Deliverables: Submit by 17:00 on Wednesday of Week-07.

Late Penalty: Late submissions will be penalised at the rate of 10% per day (including weekends). The penalty applies to the maximum available mark. For example, if you submit 2 days late, maximum available marks is 80% of the assignment marks. Submissions will not be accepted after 9am Monday of Week-08. To Submit this assignment, go to the Submission page and click the link named "Make Submission".

GENERAL RULES

- 1. This assignment is weighted at 20%
- 2. You are reminded that work submitted for assessment must be your own. It's OK to discuss approaches to solutions with other students, and to get general help from tutors or others, but you must write the python code yourself. Increasingly sophisticated software is used to identify submissions that are unreasonably similar, and marks will be reduced or removed in such cases.
- 3. The <u>Student Code</u> of Conduct sets out what the University expects from students as members of the UNSW community. As well as the learning, teaching and research environment, the University aims to provide an environment that enables students to achieve their full potential and to provide an experience consistent with the University's values and guiding principles. A condition of enrolment is that students inform themselves of the University's rules and policies affecting them, & conduct themselves accordingly. (see the course outline page with links).
- 4. You should also read the following page which describes your rights and responsibilities in the CSE context: <u>Essential Advice for CSE Students</u>

Essential

The University views <u>plagiarism</u> very seriously. UNSW and CSE <u>treat plagiarism</u> as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. [see: <u>UNSW Plagiarism Procedure</u>]

Change Log

- 1. Your functions max_peak, total_peaks, peak_list_from_file should import peak_list in your code for these three functions [Updated Oct 13th].
- 2. If there are no peaks in the inputs given max_peak should return the string "No Peaks" [Updated Oct 13th]
- 3. The text file format is one number per line (as shown in voltage_data_complete.txt in week 3) [Update Oct 13th]
- 4. For the purposes of this assignment we are only looking to identify peaks above the mean (and we will only test for such conditions) [Oct 13th]

Learning Objectives:

- 1. To apply programming concepts of variable declaration, constant declaration, assignment, selection and iteration (for loop).
- 2. To translate an algorithm described in a blend of natural language and elements of pseudocode to a computer language.
- 3. To organise programs into smaller modules by using functions.
- 4. To use good program style including comments, meaning variable names and others.
- 5. To get a practice on software development, which includes incremental development, testing and debugging.

What you must submit total_peaks.py max_peak.py peak_list.py peak_list_from_file.py

Oct 7th tip: Please read the entire practical first and think carefully about the order in which you might implement these functions!

The General Problem: Peak Detection



In this assignment, your goal is to write a python program to determine various characteristics about some time series data you are given. These types of problems are very common in many disciplines, including computer science, engineering, medicine and science. There are many different types of peak detection problems, the setting of this assignment is similar to that used in pulse oximeter data. The method described below is based on a sliding window which allows us to calculate a standard score of the number of standard deviations aways from the mean a value is.

Such peak detection has been used to understand the periodic distributions of nucleotides from the genome of SARS-CoV-2 had been sequenced and structurally annotated or pollution peaks or heart rate detection. Peak detection is a very common task in the understanding of data.

The Algorithmic Problem

What is a "peak" and how can we reliably detect them! Let's step back to some basic stats.

For a finite normal population with N members, the population mean is the following (mu):

$$\sigma^{2} = \frac{1}{N} \sum_{i=1}^{N} (x_{i} - \mu)^{2}$$

 $\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2},$

 $\mu = \frac{1}{N} \sum_{i=1}^{N} x_{i}$

Variance is a measure of dispersion (spread). For a finite normal population with N members, the population variance is the following (sigma squared):

Standard deviation is a also measure of dispersion (spread). The standard deviation of the population is the square root of the variance, it is written as (sigma):



[Basic Standard Deviation diagram: Source Wikipedia]

"A low standard deviation indicates that the values tend to be close to the mean (also called the expected value) of the set, while a high standard deviation indicates that the values are spread out over a wider range."

Observation: 1

In statistics, the **z-score** of "standard score" says, how many standard deviations is a data point above or below the mean value of what is being observed. As such, looking for values within our data with "high" z-scores might be a way to identify peaks in our data.

Observation: 2

We can determine the mean by looking at all the data. Or we could determine a moving mean value for data within a given window, not for all the data we are observing. We can refer to this "smoothing window" or lag as our "window".

Observation: 3

Finally, when we do observe something above a particular z-score we can decide how much (if at all) this should influence the moving mean value for the data within this window or subsequent windows.

Description of the peak detection algorithm

The goal is to detect does a particular part of an input signal have peaks and if so where. In the following example, the peak to be detected is a sequence of 20 numbers (see Fig 1) begins at the 9th element in the input list and ends at the 12th element.

However, if you look at the input list you can see lots of smaller up and down patterns in the data which could each be considered peaks. So, what makes 9 - 12 a peak while the others aren't. Firstly, the values 41, 38, 22 and 10 are beyond the average. How far beyond the average? Well at least 3 standard deviations in this case. The next question is,



Figure 1: Input signal. Plot of input and detected peaks (based on a given window of 8 and z-score of 3 and an input peak influence of 0)

what "average". Well we use an "moving mean" or sliding window to calculate the average within a window of the last N values (in this case 8).

However, you might be wondering, by the time the input value of 10 is considered surely the average of the last N numbers of the smoothing window (22, 38, 41, 0 ... etc.) suggests that 10 isn't that far away from the mean. This is where the notion of "**influence**" comes in. When we are considering the input we must **filter** it so that when we do detect a



Figure 2: Input and how we calculate the average and the standard deviation for the first input value in list s we consider

Original input values x



Update **list s** if a peak value is detected to alter the influence the input values may have on the moving mean

Figure 3: Formula required to filter input data and to label peaks (yi)

peak value, then those input values are treated differently than when we aren't seeing inputs from a peak value.

Implementation requirements

You need to implement the following four functions, each in a separate file.

Your functions max_peak, total_peaks, peak_list_from_file should import peak_list in your code for these three functions [Updated Oct 13th].

You need to submit these four files, each containing one function you implement

1. def total_peaks(inputs, smoothing, th, influence):

- The aim of this function is to calculate the total number of distinct peaks measured in the inputs given
- The first parameter 'inputs' is a list of (float) values, and the second parameter 'smoothing' is the size of the window used to determine the meaning average, the third parameter is the z-score (or how many standard deviations away from the moving mean does the input need to be for it to be considered a peak) and finally the 'influence' says how much should a value which is considered a peak value effect the moving average
- The function should could the number of distinct "peaks" in the inputs given (a peak should be considered as a continuous unbroken series of 1 1 1 values)
- This function can be tested using the file 'test_total_peaks.py'.

2. def max_peak(inputs, smoothing, th, influence):

- The aim of this function is to determine the maximum value of an input which is considered to be a part of a peak measured in the inputs given
- The parameters are the same as in (1) above
- If there are no peaks in the inputs given then max_peak should return the string "No Peaks" [Updated Oct 13th]
- This function can be tested using the file 'test_max_peak.py'.

3. def peak_list(inputs, smoothing, th, influence):

- The two possible outcomes are "Insufficient data" and list which is the same length as inputs but consistent of 0's and 1's where a 1 represents a peak value, given the smoothing, th and influence values from the inputs
- For the purposes of this assignment we are only looking to identify peaks above the mean (and we will only test for such conditions) [Updated Oct 13th]
- The first parameter 'inputs' is a list of (float) values, and the second parameter 'smoothing' is the size of the window used to determine the meaning average, the third parameter is the z-score (or how many standard deviations away from the moving mean does the input need to be for it to be considered a peak) and finally the 'influence' says how much should a value which is considered a peak value effect the moving average
- To code this function you should first create a copy of the inputs into a filtered_input list (which can be updated based on the values si (given in figure 3) when you determine an input value is a peak) you also need a list to hold the outputs measured (ie the 0's and 1's)
- You need to calculate the mean and the standard deviation for the smoothing window so you can test the zi value (you need to do this each time you move forward in the list)
- · When the test passes you should record a 1 into the output list
- · When the test passes you need to update the filtered input list with the value for si
- This function can be tested using the file 'test_peak_list.py'

4. def peak_list_from_file(smoothing, th, influence, file_name):

- Here you aren't provided a list of values but instead you should read in the inputs from a file
- This function can be tested using a test file you should write yourself called 'test_peak_list_from_file.py'.
- The text file format is one number per line (as shown in voltage_data_complete.txt in week 3) [Update Oct 13th]

Getting Started

- 1. Download the zip file **ass1_prelim.zip**, and unzip it. This will create the directory (folder) named 'ass1_prelim'.
- Rename/move the directory (folder) you just created named 'ass1_prelim' to 'ass1'. The name is different to avoid possibly overwriting your work if you were to download the 'ass1_prelim.zip' file again later.
- 3. First browse through all the files provided, and importantly read comments in the files.
- 4. Do not try to implement too much at once, just one function at a time and test that it is working before moving on.
- 5. Start implementing the first function, properly test it using the given testing file, and once you are happy, move on to the the second function, and so on.
- 6. Please do **not** use 'print' or 'input' statements. We won't be able to assess your program properly if you do. Remember, all the required values are part of the parameters, and your function needs to return the required answer. Do not 'print' your answers.

Testing

Test your functions thoroughly before submission. You can use the provided python programs to test your functions.

Please note that the tests provided in these files cover **only basic scenarios (cases)**, you need to think about other possible cases, modify the files accordingly and test your functions. For example, you need to **add cases** to test for "Insufficient data" scenarios.

Submission

You need to submit the following four files. Do not submit any other files. For example, you do not need to submit your modified test files.

- total_peaks.py
- max_peak.py
- peak_list.py
- peak_list_from_file.py

To Submit this assignment, go to the **Submission page** and click the tab named "**Make Submission**".

Assessment Criteria

We will test your program thoroughly and objectively. This assignment will be marked out of 20 where 15 marks are for correctness and 5 marks are for style.

Correctness

The 15 marks for correctness are awarded according to these criteria.

Criteria	Nominal marks
Function total_peaks	2
Function max_peak	1
Function peak_list	9

Case "Insufficient data"	1	
Function peak_list_from_file	2	

Style

Five (5) marks are awarded by your tutor for style and complexity of your solution. The style assessment includes the following, in no particular order:

- Use of meaningful variable names where applicable
- Use of sensible comments to explain what you're doing
- Use of docstring for documentation to identify purpose, author, date, data dictionary, parameters, return value(s) and program description at the top of the file

Assignment Originality

You are reminded that work submitted for assessment must be your own. It's OK to discuss approaches to solutions with other students, and to get help from tutors and consultants, but you must write the python code yourself. Sophisticated software is used to identify submissions that are unreasonably similar, and marks will be reduced or removed in such cases.

Further Information

- Additional Help Sessions will be available for this assignment during week-05 to week-07.
- Use the forum to ask general questions about the assignment, but take specific ones to Help Sessions.
- You can ask your tutor during your lab time any queries you may have regarding this assignment.
- Keep an eye on the class webpage notice board for updates and responses.
- This assignment has never been posed in ENG1811 previously.