**Writing MATLAB Scripts**

**Learning Objectives**

* Learn how to write and save MATLAB scripts.
* Learn how to save MATLAB plots to disk.

**Part 1 – Intro to Scripts**

Why we use scripts

.m extension

Comments first

Make separate data folder (discuss good data conventions, read only)

Reading data, analyse data

Save a script and try to run it from the wrong directory (error). Explain how to change the path

\*\*START A NEW SCRIPT CALLED analyse\_data\*\*

Write the comments in green first (this helps structure your script)

Then start filling in the code. EXPLAIN SEMI-COLONS – you don’t want heaps of lines printing out on your screen every time you run the script!

Explain the disp command – used to display things you do want to print out

%% Analyse Data

% script is to analyse patient data

%% Read in data using the code generated in last section

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ SAMPLE CODE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Import the data

[~, ~, raw] = xlsread('/Users/wsyeda/Documents/MATLAB/MATLAB\_GO/Data/Pokemon\_CP.xlsx','Form Responses 1');

raw = raw(2:end,:);

raw(cellfun(@(x) ~isempty(x) && isnumeric(x) && isnan(x),raw)) = {''};

cellVectors = raw(:,1);

raw = raw(:,[2,3,4,5]);

%Replace non-numeric cells with NaN

R = cellfun(@(x) ~isnumeric(x) && ~islogical(x),raw); % Find non-numeric cells

raw(R) = {NaN}; % Replace non-numeric cells

%Create output variable

data = reshape([raw{:}],size(raw));

% Allocate imported array to column variable names

Name = cellVectors(:,1);

Pokedex\_Num = data(:,1);

CP\_before = data(:,2);

CP\_after = data(:,3);

Level = data(:,4);

% Clear temporary variables

clearvars data raw cellVectors R;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pokemon\_CP=[CP\_before CP\_after];

%% Calculate Statistics

% find maximum Combat Power

max\_CP = max(Pokemon\_CP(:));

% find minimum

min\_CP = min(Pokemon\_CP(:));

% find std

std\_CP = std(Pokemon\_CP(:));

%% Display values

disp(['Maximum combat power: ' num2str(max\_CP)]);

disp(['Minimum combat power: ' num2str(min\_CP)]);

disp(['standard Deviation of combat power: ' num2str(std\_CP)]);

**Part 2 - Plotting**

Plotting data (just plot mean – the min, max and std are done in the challenge)

run script

%% Plot data

% plot mean

plot(mean(Pokemon\_Cp,2))

% plot min

% plot max

% plot std

Add axis labels and title

**Challenge**

%% Challenge #1

% Using analyse\_data as as model, create your own script called

% plot\_data that plots mean, min, max and std calculated over all pokemons for each day.

%% Extension - Plotting using subplots

% Type help subplot in the command window or search the documentation

% Can you work out how to create two plots in the same figure?

% Plot the four plots you have generated with your analyse.data script, but now arrange them in a subplot so that you have two plots in the first

% row and two in the second row

% Give all subplots x and y labels and a title.

**Part 3 – Printing Figures To File**

Go over subplot answers

% plot data

subplot(2,2,1)

plot(mean(Pokemon\_CP,2)) %plot mean

xlabel('Days')

ylabel('Mean inflammation')

title('Mean inflammation over time')

%plot min

subplot(2,2,2)

plot(min(Pokemon\_CP,[],2))

xlabel('Days')

ylabel('Min inflammation')

title('Min inflammation over time')

%plot max

subplot(2,2,3)

plot(max(Pokemon\_CP,[],2))

xlabel('Days')

ylabel('Max inflammation')

title('Max inflammation over time')

%plot std

subplot(2,2,4)

plot(std(Pokemon\_CP,[],2))

xlabel('Days')

ylabel('Std inflammation')

title('Std inflammation over time')

Saving your figure to file

%% Save Plot

print('Pokemon\_CP\_Plot\_1','-dpng')

**Challenge**

%% Challenge #2

% Add a line to your script that will save the plot you generated

% as a png file in the current working directory.

%% Extension

% Using the 'hold' function plot the mean, minimum and maximum inflammation

% in the same figure. See if you can use different colours to differentiate

% between the different parameters. Add a legend to your figure. Save your

% figure as a png.

% Look at the documentation for print and see if you can work out how to

% save your plot in a different format

% Can you also work out how to save it at a higher resolution?

**Part 4 – Debugging**

Break the analyse\_data script and talk them through the debugging options

**Challenge!**

%% Challenge 3

% Debugging scripts

% Debug pie\_script