1. Introduction

The aim of the laboratory was to familiarize with signal processing in Python using heartbeat. Task required to properly read the signal, convert it to frequency domain and filter out the noise. Alternatively heartpy library could be used to calculate BPM value.

What are the problems that can be met in real applications of signal processing?

- Wrong sample rate can render measurements useless (spectral leakage)
- Aliasing
- ADC errors
- Efficient processing of a lot of data

2. Results

Firstly, signal was read using wavefile module and plotted in time domain; we can see visible peaks and considerable amounts of noise

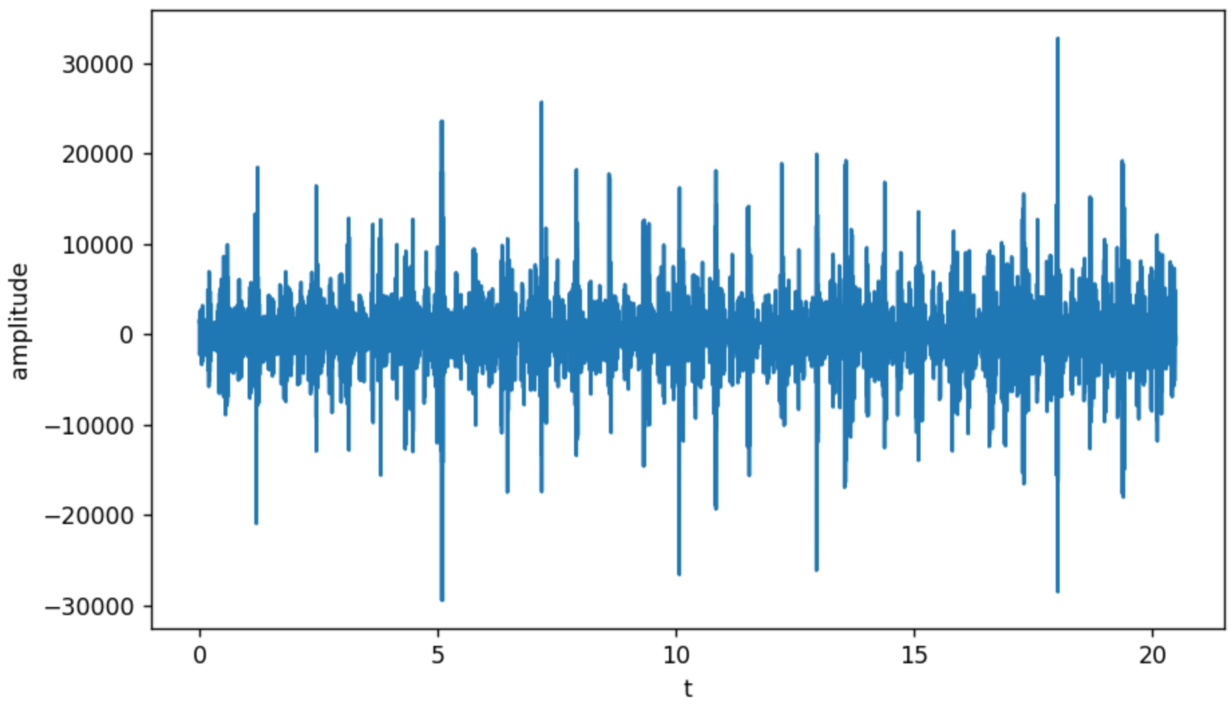

Figure 1 Raw signal in time domain

Next step was to perform FFT of signal to reveal frequencies.

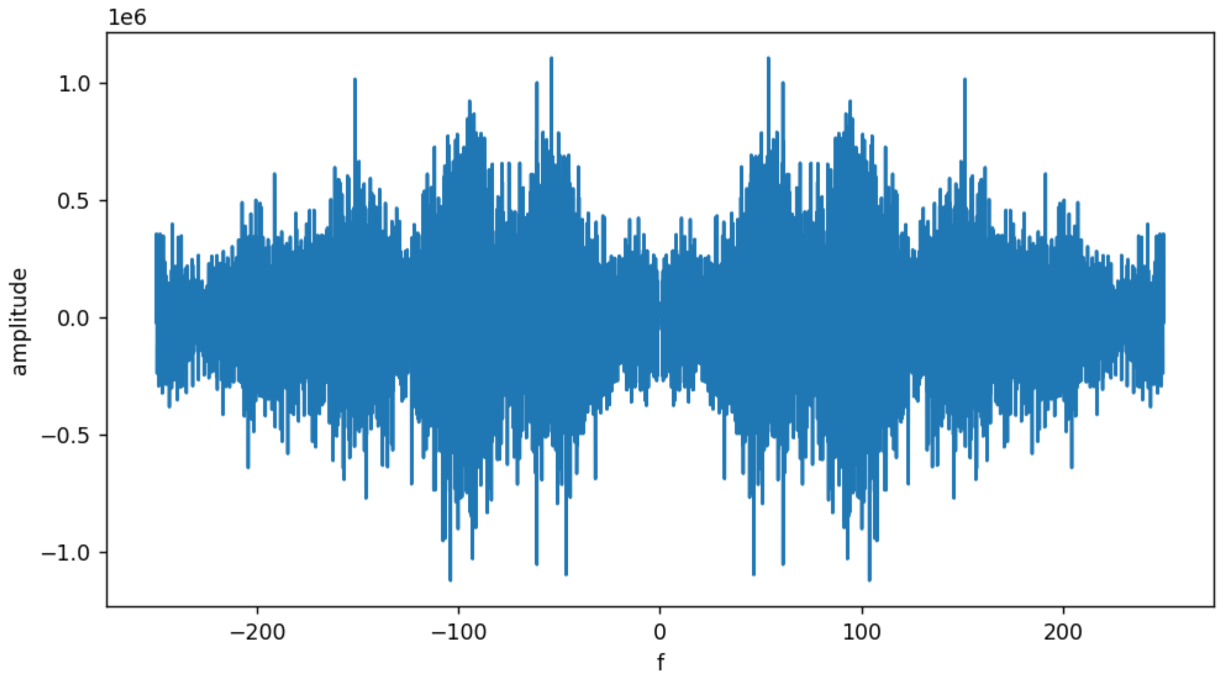

Figure 2 Unprocessed signal in frequency domain

Another 2 steps uses symmetry features of FFT to simplify the plot

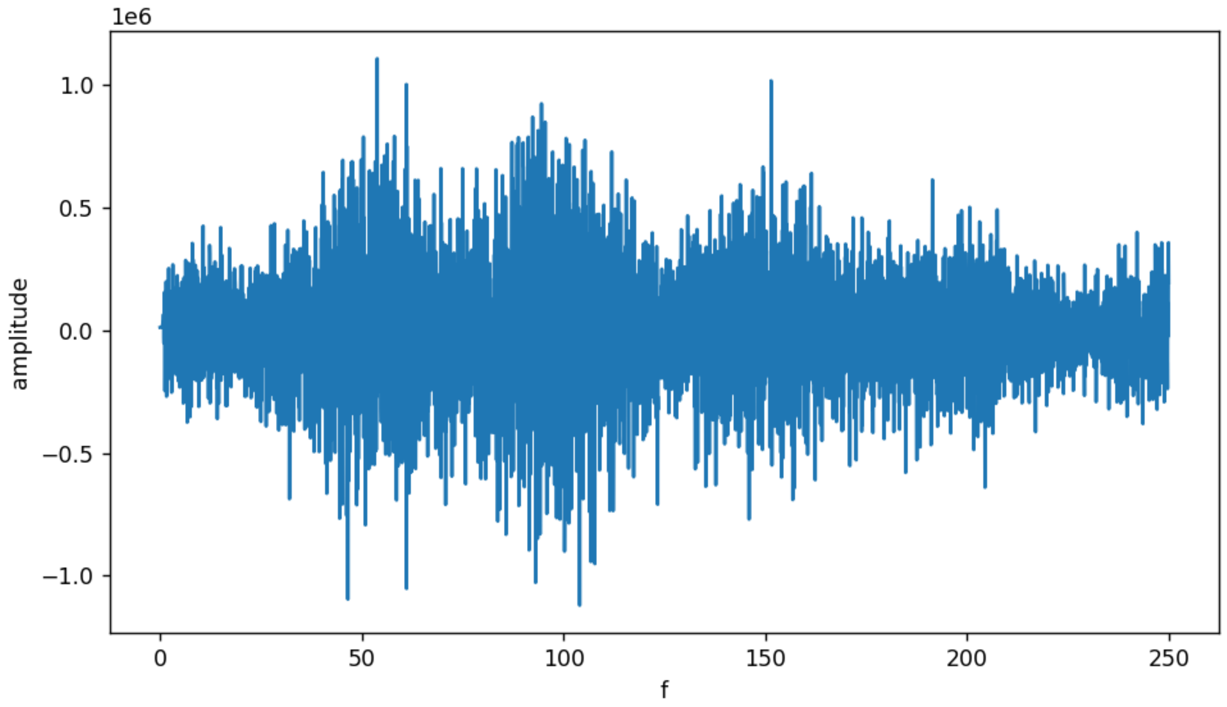

Figure 3 Signal in frequency domain without y-axis mirror

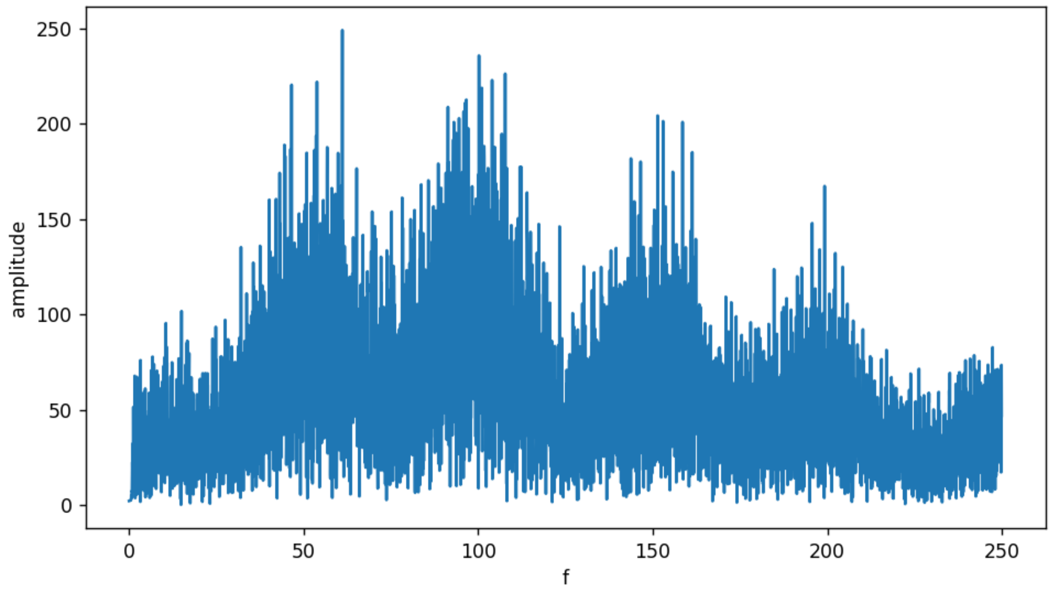

Figure 4 Processed signal in frequency domain

Next stop was to find useless range of frequencies and filter them out. Experimentally, f_L was chosen as 100 Hz and f_H as 200 Hz.

Using Butterworth bandpass filter (from scipy.signals) of 9th order with above parameters, signal was filtered. Below we can see output in time domain. As seen, a lot of noise was deleted while peaks were preserved. In other words, SNR was increased.

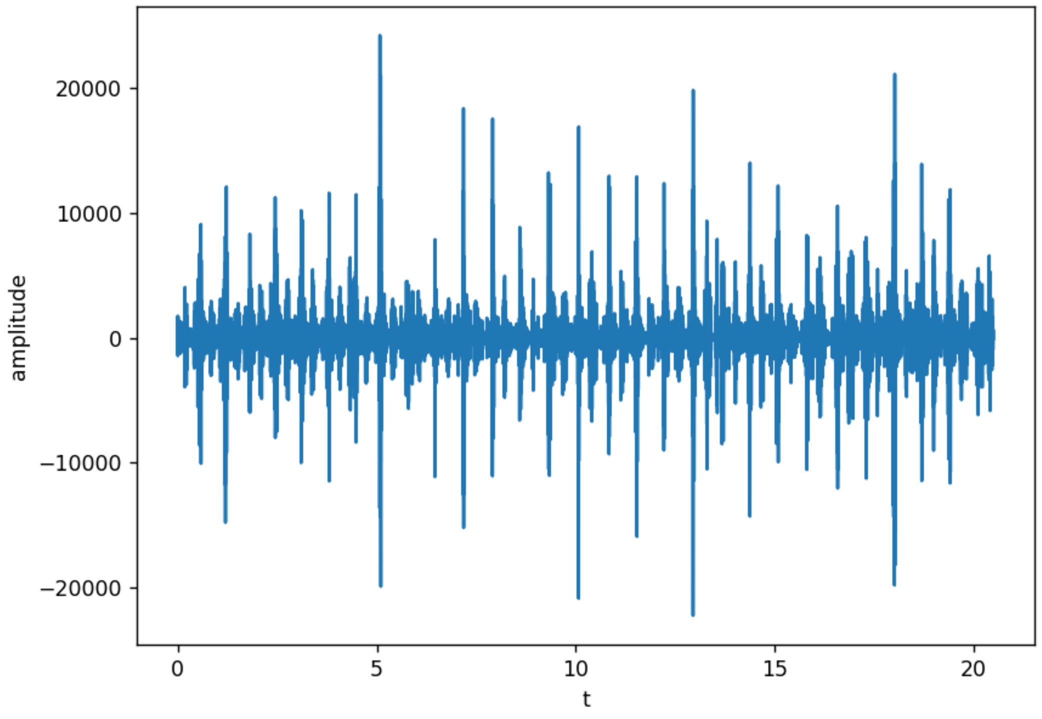

Figure 5 Signal filtered with 9th order Butterworth bandpass filter (f1 = 100 Hz, f2 = 200 Hz), in time domain

Out of curiosity, there was also “manual” approach to filter checked. Frequencies out of range were just toned down to 0.

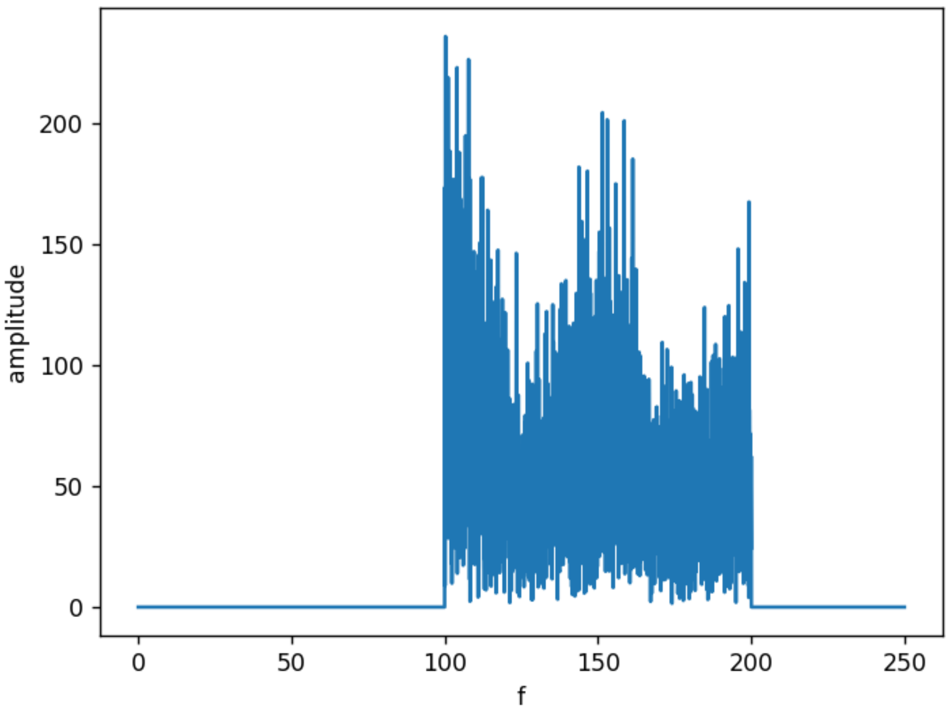

Figure 6 Signal filtered manually between 100 and 200 Hz

Below we can see time domain for “manual” filter. It does not look as clean as the proper Butterworth one.

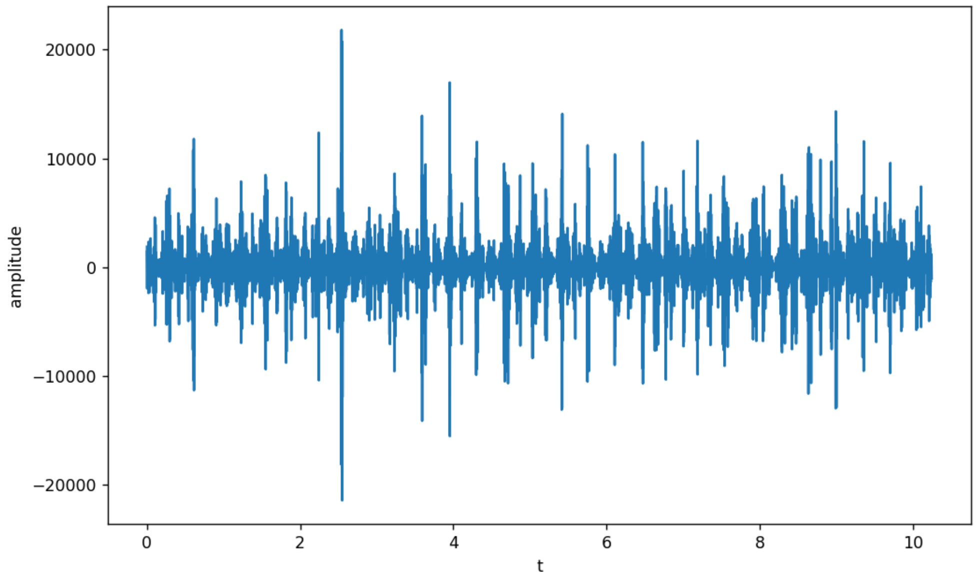

Figure 7 Manually filtered signal converted back to time domain

To sum up, below we can see comparison of original and filtered (with Butterworth) signal.

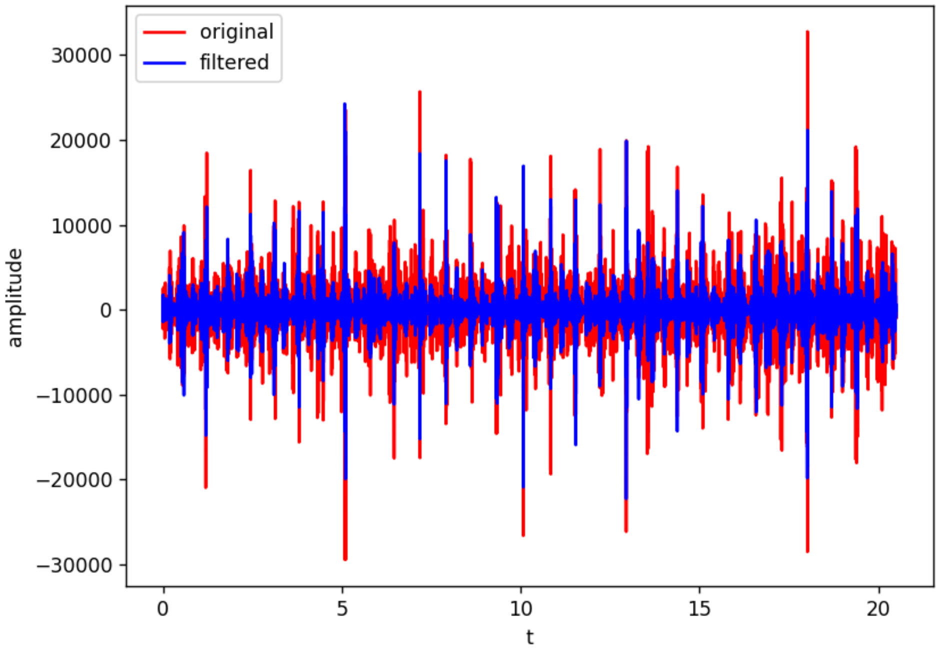

Figure 8 Original and filtered signal in time domain

Additionally, Heartpy library (specially designed to perform heart beat calculation) was used. As seen below, for unfiltered signal, it calculated heart rate as ~90 BPM (Fig. 8). Similar results were achieved for signal filtered with Butterworth (Fig. 9). Results for "manual" filter are quite different - ~124 BPM (Fig. 10)

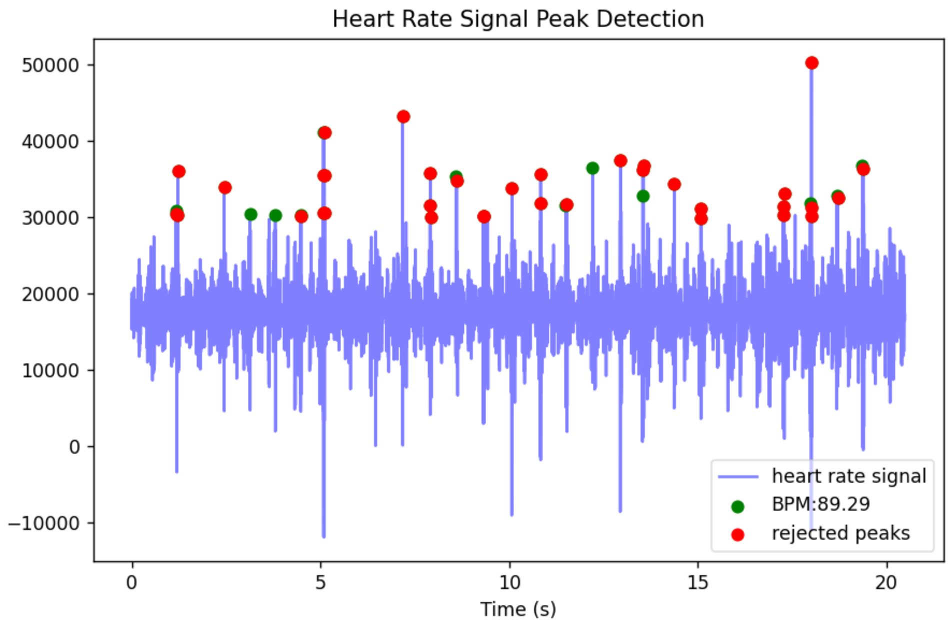

Figure 9 Unfiltered signal processed by Heartpy

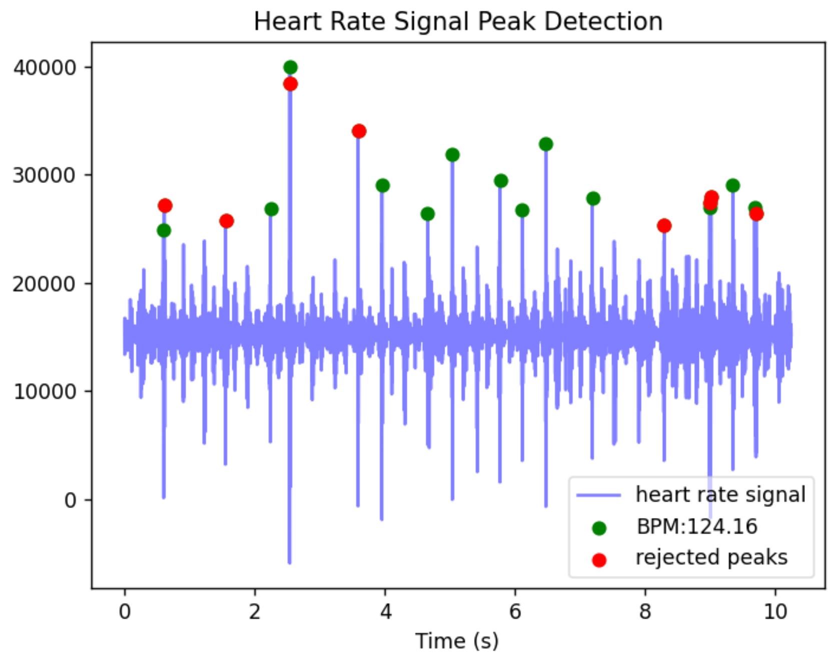

Figure 10 Manually filtered signal processed by Heartpy

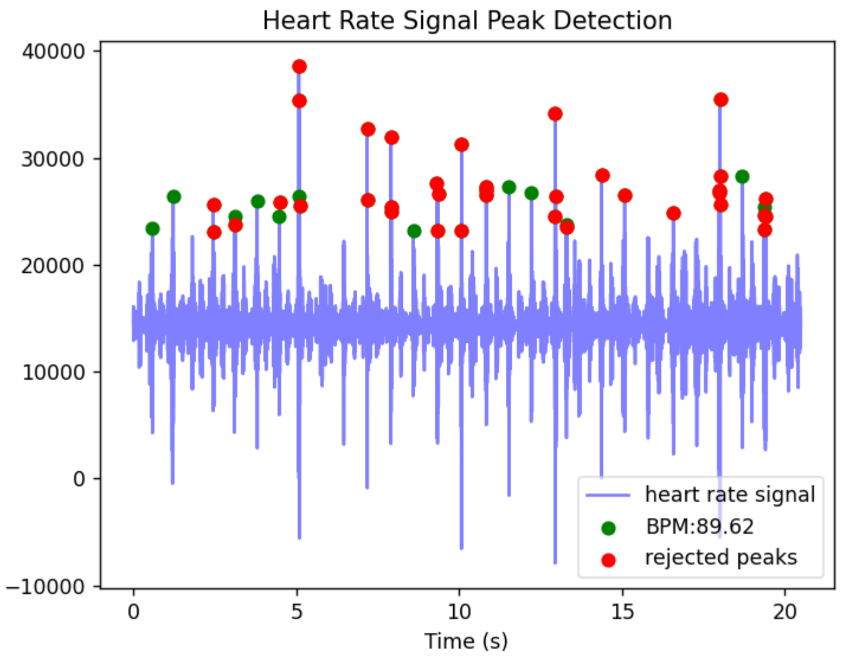

Figure 11 Signal filtered by Butterworth processed by Heartpy

In the end, .wav file was generated for filtered signal. There is a significant improvement of the audio quality. File was added to repository.