DSAA Project Report

Team 17

Mohsin Mustafa (**20161131**), Shyamgopal Karthik (**20161221**), Sumukh S (**20171023**)

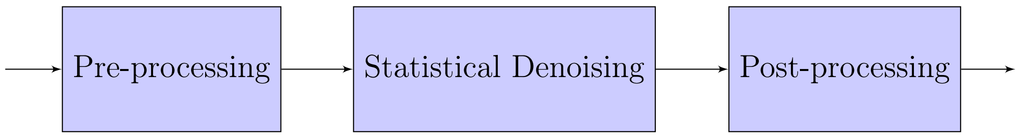
1 Overview

The algorithm used in the project consists of a pipeline of three sections. The first one is the **Pre-processing** stage, where basic methods like bandpass filter is used.

This is followed by the crux of the algorithm, where **statistical methods** are used for denoising the signal. The specific filter used is the Wiener filter. It makes use of the accelerometer data to obtain the noise portion required in the filter.

Finally there is the **post-processing** stage where the peaks are identified for estimating the heart rate in a small range determined by the previous window heart rate.

The above pipeline can be shown as in the below figure. Now each part will be focused on and explained in more detail.



2 Pre-processing Stage

The first step is to remove frequencies of values of heart rate that are outside the range of a healthy person. So only the range $[20, 240]$ **bpm** is considered. To do this a Butterworth filter of order 4 was used.

The choice of the Butterworth filter was chosen based on comparisons with other filter in the same algorithm such as Chebyshev and Bessel filters. The order for the Butterworth filter was found using trial and error on the test data provided, where 4 worked best.

Listing 1: Butterworth code snippet

```
samplingRate = 125;
bandPassRange = [24 240];
[bw1 bw2] = butter(4, bandPassRange/(125*60/2), 'bandpass');
...
for j = (1 : 5)
    initRangeData(j, :) = filter(bw1, bw2, initRangeData(j, :))
end
```

After developing the initial filter, we divide the samples as mentioned in the question - windows of length eight with step value of two. The remaining operations (in this stage as well as the following) are performed for each sample.

The Butterworth filter is applied on the three accelerometer signals and the PPG signal channels. The two channels of the PPG are combined to one by adding the z-normalizations (setting mean to zero and standard deviation to one) of the two channels.

Since the sampling rate is very high (for a human bpm estimation), we downsample all the signals (PPG and accelerometers) by a factor. Then the DFT of these downsampled signals are taken (after padding with zeroes for the required size).

With this the pre-processing stage is over, and the major denoising part starts.

3 Statistical Denoising Stage

Due to the acceleration and jerks of the object the PPG sensor is placed on, the signals are affected by noise too, which causes motion artifacts (MA) in the signal constructed by the PPG sensor.

Among the two ways of either using a Bayesian model or a frequentist model for estimating the underlying signal, we used a frequentist model, specifically the Wiener filter.

Although the Kalman filter (which is tightly related to the Wiener filter) was considered, we went on the assumption that the signal was of a stationary process.

The Wiener filter allows the original signal *bpm* to be random, which is useful in analysing the heart beat rate. Without presenting the entire derivation of the Wiener filter, the important parts are mentioned here.

Provided that the given signal is $x(t)$ and the final signal to be found (bpm) is $y(t)$, the relation between them is

$$y(t) = x(t) + w(t)$$

where $w(t)$ is a noise process which has no correlation with $x(t)$. So the optimal estimator filter is given by the LTI filter whose frequeny response is

$$H(\omega) = \frac{S_{xx}(\omega)}{S_{xx}(\omega) + S_{ww}(\omega)}$$

where S_a is the spectral density of $a(t)$. Two things to note are that the filter is causal and that the functions are to be normalized (normally-distributed) before applying the filter.

Two Wiener filters are used, whose results are then added up.

The first filter can be represented as

$$\frac{PPG - \frac{1}{3}\sum_{i=1}^{3} ACC(i)}{PPG}$$

which considers the acceleration motion artifacts to be added upon the PPG signal values.

The second filter is

$$\frac{PPG}{PPG + \frac{1}{3}\sum_{i=1}^{3} ACC(i)}$$

which considers the acceleration motion artifacts to negatively affect the PPG signal values.

The relevant code of this section is concisely shown below.

Listing 2: Wiener filter code snippet

```
for j = (2 : 4)
    normFFTMat(j, :) = normalize(FFTMatRan(j, :));
end
...
Fin1 = 1 - 1/3 * sum(normFFTMat(2:4, :))./(normFFTMat(1, :));
Denoised1 = abs(FFTMatRan(1, :)).*Fin1;
Fin2 = normFFTMat(1, :)./(normFFTMat(1, :) + ...
       sum(normFFTMat(2:4, :))./3);
Denoised2 = abs(FFTMatRan(1, :)).*Fin2;
Denoised = Denoised1 + Denoised2;
```

The causality of the Wiener filter allows this to be used real-time too, which is a huge bonus.

4 Post-processing Stage

For post-processing, the main idea used is that the heart beat rate of a human being, across a small window, only changes by a small range.

However, such a range would also be affected by the previous changes. This means that a person sitting down with x **bpm** would have his next recorded bpm (in the next window) to change only by a small window of, say, $x \pm 5$ **bpm**. However, when the person starts to run, his rate of change of bpm across windows would be larger. So the rate of change of the range size ($\pm R$ **bpm**) should also be a function of the previous estimated bpms.

Hence, for our case, we first assume such a range to be ± 25 **bpm** for the initial few estimations. This is to take into account the case that the man could be sitting or running from the first recorded PPG. Then on, the heart beat estimate range for a window is taken as the maximum bpm change in the previous recordings. Mathematically,

$$\text{Range}(n) = \begin{cases} 25\ \textbf{bpm} & \text{if } n \leq 15 \\ \max(bpm(x) - bpm(x-1))\ \forall\ x \in [2, n-1] & \text{if } n > 15 \end{cases}$$

where the current iteration of the estimate is taken as n, the estimates are stored in *bpm*, and the current range is $\pm\text{Range}(n)$.

With the range length available from the above calculations, finding the heart beat estimate is merely finding the peak of the range we are focused on, and mapping this to the frequency per minute by multiplying by 60. Thus the beat estimate for the n^{th} window lies in

$$[bpm(n-1) - \text{Range}(n), bpm(n-1) + \text{Range}(n)]$$

where the Range values must be appropriately transformed to bps (beats per second, or just Hz) values. The code relevant to this section is presented concisely below.

Listing 3: Range and *bpm* estimation code snippet

```
beatRange = 25;
if i > 15
    beatRange = max(abs(diff(pred(1:i-1)))) + 5;
end
...
pred(i) = mapFreq(curRange(ind(1))) * 60;
```

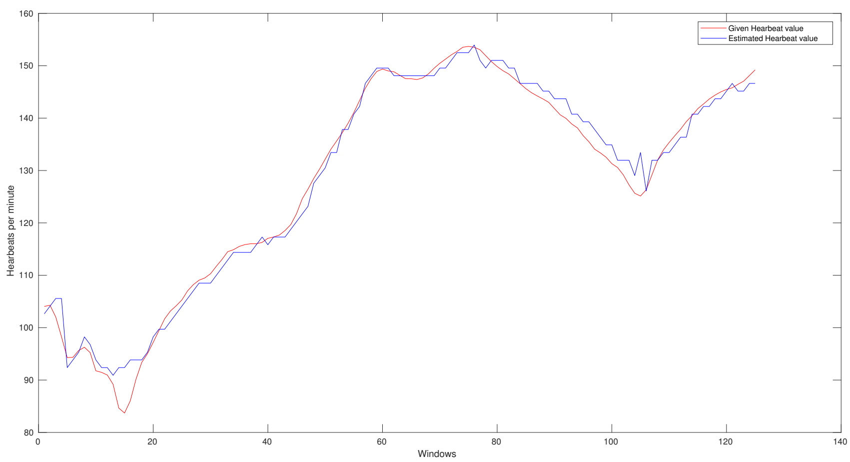
5 Results

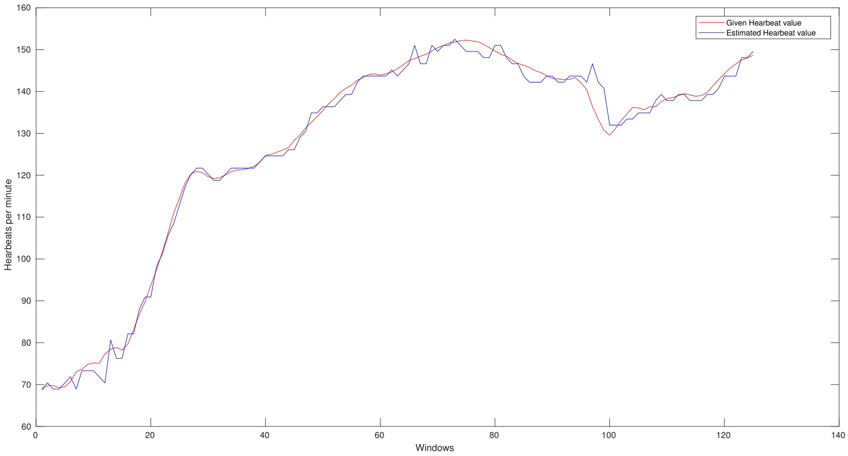
File Name	Average Value
$DATA\_01\_TYPE01$	4.27
$DATA\_02\_TYPE02$	1.69
$DATA\_03\_TYPE02$	1.69
$DATA\_04\_TYPE02$	12.33
$DATA\_05\_TYPE02$	9.81
$DATA\_06\_TYPE02$	1.48
$DATA\_07\_TYPE02$	1.33
$DATA\_08\_TYPE02$	13.91
$DATA\_09\_TYPE02$	1.34
$DATA\_10\_TYPE02$	10.02
$DATA\_11\_TYPE02$	2.47
$DATA\_12\_TYPE02$	5.95
$AVERAGE$	5.52

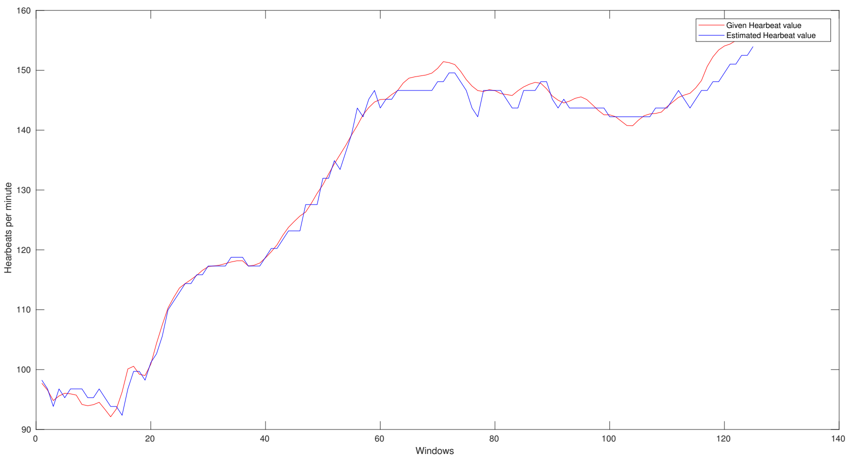
Shown above are the results of running the code on the given testing data. The average error is calculated as

$$\frac{1}{S}\sum_{i=1}^{S}|bpm(i) - actual(i)|$$

where $actual(i)$ is the given heart beat rate for the testing data for the i^{th} sample. Some plots for random test data used are attached below. The red line is for the original values given to us, and the blue lines are for the estimate heart rate.

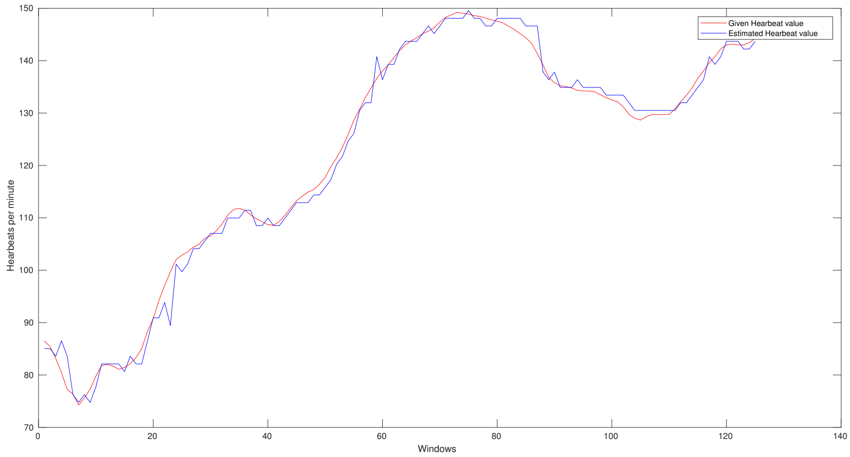


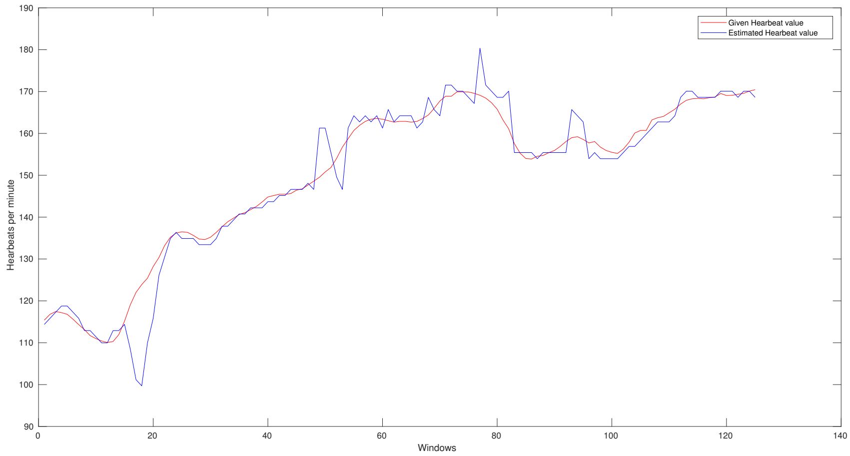




6 Conclusion

Here, we use a statistical denoising approach to clean up the noisy input to predict the heart rate. The accuracy that we obtain is fairly good. However, it can be improved further by using more post processing techniques to smoothen out the predictions as well as improving the initial estimate, and by checking Kalman filter estimates too as a comparison. The main motivation behind this approach was to use a causal filter so that the entire pipeline can be used to predict the heart rate in real-time.





7 References

Apart from the PPG estimation research papers we went through for the project, here are some links we used for understanding Wiener filters:

- `https://ocw.mit.edu/courses/earth-atmospheric-and-planetary-sciences/12-864-inference-from-data-and-models-spring-2005/lecture-notes/tsamsfmt2_6.pdf`
- `http://sharif.edu/~mbshams/files/13.pdf`
- `http://nptel.ac.in/courses/117103018/29`