Maya Hussein, Farah Awad, Farah Fathy

CSCE-3611

May 2023

Analyzing ECG Signals using Autocorrelation and Identifying Atrial Fibrillation

1 Description of Approach Used

First, we used the FivePointDiff function to read the data from the test file, to compute the derivative of the signal using the 5-point difference equation provided in the slides, and to square the signal's derivative.

Then, we used the smoothing function to smooth the squared derivative signal, using a moving average window of size 31 samples.

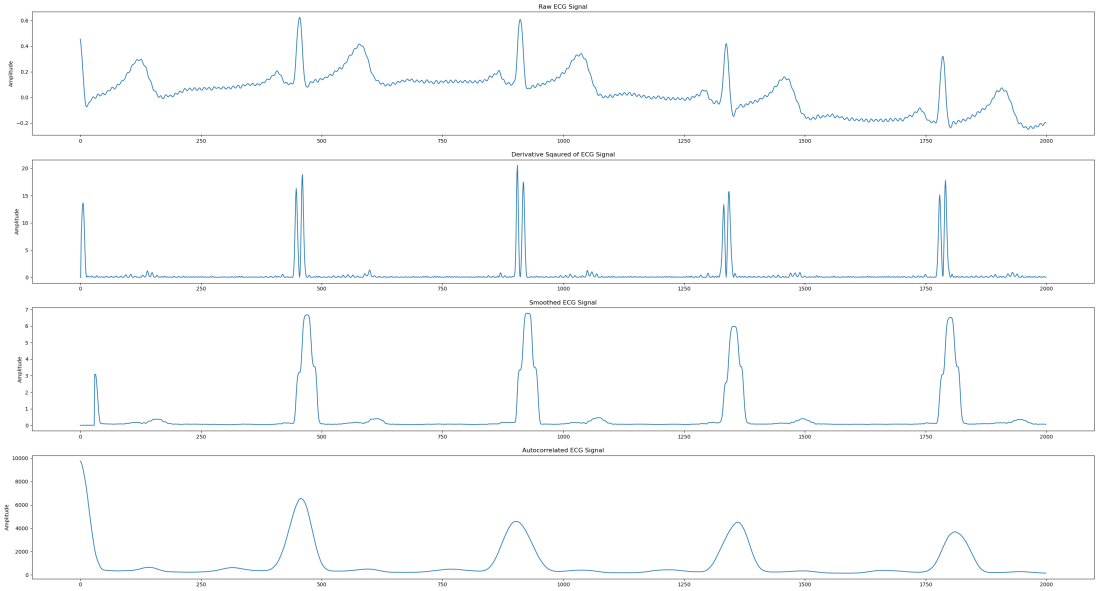
Then, we used the autocorrealation function to compute the autocorrelation of the smoothed signal.

Then, we used the avg_time_between_beats function to find the average time between beats.

Finally, we used the plot_signal_and_autocorrelation function to conduct a figure showing the first 2000 samples of the ECG signal after smoothing and to conduct a plot of the autocorrelation showing the lag on the x-axis and the autocorrelation value on the y-axis.

2 Output of Experiment I

2.1 Plots



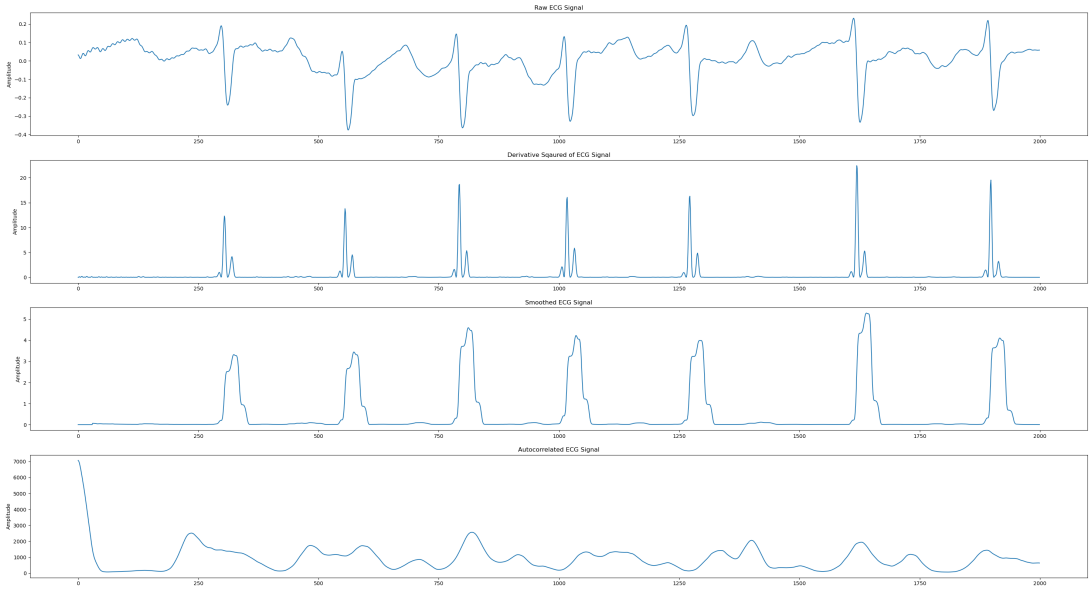
2.2 Results

The Average Time between Beats is 0.87.

The Average Heart rate is 68.46.

3 Output of Experiment II

3.1 Plots



3.2 Results

The Average Time between Beats is 0.418.

The Average Heart rate is 143.55.

4 Discussion and Comparison

4.1 Observation About Autocorrelation

It can be observed that the autocorrelation plot of dataset 2 has higher frequencies and lower peak amplitudes.

It can be noted that increasing the amount of Atrial Fibrillation increases the average frequency of the ECG signal. Thus, we decided that we will use the average frequency as the measure that will be computed from the autocorrelation.

To calculate the average frequency, the beat times between peaks is calculated, then they are converted to frequencies by taking the reciprocal. To elaborate, We use the find_peaks function from scipy.signal to find the peaks in the autocorrelation array A. The height parameter is set to np.percentile(A, 92), which means it finds peaks that are higher than 92% of the values in A. The sample period T is calculated as the reciprocal of the sample rate. The time differences between consecutive peaks are computed using np.diff(peaks) and then multiplied by T to convert them to time durations between beats. The result is stored in the beat_times array. Then, The frequencies are calculated by taking the reciprocal of the beat times, resulting in an array of frequencies. This is done using 1 / beat_times if beat_times is not empty, otherwise an empty array is returned. Finally, The average frequency is computed by taking the mean of the frequencies using np.mean(frequencies).

4.2 Measure

Dataset 1 Average Frequency	Dataset 2 Average Frequency
1.143	6.350