Heart Rate Monitoring using Adaptive Noise Cancellation

2015-2016 Q4

Bachelor Thesis

by

Bas Generowicz, 4029542
and
Xenia Wesdijk, 4144074

Supervisors:
R.C. Hendriks
and
S. Khademi

at

Delft University of Technology
15 June 2016

Contents

Abstract

This bachelor thesis proposes a solution on how to reduce motion artifacts in photoplethysmographic (PPG) signals and therefore make the retrieving of the heartbeat more accurate. This can be used to further improve the accuracy of wearable sport tracking devices and smartwatches. The recording of the PPG signals was done with two Infra-Red light emitting diodes and two sensors that measured the changes in the blood volume. Since the measurements were done at the wrist, the PPG signal is corrupted with motion artifacts. In this thesis different techniques are used to filter these motion artifacts out of the PPG signal. The two types of adaptive filters that were tested were the Least Mean Squares and Recursive Least Squares filters. They were tested on 12 different data sets containing data from different test subjects running at various speeds.

The results show that between these two filters the least mean squares filter with spectral peak tracking produces the best results. The system achieves a mean absolute error of 1.66 and has a computation time of 42 seconds over all the data sets. The Sports Motion Artifact Removal Technique (SMART) was presented as a combination of multiple techniques and provides the overall best results with a mean absolute error of 1.42 and a computation time of 37 seconds. Since the measurements of the sport devices needs to be real time, a low computation time is important and SMART provides a robust and fast solution.

Abbreviations

- ANC - Adaptive Noise Cancellation
- BPM - Beats per Minute
- ECG - Electrocardiography
- FFT - Fast Fourier Transform
- GW RLS - Growing Window RLS
- HR - Heart Rate
- Hz - Hertz
- IR - Infrared
- LED - Light Emitting Diode
- LMS - Least Mean Squares
- MA - Motion Artifact
- MAE - Mean Absolute Error
- NLMS - Normalised Least Mean Squares
- PPG - Photoplethysmography
- PSD - Power Spectral Density
- RLS - Recursive Least Squares
- SNR - Signal-to-Noise ratio
- SpO2 - Peripheral capillary oxygen saturation
- SPT - Spectral Peak Tracking
- SSA - Signal Decomposition
- SSR - Sparse Signal reconstruction

1

Introduction

In the past decades, electronic devices have becrme smaller and faster due to the great development in the field of transistor design. This means that it is possible to make more powerful devices and make them smaller. Good examples are smartphones, smartwatches and sport devices. Sport devices are bracelets or watches you can wear, while working out, that keep track of vital signals. These watches use the non-invasive photoplethysmographic (PPG) technique, which is frequently used in clinical investigations for the heart rate, blood oxygen (arterial oxygen saturation SpO2), blood pressure, cardiac output, and respiration measurement [1].

PPG is a low-cost optical technique that can be used to detect changes in the blood volume [2]. Although PPGs are sometimes used in clinical observations, it is difficult to obtain reliable and accurate measurements. This is because PPGs are very sensitive to Motion Artifacts (MA). So only when a subject refrains from motion, during the entire measurement, is it possible to get an accurate heart rate (HR) out of the PPGs. When a subject starts to move the PPG signals get contaminated with MA. It is fairly difficult to filter out these motion artifacts afterwards, because the major frequency components overlap with those in the PPG signal.
The PPG signals are acquired from a pulse oximeter, which uses two Light Emitting Diodes (LEDs) and photo detectors to observe the changes in the light absorption measured at the wrist. Usually infra-red (IR) light is used for this since the blood absorbs this light really well, but the skin does not, making it relatively easy to observe the blood volume changes. The wavelength of the LEDs are 725nm.

Electrocardiography [3] or ECG is the technique used by medical doctors to measure HRs. The device records the electrical activity of the heart over a certain period of time. Even though the alternative technique, PPG, is a lot cheaper than ECG, ECG is still a lot more accurate because motion doesn't have a big influence on the measurement. Besides being low cost, PPG measurements are also a lot more comfortable. With ECG, the subject is hooked on to a machine so that he is

1

restricted in his movement.

In this Bachelor thesis we and our four teammates focus on finding the best technique to retrieve the heartbeat from PPG signals measured at the wrist. The focus will be on finding a method with the smallest Mean Absolute Error (MAE). Since the monitoring is in real time, it is also necessary to keep in mind that the algorithm works fast on real time applications and not just on (fast) PC's.

The Bachelor project is inspired by the signal processing cup. This is a competition for bachelor students in which they try to solve an existing and challenging problem using signal-processing techniques [4]. The databases that were used throughout this thesis are the same as the ones used in [5]. All data sets contained 6 different signals:

- ECG signal (from the subject's chest)
- two PPG signals, PPG1 and PPG2 respectively (recorded at the wrist)
- Acceleration in the X, Y and Z direction (recorded at the wrist)

These signals were measured simultaneously and have a sampling rate of 125Hz while the test subjects were performing different exercises. The specific data sets used for this investigation are the so called $Training\_Data$. The data in this set was recorded while the subjects were running at different speeds. The recording of this data took about 5 minutes and consists of roughly 35.000 samples.
In order to simulate the real time heart rate monitoring windows are used. These windows are 8 seconds long with incremental steps of 2 seconds so that all the windows overlap 6 seconds with the previous window.

The main goal of the whole Bachelor project is to design an algorithm that can estimate the HR, for all windows, using the signals mentioned above. The ECG signal here is used as a reference signal.
A subdivision was made to make the project more approachable. The project was divided into three subgroups of two students. The different topics that were investigated by the subgroups are listed below.

- Adaptive Noise Cancellation (ANC) - subgroup 1
- Spectral Peak Tracking (SPT) and Sparse Signal Reconstruction (SSR) - subgroup 2
- Signal Decomposition (SSA) - subgroup 3

In this thesis the first method, ANC, will be thoroughly discussed. The main goal is to find a way to reduce the motion artifacts (MA) caused by (involuntary) movement. A sub target is to keep in mind the speed of the algorithm.

Figure 1.1 shows two extreme situations that occur with PPG signals [4], in parts 'a' and 'b' the PPG signal has no MA present and from the corresponding spectrum in 'b' the HR can be easily read in the frequency domain. The actual HR is given by the red circle.

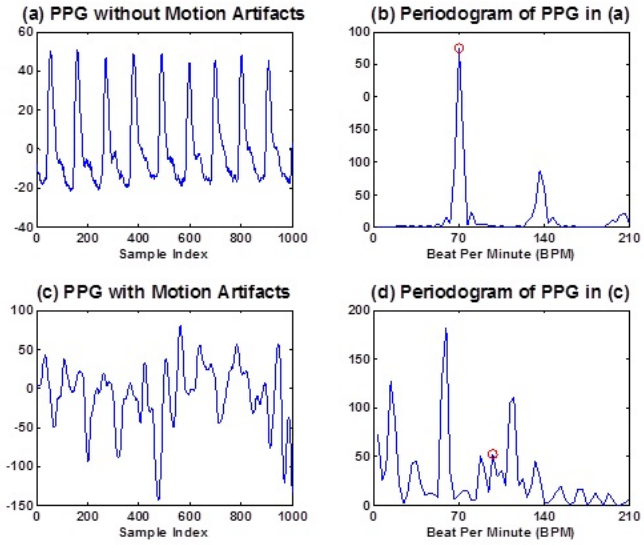

Figure 1.1: PPG signals and their corresponding spectra

Once MA components are present, as is the case in 'c' and 'd', the time dependent signal in 'c' is clearly distorted. In 'd' the actual HR is around 100 Beats Per Minute (BPM), but unlike the previous example there is no prominent peak present at the actual HR. Using a variety of techniques it is still possible to read the correct HR, even in cases like 'c'. These techniques, and the accuracy of them, will be discussed in this thesis.

While many techniques for removing MA have been investigated before, a comparison between these different techniques could prove useful when compared under the same circumstances.

Finally a robust solution will be presented in this thesis called SMART (Sports Motion Artifact Removal Technique), this framework combines the work of each subgroup to achieve robust performance while also optimising computation time.

2

Existing techniques

There are multiple methods designed to measure HR by using PPG signals. The real challenge comes when MAs are introduced that distort the PPG signal. This thesis will compare the different methods of removing the MA components by comparing the MAE.

2.1. TROIKA framework

TROIKA [5] was introduced to offer a robust solution. This framework forms the basis of our research on the subject and the steps are displayed in Figure 2.1.

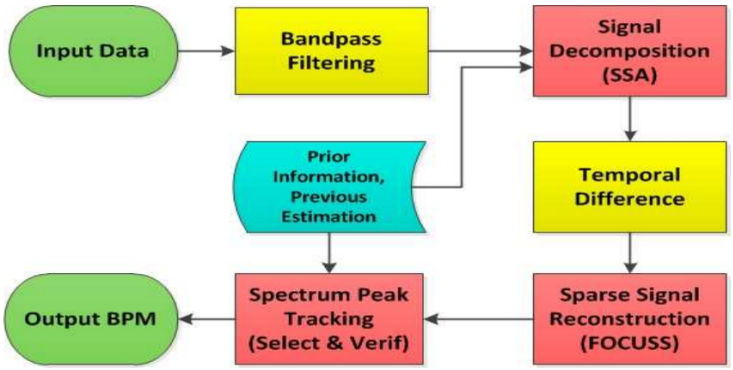

Figure 2.1: TROIKA framework

The three main steps, displayed in red, are signal decomposiTion (SSA), sparse signal RecOnstructIon (SSR) and spectral peaK trAcking (SPT), the capitalised letters forming the name TROIKA. In TROIKA the input PPG signals are first put through a bandpass filter, this is done because the HR can only be within the human

heartbeat range so values that are not feasible are ignored. This range was chosen to be between 30 and 180 BPM because professional athletes can have resting HRs as low as 30 [6].The signal is then delivered to the signal decomposition where MA is removed. After this, temporal difference is done to remove non-periodic noise to prepare the signal for the SSR. SSR in TROIKA is done by using the FOCUSS algorithm [7] and makes the signal from the temporal difference more sparse for easier peak detection. The SPT uses prior information and the current data to best make an estimate of the current HR.

2.2. Adaptive Noise Cancellation (ANC)

ANC will be the main theory discussed in this thesis. This type of filter is proven technology for cancelling noise from a PPG signal [8] [9]. There are 2 types of filters named Fixed filters and Adaptive filters. An adaptive filter is used when there are time-variant input signals (linear filter). The parameters of the filter are ”adaptive” meaning that they change according to the characteristics of the system. This is advantageous as it means that no prior knowledge of the signal or noise is needed [10]. Two types of adaptive filters that are discussed in this report are the Least Mean Squares filter (LMS) and the Recursive Least Squares (RLS) filter.

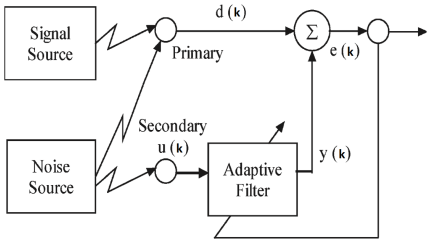

Figure 2.2: ANC Diagram [11]

The basic functionality of an adaptive filter can be seen in Figure 2.2. The signal d_k contains the desired signal and noise at point k, given as s_k and n_k respectively in Equation 2.1.

$$d_k = s_k + n_k \tag{2.1}$$

The secondary input u_k contains noise that is not exactly the same noise as in d_k or it could just be subtracted but is instead correlated to the noise of d_k and hopefully not correlated to the desired signal in d_k. u_k is used to estimate the noise called $\hat{n}_k$ after which the desired signal is estimated by subtracting the estimated noise from d_k as seen in Equation 2.2.

$$\hat{s}_k = d_k - \hat{n}_k \tag{2.2}$$

Equations 2.1 and 2.2 are combined to give:

$$\hat{s}_k = s_k + n_k - \hat{n}_k \tag{2.3}$$

Next Equation 2.3 is squared and the mean is taken giving:

$$E[\hat{s}_k^2] = E[s_k^2] + E[(n_k - \hat{n}_k)^2] + 2E[s_k(n_k - \hat{n}_k)] \tag{2.4}$$

The term $2E[s_k(n_k - \hat{n}_k)]$ is equal to zero since s_k is uncorrelated to both n_k and $\hat{n}_k$, which leaves:

$$E[\hat{s}_k^2] = E[s_k^2] + E[(n_k - \hat{n}_k)^2] \tag{2.5}$$

By trying to minimise each side of the equation the signal to noise ratio is maximised, since $E[s_k^2]$ is not variable (it is simply the desired signal) the equation becomes the following:

$$minE[\hat{s}_k^2] = E[s_k^2] + minE[(n_k - \hat{n}_k)^2] \tag{2.6}$$

From this it can be concluded that minimising the total power at the output the maximum signal to noise ratio is achieved [12].

2.2.1. Least Mean Squares (LMS) - filter

The LMS algorithm is defined as follows [13]:

$$y_k = \mathbf{w}_{k-1}^T \mathbf{u}_k \tag{2.7}$$

$$e_k = d_k - y_k \tag{2.8}$$

$$\mathbf{w}_k = \mathbf{w}_{k-1} + f(\mathbf{u}_k, e_k, \mu) \tag{2.9}$$

Using Figure 2.2, the variables are taken as follows: u_k is the acceleration data, w_k is the filter weight vector, d_k is the signal that contains the PPG with MA and μ is the step size. The error e_k is estimated by using the tap weight vector and the MA signal and subtracting this from the noisy input signal. The tap weights are then updated for the next iteration.

There are different methods of calculating the LMS, where the $f(\mathbf{u}_k, e_k, \mu)$ is different. These different algorithms are as follows:

LMS:

$$f(\mathbf{u}_k, e_k, \mu) = \mu e_k \mathbf{u}_k^* \tag{2.10}$$

Normalised LMS:

$$f(\mathbf{u}_k, e_k, \mu) = \mu e_k \frac{\mathbf{u}_k^*}{\epsilon + \mathbf{u}_k^H \mathbf{u}_k} \tag{2.11}$$

Sign-Error LMS:

$$f(\mathbf{u}_k, e_k, \mu) = \mu sign(e_k) \mathbf{u}_k^* \tag{2.12}$$

The normalised LMS algorithm is potentially faster at converging than the standard LMS filter [14] because it can use a time-varying step size [15], so the choice of which type to use has to be made by looking at which algorithm achieves the best results. The sign LMS is simplified implementation of LMS which is easy to implement on hardware such as a Field Programmable Gate Array (FPGA).

2.2.2. Recursive Least Squares (RLS) - filter

The idea behind the RLS filter is to minimise the cost function, also known as error function, by appropriately selecting the filter coefficients w_k, updating the filter as new data arrives.

The RLS algorithm can be defined as [16]:

$$\mathbf{k}_k = \frac{\lambda^{-}1 P_{k-1}\mathbf{u}_k}{1 + \lambda^{-}1\mathbf{u}^T P_{k-1}\mathbf{u}_k} \tag{2.13}$$

$$e_k = d_k - \mathbf{u}_k^T \mathbf{w}_{k-1} \tag{2.14}$$

$$\mathbf{w}_k = \mathbf{w}_{k-1} + \mathbf{k}_k e_k \tag{2.15}$$

$$P_k = \lambda^{-}1 P_{k-1} - \lambda \mathbf{k}_k \mathbf{u}_k^T P_{k-1} \tag{2.16}$$

Just as with the LMS filter u_k and d_k represent the acceleration data and the desired signal, which is the PPG signal including the MA, respectively. In this case the error (e_k) is measured using the filter coefficient w_k. The filter coefficient is initially set to zero. Furthermore, P_k is the inverse of the correlation matrix and k_k is the Kalman gain of the RLS algorithm.

The RLS algorithm, like the LMS algorithm, is a mean square error reduction algorithm. Since the RLS is a lot like the LMS it operates practically in the same way, except that the scaling factor of the LMS (also known as the step-size), is better known as the Kalman gain of the RLS algorithm. Supposedly, an RLS based filter implementation can obtain a smaller error signal. However, this smaller error is thanks to a high computational complexity which results in a longer execution time. Running the RLS algorithm also requires more process power consumption. This last aspect is important to keep in mind when designing something that should be used in real time application.

Like the LMS filter, the RLS filter has two important parameters as well. The first parameter is again the length or order of the filter and the second one is the forgetting factor of the filter. The forgetting factor is usually between 0.98 and 1 λ. The smaller the forgetting factor, the smaller the contribution of previous samples will be. This makes the filter more sensitive to recent samples, which means more fluctuations in the filter co-efficients.

There are different variations on the conventional RLS algorithm. A special form of the conventional RLS algorithm is called the growing window RLS algorithm. It works in the same way as the conventional RLS does, except that λ is fixed and equal to 1 for this particular filter.

2.3. Performance

The performance of the system will be evaluated using the MAE [5] given by the following equation.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} |BPM_{estimated}(i) - BPM_{true}(i)| \tag{2.17}$$

This compares the estimated BPM to the actual BPM that is found by using an ECG reference signal.

3

Programme of requirements

Currently there are many wearable devices that can keep track of ones health. One of the features most of them have is the monitoring of the heart rate, which is typically done by using photoplethysmographic (PPG) signals that are measured with a watch on the wrist. The problem is that the measurements of the heart rate can be distorted by the movement of the watch when the wearer is doing physical exercise. This makes monitoring of the heart rate much harder and consequently less accurate. This is why there is need for an algorithm that can overcome these problems. This system will be marketed to businesses that produce wearable devices that have a heart monitoring function.

The requirements that are set for the system are based on given data sets. These data sets consist of a subject doing running exercises. The duration of the measurements of each data set is around five minutes. There are 12 data sets in total.

3.1. Functional requirements

[1.1] The system must give a heart rate in BPM of the wearer.
[1.2] The system must be able to be implemented in a wearable device that can be used during exercise.
[1.3] The system must give accurate heart rate values during rest and exercise.

3.2. System requirements

[2.1] The average absolute error on each data set must be below 10 BPM.
[2.2] The average absolute error over all data sets must be below 5 BPM.
[2.3] The system must be able to calculate the heart rate within 2 seconds on a wearable device.

3.3. Development of manufacturing methodologies

[3.1] The algorithm is developed in Matlab, which has a rich set of toolboxes and functions that allow for easy implementation of the system.

3.4. Business strategies, marketing and sales opportunities

[4.1] The system will be marketed in a business to business model to smart-watch manufacturers.

[4.2] Licensing will be used to make profit on the system.

4

Methods and techniques

Before working on retrieving a heartbeat from the PPG signal, tests were performed on the ECG-signal. ECG signals are influenced a lot less by motion artifacts and has much better defined heart rate peaks, therefore it can be used as a reference to find the actual heart rate.

4.1. ECG signal

When looking at an ECG signal, there are positive and negative peaks. In most cases it is simple enough to find the positive peaks like is the case in Figure 4.1, however sometimes those peaks are less easy to identify. An example of this can be seen in Figure 4.2, here the positive peaks are less explicit which makes it more difficult to detect the correct HR. The second graph of this figure illustrates this. Doing a direct Fourier transform on the non acceptable signal leads to detecting the highest peak which has a BPM of around 172. The real BPM however is on the second highest peak and has a BPM of 135.

4.1.1. Splitting the ECG signal

In order to find the correct BPM and thus make all the peaks easy to detect, even in the case of Figure 4.2, an idea was to split the ECG signal into two separate signals, one being the positive values and the other being the negative values. For ease of use the absolute value of the negative part was taken. The $findpeaks$ function in Matlab was then used on both parts to obtain the peaks in the time domain, see also (a) and (c) in Figure 4.3 and Figure 4.4.

As mentioned the Matlab $findpeaks$ function was used to detect the peaks. In order to make sure the function detects all the peaks, even when the signal looks distorted, a few parameters were added. The $MinPeakHeight$, which simply only makes peaks above a certain amplitude detectable, was set to an amplitude of 50. Another used parameter is $MinPeakDistance$ and was set to 30, this makes sure that smaller peaks next to the highest peak are ignored. 30 was chosen

because there were no data sets in which the peaks correlated to the heartbeat appeared in a closer proximity. Then $MaxPeakWidth$ was used, this could be kept relatively small, 10, since the peaks of the ECG signal are steep. Lastly, the $MinPeakProminence$ was used, this measures how much a peak stands out due to intrinsic height and its location relative to other peaks.

In 4.3 (c) it can be seen that by using this combination of parameters not all peaks are detected. It looks like the peaks could have been detected easily by only setting the parameter $MinPeakHeight$ to 150, however this would mean that a lot of other signals would not have been able to detect their peaks since some are below 150. So the current trade off was made to ensure the best results for all data sets

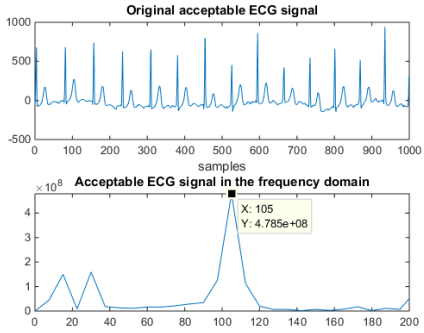

Figure 4.1: First graph: Original ECG signal.
Second: Fourier transform of the ECG signal above.

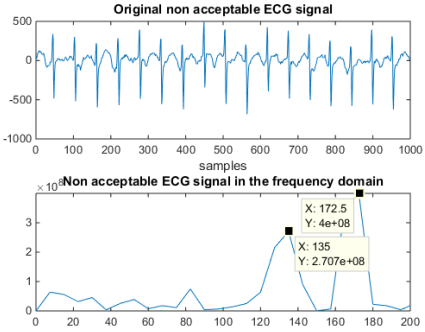

Figure 4.2: First graph: Non acceptable original ECG signal.
Second: Fourier transform of the ECG signal above.

Having two signals means that a choice has to be made on which signal to use. For this the function *kurtosis* is used. Kurtosis is the degree of peakedness of a

distribution [17]. This means that the steeper the peaks of a signal, the higher the kurtosis will be. In this situation it is desirable to have a higher kurtosis, because in a signal with a higher kurtosis the peaks can be found more easily. So the signal (positive or negative) with the highest kurtosis will be denoted as the 'best' signal. This so called best signal will then undergo a transformation to make the signal more 'clean'. Then this clean signal is simulated by giving a value of 1 to all locations where a peak was found and setting all other values to 0. This is done for both signals in (b) and (d) of Figure 4.3 and Figure 4.4.

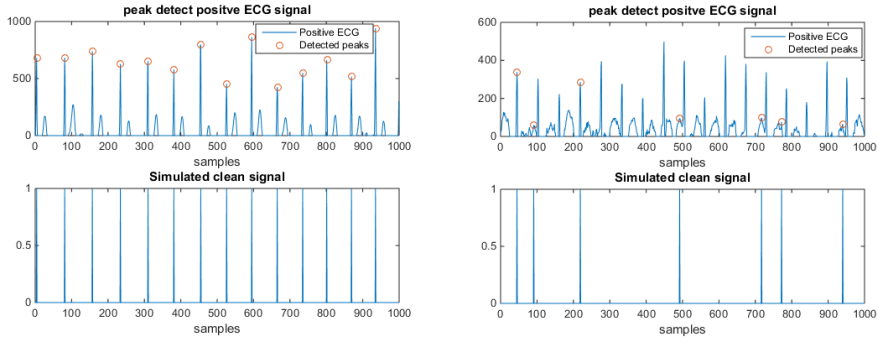

Figure 4.3: (a): Positive part of the original ECG signal in 4.1.
(b): The peaks from (a) have been assigned a 1, the rest of the signal has been made 0, which results in a clean signal.
(c): Positive part of the ECG signal in 4.2.
(d): The peaks from (c) have been assigned a 1, the rest of the signal has been made 0, which results in a clean signal.

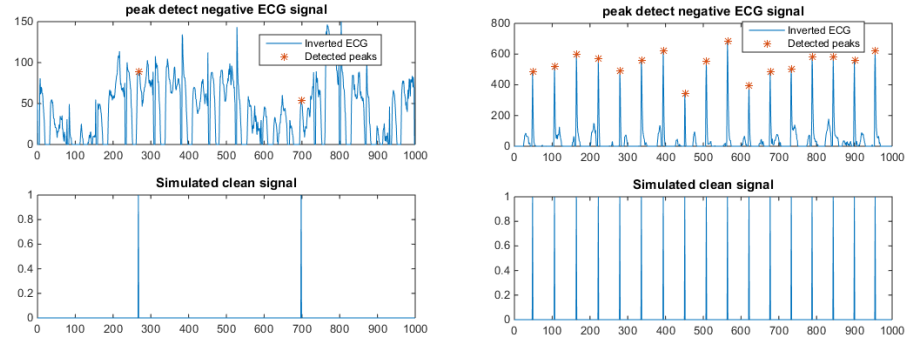

Figure 4.4: (a): Absolute value of the negative part of the original ECG signal in 4.1.
(b): The peaks from (a) have been assigned a 1, the rest of the signal has been made 0, which results in a clean signal.
(c): Absolute value of the negative part of the ECG signal in 4.2.
(d): The peaks from (c) have been assigned a 1, the rest of the signal has been made 0, which results in a clean signal.

Then the clean signal, the one previously marked with the highest kurtosis, under goes the fast Fourier transform (FFT) which results in Figure 4.5 for the non acceptable signal in 4.2.The highest peak in the Fourier domain represents the heartbeat at a certain moment in time. When comparing Figure 4.2 and Figure 4.5 it shows that doing a peak detect over the signal with the highest kurtosis and then taking the Fourier transform results in finding the correct heartbeat. Whereas just doing a Fourier transform over the original ECG results in a highest peak that is not equal to the correct BPM. In the case of 4.1 it turns out that the splitting of the signal is not necessary, because Figure 4.6 shows the exact same peak as the one already found in the bottom graph of Figure 4.1.

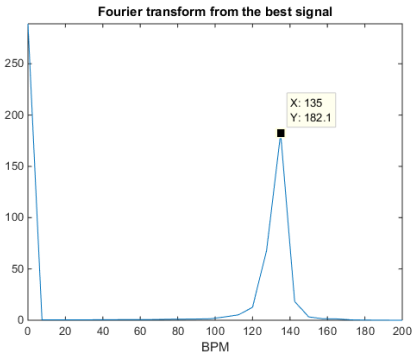

Figure 4.5: Fourier Transform over the signal with the highest kurtosis ((d) of Figure 4.4).

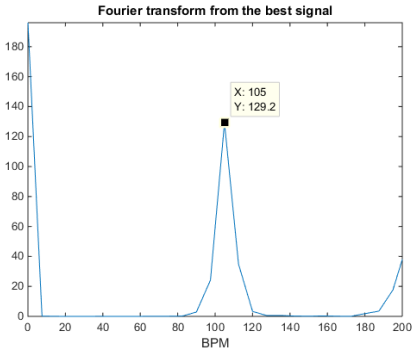

Figure 4.6: Fourier Transform over the signal with the highest kurtosis ((b) of Figure 4.3).

From this it can be concluded that this technique looks promising for finding the right heartbeat. However, this is only works well for the ECG signal and not the PPG signals. A reason for this are that the PPG signals are distorted with a lot of MA, but also because the ECG signal is more peaked than the PPG signal, which makes it more difficult to easily detect the peaks in a certain window.

4.1.2. Slope detection

Another technique that was investigated was to use the slope of the signal. This is an interesting and intuitive method of finding the HR. The cardiac cycle consists of two phases, diastole and systole. Diastole is when the heart relaxes and fills with blood whereas systole is when the heart contracts. Within a window of data the slopes of these two phases are recognisable and can be extracted from the data.

The slope is given by:

$$m(i) = \frac{y(i+1) - y(i)}{x(i+1) - x(i)} \tag{4.1}$$

Where $y(i)$ is the amplitude of the signal at point i and $x(i)$ is the sample at point i. This slope is evaluated between every successive sample, making the denominator always equal to one. This can be done using Equation 4.1.

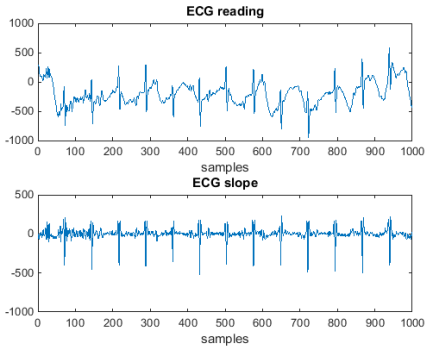

Figure 4.7: ECG reading and ECG slope

From Figure 4.7 it can be seen that the peaks are a lot more defined than in the original ECG reading while the parts in between the peaks are filtered out because the difference between 2 following points is small. It is easy to determine the heart rate without even having to do a Fourier transform using the ECG slope.

4.2. Implementing Adaptive Filters

Before testing the LMS and RLS filter on real gathered data, tests were performed to simulate acceleration data and periodic HR. For the HR (clean signal in Figure 4.8) a simple periodic signal was taken since within such a small window the HR is usually stable. The MA were simulated by adding randomly distributed noise and a combination of different periodic sine waves, including one that overlaps the frequency of the HR. The results are easiest to interpret in the frequency domain displayed in Figure 4.9.

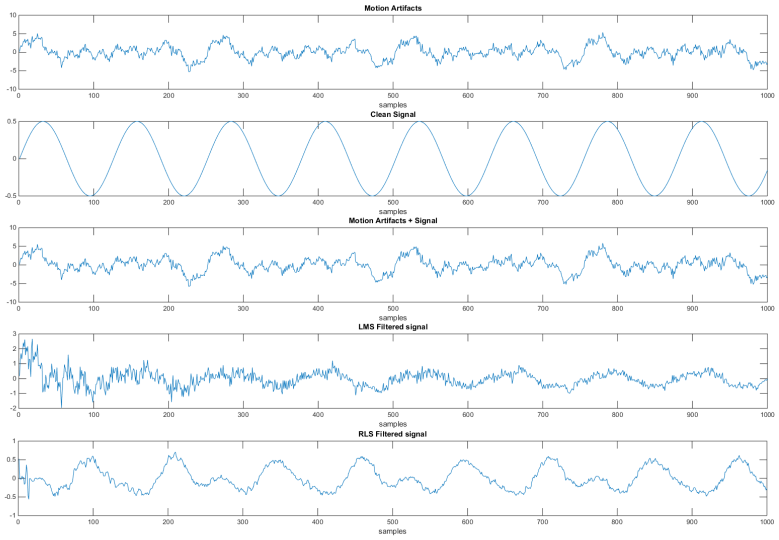

Figure 4.8: Testing the LMS & RLS filter in time domain

When the measured accelerometer data was analysed, similar periodic signals appeared so the simulated MA are realistic. One of the problems is that the frequency of the MA overlap with the frequency of the HR. If a simple notch filter was to be implemented to remove frequencies around the MA components, you could also remove the HR peak if they have a similar frequency. As can be seen in Figure 4.9, using an LMS filter even when there are overlapping frequencies, the heart rate peak is not filtered out and the filtered signal is close to the wanted clean signal.

This filter is applied to PPG signals in section 4.3.1.

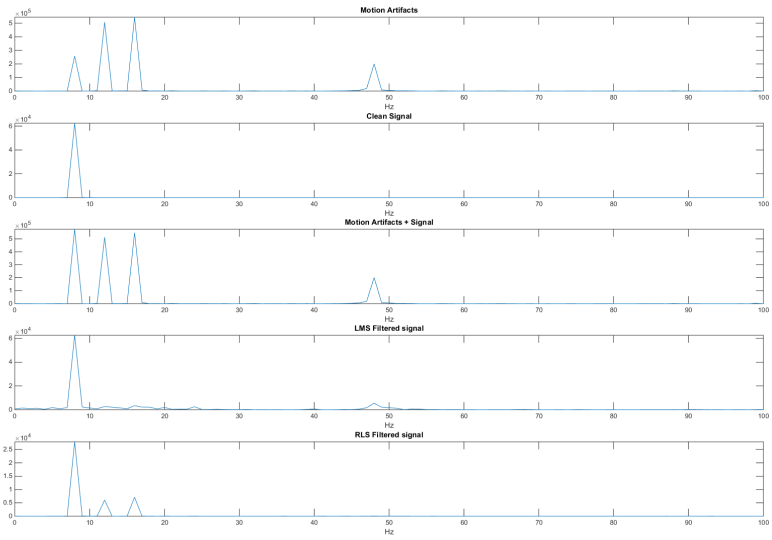

Figure 4.9: Testing the LMS & RLS filter in frequency domain

4.3. PPG signal

There are multiple situations that can occur that can cause the estimated BPM to differ from the actual BPM. These situations can be categorised as follows.

1. The MA peak is far away from the BPM peak
2. The MA peak is close to the BPM peak
3. The MA peak is on the BPM peak
4. There is no heart rate peak

The best way to analyse this is to look at the power spectral density of the PPG signals, this is done using Equation 4.2.

$$Y(f) = abs(fft(y(t)))^2 \tag{4.2}$$

4.3.1. MA peak far from BPM peak

The most common situation is when the MA peaks are far away from the BPM peak, one of these situations is displayed in Figure 4.10. The peaks at 70 and 150 in the power spectral density (PSD) of the PPG are caused by MA while the peak at 110 is from the actual BPM. In this specific window the MA components are larger than the HR peak meaning that a simple $Max(window)$ function would read 150 as a HR instead of the actual BPM of 110.

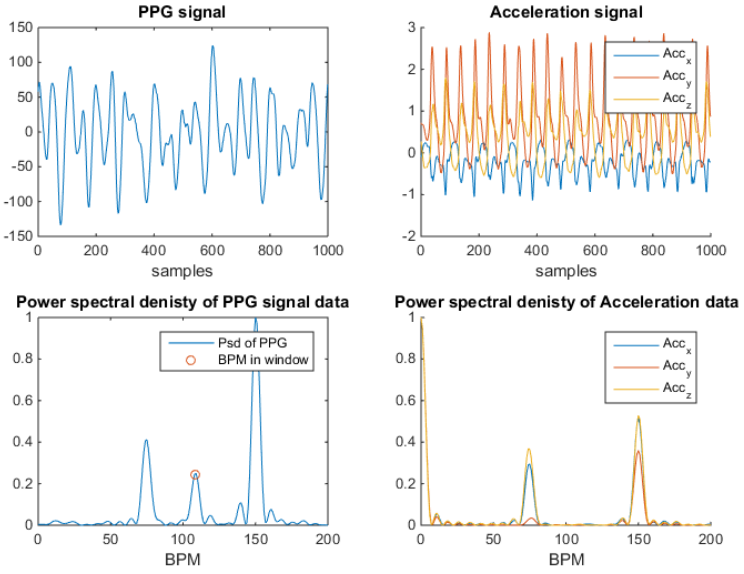

Figure 4.10: Comparing PPG data with accelerometer data where the MA peak is far from the BPM peak

Applying LMS Filter

The example displayed in Figure 4.11 shows the initial unfiltered PPG signal (in blue) that is influenced by MA at 75 and 150 BPM. The filtered signal (in yellow) clearly reduces the peaks where the MA is present, this results in the peak at the HR being relatively more pronounced than pre-filtering.

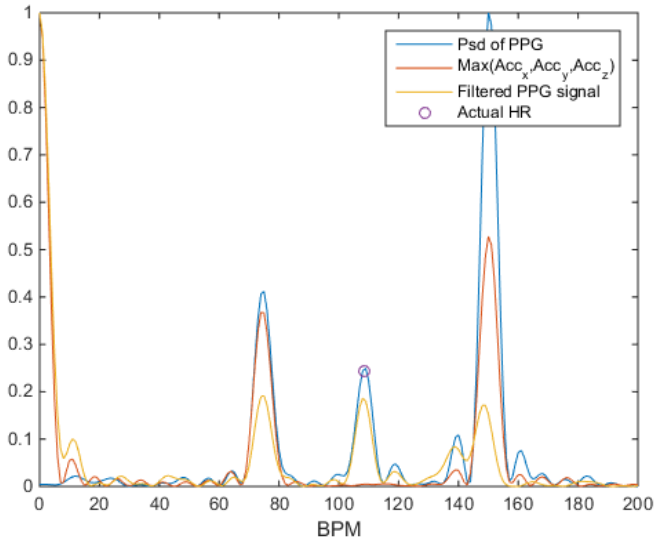

Figure 4.11: LMS filter when MA peak is far from the BPM peak

4.3.2. MA peak close to the BPM peak

Another situation is when the MA peak is close to the BPM peak. In Figure 4.12 the actual heart rate is 139 BPM and there are MA peaks at 150. This peak almost overtakes the real HR peak but in this case it is still possible to simply read the maximum value from the PSD.

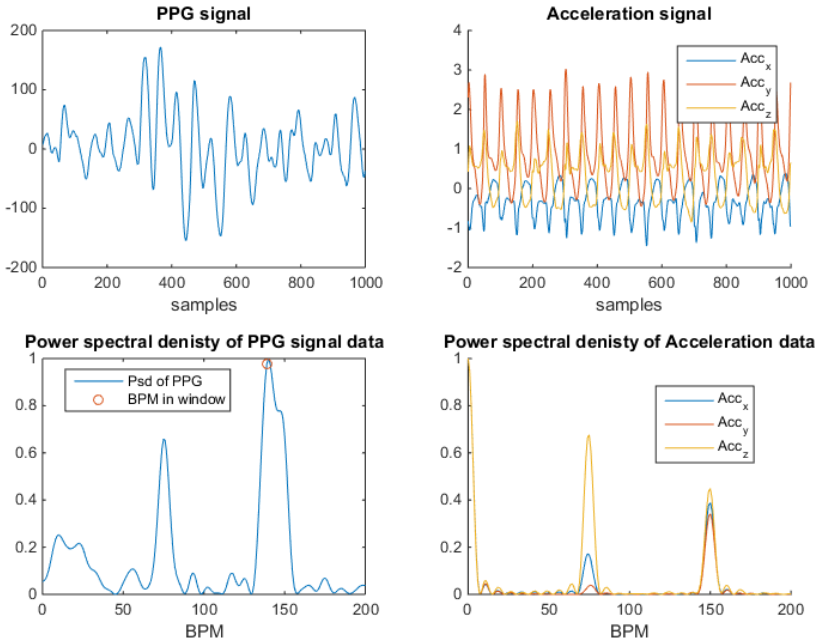

Figure 4.12: Comparing PPG data with accelerometer data where the MA peak is close to the BPM peak

Applying LMS Filter

When the MA peak is close to the BPM peak as is the case in Figure 4.13 the filter manages to lower the MA influence and leaves the peak at the HR in tact.

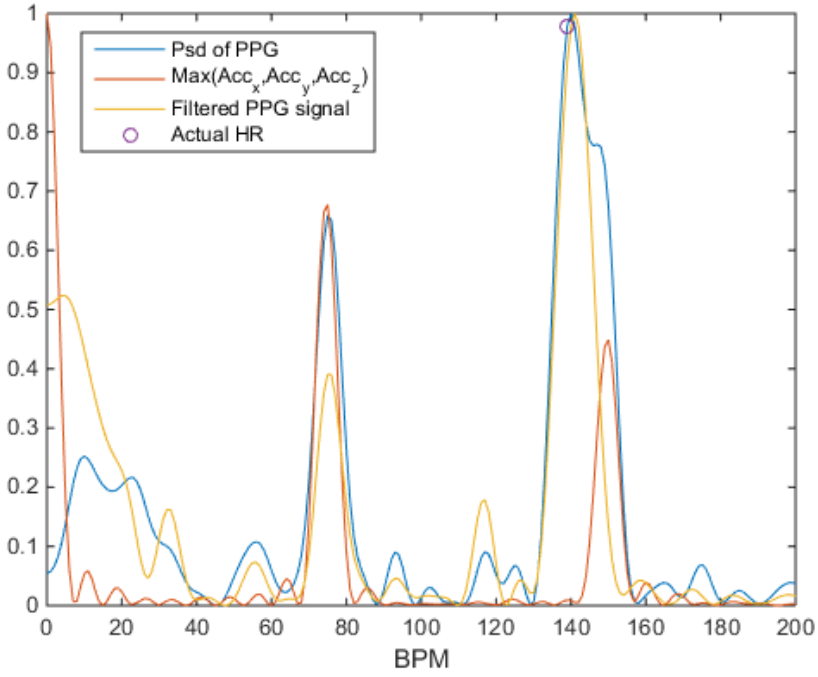

Figure 4.13: LMS filter when MA peak is close to the BPM peak

4.3.3. MA peak is on the BPM peak

In this situation the MA and the PPG share a peak at 150 BPM. Since the MA components are added constructively the HR peak is enlarged and in this case the correct HR would also be read by taking the maximum value.

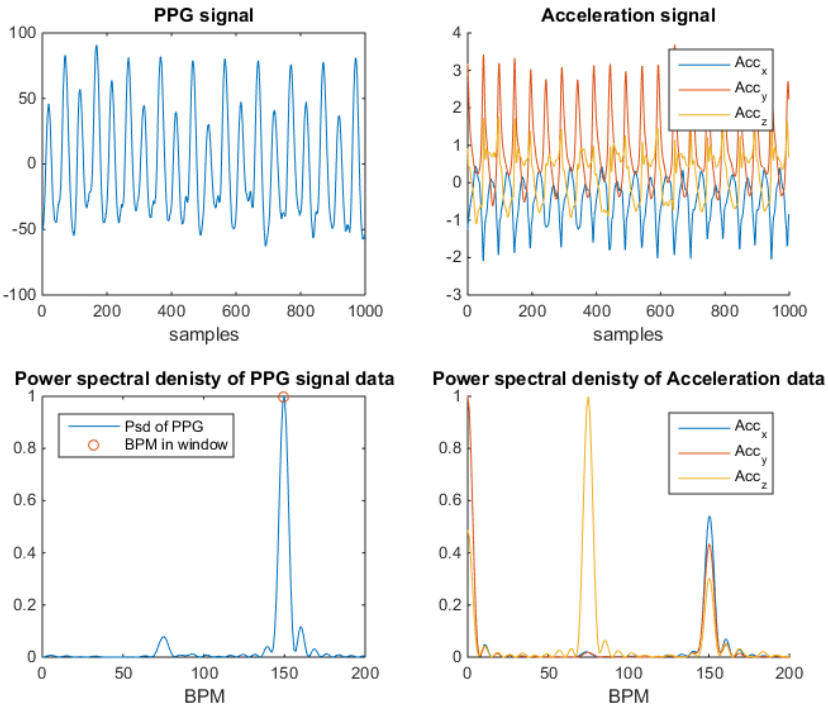

Figure 4.14: Comparing PPG data with accelerometer data where the MA peak and BPM peak are at the same frequency

Applying LMS Filter

If the MA peak is on the BPM peak as in Figure 4.15 the peak is not removed as would be the case with a notch filter (or band-stop), the peak is lowered corresponding to the influence of the MA.

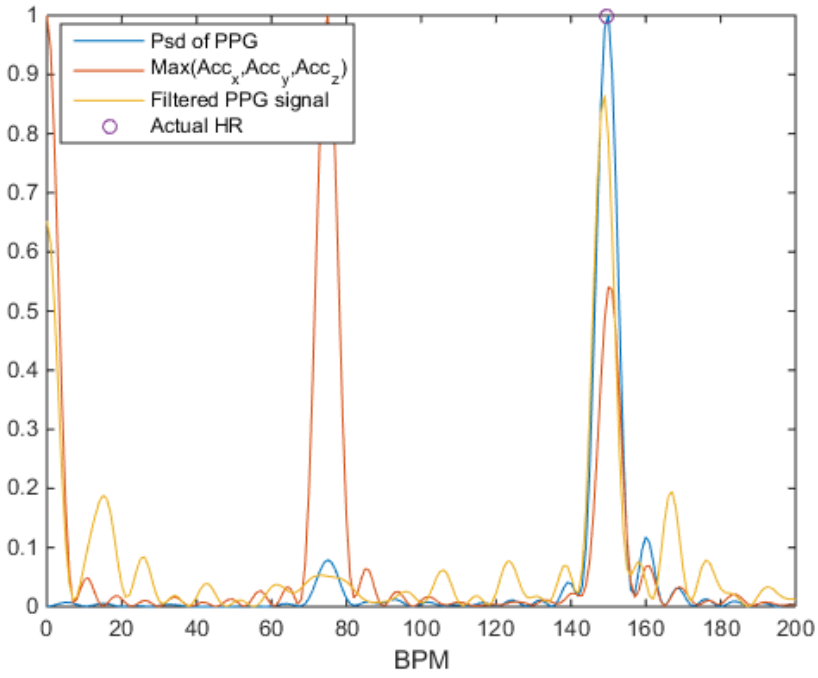

Figure 4.15: LMS filter when MA peak is on the BPM peak

4.3.4. There is no heart rate peak

Finally, a situation can occur that the PPG signal is simply too distorted that there is no way to read the HR from it. In these cases another technique has to be applied to work around this problem, one way is to estimate the HR from previous values if no prominent peak is present within an expected range.

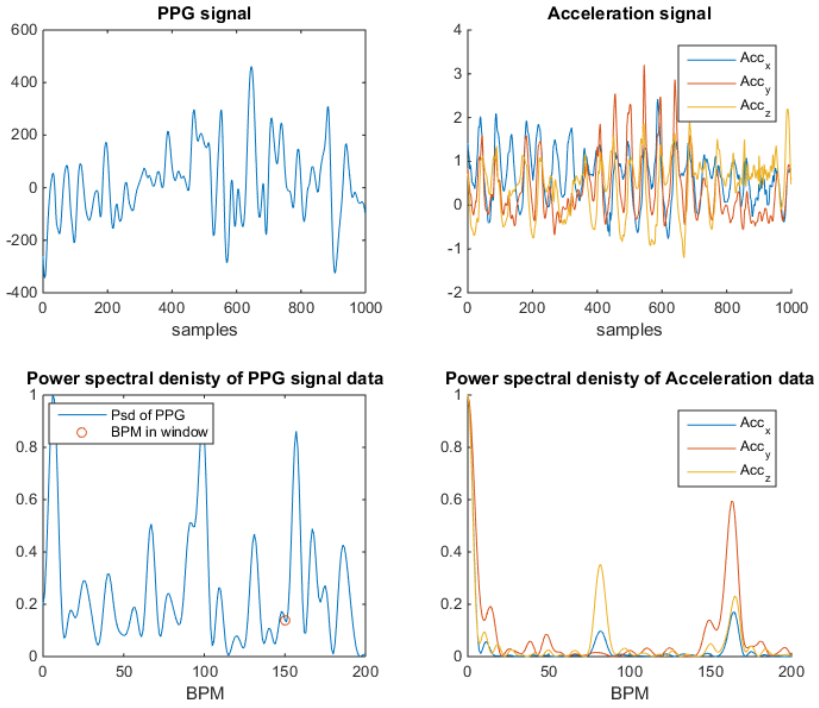

Figure 4.16: Comparing PPG data with accelerometer data where no HR peak is present

Applying LMS Filter

When the PPG signal is very bad one is unlikely to get any useful information from it. At the HR in Figure 4.17 there is actually a small peak in the filtered signal. With a smart spectral peak tracking algorithm this peak can still be extracted as long as the previous estimated HR is correct.

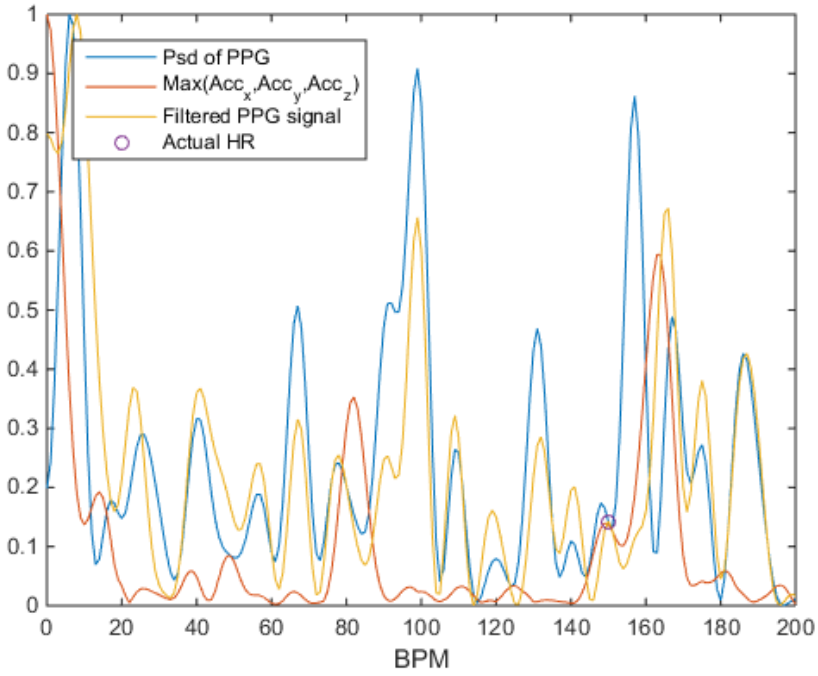

Figure 4.17: LMS filter when no BPM peak is present in PPG

5

Optimisation

Since the LMS and RLS filters have been implemented, multiple methods were used to improve on the result in Chapter 4.2.

5.1. Peak Detection

Up until now the HR was always found by looking at the highest peak in the PSD of the PPG signal.

5.1.1. Slope Detection

One method that was also investigated for ECG in section 4.1.2 was using knowledge on the cardiac cycle to look at the slope of the signal. This technique can also be applied to PPG signals. Immediately it must be noted that ECG peaks are a lot more pronounced than PPG peaks meaning that it is likely harder to implement.

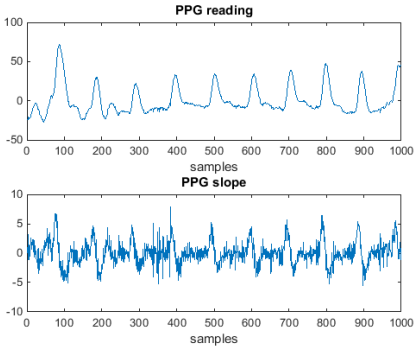

Figure 5.1: PPG reading and PPG slope

At first glance the PPG slope data from Figure 5.1 is not very helpful as was

expected due to the less well defined peaks. A Histogram was used to see if there were specific slopes that occurred more often for PPG.

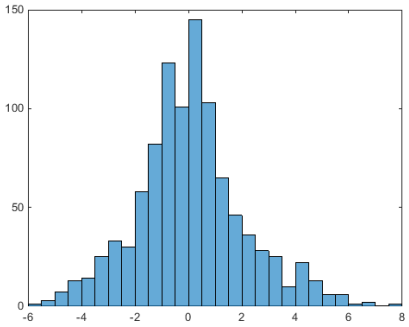

Figure 5.2: Histogram of PPG slope in 5.1

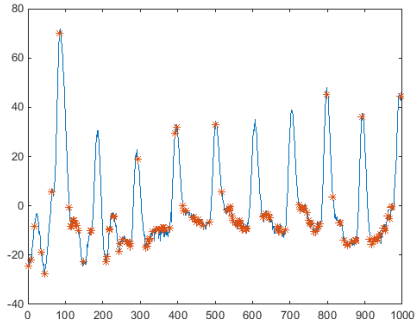

Figure 5.3: Corresponding slopes with PPG locations

Figure 5.2 shows the distribution of PPG slopes, the most common slope ranges between 0 and 0.5, this is clearly not the slope created by diastole or systole. Figure 5.3 shows which places in the PPG data have a slope between this range. After some analysis it was noted that the diastole slope for this data set ranges between 4.4 and 6.5. For these ranges the data points of Figure 5.4 are found.

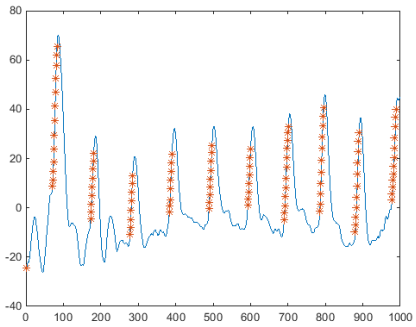

Figure 5.4: Corresponding slopes with PPG locations

The problem with this approach is that it requires the slope to remain within a certain range. During rest this technique managed to work with reasonable accuracy but as soon as movement was introduced the slope related to diastole no longer fit within these ranges. It was therefore decided this method was not suitable for reliable results.

5.1.2. Spectral Peak Tracking

Spectral Peak Tracking (SPT) was implemented by subgroup 2 by Julio and Tiamur and is used to enhance the peak detection process. SPT uses previous values of the heart rate to only look for peaks within a range around it.

In the example from Section 4.3.1 where the MA peak is far away from the BPM peak, Figure 4.10 shows that the real heart rate is at 110. While there is a peak present at this value the largest peak (and thus the faultily measured HR) is caused by MA at 150 BPM. If SPT were to be used, and assuming that the previous BPM was estimated correctly, the system would only look for peaks in the range of 110 BPM ensuring that the right peak was still found.

5.2. LMS Filter Optimisation

With each of the adaptive filters there were multiple values that could be altered.

5.2.1. Adjusting filter weight vector length

One of the filter parameters that had to be analysed was the filter weight vector length from Equation 2.9. The influence on computation time is minimal using Desktop PC's so it can not be analysed what the effect on it is.

Figure 5.5 shows the relation between the filter weight length and the MAE over all the data sets. At low orders the error is more unstable and after a length of 100 the changes in MAE were minimal. The lowest MAE occurs when a weight vector length of 74 is taken with an MAE of 5.43.

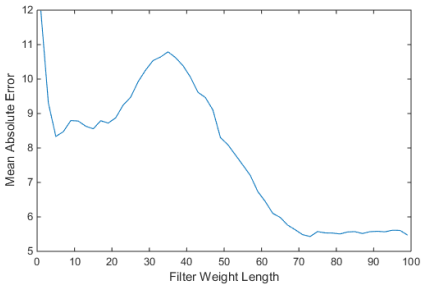

Figure 5.5: Influence on MAE by varying the filter weight

Table 5.1: Comparing filter weight length vs MAE between lengths 66-76

Filter Weight Length	66	68	70	72	74	76
MAE	5.98	5.76	5.63	5.48	5.43	5.57

5

Having a higher filter weight vector length leads to more knowledge on previous iterations, so when this length is low it could lead to unstable results. The trade-off is that using a higher filter weight increases the computational complexity which increases the convergence time.

5.2.2. Adjusting the step size

The step size effects the convergence speed, steady state error and stability of the adaptive filter [18].

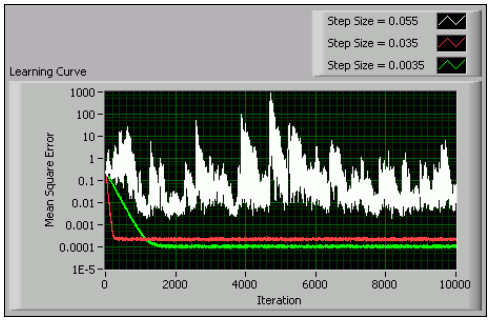

Figure 5.6: Comparing the performance of an adaptive filter with different step size values [18]

Figure 5.6 shows that that when the step size is large the convergence speed is higher, however in the case for step size = 0.055 the system has become unstable. A smaller step size takes longer to converge but has a smaller steady state error. The step size does not influence the computation time as the same number of iterations are computed regardless of step size.

Figure 5.7 shows that as the step size is increased the MAE is lower up until a certain point. Before this point the system doesn't fully converge within the

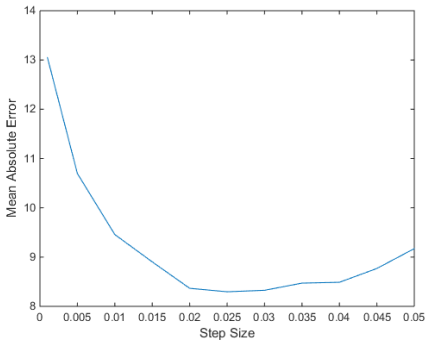

Figure 5.7: Comparing Step size vs MAE over all the data sets

iterations but after this point is where the system has completely converged. Increasing the step size beyond this point further increases the convergence speed but negatively influences the MAE.

Table 5.2: Step size vs MAE using smaller step sizes

Step Size	0.020	0.021	0.022	0.023	0.024	0.025	0.026	0.027	0.028	0.029	0.030
MAE	8.37	8.28	8.28	8.26	8.20	8.30	8.24	8.22	8.23	8.24	8.33

From Table 5.2 the optimal step size for the given data can be set at 0.024 with a corresponding MAE of 8.20.

5.2.3. Choosing LMS algorithm

There are multiple LMS algorithms that can be used. These were laid out in Section 2.2, and were tested to see which algorithm gives the best results. The step size and filter weight length were optimised for each of the filters and the results are displayed in Table 5.3

Table 5.3: Comparing different LMS algorithms on MAE and computation time

Algorithm	**MAE**	**Time (s)**
LMS	6.00	38
Normalised LMS	4.64	38
Sign-Error LMS	6.10	37

The best result was obtained using the Normalised LMS filter using a step size of 0.024 and a filter weight length of 74.

5.3. RLS Filter Optimisation

As mentioned in Section 2.2.2 the two important parameters that can be adjusted are the filter weight length and the forgetting factor.

5.3.1. Adjusting filter weight vector length

In Section 5.2 the optimal filter weight vector length of the LMS filter was found. In the same way this has been done for the RLS filter. Starting with the growing window RLS, with non variable λ, the results are shown in Figure 5.8. In this Figure it can be seen that the error gets lower until a filter weight order length of about 70 after which it becomes relatively stable.
In Table 5.4 it is shown that the lowest MAE is 5.16 at a filter weight length of 69.

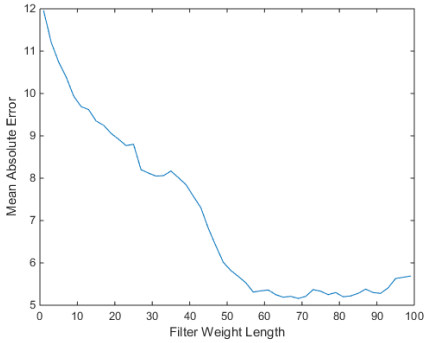

Figure 5.8: Influence on MAE by varying the filter weight

Table 5.4: Filter weight length vs MAE

Filter weight length	59	61	63	65	67	69	71	73	75
MAE	5.34	5.36	5.25	5.19	5.21	5.16	5.21	5.37	5.33

The time it took for Matlab to run the script increased with increasing filter weight length. However, this is well worth it, since MAE also increased significantly with increasing filter order weight.

5.3.2. Adjusting the forgetting factor

The value of the forgetting factor (λ) defines the system memory and affects:

- The convergence and the ability of the filter to track time-varying statistics in the input sequence
- the stability of filter coefficients

For the conventional RLS filter, λ is variable and needs to be between 0.98 and 1. As mentioned in 2.2.2 a forgetting factor closer to 1 should give a smaller MAE.

Figure 5.9 shows that this is true for the given data. Tabel 5.5 zooms in on the part $0.98<\lambda<1$. This results show that the smallest error is 11.75 for λ=0.990.

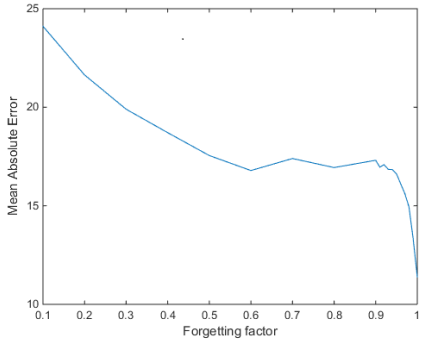

Figure 5.9: Influence on the MAE by varying the forgetting factor

Table 5.5: Comparing the AME and the forgetting factor. Zooming in on 0.995-1

Forgetting factor	0.980	0.985	0.990	0.995	0.999
MAE	11.87	11.92	11.75	11.82	11.93

5.3.3. Choosing RLS algorithm

The different RLS filters, that have been tested, were mentioned in Section 2.2.2, the results of this are shown in Table 5.6. For each filter the best filter weight length and forgetting factor were computed. In Section 5.3.1 this was done for the growing window RLS. For the conventional RLS the best forgetting factor was computed in Section 5.3.2. In the same way as for the growing window RLS, the filter weight length with the lowest MAE was found for the conventional RLS. This turned out to have an error of 8.53 with a filter weight length of 58. Combining these optimal parameters gives the results shown in Table 5.6

Table 5.6: Comparing different RLS algorithms on MAE and computation time

Algorithm	**MAE**	**Time (s)**
Conventional RLS	6.52	176
Growing Window RLS	5.16	197

As can be seen, the best result comes from using the growing window RLS with an error of 5.16 and a filter weight length of 69.

5.4. Using multiple PPG signals

One way to improve the HR estimation is to use multiple PPG input signals. Usually one PPG sensor is used as under normal circumstances this works well enough and adding more sensors would shorten battery life. Knowing that during exercise the PPG measurements are not good enough for accurate readings, more PPG sensors could be added and only used when the system notices that it is required.

Multiple PPG signals can be combined to form a more clean signal. Firstly the mean of the 2 PPG signals was taken with the idea that common peaks would remain and peaks created by noise would not be common and therefore become reduced. An important aspect to take into account was the time delay between the 2 PPG signals, if the sensors were not placed at the same point on the wrist the signals could be shifted. Since for this experiment all the data was already given this was not in our control but upon investigation there was no delay present, implying that the PPG signals were recorded on a similar location on the wrist. Since the noise from MA is added constructively, and is present in both the PPG signals, taking the mean between multiple signals would not remove the MA. However this would be able to reduce the non-MA related noise significantly.

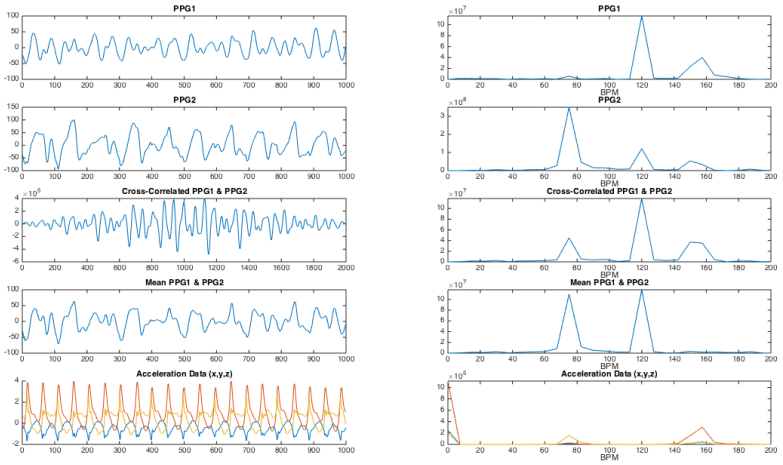

Figure 5.10: Mean and cross-correlation results

Another technique that is investigated is using the cross-correlation function due to the correlation between two PPG signals. In Figure 5.10 we can see peaks in the acceleration data at roughly 75 and 160 beats per minute (BPM), the influence that these have on the PPG signals can be seen here too. The peak at 75 on PPG2 is larger than the peak created by the HR, whereas PPG1 is more influenced by the acceleration peak at 160 BPM. In this situation, with data sampled over a 10 second interval, if the peak was detected using PPG2 the wrong HR would have been found but luckily the mean and cross-correlation can help us in these situations. The cross-correlation proved particularly helpful when the interference from MA obscured simple readings.

It can also be noted that the MA components are still visible in both the mean and the cross-correlation, therefore both signals were put through the filter separately before other mathematical operations were applied. In Figure 5.11 the results are shown. It is important to note that one downside of using the cross-correlation or the mean is the computation time, in both these cases the two PPG signals are filtered separately which requires double the time per window. The cross-correlation function also takes significant time and doubles the samples in the window, so unless down sampling takes place back to the original length, further computations done on the longer signal will also take more time.

Lastly diversity combining was investigated, this technique has multiple solutions to combine multiple signals into an improved signal. One of the solutions of diversity combining is the equal-gain method, here the signals are simply summed. An interesting theory is Maximal-ratio combining, here the signals are weighted based on their Signal-to-Noise ratio (SNR).

$$SNR = \frac{P_{signal}}{P_{noise}} \tag{5.1}$$

The problem with this is that you need a clean signal with which you can calculate P_{signal}. This assumes that the PPG signal only contains the HR and is no longer influenced by any noise or MA. When looking at the frequency spectrum only one peak should be visible at the correct HR. The MA component is not perfectly additive or it would also be easy to remove it, so in this situation our ”clean signal” is the output of the filter even though this is not actually ”clean”, this causes the SNR to often pick the worse signal and therefore was not used.

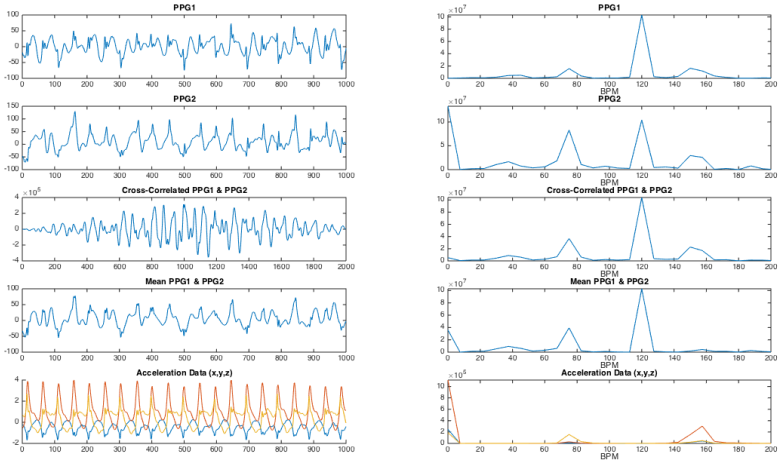

Figure 5.11: Mean and cross-correlation including filtering results

5.5. SMART

By combining the knowledge gathered by the three subgroups the Sports Motion Artifact Removal Technique (SMART) was made. This is the final product presented as a whole group. JOSS, implemented by Julio and Tiamur in subgroup 2, manages to achieve the lowest MAE as a standalone system however the problem is the large amount of computation power required to use it. ANC offers a less robust system but it makes up for that in computation time. The exact details of the SMART framework can be found in Julio and Tiamur's thesis though the results will be discussed here.

6

Results

6.1. ECG signal

Using the Slope Detection discussed in Section 4.1.2, the HR was estimated over all the 12 data sets. An example of data set 8 can be seen in Figure 6.1.

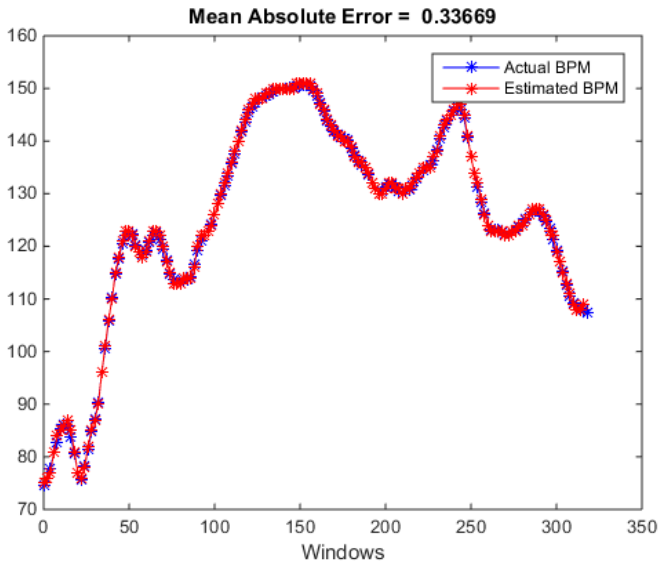

Figure 6.1: ECG estimated heart rate vs actual heart rate

Table 6.1: Mean Absolute Error of estimated ECG BPM over all data sets

Data set	1	2	3	4	5	6	7	8	9	10	11	12	Average
MAE	0.416	0.635	0.431	0.416	0.412	0.588	0.365	0.337	0.320	0.337	0.432	0.356	0.420

The data in Table 6.1 shows that the average error between the estimated and true BPM is 0.42. When looking at the example data set in Figure 6.1 the estimated BPM follows the actual BPM curve very closely, in practice this would be sufficiently accurate.

An important aspect to note is that the BPM scale used only contains integer values of BPM whereas the true BPM often lies between these values. As an example if the true BPM is 110.5, the closest next integer is 0.5 away. Using Equation 2.17 the MEA for this window would be 0.5. This is why there is still an error present even while using ECG.

6.2. LMS results

Using the optimised parameters from Section 5.2, the NLMS filter was tested over all the data.

Table 6.2: Comparing the MAE of different algorithms over all the data sets

Data set	1	2	3	4	5	6	7	8	9	10	11	12	Average
Normalised LMS	10.66	2.46	1.31	1.37	2.06	6.95	7.01	0.70	0.46	11.61	6.76	1.19	4.38
NLMS with SPT	1.79	1.09	0.66	1.09	0.84	1.14	0.95	0.62	0.44	3.92	5.87	1.46	1.66

The results for data set 10 using NLMS are displayed in Figure 6.2. While on average over this data set the error is 11.61 the estimated BPM follows the true BPM reasonably well. The problem is that this data set is influenced by MA that corresponds to BPM of 80 that the filter does not get rid of.

This is why SPT from Section 5.1.2 was applied. This would not allow for big jumps is the BPM and would only look for the heart rate peak around the previous BPM. Figure 6.2 shows that using SPT the MAE is decreased to 3.92 over the troublesome data set 10. While SPT does not appear to find the right peak all the time it does stay within range of the previous BPM which is advantageous for the MAE.

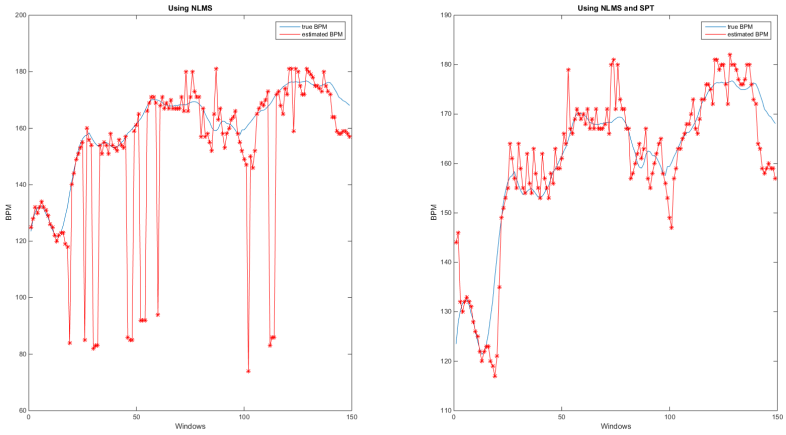

Figure 6.2: Comparing data set 10 using NLMS with a MAE of 11.61 and NLMS + SPT with a MAE of 3.92

6.3. RLS results

From Section 5.3 it became clear that the growing window RLS showed the best results, so this filter, with its optimal parameters, was tested over all the data. The results of this are displayed in Table 6.3.

Table 6.3: Comparing the MAE of different algorithms over all the data sets

Data set	1	2	3	4	5	6	7	8	9	10	11	12	Average
Growing window RLS	13.36	7.69	5.75	4.16	2.76	3.60	2.73	1.95	2.44	10.64	2.79	4.01	5.16
Growing window RLS with SPT	2.50	1.09	2.02	1.02	1.07	1.47	1.49	0.71	0.69	3.41	6.37	1.13	1.91

As can be seen from this Table the data set with the biggest error is data set 1. In Figure 6.3 it is shown that even though the error over this data set is 13.36 it still follows the true BPM curve really well, it just has some outliers. With the help of the SPT, which is explained in Section 5.1.2, this problem is mostly fixed. The second graph in Figure 6.3 shows that with this SPT the MAE drops to 2.50. The SPT has effect on the RLS filter in the same way as on the LMS filter which is explained in Section 6.2.

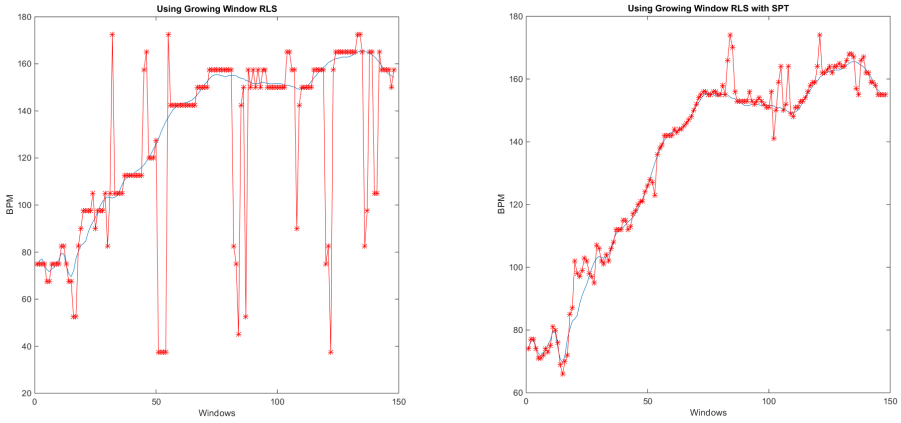

Figure 6.3: Comparing data set 1 using GW RLS with a MAE of 13.36 and GW RLS with SPT having a MAE of 2.50

6.4. Combined Results

Within this thesis the LMS and RLS filters were compared on all the data sets. To make the comparison fair no special peak tracking was added to either of the filters. The parameters were optimised in Section 5.2. These algorithms were then also compared to the implementations of the other sub groups, namely JOSS and the combined product named SMART. The results are displayed in Table 6.4.

Table 6.4: Comparing the MAE of different algorithms over all the data sets

Data set	1	2	3	4	5	6	7	8	9	10	11	12	**Average**
NLMS with SPT	1.79	1.09	0.66	1.09	0.84	1.14	0.95	0.62	0.44	3.92	5.87	1.46	**1.66**
GW RLS with SPT	2.50	1.09	2.02	1.02	1.07	1.47	1.49	0.71	0.69	3.41	6.37	1.13	**1.91**
TROIKA	3.01	2.50	2.22	1.48	0.68	1.13	0.66	0.73	0.51	6.50	1.13	1.77	**1.86**
JOSS	2.26	2.65	1.03	1.21	0.59	1.18	0.67	0.55	0.57	3.67	0.78	1.48	**1.39**
SMART	2.59	2.52	0.74	1.06	0.73	1.19	0.61	0.47	0.46	4.09	0.88	1.68	**1.42**

Table 6.2 shows that SMART, which combines different algorithm manages to achieve the best results with an MAE 1.42.

Table 6.5: Comparing algorithm computation time over all data sets on Desktop PCs (specs: [19])

Algorithm	**Execution Time (s)**
NLMS	38
NLMS with SPT	42
GW RLS	212
GW RLS with SPT	358
TROIKA	183
JOSS	186
SMART	37

Table 6.5 shows that there is a large difference in computation time between the different algorithms. SMART combines JOSS and NLMS to create a middle ground between computation time and accuracy.

7

Conclusion

The results show that there is a trade-off between computation time and accuracy. While JOSS has an MAE of only 1.39 the computation time is 186 seconds, NLMS has a larger error of 4.38 however the computation time is almost 5 times less. It is important that the computation time is fast enough to be implemented on a wearable device which is a lot less powerful than the desktop PC's that were used for these results. There are roughly 150 windows per data set and 12 data sets which means that the BPM was calculated for roughly 1800 windows. From this it can be concluded that JOSS needs roughly 0.1 seconds per window. Since every window needs to be calculated in 2 seconds, this means that a wearable device can only be a factor 20 slower than the used Desktop PCs and for NLMS the wearable device can be 95 times slower.

From the results of NLMS in Table 6.2 the MAE is clearly a lot worse for specific data sets that skew the average. For data set 10 the MAE is 11.61, this data set is also the worst result for JOSS and SMART. By adding SPT to NLMS the error is greatly reduced to a MAE of 1.66 with the downside being a slightly higher computation time relating to the added SPT. SPT only adds roughly four seconds to the computation time which taken over one window is negligible.

SMART takes the best features of NLMS and JOSS. At 37 seconds the computation time is much lower than when just using JOSS and the MAE is significantly lower than when just using NLMS, at 1.42. This makes SMART fast and functional even under the stress of sports with a lot less burden on computational power than JOSS and more accuracy than using NLMS.

It is important to ensure that SMART satisfies the Programme of Requirements. The functional requirements are satisfied by optimising SMART for computational complexity and while it was not tested on an embedded system it looks promising. SMART falls well within the system requirements (worst data set MAE: 10, average over all data: 5) as the worst data set is data set 10 at a MAE of 4.09 and an average of 1.42.

7.1. Future Work and Recommendations

7.1.1. Improving LMS algorithm

An idea that was proposed during our work was to apply the LMS algorithm in the frequency domain. This has been said to reduce computational complexity [20].

7.1.2. Improving RLS algorithm

While researching different RLS filters a few other names popped up, for example; the Householder RLS, the Sliding Window RLS, a combination of these two: Householder Sliding Window RLS and QR decomposition [21]. There was not enough time to implement all of these different RLS filters, so the most promising two were chosen for this thesis. However, in future, it might be useful to implement these as well.

7.1.3. Using multiple PPG sensors

While adding multiple sensors can have a negative effect on the computation time it could improve the results under more extreme circumstances. While this was investigated in Section 5.4 the second PPG signal that was given in our data sets almost always performed worse than the first PPG signal, sometimes even making it disadvantageous to use both PPG signals.

Appendix

```
% Function to find the heart rate
function [BPM, loc] = findBPM(signal,Fs,k,step_size,loc)
    %signal: size 1x6, contains ECG,PPG1,PPG2,Acc_x,Acc_y,Acc_z
    %k: filter weight length
    %step_size: filter step size
    %loc: location of previous BPM peak
    %output:
    %   BPM: Heart rate in BPM
    %   loc: location of current BPM peak


    %%% Variables %%%
    N = 60*Fs;
    f = 60*Fs*[0:1:N-1]./N;
    srate = Fs;                         % 125 Hz
    window    = 8 * srate;              % window length is 8 seconds
    step      = 2 * srate;
    MaxHRChange = 7;                    %BPM/s
    delta = MaxHRChange*2;              %Maximum BPM deviation per step

    %Extract relevant data
    ppg1 = signal(2,:);
    acc_data = signal(4:6,:)

    %Filter signal
    clean_sig = LMS_sig(ppg1,acc_data,k,step_size);

    %Bandpass signal
    BPfilt_sig = BP(clean_sig',Fs);

    %Compute power spectral desity
    Y_clean = abs(fft(BPfilt_sig,N)).^2;

    %OPTION 1 - uncomment to use
    %Find peaks and BPM without SPT
    [pks, locs] = findpeaks(Y_clean,f);
    [,index] = max(pks);
    BPM = locs(index);

    %OPTION 2 - uncomment to use
    %Find peaks and BPM with SPT
%     [ BPM, loc ] = SPT( Y_clean, delta, loc, N, f, f(N) );

end
```

```
%Function for LMS algorithm
function [ clean_signal ] = LMS_sig(signal,acc_data,k,step_size)
    %signal: PPG signal of 1 window
    %acc_data: Acc data of same window (x,y,z)
    %k: filter weight length
    %output: filtered time domain signal

    temp = zeros(1,length(signal))';
    mu = step_size; % step size

    lms2 = dsp.LMSFilter('Length',k, ...
   'Method','Normalized LMS',...
   'StepSizeSource','Input port', ...
   'WeightsOutputPort',false);

    for i = 1:3
        x = acc_data(i,:)';
        d = signal';
        [, err] = step(lms2,x,d,mu);
        temp = temp+err;
    end
    clean_signal = temp;
end


%%Function for the RLS algorithm
function [ clean_signal ] = RLS_sig(signal,acc_data,h,p)
    %signal = PPG signal per window
    %acc_data = acceleration data in the X, Y and Z direction for the same window
    %h = filterorder/ filter weight length
    %p = forgetting factor

    temp = zeros(1,length(signal));

    %Without selecting a specific RLS method, the conventional RLS is used
    hrls1 = dsp.RLSFilter(h, 'ForgettingFactor', p);

    for i =1:3
        x = acc_data(i,:)';
        d = signal';
        [, err] = step(hrls1,x,d);
        temp = temp+err;
    end

    clean_signal = temp;

end
```

```
% Sectral Peak Tracking Fuction
function [ BPM, new_loc ] = SPT( spec, delta, old_loc, N, f)
    %spec: power spectral density of filtered signal
    %delta: search range around previous BPM
    %old_loc: location of previous BPM



    %old loc is -1 when initialising, then search whole range
    if old_loc < 1
        delta = 200;
    end

    %Apply filter around previous BPM
    H = Hfilter( delta, N, old_loc );
    spec = spec.*H;

    %Find peaks
    [,ind] = max(spec);
    new_loc = ind;
    BPM = f(ind);

end


%Function to filter around previous BPM
function [ H ] = Hfilter( delta, N, old_loc )
    %delta: BPM Deviation from old BPM
    %old_loc: Location of previous BPM

    %If not in initialisation state:
    if old_loc > 0
        Ndelta = delta;
        Ndelta = ceil(Ndelta);
        Nmin = old_loc - Ndelta;
        Nmax = old_loc + Ndelta;

        if Nmin <= 0 && Nmax > N
            range1 = 1:N;
        elseif Nmin <= 0
            range1 = 1:Nmax;
        elseif Nmax > N
            range1 = Nmin:N;
        else
            range1 = Nmin:Nmax;
        end
        H = zeros(1,N);
        H(range1) = 1;
    %When initialising check whole viable spectrum
    else
        range1 = 41:191;
        H = zeros(1,N);
        H(range1) = 1;
    end
end
```

References

1. **H. Han, J. Kim** *"Artifact in wearable photoplethysmographs during daily life motions and their reduction with least mean square based active noise cancellation method"*, Computers in Biology and Medicine, vol. 42, no 2, pp. 387-393, April 2012

2. **John Allen**, *"Photoplethysmography and its application in clinical physiological measurement"*, Physiol. Meas. 28 R1, 2007

3. **Gu-Young Jeong**, *"Development of portable ECG measurement device and PC software for automatic ST analysis"*, IEEE, October 2010.

4. **Zhilin Zhang** *2015 IEEE Signal Processing Cup: Heart Rate Monitoring During Physical Exercise Using Wrist-Type Photoplethysmographic (PPG) Signals*

5. **Zhilin Zhang, Zhouyue Pi,Benyuan Liu**, *"TROIKA: A General Framework for Heart Rate Monitoring Using Wrist-Type Photoplethysmographic signals During Intensive Physical Exercise"*, IEEE Trans. on Biomedical Engineering, vol. 62, no. 2, pp. 522-531, February 2015

6. **Dr John Mandrola, MD** *What's a normal heart rate?*, Dr. John M, August 2011.

7. **I. F. Gorodnitsky and B. D. Rao** *"Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm,"* IEEE Trans. on Signal Processing, vol. 45, no. 3, pp. 600–616, 1997.

8. **M. Raghu Ram, K. Venu Madhav, E. Hari Krishna, Nagarjuna Reddy Komalla, K. Ashoka Reddy** *"On the performance of AS-LMS based adaptive filter for reduction of motion artifacts from PPG signals"*, Instrumentation and Measurement Technology Conference (I2MTC), IEEE, pp. 1-4, 2011.

9. **Levi B. Wood, H. Harry Asada** *"Noise Cancellation Model Validation for Reduced Motion Artifact Wearable PPG Sensors Using MEMS Accelerometers"*, Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE, pp. 3525-3528, 2006.

10. **Willis J. Tompkins** *eBook: Biomedical digital signal processing, Chapter 8 - Adaptive Filters*

11. **M. Raghu Ram, K. Venu Madhav, E. Hari Krishna, Nagarjuna Reddy Komalla, and K. Ashoka Reddy** *A Novel Approach for Motion Artifact Reduction in PPG Signals Based on AS-LMS Adaptive Filter*, IEEE Trans on Instrumentation and Measurement vol. 61 , no. 5, pp. 1445-1457, 22 December 2011

12. **Abhishek Chander** *Online Lecture, Churchill College, Adaptive Filters*

13. **Mathworks: dsp.LMSFilter System object** *http://nl.mathworks.com/help/dsp/ref /dsp.lmsfilter-class.html*

14. **Dirk T. M. Slock** *On the convergence behavior of the LMS and the normalized LMS algorithms*, IEEE Trans. on Signal Processing, vol. 41, no.9, pp. 2811 - 2825,Sep 1993

15. **Least Mean Squares (LMS) Algorithms (Adaptive Filter Toolkit)** *http://zone.ni.com/reference/en-XX/help/372357A-01/lvaftconcepts/ aft_lms_algorithms/*

16. **Lecture Slides: Recursive Least Squares Estimation** *http://www.cs.tut.fi/ tabus/course/ASP/LectureNew10.pdf*

17. **Weisstein, Eric W.** *"Kurtosis"*, MathWorld - A Wolfram Web Resource.

18. **Choosing a Step Size (Adaptive Filter Toolkit)** *http://zone.ni.com/ reference/en-XX/help/372357A-01/lvaftconcepts/aft_choose_stepsize/*

19. **PC specifications: Dell OptiPlex 9020**, *http://www.emc.com.br/ PDF_Specs/DELL_OptiPlex_9020.pdf. CPU = i5-4690, Memory = 8 GB, Windows 7 Enterprise 64-bit.*

20. **Boaz Rafaely,Stephen J. Elliott** *"A computationally efficient frequency-domain LMS algorithm with constraints on the adaptive filter"*, IEEE Trans. on Signal Processing vol. 48 no. 6,pp. 1649 - 1655, Jun 2000

21. **Anthanasios A. Rontogiannis, Sergios Theodoridis** *QRD-RLS Adaptive Filtering*, Springer US, chapter 7, pp. 181-202, December 2008