Ruairidh Barlow

12/16/2019

**Fall 2019 Independent Study Progress Report**

**Abstract**

The goals of this project were to develop a proper understanding of the working principles of both electrocardiograms (ECG) and accelerometers, to learn about the unique physiological signals that comprise each time series datasets collected by these two sensors, and to develop programming skills in MATLAB in order to process these two signal types collected by the Sensor Dot wearable sensor. Results from this project included the visualization of the subjects change in BPM over time, a calculated average BPM, and the visualization of the subjects change in respiration rate over time.

**Introduction**

ECGs show the movement of waves of depolarization (positively charged) through the heart. The appearance of the wave depends on the set of electrodes that are being used to detect it. In Figure 1, the lead II set of electrodes are being used to visualize a wave of depolarization. This wave is traced as a positive deflection on the ECG tracing. These electrodes measure the charge on the outside of the heart cell.

Resting cells are polarized, meaning they have a negative charge. When a wave of depolarization passes through them, their charge becomes positive. If the wave of depolarization is feezed halfway while traveling though cells, a dipole is created (Figure 2a). In this case, the dipole vector will point towards the positive electrode. This is traced by an ECG as a positive deflection.

The greater the magnitude of the wave, the greater the spike on the ECG tracing. When the wave of depolarization has passed completely through the heart cells, all cells now have a positive charge. There is no dipole, resulting in a flat line tracing on the ECG (Figure 2b). A wave of repolarization will soon follow. The dipole vector will point towards the negative electrode and will be traced as a downward deflection; the greater the magnitude of the wave, the greater the tracing (Figure 2c). This results in the cells regaining their resting negative charge.

In a standard ECG, there are a total of 10 electrodes that are being used to monitor waves of depolarization. There are 4 limb electrodes on the coronal plane of the human body. They are comprised of the right arm (RA), left arm (LA), left leg (LL), and right leg (RL) electrodes (Figure 3). There are 3 limb leads that detect positive waves and they are only comprised of one electrode (Figure 3a). Additionally, there are 3 bipolar limb leads, that use 2 electrodes (Figure 3b). These make up a total of 6 limb leads.

There are 6 precordial electrodes on the chest, which are V1 – V6. These electrodes are used to make up the chest leads (Figure 4). There are a total of 6 leads on the transverse plane of the human body. It is important to understand that although there are 10 electrodes in a standard ECG, they make up a total of 12 leads (Figure 5).

Typically seen on an ECG tracing is a QRS complex (Figure 6). A QRS complex is usually the main spike. It is comprised of 3 graphical deflections, the Q, R, and S waves. The QRS complex represents a heartbeat, and usually lasts between 0.06 and 0.12 seconds in adults. The QRS complex is best visualized with the II axis (Figure 7). It is important to have a clear and accurate tracing for signal processing. The wearable sensor used for data collection was the Byteflies Sensor Dot (Figure 8). Each capsule contained an array of sensors; however, for the purpose of this experiment, the II axis ECG and tri-axial accelerometer were used.

A tri-axial accelerometer records movement simultaneously in the x-, y-, and z-axes. This type of accelerometer is more accurate than a two-axis accelerometer since it can accurately detect when it is inverted. This sensitivity is crucial when collecting data from an exercising subject. Sensor Dots can be used in tandem to collect multiple types of data at the same time. This data is collected in the form of physiological waveforms. The ECG and accelerometer waveforms, or signals, were processed in MATLAB. First, the signals were filtered. Unwanted segments and frequencies were removed from the signals. Features, reoccurring shapes that give physiological information, were then detected. Detected features were then used to calculate beats per minute (BPM) in 4 different ways. A respiration signal was then derived from the ECG signal and used to calculate the subject’s rate of respiration.

**Materials and Methods**

All programming was done in MATLAB. ECG data collected from the Sensor Dot was in a .CSV format. Scripts listed below were created for this project.

*Data Collection Protocol*

This data collection protocol was adapted from the protocol used in Boyle’s “Automatic Detection of Respiration Rate from Ambulatory Single-Lead ECG” (Boyle et al., 2009). Data collection was performed on September 27, 2019, in the VCU Medical Center’s Metabolic Chamber and lasted for 40 minutes. A Sensor Dot was applied to the subject along the II axis. Data collection began when the subject laid down on a bed for 10 minutes. This was done to establish a baseline for resting heart activity and breathing rate.

Next, the subject sat in a chair for 5 minutes in order to establish a second baseline. There were three exercise sessions done on an exercise bike. Each session lasted 5 minutes and the exercise bike’s resistance was increased each time (0.75 of the subject’s body weight, 1.5 of the subject’s body weight, and 1.9 of the subject’s body weight). Between each exercise session the subject rested in a chair for 5 minutes. This was done to create breaks in the dataset to help differentiate the activities in the collected datasets.

*Signal\_Filtering.m*

This is the main script for processing the ECG signal. First, it imports the ECG signal, stopping at index 286571, to eliminate a data artifact caused by when the sensor fell off near the end of the data collection. Next, the entire accelerometer x-axis signal was imported. Variables were created to store the sampling frequency of the ECG signal (125 Hz), the sampling frequency of the accelerometer signal (50 Hz), and the lengths of both imported signals. The ECG signal was then transformed into the domain of frequency by calling the SpectraPlot function. The output of this function was then used to determine the frequency where the signal drift laid. This frequency was then used to build a high pass Butterworth filter using MATLAB’s built-in function, butter. This function returned a filtered ECG signal that no longer had a drift in it.

The R peaks in the ECG signal were then detected using MATLAB’s built-in function called findpeaks. This function returned the x and y coordinates of detected peaks that were then overlaid onto the filtered signal in order to double-check if all peaks were correctly identified. The BPM of the entire signal was calculated using 4 different methods. Overall signal BPM, an average BPM calculated by a windowed method (overlapping and non-overlapping), and an average BPM calculated by peak frequency. The change of BPM overtime for the overlapping windowed, non-overlapping windowed, and frequency-based calculations were then plotted and compared.

Next, a respiration signal was derived by first creating a new signal that was based upon interpolating the space between the detected R peaks using MATLAB’s built-in function called interp1. This interpolated signal was then down sampled using MATLAB’s built-in resample function in order to match the sampling rate of the accelerometer. The accelerometer signal length was then shortened to account for the data fragment that was initially removed when importing the ECG signal. The interpolated signal was then plotted and overlaid on the x-axis accelerometer signal. It was clear to see that the location of the curves in the interpolated signal lined up with the different activities that were detected by the accelerometer. These coordinates of the different regions of activity in the accelerometer signal were then used to segment the interpolated signal. The interpolated signal had to be segmented in order to allow for proper signal processing. Each region represented a different “amount” of activity, which necessitated different high pass, low pass, and peak detection parameters to properly process each segment. These segments were then processed using the Segment\_Processing function which returned fully processed segments. These segments were then combined back together to recreate the original signal. The now filtered and standardized derived respiration signal was used to calculate and plot the change in the respiration rate over time. This was done by using the resp\_rate function.

*Importfile.m*

This was an auto-generated function made by MATLAB. The purpose of this function was to import a .CSV file. The user could specify the indices of what parts of the files should be imported.

*SpectraPlot.m*

This was provided by Dr. Chen. The purpose of this function was to transform an input signal from the time domain into the frequency domain using the Fourier Transform. This function then output a single-sided amplitude spectrum that was used to determine where low frequencies lie in order to eliminate signal drift.

*Overall\_BPM.m*

The purpose of this function was to calculate and an overall BPM for an ECG signal. This was done by taking the length of an ECG signal, dividing it by a user-defined sample rate, and then converting it into the unit of seconds. Seconds were then converted to the unit of minutes, and finally, the total count of beats were then divided by the minutes to get BPM.

*Buffer.m*

This was the first step for the windowed BPM method. This function took a signal and broke it up into windows of a specified length. This function also allowed for creating both a non-overlapping and overlapping file type, with an overlap that could be specified. This function then called the BPM\_windowed function and passed the window size, the overlap length, the sampling frequency, and a choice variable (whether the buffered dataset contains an overlap or not).

*BPM\_windowed.m*

The purpose of this function was to take the windowed versions of the ECG signal and calculate the BPM of each window. This was done by iterating over each window and for each index, determining if it was an x coordinate of a peak (these x coordinates were from the find peaks function). At the end of each window, a count of detected peaks and the window length was passed into the overall\_BPM function. The calculated BPM for each window was then interpolated for the length of the given window. This process was repeated for each window.

Depending on the choice variable (0 for non-overlapping and 1 for overlapping), one of two things happened. If the current data frame window type was non-overlapping, an average BPM was calculated for the entire signal, and the interpolated BPM signal was returned. If the data frame window type was overlapping, the interpolated BPMs were then corrected to account for the overlap between windows. The BPMs that are in the overlap region were replaced with an average of the BPMs of the two overlapping windows. This corrected BPM data frame was then returned along with an average BPM of the entire signal.

*BPM\_freq.m*

The purpose of this function was to calculate the BPM of a signal based on an R – R interval, or the distance between R peaks. This was done by taking the x coordinates from the findpeaks function and used the diff function to create a vector containing the distance between peaks. These distances were then converted into instantaneous BPMs by first dividing the distance by the sampling frequency and then dividing 60 by the distance. The instantaneous BPMs were then used to recreate the entire length of the original signal by interpolating them for the length of the distance between the peaks that were used in the calculation. An average was then calculated for the entire signal.

*Segment\_Processing.m*

The purpose of this function was to replicate the method that was used in Cysarz’s “Comparison of Respiratory Rates Derived from Heart Rate Variability, ECG Amplitude, and Nasal/Oral Airﬂow” to derive a rate of respiration signal from an ECG signal (Cysarz et al., 2008). This function took in a segmented signal of the already interpolated R-R distance, the starting coordinate of that signal, the sampling frequency of the segmented signal, a high pass filter cutoff, a low pass filter cutoff, peak height, and peak distance.

This function first filtered the given segment using both a high pass and low pass Butterworth filter, in that order. The parameters for these filters would vary depending on the segmented signal that was being processed. Next, the local maxima were detected across the entire signal. Once these peaks were detected at intervals of 250 samples (representing the approximate time between breaths), the entire signal was standardized by dividing the entire signal by the 75th percentile of all local maxima to avoid the influence of single deep breaths. Next, the local maxima were detected in the now standardized signal. This time the distance and the height parameters of the peak detection function differed depending on the segment. Once all peaks were detected their amplitudes were tested to see if they were greater than the 30th percentile of all detected peaks. This was done to ensure that a detected peak was a valid respiration cycle. Finally, this function modified the x coordinates of valid peaks to match the original starting coordinate of the originally imported segment. This function then returned the x- and y-coordinates of valid peaks.

*resp\_rate.m*

The purpose of this function was to calculate the number of breaths the subject took in a given window of time. This respiration rate was calculated using a similar method to that of the overlapping BPM calculation – a sliding window. For each window of a set length, the number of valid peaks was counted. This count was then interpolated for the length of that window. This interpolation was repeated for each window until the original length of the signal was recreated. Regions of the signal that represented an overlap were modified. The number of breaths was an average of the number of breaths of the two windows that were part of the overlap.

**Results**

The original ECG signal had a data artifact at the end, which had to be removed to ensure that the signal could be processed properly. This was done by changing the index end position of what portion of the dataset the signal was imported from, resulting in a shorter signal that allowed for accurate signal processing (Figure 9). Signal drift was eliminated using the Fourier Transformed method in which the signal was transformed from the time domain into the frequency domain. The frequency where the signal drift laid was approximated and was used to create a high pass Butterworth Filter. The resulting filtered signal had a desired flat response which now made it possible to detect R peaks. Peaks were detected using MATLAB’s built-in function called findpeaks. This function detected a total of 4595 peaks within the processed signal (Figure 10). The x-axis coordinates of these peaks were then used to calculate four types of BPMs. Results for these calculations are as follows: Overall: 120.2590, non- overlapping: 120.0784, overlapping: 119.900, frequency: 120.2612. When plotting the interpolated BPM signals for the frequency-based and both overlapping and non-overlapping BPM calculations it was noticed that they have similar overall trends, suggesting that the calculations were done correctly (Figure 11). Despite accurately detecting peaks in the filtered ECG signal, there were areas of the plotted frequency-based BPM that had sudden spikes in BPM (Figure 12).

When overlaying the standardized derived respiration signal over the accelerometer signal, it was noticed that there were similar trends. During sessions of exercise, the accelerometer signal became more erratic, due to the whole body moving. This change in activity was also present in the derived respiration signal. In regions where exercise was occurring, the respiration signal became much denser, representing the increase in the rate of breathing. The plotted respiration rate (Figure 13) had similar trends when compared to the different segments of activity in the accelerometer signal. When plotting the respiration rate and frequency-based BPMs over time, it was observed that there was a similar trend between the rate of breathing and the BPM in the same window of time (Figure 14). Again, if a subject was exercising and their heart rate was increasing, the number of breaths during that time increased as well. When the subject was resting and their heart rate decreased, the number of breaths during that time decreased as well.

**Discussion**

A flow chart outlining the major steps of this project has been added to the appendix to hopefully make the general steps that are taken for signal processing easier to understand (Figure 15). The removed artifact at the end of the ECG signal was most likely caused by the Sensor Dot completely falling off due to the subject’s sweat. This artifact had to be removed because it introduced outliers into the dataset. In order to achieve a flat as possible frequency response from the Butterworth Filter, the cutoff frequency had to be adjusted multiple times. Initially, it was thought the drifts lied at the frequency of 0.04. This cutoff frequency did not yield a desired frequency response. The cutoff threshold had to be changed until a flatter frequency response was attained. 0.2 yielded an acceptable frequency response. The parameters used to detect peaks had to also be set manually and the plotted peaks had to be checked in order to ensure accuracy. The minimum peak distance and height had to be changed multiple times to ensure this. This process was quite tedious (Figure 16).

When plotting the change in BPM over time for each of the three methods (frequency-based, overlapping, and non-overlapping), the plotted data all followed a similar trend (Figure 11). It was clear to see the change in heart rate over time, as well as where it increased due to exercise, and where it decreased when the subject sat in a chair. As mentioned, there were sudden spikes near the end of the signal (second and third round of exercise) (Figure 12). These were caused by the sensor slipping, due to the subject’s sweat. At that time in the data collection, the sensor had not fallen off completely and was pushed down on in an effort to reapply it. The newly collected accelerometer data was not adequate enough to be used in order to study the subject’s respiration rate, which was the same issue that prompted the second round of data collection. This was due to the accelerometer not being able to distinguish different activities that were occurring at the same time, such as breathing and cycling. When the subject was resting on the bed, a breathing pattern was seen. However, when exercising, this pattern was horribly disrupted.

A literature search was conducted in order to find a tested method that would derive the rate of respiration from an ECG signal. The method used in Cysarz et al., 2008 was deemed suitable since it relied on RR intervals (the distance between R peaks) which was something that was already calculated. A new function was created in MATLAB in order to replicate the method that was briefly described in the literature. The resulting derived respiration rate from the ECG signal lined up well with the changes in activity found in the accelerometer signal (Figure 17). When plotting the respiration rate and the accelerometer signal, it was observed that an increase in activity recorded by the accelerometer corresponded with an increase in the number of breaths that were taken by the subject (Figure 13). When plotting the respiration rate and the frequency-based BPMs over time, it was clear to see that during time periods of more strenuous exercise that the subject’s BPM and the number of breaths increased, and during times of resting the subject’s BPM and the number of breaths decreased (Figure 14). The similar trends seen in these plots suggested that the derivation was successful.

Potential experimental errors arose when choosing the values of the cutoff frequency and the minimum peak height and distance. There were many parts of this project that depended on the accuracy of these parameters and they were different for each signal type. It was important to ensure that the flattest frequency response had been achieving and that all peaks in the signal had been successfully identified. If not, this could cause the incorrect calculations of the various BPMs and an incorrect derived respiration signal. There were a couple of ways to check this visually by plotting the peaks onto the original signal and seeing if the overlaid peaks were correct and that nothing was missing, or by plotting the distances between R peaks and checking if there were any outliers. All of these could help determine if there were any issues and which peaks were causing the issue in the filtered signal. Further experimental error could have been caused by the Sensor Dot slipping off of the subject’s skin due to sweat. This could have been avoided by using more tape to hold the sensor in place.

**Next Steps**

The next step for this project would be an attempt to automate this process – specifically, picking thresholds for filters and peak detection parameters. This would be very challenging. Each ECG signal would be different depending on the weight, age, and physical fitness of the subject.

**Works Cited**

* Boyle, J., Bidargaddi, N., Sarela, A., & Karunanithi, M. (2009). Automatic Detection of Respiration Rate From Ambulatory Single-Lead ECG. *IEEE Transactions on Information Technology in Biomedicine,13*(6), 890-896. doi:10.1109/titb.2009.2031239
* Documentation. (n.d.). Retrieved from https://www.mathworks.com/help/
* A Better Way To Learn. (n.d.). Retrieved from <https://www.osmosis.org/prime-ecg-series?utm_source=youtube&utm_medium=video_description&utm_campaign=ecg_basics&utm_content=ecg_basics>
* Cysarz, D., Zerm, R., Bettermann, H., Frühwirth, M., Moser, M., & Kröz, M. (2008). Comparison of Respiratory Rates Derived from Heart Rate Variability, ECG Amplitude, and Nasal/Oral Airflow. Annals of Biomedical Engineering, 36(12), 2085–2094. doi: 10.1007/s10439-008-9580-2

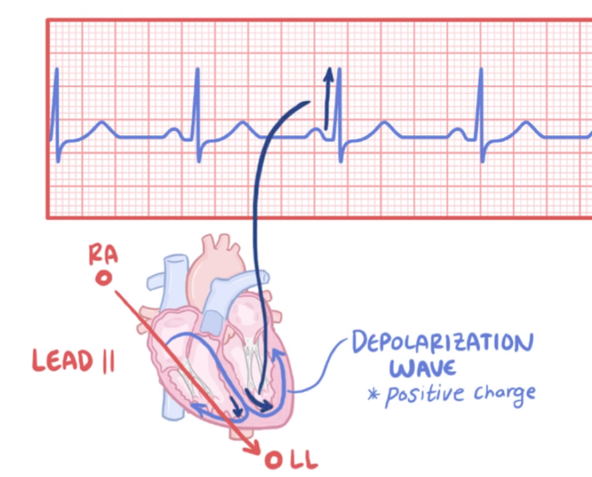
**Appendix**

Figure 1: Movement of wave from right arm (ra) electrode to left leg (ll) electrode you get a positive (upward) deflection

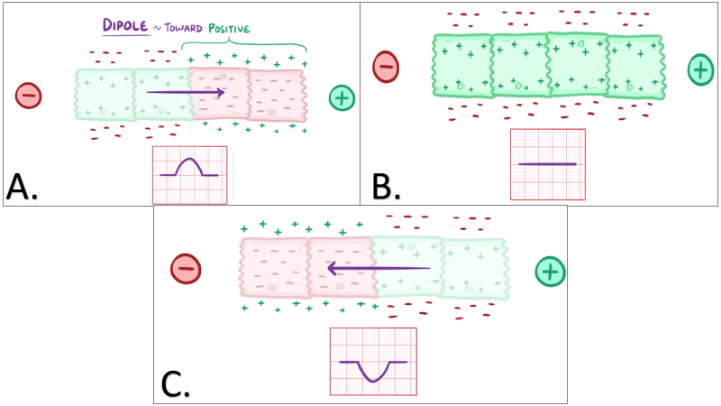


Figure 2 A: Dipole vector (arrow) points towards the positive charge. The ECG tracing shows this as a positive (upward) deflection. B: No dipole resulting in a flat line on ECG tracing. C: Dipole vector (arrow) points towards the negative charge tracing

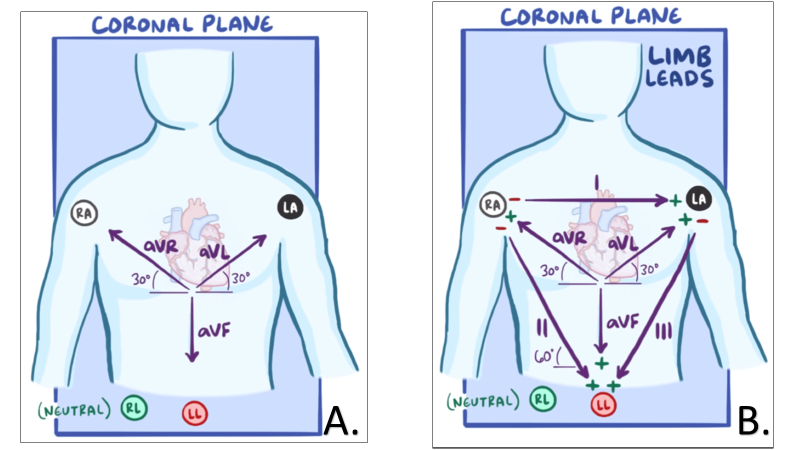


Figure 4: Chest electrodes V1 - V6

Figure 3 A: 4 limb electrodes: RA, LA, LL, RL (right leg neutral). Limb leads aVR, aVL, aVF B: Bipolar limb leads I II, III. All leads detect positive waves and are depicted in purple.

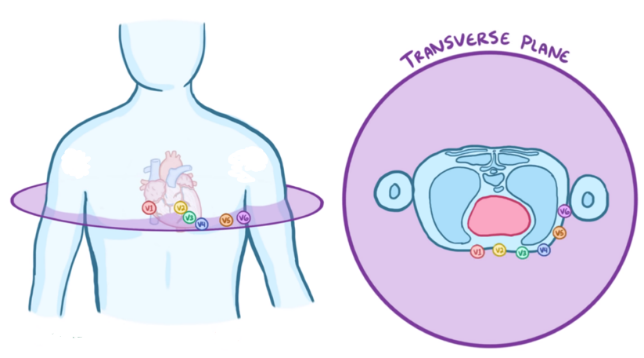
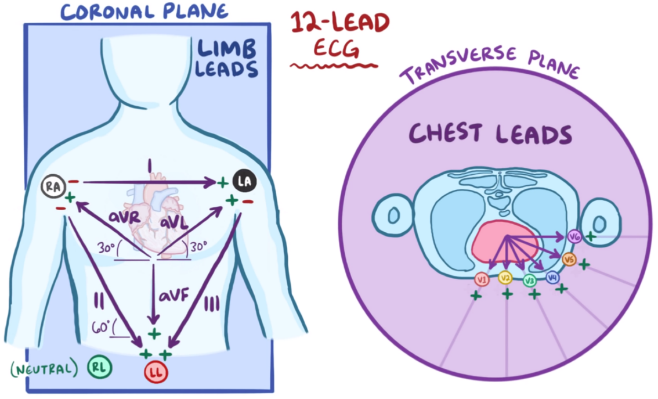


Figure 5: 12 lead ECG. Leads are in purple



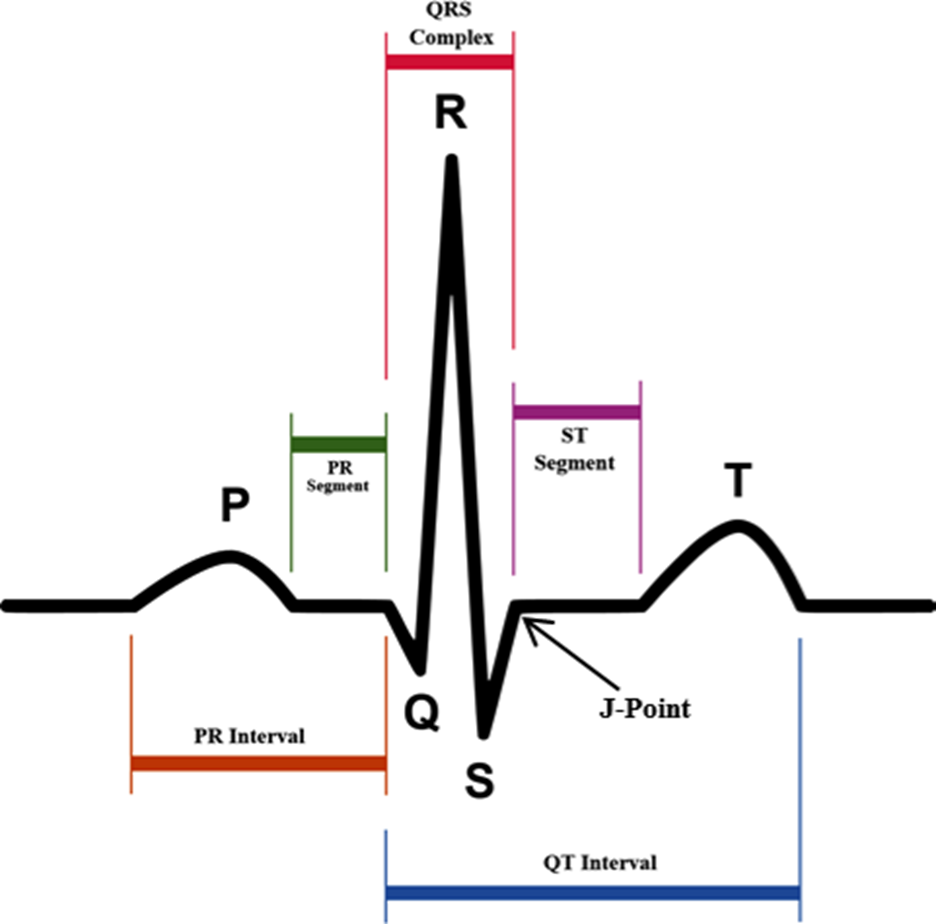


Figure 6: QRS complex, the main spike on an ECG tracing

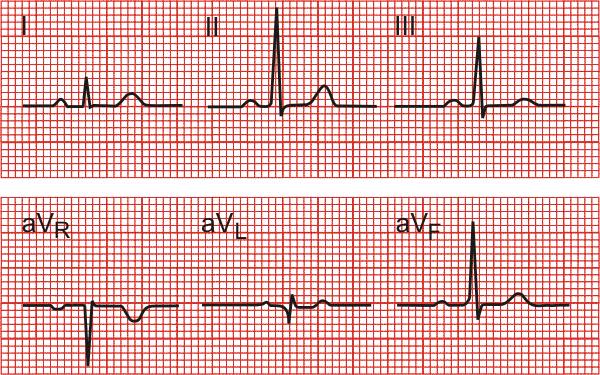
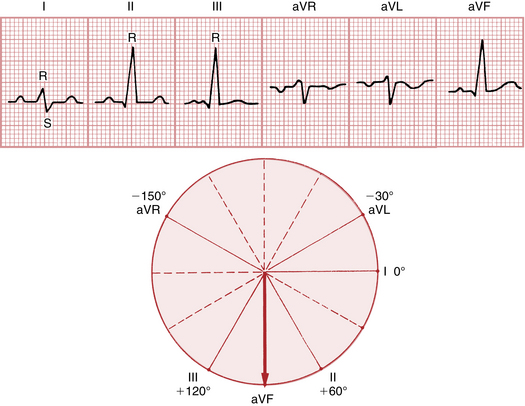


Figure 7: Note the similarity of the waveform when compared to Figure 6. II axis has the clearest QRS tracing

Figure 8: Byteflies Sensor Dot

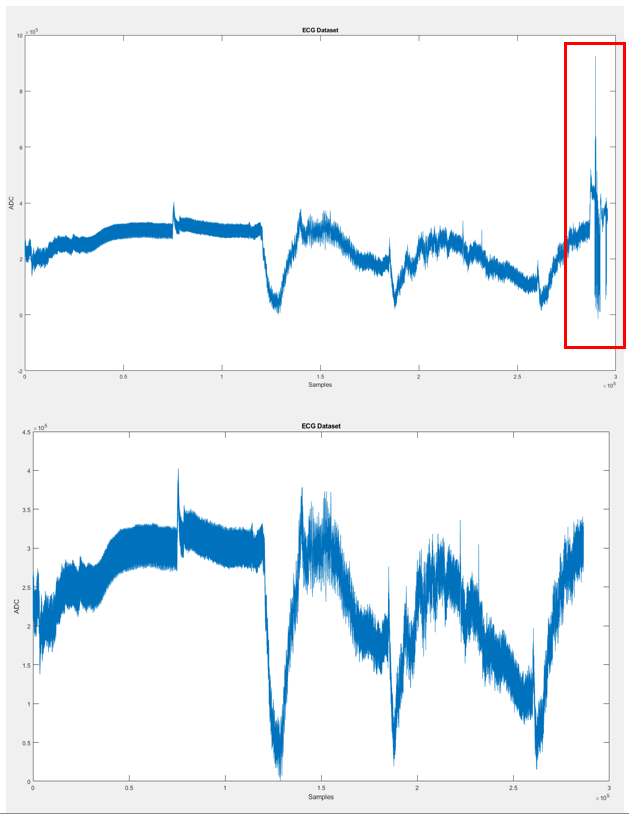
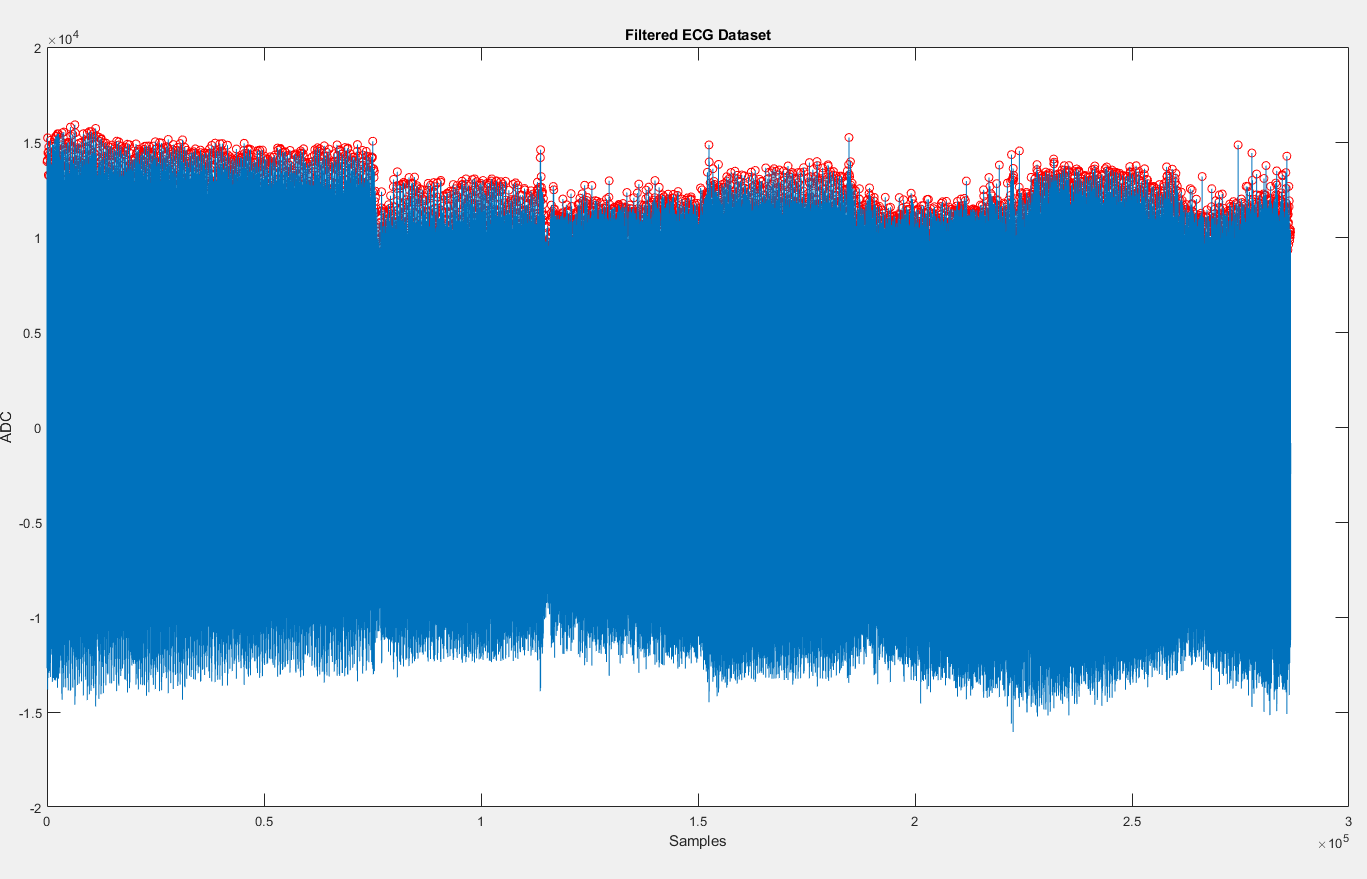


Figure 9: Removal of data artifact (highlighted in red) from the ECG signal

Figure 10: Detected peaks of filtered ECG signal denoted in red

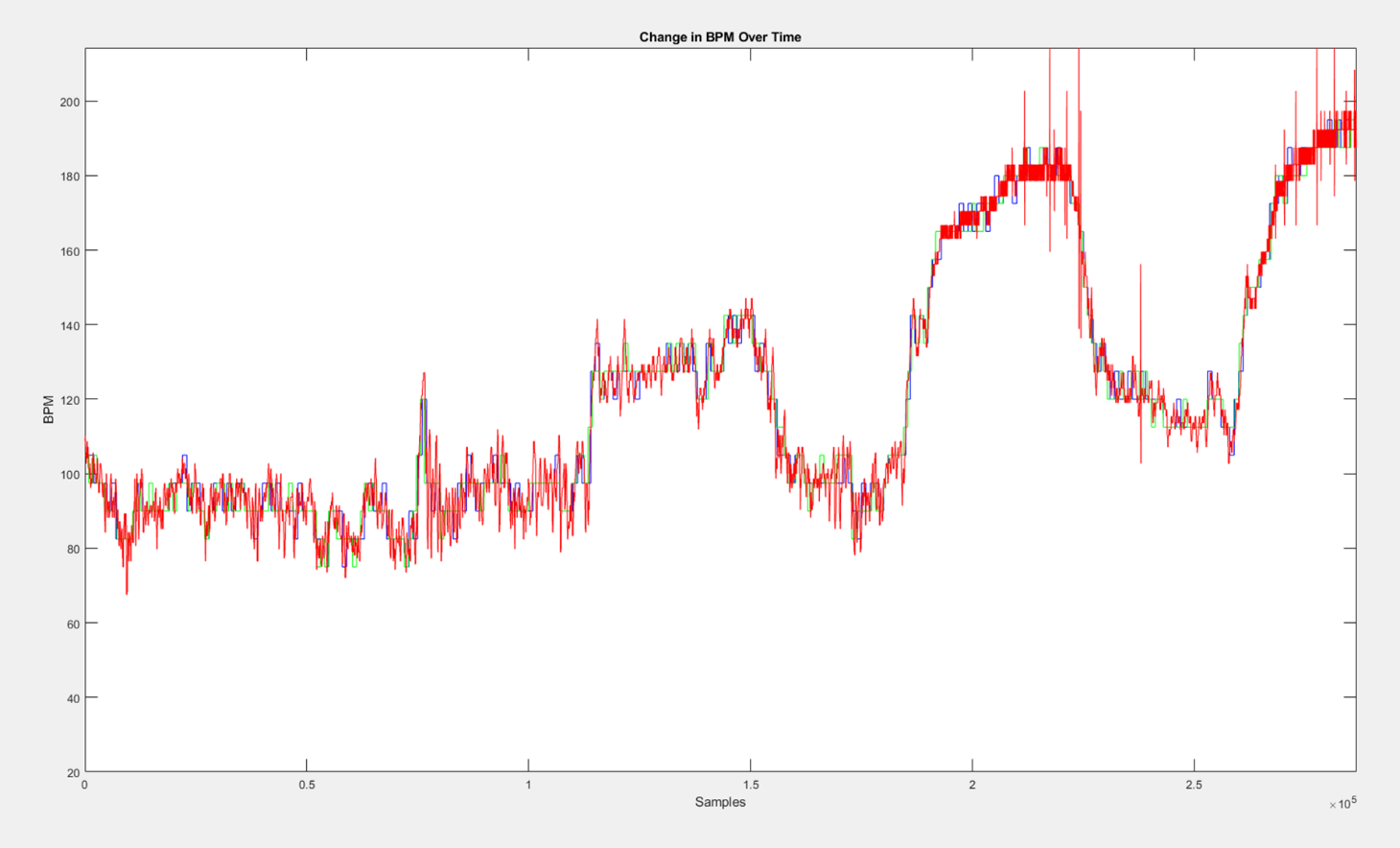


Figure 11: Plotted here is the change in BPM over time. The red line represents the frequency-based BPM method, the blue represents the non-overlapping windowed method, and the green represents the overlapping windowed method. Note the similarity between the three lines.

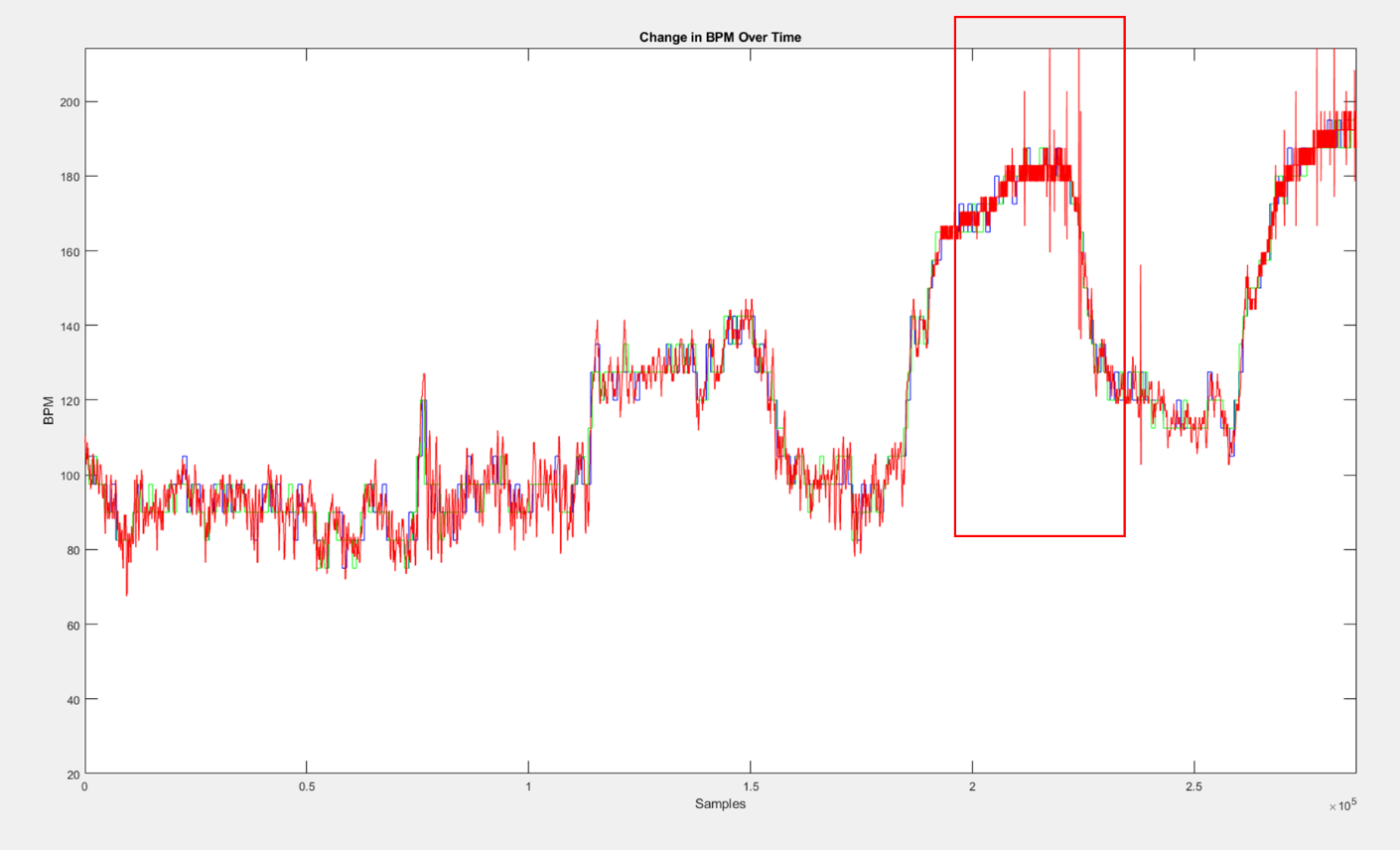


Figure 12: Note spikes in BPM in the red box. These sudden increases and decreases in BPM are likely caused by the Sensor Dot starting to fall off during exercise due to sweat.

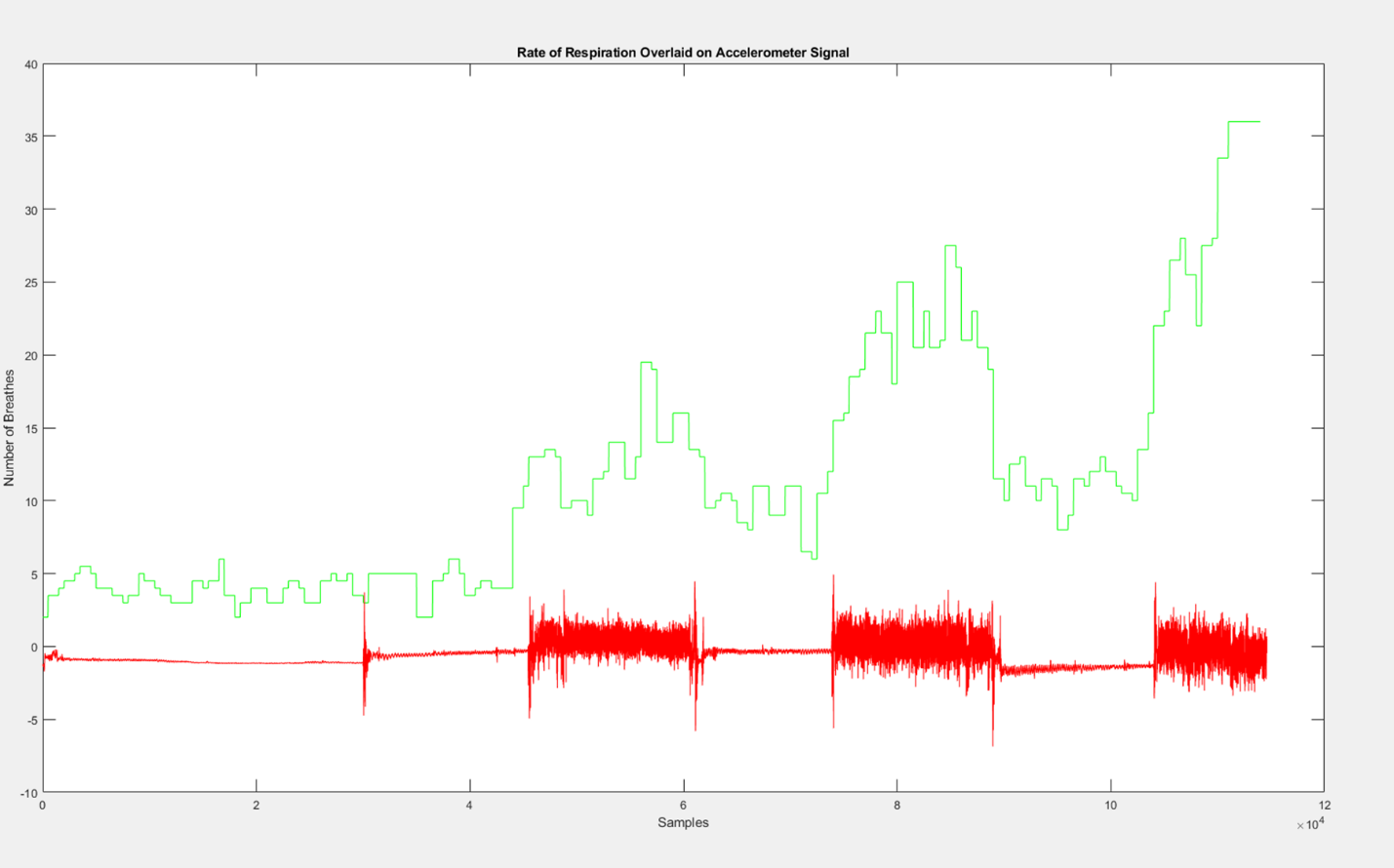


Figure 13: Plotted in green is the subject’s rate of respiration and plotted in red is the accelerometer signal. Note When activity increases the amount of breathes need increases as well.

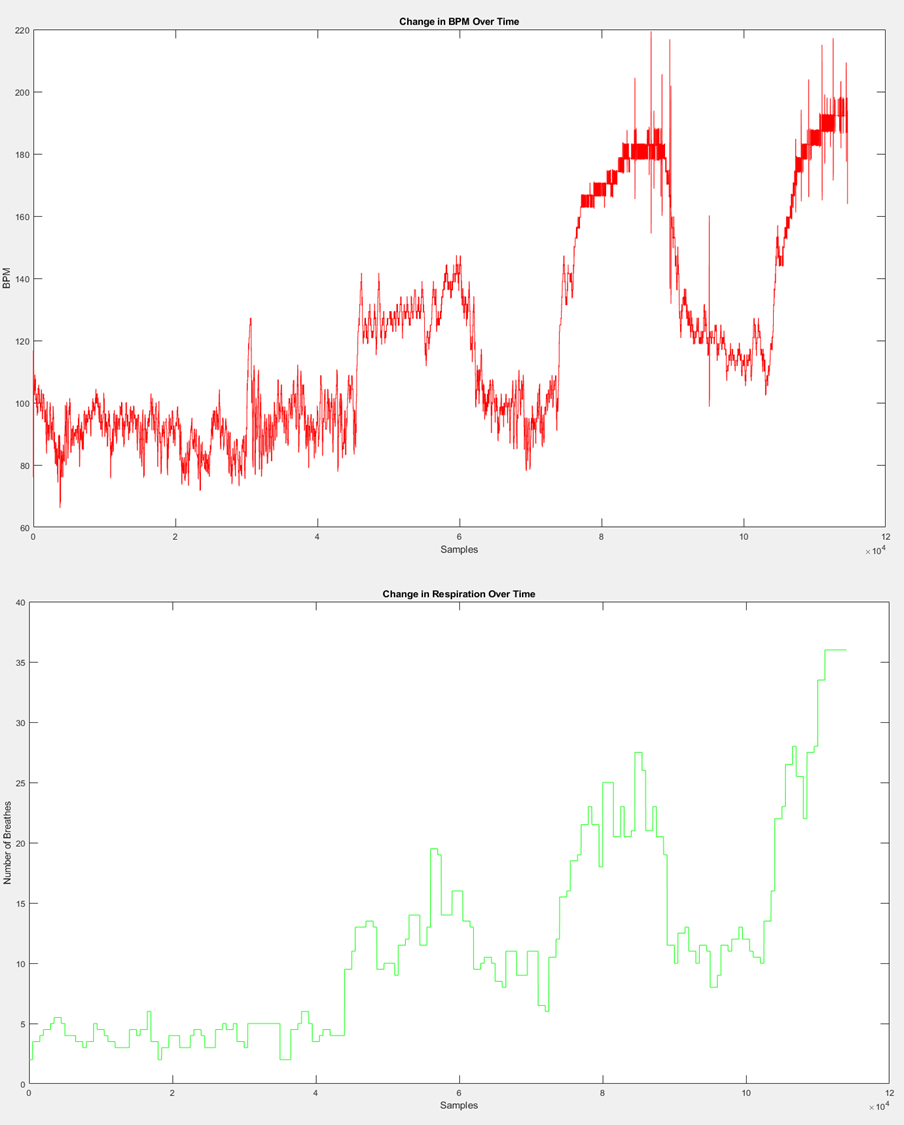


Figure 14: The first plot depicts the change in the subject’s heart rate over time. The second plot depicts the subject’s rate of respiration over time, each measurement was taken in a 30second window. Note when heart rate increases due to exercise the rate of respiration increases as well.

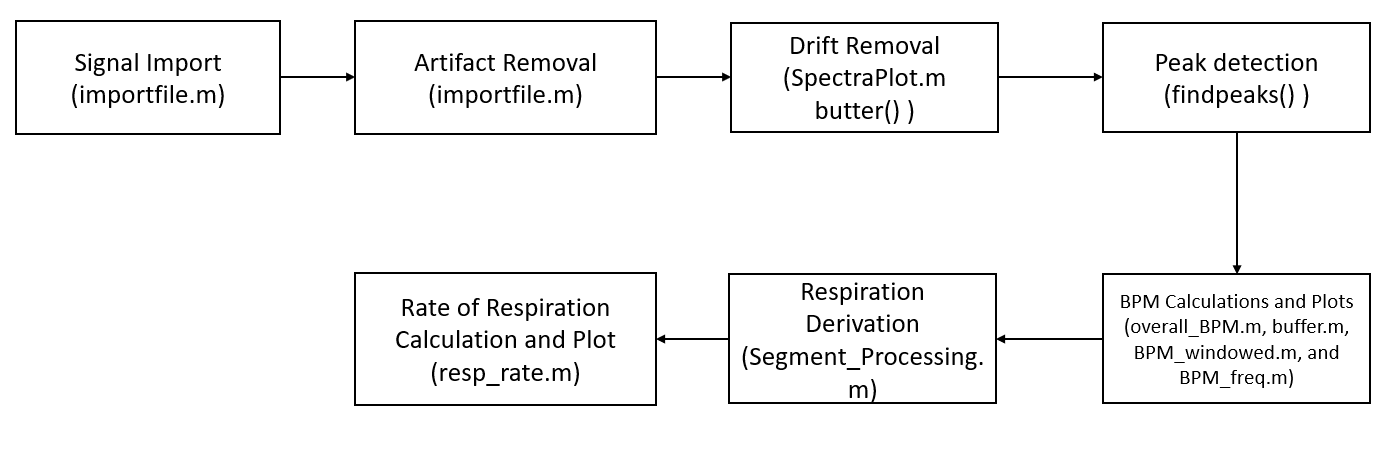


Figure 15: Flow chart of signal processing steps with corresponding functions

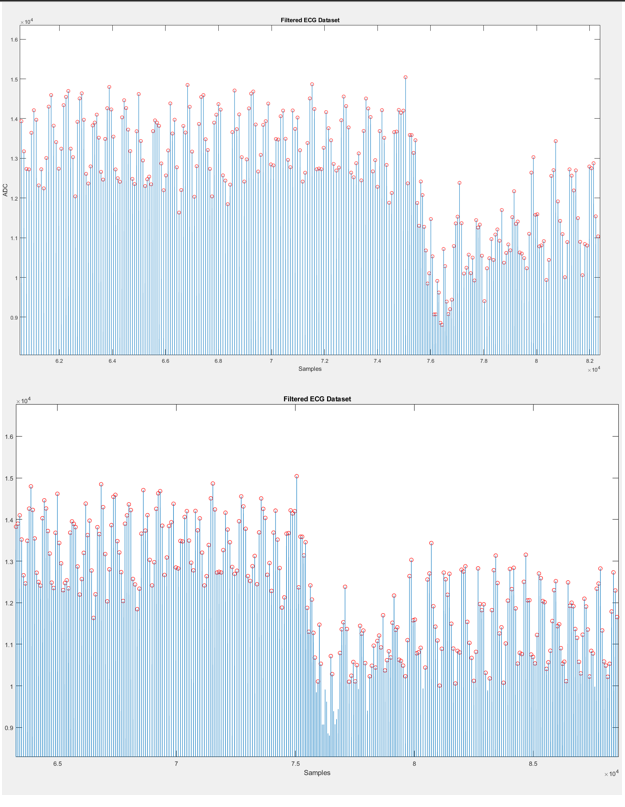


Figure 16: Top plot reflects the first attempt of using the findpeaks function in MATLAB. Note, there are many undetected peaks. Bottom plot reflects adjusted parameters for the find peaksfunction. Note how peaks are properly detected in comparison.

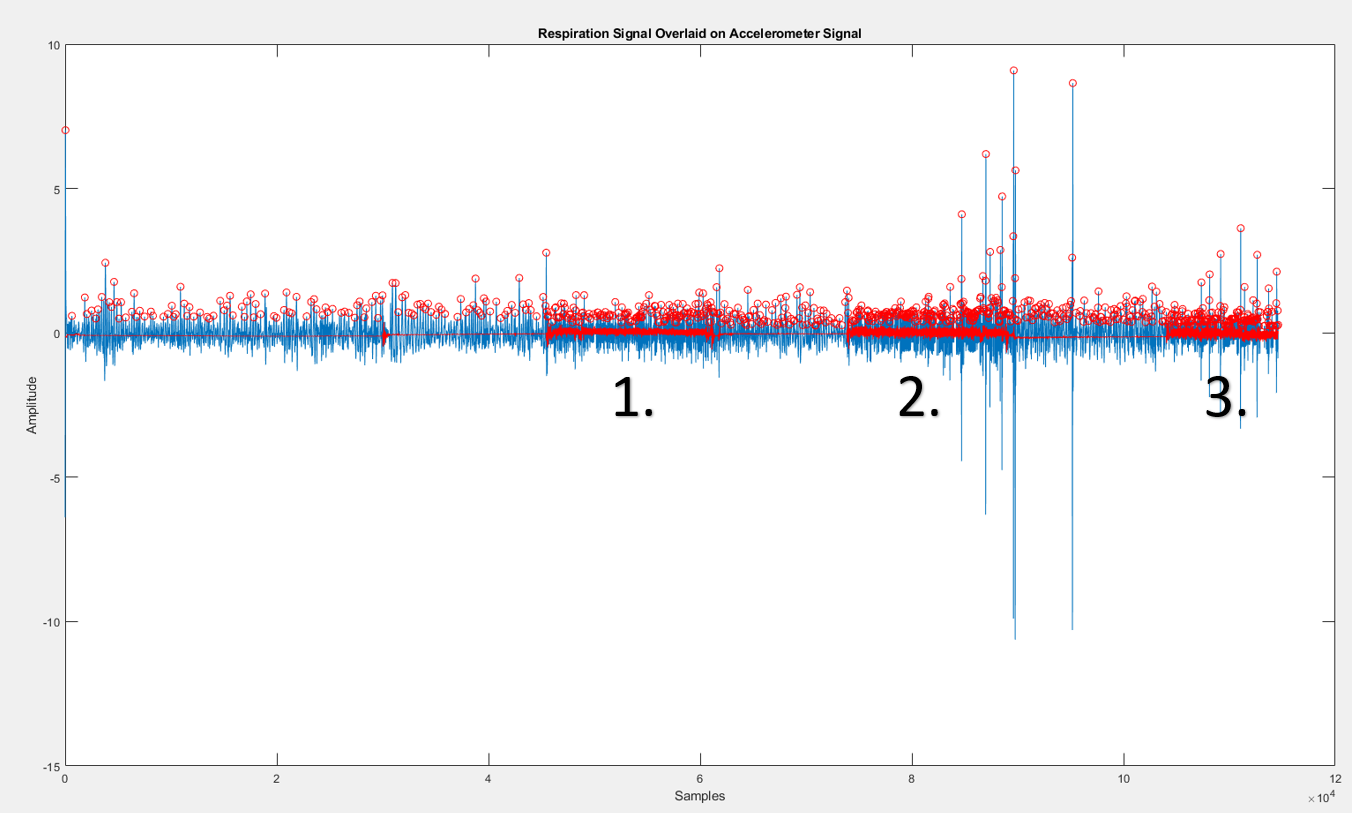


Figure 17: In blue is the derived respiration signal and in red is the accelerometer signal. Note segments 1, 2, and 3. All represent segments when exercise was occurring. Note how both signals in these segments are a lot denser and darker. That represents more data being recorded. That data is movement while exercising (red) and more breathes being need (blue).