU_fb, MSU_bb ]
```

If you have your data for each sport in rows, you may want to use "`'`" to transpose it.

```
fb_avgs = fb_avgs';       % move data from 1 row to 1 column
bb_avgs = bb_avgs';       % move data from 1 row to 1 column
cat(2,fb_avgs,bb_avgs);   % concatenate 2 columns into 1 array
```

Make sure your plot has:

1. The school names as **labels for the bins:** if you pass `bar()` 2 input arrays, the first will become the labels for the data bins. These will need to be of type "categorical". You can convert data to that type using the function `categorical()`. (Documentation linked above.)
2. **Axis labels** (doesn't matter what they're called; just make sure they're descriptive)
3. A **title** (doesn't matter what it's called; just make sure it's descriptive)
4. A **legend** (i.e. football and basketball)

Code Submission Guidelines

See guidelines listed under the Canvas MATLAB Autograder page!!