Automated Robust First Peak Detection in a Time Signal Using Computational Intelligence

Information Technology (M.Eng.)
Computational Intelligence
Mashnunul Huq
1384042
(mashnunul.huq@stud.fra-uas.de)

***Abstract*— This paper represents computational approach to find first peak in a time series signal using statistical standard score. By using the standard deviation of a portion of the signal selected by window technique and measuring the dispersion of whole signal threshold for peaks at points are detected. From the continuous threshold points throughout signal, peaks are obtained as crossing the threshold. This technique is implemented on a specific dataset of 50 signals.**

Keywords—dispersion, influence, lag, robust algorithm, signal, statistical average, threshold, window, z-score

I. INTRODUCTION

The engineering field of signal processing focuses on analyzing analog and digital signal with respect to time. Here the term time series stands for a data sequence recorded at regular intervals of time. In today's artificial intelligence projects time series analysis is an important step in the development phase of a program for forecasting data series, frequency and prediction of occurrence of a particular point. This analytic process helps to extract important statistics and characteristics of data to create accurate forecasting. Time series analysis is important in everyday life science. We can use it to measure patient's heart condition to the distant and deep earth gas geochemistry analysis to predict an earthquake.

Now signals are time varying data that represents physical events and belongs to time series data. Every signal has two major components, amplitude and frequency. By definition, frequency is the number of cycles a wave or signal completes in one second and it is always defined as the inverse of time (unit is Hz which is equal to 1/s). Where as amplitude is related to the maximum absolute value reached by a waveform at a particular point. As mentioned the absolute value which in actual can be in both direction negative and positive. In this project our working area is the peak or maximum amplitude. Peak is defined as the maximum positive or negative deviation of a waveform from the stationary reference level. In this project the positive maximum deviation is counted as the peak. But incase of maximum value finding there can be two types of maximum values, Local Maximum and Global Maximum. In a sense local maximum is the greatest value for the given subset but there can be larger values outside noted subset. Global maximum is the largest value in the whole set of data. So in a sense, inside a data set or signal amplitudes there can be many local maxima but only one global maxima. Global maxima itself can be a local maxima for a subset. If we take the window technique to find all the local maxima for whole signal data then eventually we will find the global or absolute maxima. But the project is finding the first peak or the first local maxima. Local maximum can be found from the equation of curve of a portion of the signal. If we find the first derivative of that curve equation and equals it to zero, then solving this new equation will give us the local extreme which can be local maximum or local minimum. "An interior point of the domain of a function f where f- is zero or undefined is a critical point of f." (Thomas, 2013)

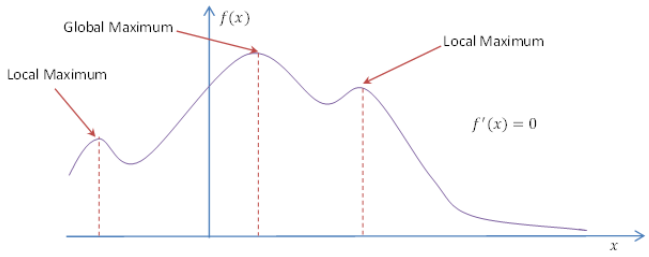


Figure 1: Global and Local Maximum

In the realm of statistics two terms are frequently used population and sample. Population is the term to describe the entire data set and sample is the specific group of data selected from that data set. E.g., the population of Frankfurt is the population for collecting data on monthly expenditure and if from the data collected one choose to analyze expenditure nature of people aged between 10 to 30 then that is the sample. In our case the whole signal data is the population e.g. signal number 5, and the current window working with one or five data points is the sample data. In conclusion we can find local maximum values by calculating from these samples.

Mean and median are used to measure the central tendency of a data set. Mean is the average of a group data obtained by summing up all the data and dividing it by the number of data. Mean is always sensitive to large values when the sample is small. E.g. if we take 20 students' exam score, two higher scores will shift the mean to the high score portion as well as two lower scores can shift it to the failing score. Median is the point where half position of the data can be determined. From Bell's curve (normal distribution) theorem median can still be found at the half point even the curve is positively or negatively skewed. If the data set is negatively skewed then the distribution of data have a tail on the left side. If the dataset is positively skewed then the bell curve will have a tail on the right side. (Gregersen, 2020)

In order to know better about a data set understanding standard deviation and variance is a must. Both of them refers to the dispersion characteristics of the signal data. Variance is the average of the squared differences from the mean. That puts forth on how far a set of data is spread out from their average value and how steep the bell curve becomes. The lower the value of variance means the more

steep normal distribution curve becomes. Standard deviation is obtained by square rooting the variance. (Bland & Altman, 1996)

The standard score or the z-score is the number of standard deviation that defines the raw data point is how much above or below the mean value. It is calculated by subtracting the population mean from an individual raw data and dividing the answer by population's standard deviation. For computing the z-score, mean and standard deviation of the complete population is required. Z-score is often used in standardized testing called z-test. It is a test where null hypothesis can be approximated by a normal distribution. Z-score is used in calculation of prediction intervals of the z-test. Main difference between z-test and t-test is that z-test is the standardized testing for a sample of the population and t-test is the standardized testing for the whole population. (Carroll & Carroll, 2009)

When the signal is real, data set becomes enormous. In that case manipulating or calculating parameters and analyzing for the whole signal is inefficient and cumbersome. To avoid this situation window technique is used where at a time a small subset of the whole signal is manipulated. E.g. a rectangular window where by truncating the dataset before and after the window only manipulation takes place for the rectangular section. More efficient windowing techniques are hamming or blackman window.

Lag is referred to the delay of starting any process. In our project calculation of z-score starts from a certain point where main signal data is more visible otherwise the computational power will be wasted calculating the low level noises. It saves time, power and also from computational hazards to manipulate the signal data with ease.

The term influence stands for the affect of something on the situation or over another thing. In our project influence of previous data points are used on future windows. It is useful for reducing computational time by storing the average point value to a certain level while calculating for the next window of the whole signal.

II. Method

A. Reading Data From File

To start with the data processing first step is to read from the data file. As per the data file of our project has an extension of 'xlsx', it can be opened using Microsoft excel application. There are plenty of libraries to open an excel file but pandas library is the best and mostly used among them. We read from the file using the method called pandas.read_excel. As there are no heading inside the file we initiated the file using header attribute equals to None. After reading from the file we added column heading as convention "col#"(Table 1). The file path is read in a tkinter StringVar object and the entry for the path is created using ttk.Entry object (Table 2). In the file we can find out that every row is an individual signal data and first 6 columns (for second file 17 columns) are not part of the signal. For that reason we dumped those columns using pandas.Dataframe.drop method.

Table 1: File reading

```
df = pd.read_excel(file_path_text.get(), header=None)
```

```
columnName = []
for i in range(len(df.axes[1])):
    name = "col" + str(i)
    columnName.append(name)
df.columns = columnName
```

Table 2: Entry for file reading

```
file_path_entry = ttk.Entry(insiderFrame, textvariable=file_path_text, width=100)
file_path_entry.grid(row=0, column=1)
```

B. Signal Manipulation

All the signals have a data range from -0.33 to +0.33. So, the peak detection for our algorithm is difficult if the data span is small and also the peak is smoothed peak. A smoothed peak is where the slope of the peak is very small (Figure 2). As our algorithm changes the threshold as the signal proceeds with the forwarding windows, smooth peaks are hard to be detected as peaks. For this reason, we normalized the amplitude of the signal. While normalizing we took the difference on higher values magnified in higher scales. E.g., the amplitude between 0.3 and 0.31 has a magnified amplitude of 1 whereas between 0.324 and 0.325 has a magnified value of 1000000. Moreover, we magnified only the positive side of the signal as we wanted the positive first peak of the signal (Table 2). While magnifying amplitudes we used the technique of multiplying each interval after 0.3 value 10 times than the previous interval.

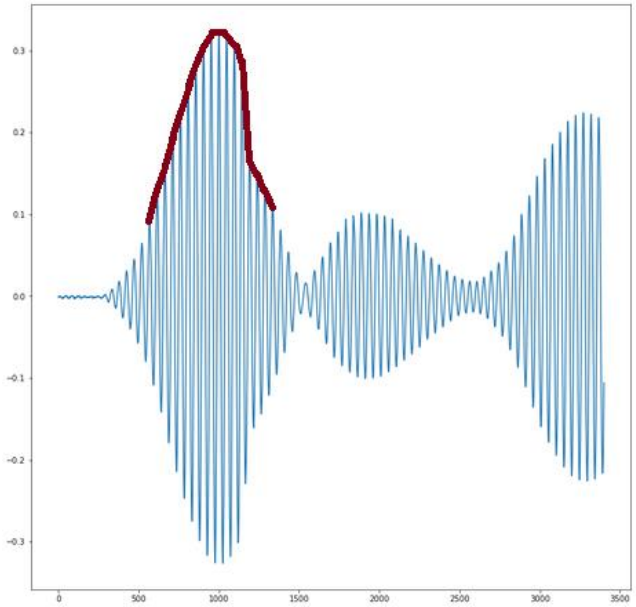


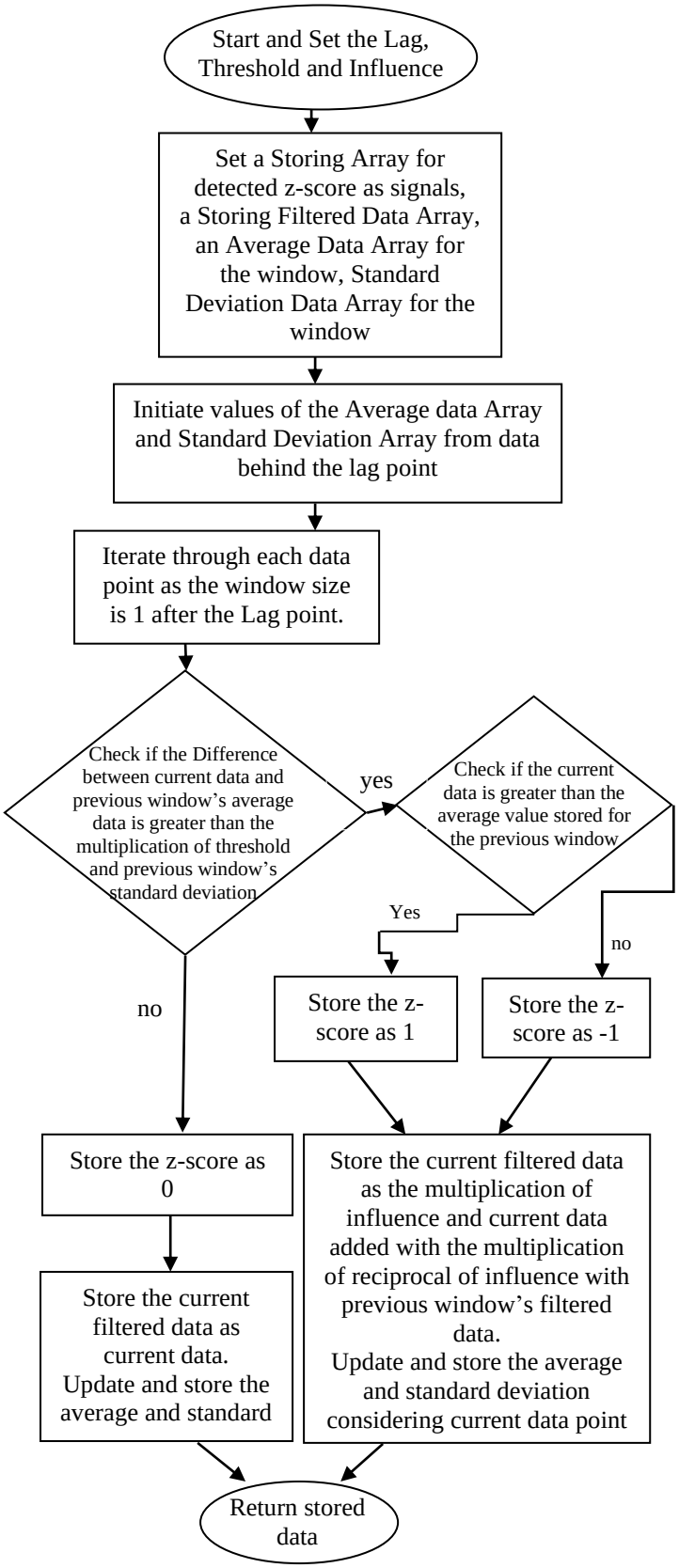
Figure 2: Smooth peak of the signal no.1

Table 3: Magnification of amplitudes

```
y = np.where((y >= 0.1) & (y <= 0.2), 0.001, y)
y = np.where((y >= 0.2) & (y <= 0.3), 0.005, y)
y = np.where((y >= 0.3) & (y <= 0.31), 1, y)
```

```
y = np.where((y >= 0.31) & (y <= 0.32), 10, y)
y = np.where((y >= 0.32) & (y <= 0.321), 100, y)
y = np.where( (y > 0.321) & (y <= 0.322), 1000, y)
y = np.where( (y > 0.322) & (y <= 0.323), 10000, y)
y = np.where( (y > 0.323) & (y <= 0.324), 100000, y)
y = np.where( (y > 0.324) & (y <= 0.325), 1000000, y)
y = np.where( (y > 0.325) & (y <= 0.326), 10000000, y)
y = np.where( (y> 0.326), 100000000, y)
```

III. Z-Score Algorithm



To start with this algorithm we need to see the thresholding_algo function of the code. First we need a lag point from where the window of calculating average mean and standard deviation will start. We decided the lag point is at the point of the signal where the value of the signal is 0.1. This Lag point can also be decided by the user putting the value in the field of noise amplitude which is by default 0.1. As the data of the second file is more than 4000 in values we divided the value with 1000 to match the fractioned value like the first data file. But there are spikes present in the second data file. That is why only for only the second file's data spikes are needed to be cleared first. This is done with the function called despike. With this function the unwanted spike is found comparing with the near signal points difference with a given threshold (th = 20). The area of the spike can be narrowed or widened by changing the value of threshold and then with fitting technique the concatenated spike portion is filled with interpolated values.

After cleaning the signal we can use the signal for our main peak finding algorithm. This algorithm in thresholding_algo finction can detect the peaks that is higher than the moving average threshold of the signal. As the window forwards along the signal counting each points, the threshold changes with the updating avg and standard deviation value (Figure3 Green line). Normally Z-score is calculated using the formula of

z-score = (raw-score – population mean) / population standard deviation

This is an standardization technique of a normal distribution data. But for our case the z-score is limited between -1, 0 and 1. So the data above the moving threshold is 1 and those are the point where the peak can be found.

As we can see with this algorithm the thresholding point takes almost the same shape of the signal so we came up with normalizing the gap with higher values to make the peak higher than the threshold value (Figure 4). The threshold is calculated with the following equation,

Threshold=Influence x Point Data + (1 - Influence) x Previous Thresh.

From the z-score we can find out the points where the peak can be situated (Figure 5). Among these points the highest is the peak.

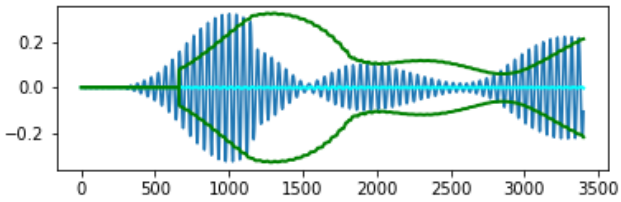


Figure 3: Threshold changing with the consecutive windows

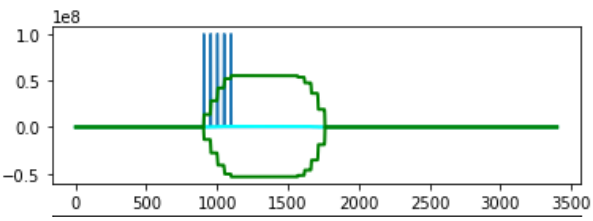


Figure 4: Peak values crossing the signal's moving threshold

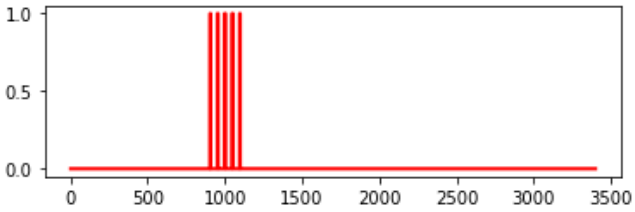


Figure 5: Z-Score as 1 where the elaborated signal peak is higher than threshold

IV. How to use the application

As the application's user interface is built on tkinter, the fields to be filled mandatorily by the user is marked with "*" at the end of the label. The first mandatory field is the file path. Either giving the full directory path with file extension or just giving only the file name where the file is already in the same application directory the program will work fine. The second mandatory field is the signal number. As the application shows one signal at a time, the signal number has to be given. The processing of the two data files is different and it is automatically done inside. The second data file is big and requires much more time than the first data file.

All the other fields are not mandatory but user can use these fields to find different outcomes and optimize the application's automation. Noise Ignoring Amplitude is the amplitude where the signal is very small. By default, this value is set to 0.15 as per the data optimized for the first data file. The initial threshold value is the threshold set for the z-score algorithm. Default value is set to 25 and for signal number 15 24 based on the first data file. The second data file is overall set to 25. The optimization of the second file's data yet to be done. The last field is the spike removal threshold which only works with the second data file determines the width of the spike area to be removed and interpolated from average data. The default value for this field is 15 but it needs to be changed with every signal as the spike for different data is different in area and amplitude level also.

The result is shown in text in the bolded text box after the start button. When the two mandatory fields are given by the user and the start button is pressed, data processing starts and for the first data file it takes less than 1 minute to produce the result and for the second data file it takes 5-10 minutes as loading the file takes time.

For the first data file there are 3 graphs. 1st graph shows the Main Signal with the detected peak marked as red dot. 2nd graph shows the z-scores or detected peak points. 3rd graph shows the threshold amplitude for the whole signal. For the second data file there is another graph showing the cleaned signal without the spike area and interpolated with average data. If the user is experimenting with the data files these four graphs are important to know about the signal.

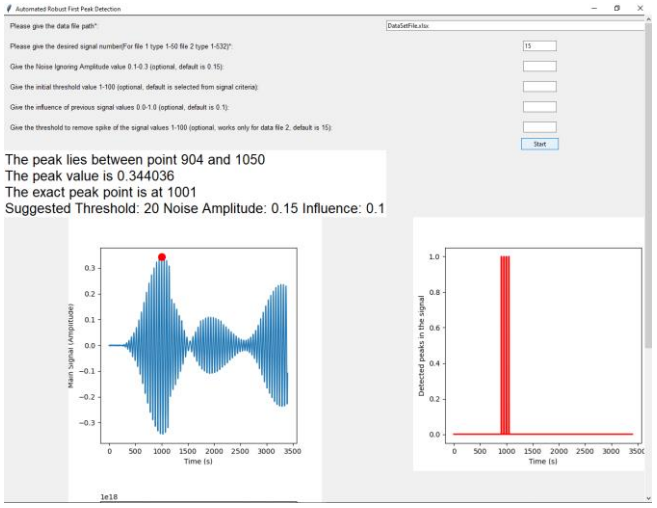


Figure 6: Successful run with first data file's signal no. 15

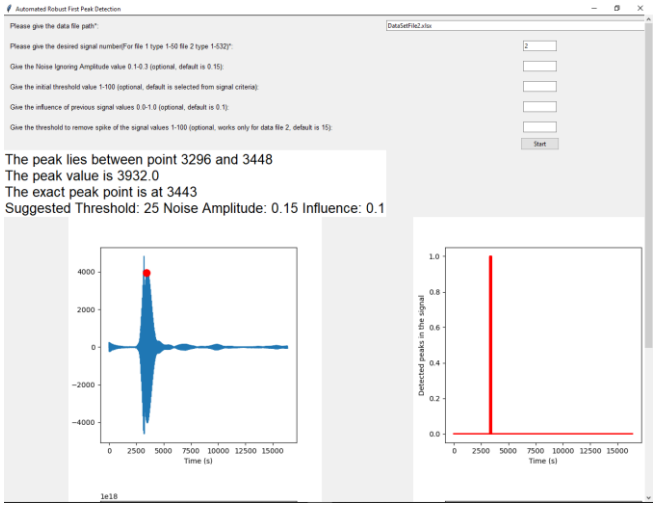


Figure 7: Successful run with second data file's signal no. 2

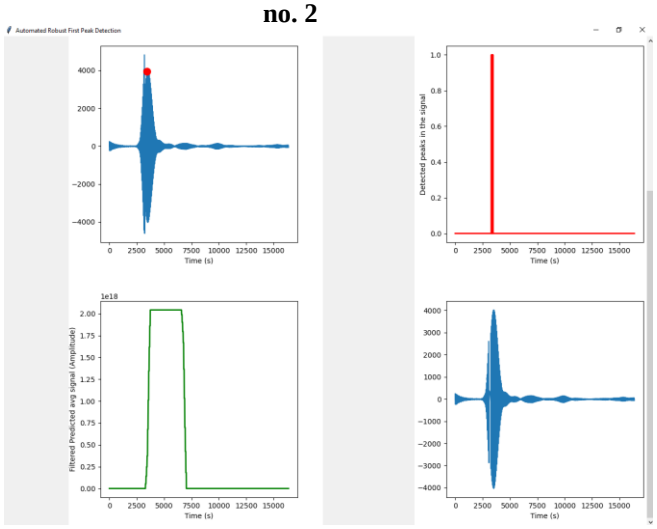


Figure 8: Graphs of successful run with second data file's signal no. 2

V. Results and future modifications

As our goal for this project is to find the first peak in robust automated detection technique, we made a user interface to swiftly operate for each of the signals for both data files. The result of all the signals of first data file matches with the actual peak. There are no sudden spikes in the first data file. For that reason the results of the first data file's signals are accurate.

The second data file is with more than 500 signals. All the signals have different level of spikes. These spikes are not easy to remove from the signal. In some cases the signal spike removal is not possible with the interpolation technique we used and this spike removal has to be improved with future works of this project. We have worked on first two signals and found the required threshold for these are 25 like the first data file's signals. But there are more signals with different shapes and different levels of amplitude. So this file needs to be more experimented in future to find a threshold common for most of the signals. Also, the threshold dictionary object is built for the first data file signals. Similar threshold dictionary should be built for the second data file signals.

Another big experimenting field for the second data file is the threshold of interpolation spike removal. If in future this technique is used to remove the spikes from the signals, threshold for removal portion needs to be experimented for each signals. Also the amplitude magnification needs to be smoothed and experimented to find better multiplying factor and intervals of the amplitudes.

In a sense robust z-score technique for small amount of signals are easy to be implemented and managed. But when it comes to bigger data files with huge amount of signals then going for Convolutional Neural Network or Recurrent Neural Network is more suitable. Here by selecting half of the signals of second data file as pre-determined peaks and throwing the signals' spectrogram in a 32 x 32 or higher frames division at the learning stage of the application can be a greater and faster solution which can bring more accurate result with less computational resources.

Acknowledgment

First, we want to acknowledge Professor Andreas Pech who supported us on this project giving theories and data files. Next, we want to acknowledge Marc Compere for sharing the idea and pseudo code of Z-Score algorithm in Stack Overflow website.

References

Bland, J., & Altman, D. (1996). *Statistics notes: measurement error.*

Carroll, S., & Carroll, D. (2009). *Statistics Made Simple for School Leaders.* Rowman & Littlefield.

Gregersen, E. (2020, 08 21). *Mean mathematics.* Retrieved from Encyclopedia Britannica: https://www.britannica.com/science/mean

Thomas, G. B. (2013). *THOMAS' CALCULUS.* Boston: PEARSON.