Instantaneous Amplitude Envelope Estimator using the Hilbert Transform

INDIAN INSTITUTE
OF TECHNOLOGY
PALAKKAD

Term Project Report

Shoeb Mohammed Ziauddin 121901038

April 25, 2022

Contents

1 Problem Statement

Design and Implement instantaneous amplitude envelope estimator by generating analytical signal using the Hilbert transformer. Use DFT for design of Hilbert transformer. Use heart sound signal with sampling rate of 2000 Hz. and Assume block duration of 10 seconds.

2 Solution

2.1 Introduction

The heart sound signal carries important information about the heart's functioning and its state of health, and it can be analysed using signal processing techniques to identify various cardiac disorders before the situation goes out of hand. And hence, the signal may be processed using a variety of signal processing techniques. The collection of the heart sound signal, noise reduction, sampling the signal at a certain frequency rate, feature extraction, training, and classification are the steps involved in processing and diagnosing the heart sound data. In this report, the steps are followed until the feature extraction stage, and the feature of interest is the instantaneous amplitude envelope of the heart sound signal. And the signal processing technique applied to extract the said feature is by using a DFT based Hilbert Transformer.

2.2 Theory

2.2.1 Hilbert Transform

The Hilbert transform is a linear operator that takes a real-valued function, $u(t)$, and transforms it into another real-valued function, $H[u(t)]$ or $\hat{u}(t)$ in the same domain. Hilbert transform of a signal is also defined as a transform that shifts the phase angle of every component of the signal by 90°. The Hilbert transform is also a component of the analytical representation of a real-valued signal, hence its importance in signal processing. The Hilbert Transform of $u(t)$ is given as,

$$\hat{u}(t) = u(t) * \frac{1}{\pi t} \tag{1}$$

$$DFT[\hat{u}(t)] = U(k) \cdot (-j \cdot sgn(k)) \tag{2}$$

Basically, to obtain the Hilbert transform in frequency domain, the positive frequencies of the spectra of a signal are multiplied with $-j$ and the negative frequencies are multiplied with j. And the IDFT of the resulting spectra will give us the Hilbert transform in time domain.

2.2.2 Analytical Signal Generation

After obtaining $\hat{u}(t)$, we can generate the analytical signal $z(t)$ by the following equations,

$$z(t) = z_r(t) + jz_i(t) = u(t) + j\hat{u}(t) \tag{3}$$

It is also interesting to note that, on taking the DFT of the (3) we obtain,

$$z(t) = IDFT[U(k) + j\hat{U}(k)] \tag{4}$$

That is, instead of taking IDFT of the frequency domain Hilbert transform obtained in (2), we can compute the equation (4) while working in the frequency domain, and directly obtain the time domain analytical signal $z(t)$ upon performing the IDFT.

2.2.3 Obtaining the Instantaneous Amplitude Envelope

Now that we have obtained the analytical signal $z(t)$, the instantaneous amplitude envelope is just the magnitude of $z(t)$. That is,

$$\text{instantaneous amplitude} = |z(t)| \tag{5}$$

But, upon plotting the obtained Instantaneous Amplitude Envelope, it can be observed that it is not 'smooth'. Hence, in order to isolate the effects of periodic component and to remove unwanted noise from the signal, we use a 100-tap moving average filter. And also for better legibility we plot a section of the signal rather than the entire 10 seconds block.

2.3 MATLAB Code

The above theory was implemented in MATLAB and the following code was written.
Drive Link: DSP

```
%Design and Implement instantaneous amplitude envelope estimator
%by generating analytical signal using the Hilbert transformer.
%Use DFT for design of Hilbert transformer. Use heart sound signal
%with sampling rate of 2000 Hz. and Assume block duration of 10 second

fs = 2e3;                                   % Sampling Frequency in Hz

[y, Fs] = audioread("beats.wav");           % Obtaining the signal from ...
    the audio file
y = y';

t = (0:length(y)-1)/Fs;                     % Time Domain of the ...
    obtained signal

y_fft = R2_DIT(y, -1);                      % Obtaining the analytical ...
    signal
y_fft_ideal = fft(y);                       % Ideal FFT for verification

complex_y_fft = 1j*y_fft;
n = length(y_fft);

positive_frequencies = 2:floor(n/2)+mod(n, 2);
negative_frequencies = ceil(n/2)+1+~mod(n, 2):n;

y_fft(positive_frequencies) = y_fft(positive_frequencies) + ...
    -1j*complex_y_fft(positive_frequencies);
```

```
y_fft(negative_frequencies) = y_fft(negative_frequencies) +  ...
    1j*complex_y_fft(negative_frequencies);

z = R2_DIT(y_fft, 1);

inst_amp = abs(z);                              % Obtaining instantaneous ...
    amplitude envelope

z_ideal = hilbert(y);                           % Obtaining the analytical ...
    signal using in-built function

inst_amp_ideal = abs(z_ideal);                  % Obtaining instantaneous ...
    amplitude envelope of ideal analytical signal

L = 100;                                        % Designing an MA filter to ...
    smoothen the envelope
b = (1/L)*ones(1, L);
a = 1;

inst_amp_smooth = zeros(n, 1);

for i = 1:n
    temp = 0;
    for j = 1:L
        if (i-j > 0)
            temp = temp + inst_amp(i-j);
        end
    end
    inst_amp_smooth(i) = temp/L;
    temp = 0;
end

delay = (L-1)/2;                                % Delay introduced due to ...
    the MA filter

a0 = 2*fs;                                      % Taking a section of the ...
    entire signal for better legibility
b0 = 4*fs;

figure
plot(t(a0:b0), y(a0:b0));
hold on;
plot(t(a0:b0)-(delay/fs), inst_amp_smooth(a0:b0), "r");
xlabel("Time (s)");
xlim([a0/fs b0/fs]);
title("Smooth Instantaneous Amplitude Envelope using Moving Average ...
    Filter");

figure
plot(t(a0:b0), y(a0:b0));
hold on;
plot(t(a0:b0), inst_amp(a0:b0), "r");
xlabel("Time (s)");
title("Instantaneous Amplitude Envelope using Hilbert's Transform");
```

```
69
70 figure
71 plot(t(a0:b0), y(a0:b0));
72 hold on;
73 plot(t(a0:b0), inst_amp(a0:b0), "g");
74 hold on;
75 plot(t(a0:b0), inst_amp_ideal(a0:b0), "r");
76 title("Instantaneous Amplitude Envelope from both methods");
77 xlabel("Time (s)");
78 legend("Written Code", "In-built Function", "location", "best");
79
80 figure
81 plot(t(a0:b0), y(a0:b0));
82 hold on;
83 plot(t(a0:b0), inst_amp_ideal(a0:b0), "r");
84 xlabel("Time (s)");
85 title("Instantaneous Amplitude Envelope using in-built function");
86
87 figure
88 plot(t(a0:b0), y(a0:b0));
89 xlabel("Time (s)");
90 title("The Section of Interest");
91
92 figure
93 plot(t, y);
94 xlabel("Time (s)");
95 title("Heart Sound Signal sampled at 2kHz");
96
97 function [y] = R2_DIT(x, type)              %function computes FFT when ...
      type = -1 and IFFT when type = 1
98 p=2^nextpow2(length(x));
99 x=[x zeros(1,p-length(x))];
100 N=length(x);
101 S=log2(N);
102 Half=1;
103 x=bitrevorder(x);
104 for stage=1:S
105     for index=0:(2^stage):(N-1)
106         for n=0:(Half-1)
107             pos=n+index+1;
108             pow=(2^(S-stage))*n;
109             w=exp((type*1j)*(2*pi)*pow/N);
110             a=x(pos)+x(pos+Half).*w;
111             b=x(pos)-x(pos+Half).*w;
112             x(pos)=a;
113             x(pos+Half)=b;
114         end
115     end
116 Half=2*Half;
117 end
118 if type == 1
119     y = x/N;
120 else
121     y = x;
```

```
122 end
123 end
124
125 function [y] = DFT(x, type)      %#ok<*DEFNU> %function computes DFT ...
       when type = -1 and IDFT when type = 1
126     N = length(x);
127     y = zeros(N, 1);
128     for k = 1:N-1
129         for n = 1:N
130             y(k+1) = y(k)+x(n)*exp(type*1j*2*pi*n*k/N);
131         end
132     end
133     if type == 1
134        y = y/N;
135     end
136 end
```

2.3.1 Note

In Line 16: Here, the multiplication with j is not a part of obtaining Hilbert transform, but instead multiplication of j in (4). As the Hilbert Transformer is a linear operator, multiplying constants to the signal in any order will not affect the end result.

In Line 22: Here is where the Hilbert transform takes place, as it can be seen here, the signal "complex_y_fft" was obtained before hand so the expression in this line does not becomes confusing.

In Line 33: The order of the moving average filter is taken arbitrarily.

In Line 50: Since we are passing the obtained instantaneous amplitude envelope through a moving average filter, a delay of $\frac{(M-1)}{2}$ samples is introduced in the signal, this term will later be subtracted from time array while plotting the delayed signal.

Functions at the End: The problem statement asked to use DFT, but since it was taking too long to run, I wrote the code for 2-Radix DIT FFT and used that instead of the DFT function written here.

3 Results

3.1 Plots

The following plots were obtained by simulating the code,

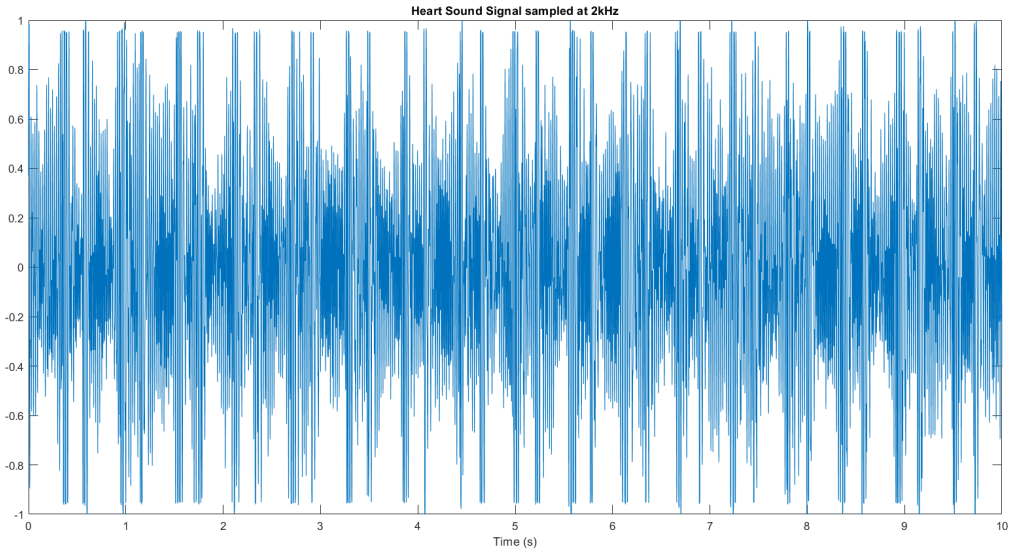


Figure 1: Heart Sound Signal Sampled at 2kHz

Plotting the entire block of 10 seconds was giving illegible graphs, hence we take a section of interest.

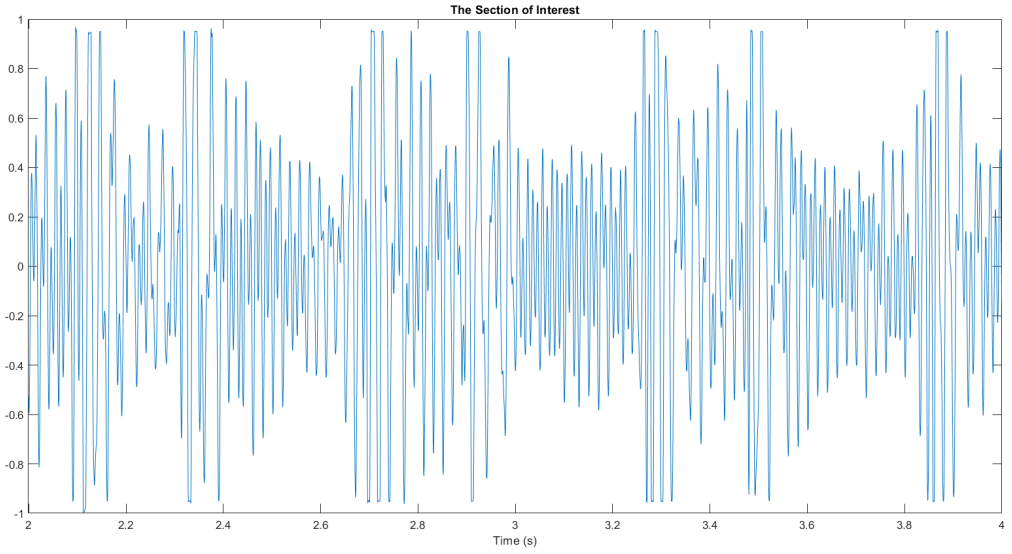


Figure 2: The Section of Interest

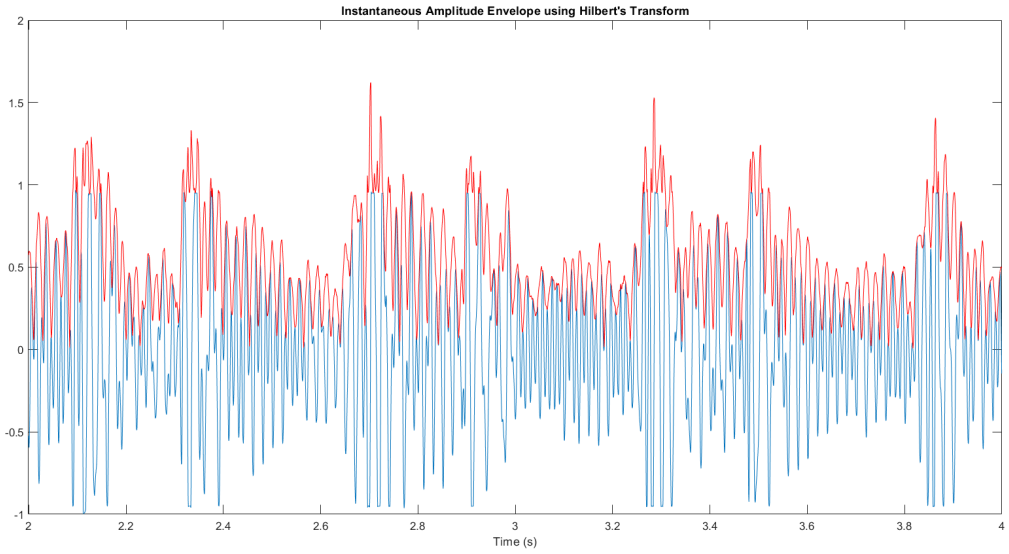


Figure 3: Instantaneous Amplitude Envelope using Hilbert's Transform

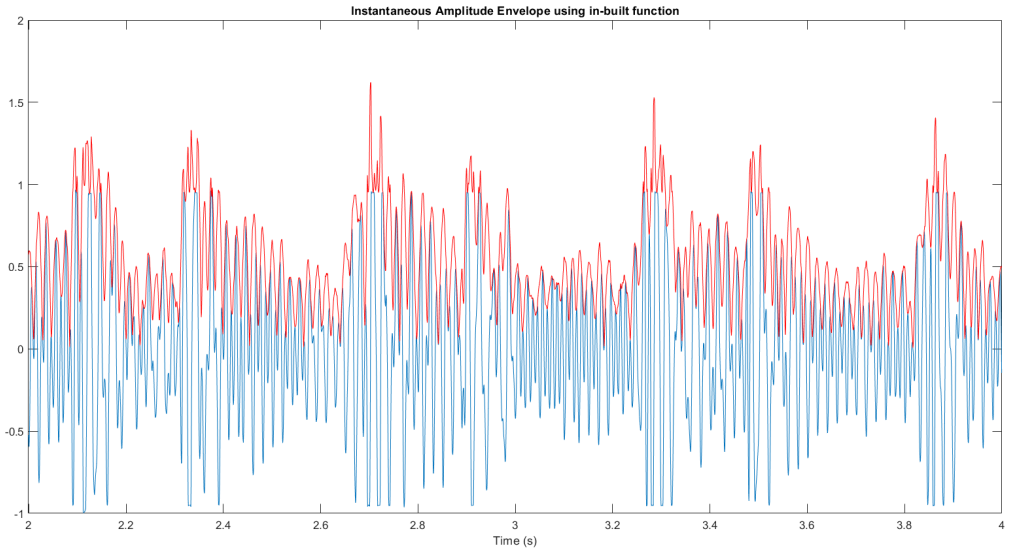


Figure 4: Instantaneous Amplitude Envelope using in-built function

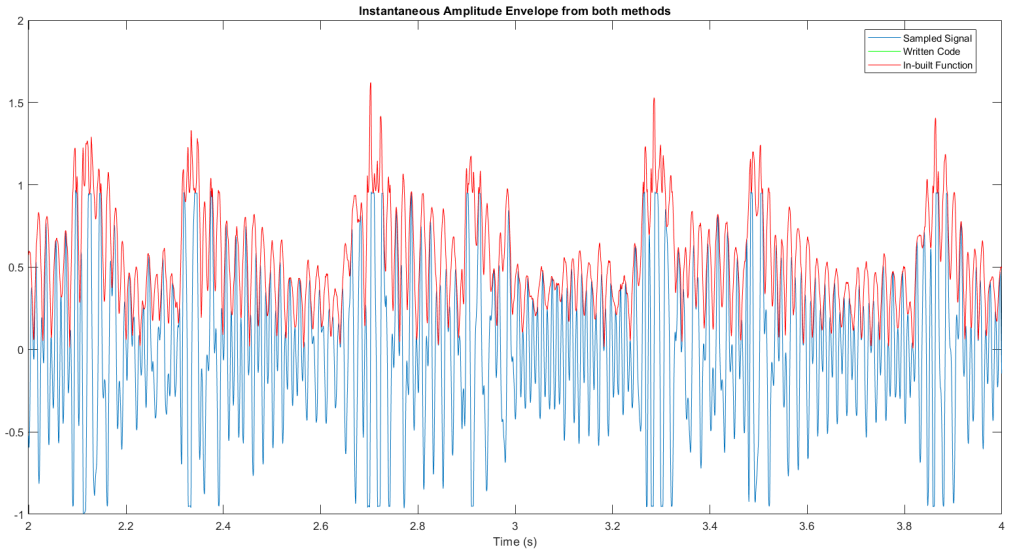


Figure 5: Instantaneous Amplitude Envelope from both methods

Note: The two envelopes almost completely overlap each other, hence they look like a single curve in this plot.

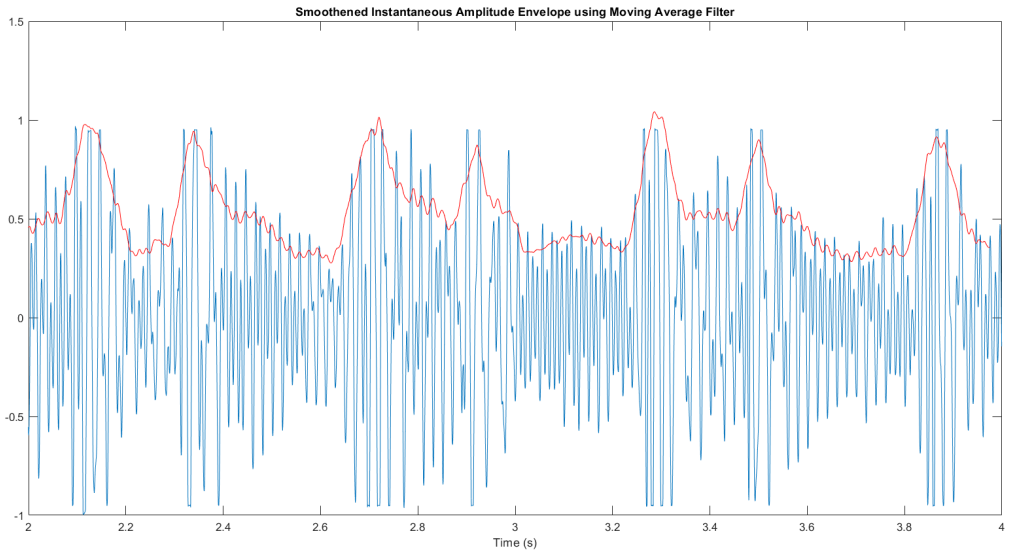
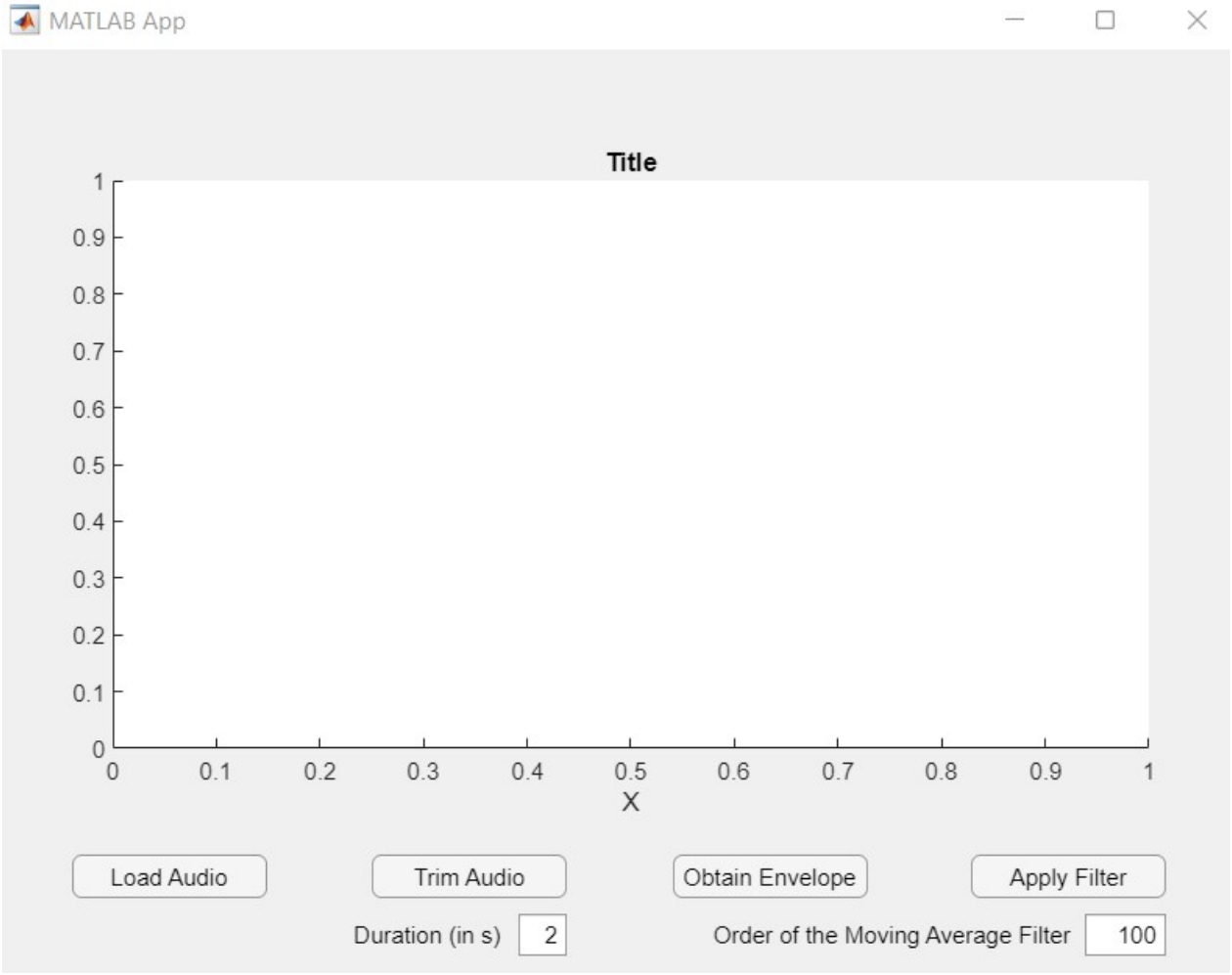


Figure 6: Smooth Instantaneous Amplitude Envelope using Moving Average Filter

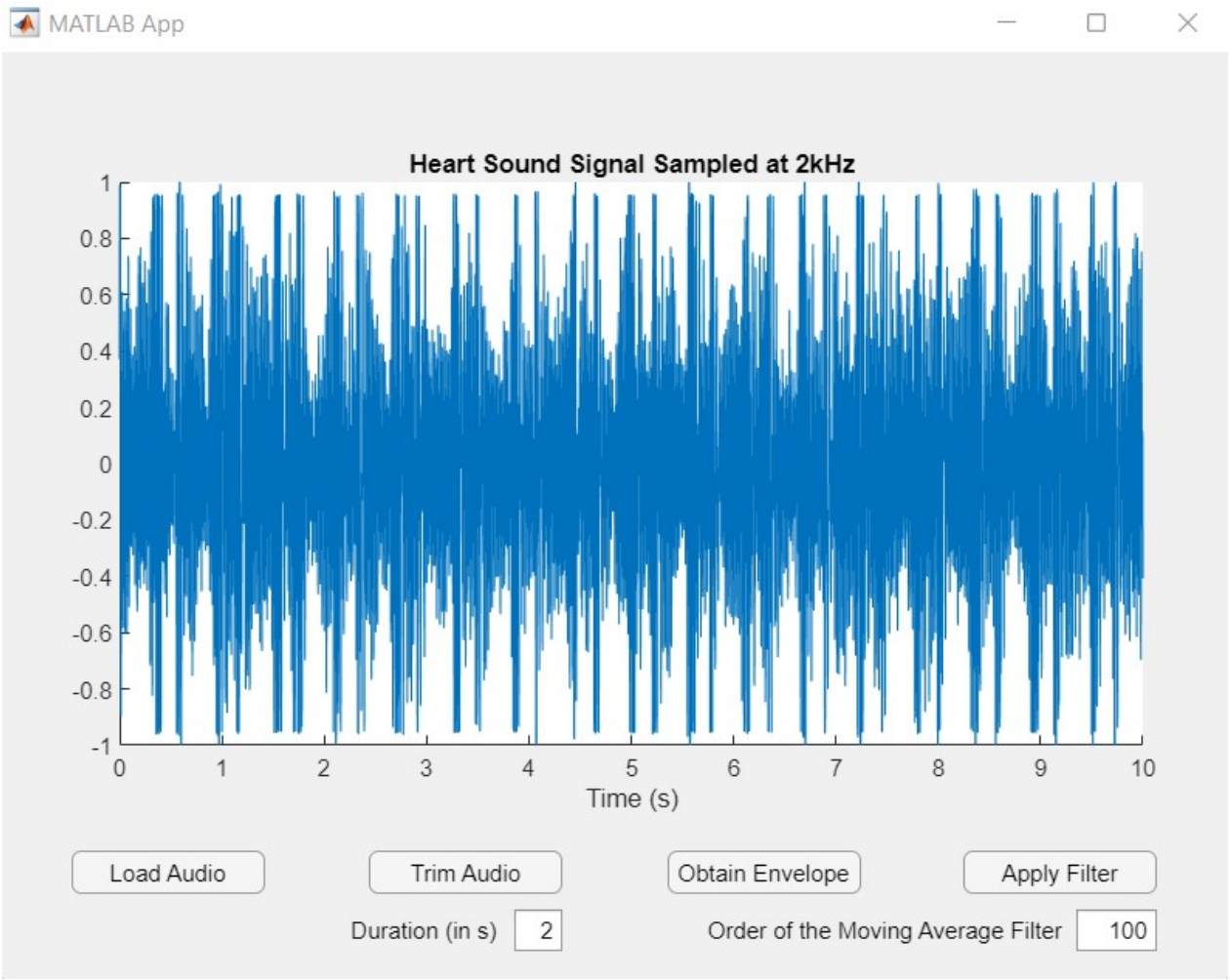
3.2 GUI

The following GUI was implemented using MATLAB App Designer,
Drive Link: Packaged App

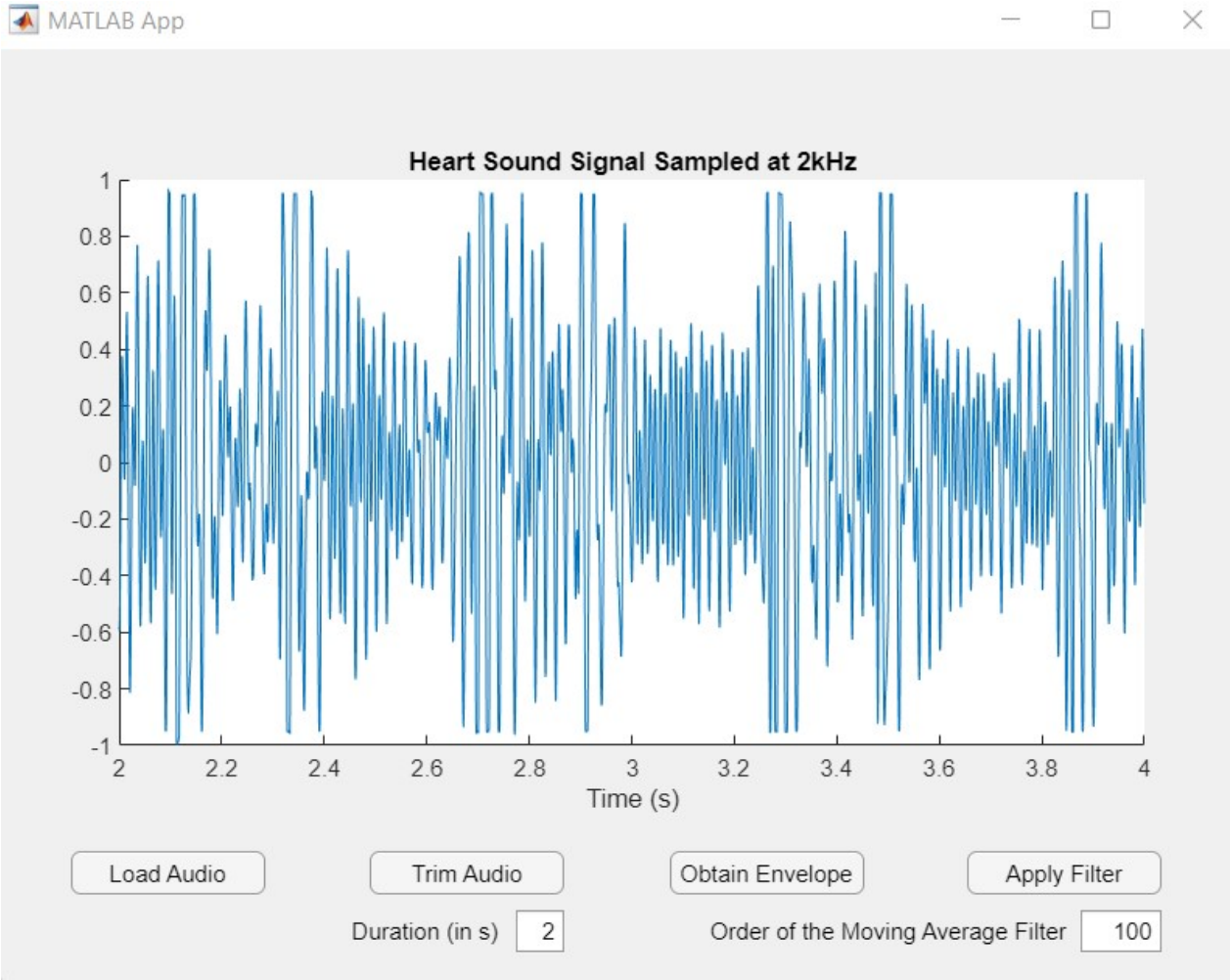
GUI when the code is run:



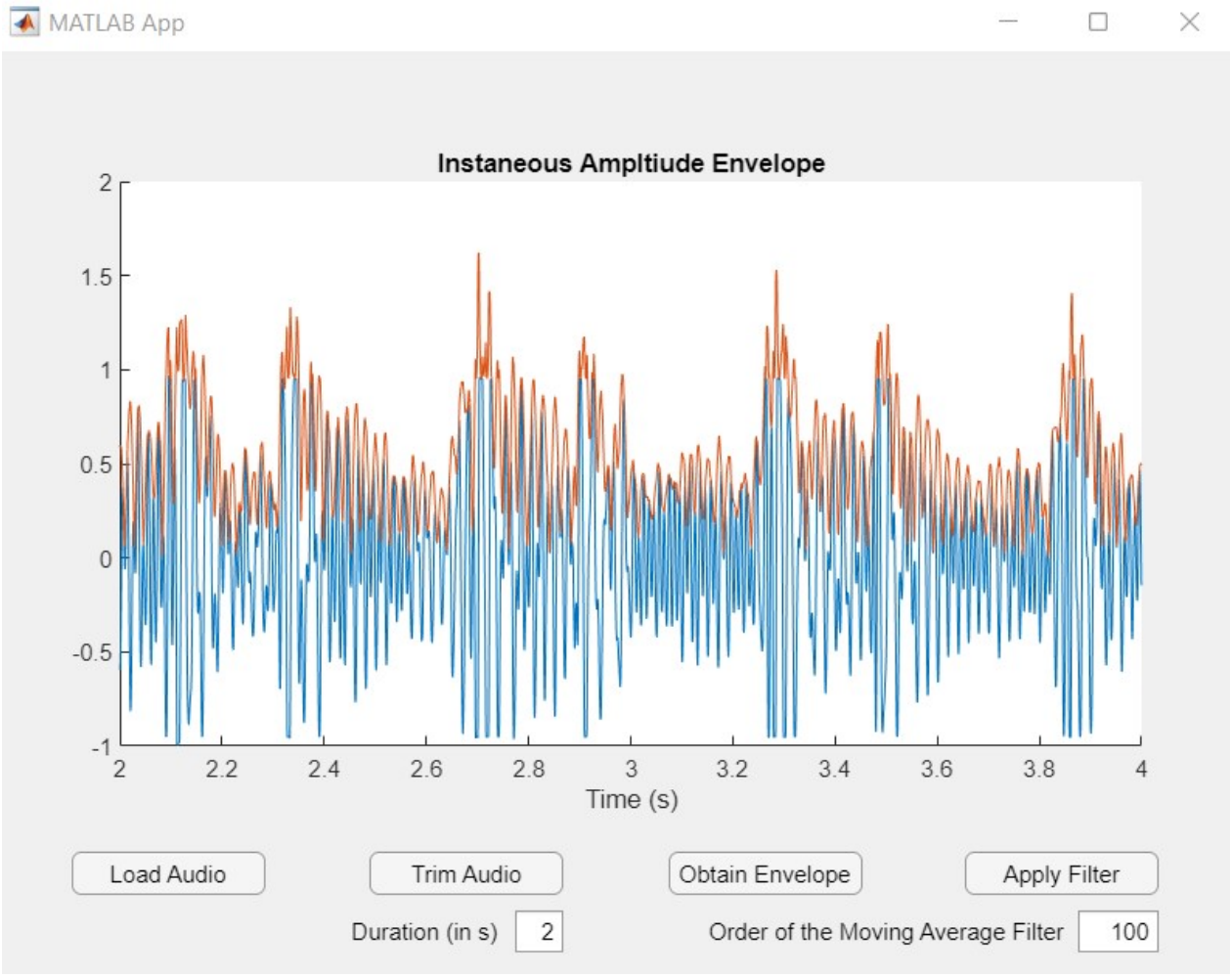
When *Load Audio* is clicked:



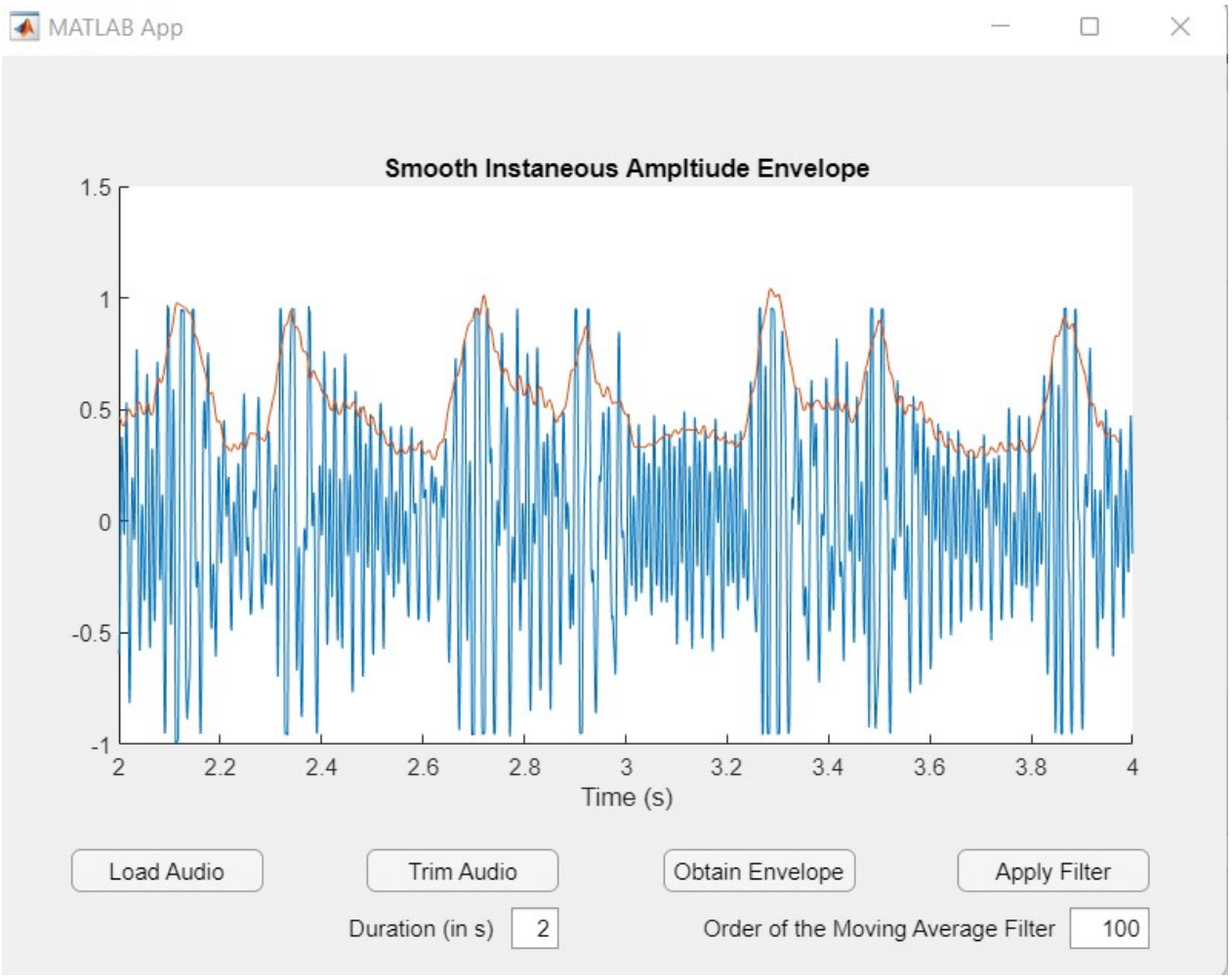
When *Trim Audio* is clicked:



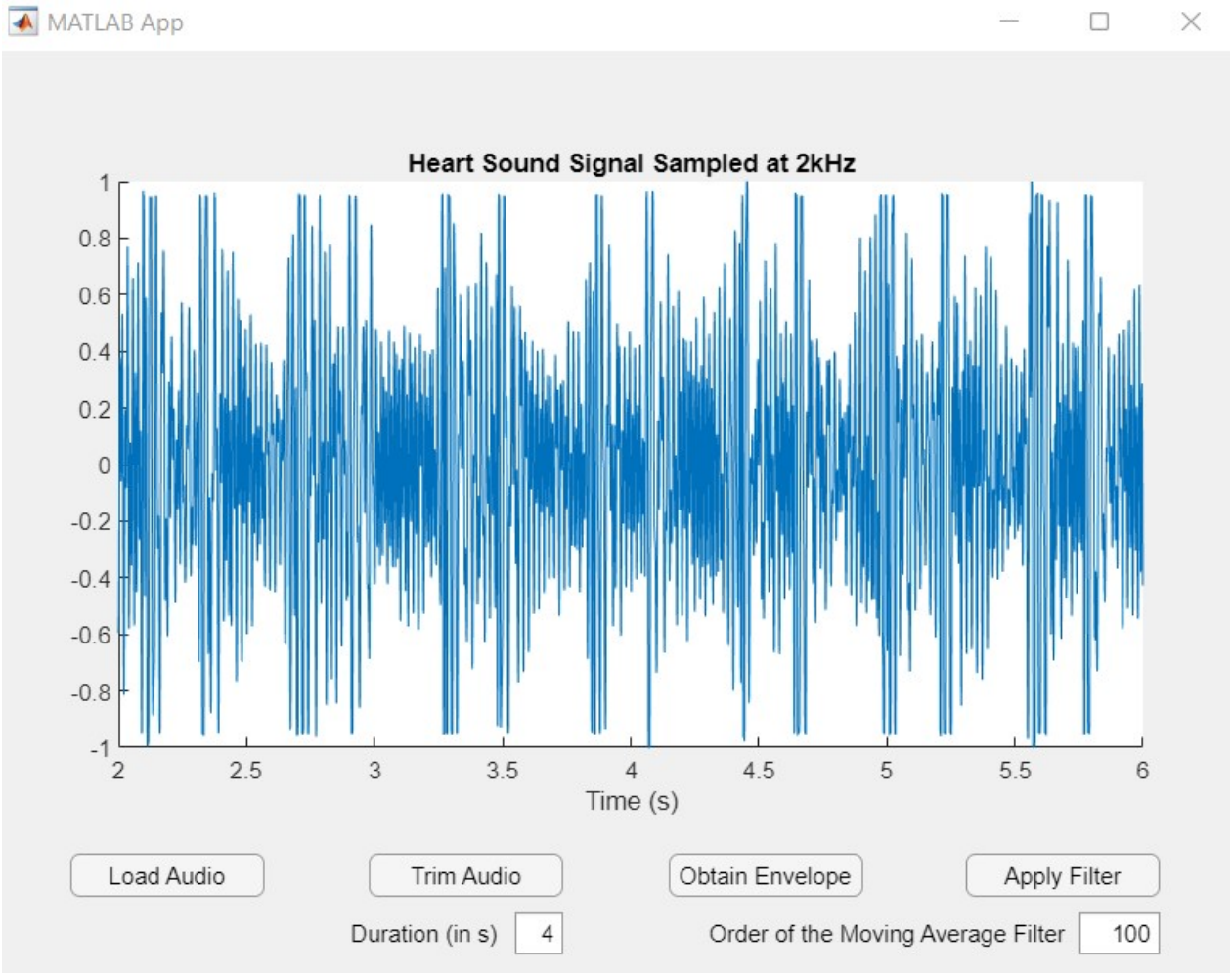
When *Obtain Envelope* is clicked:



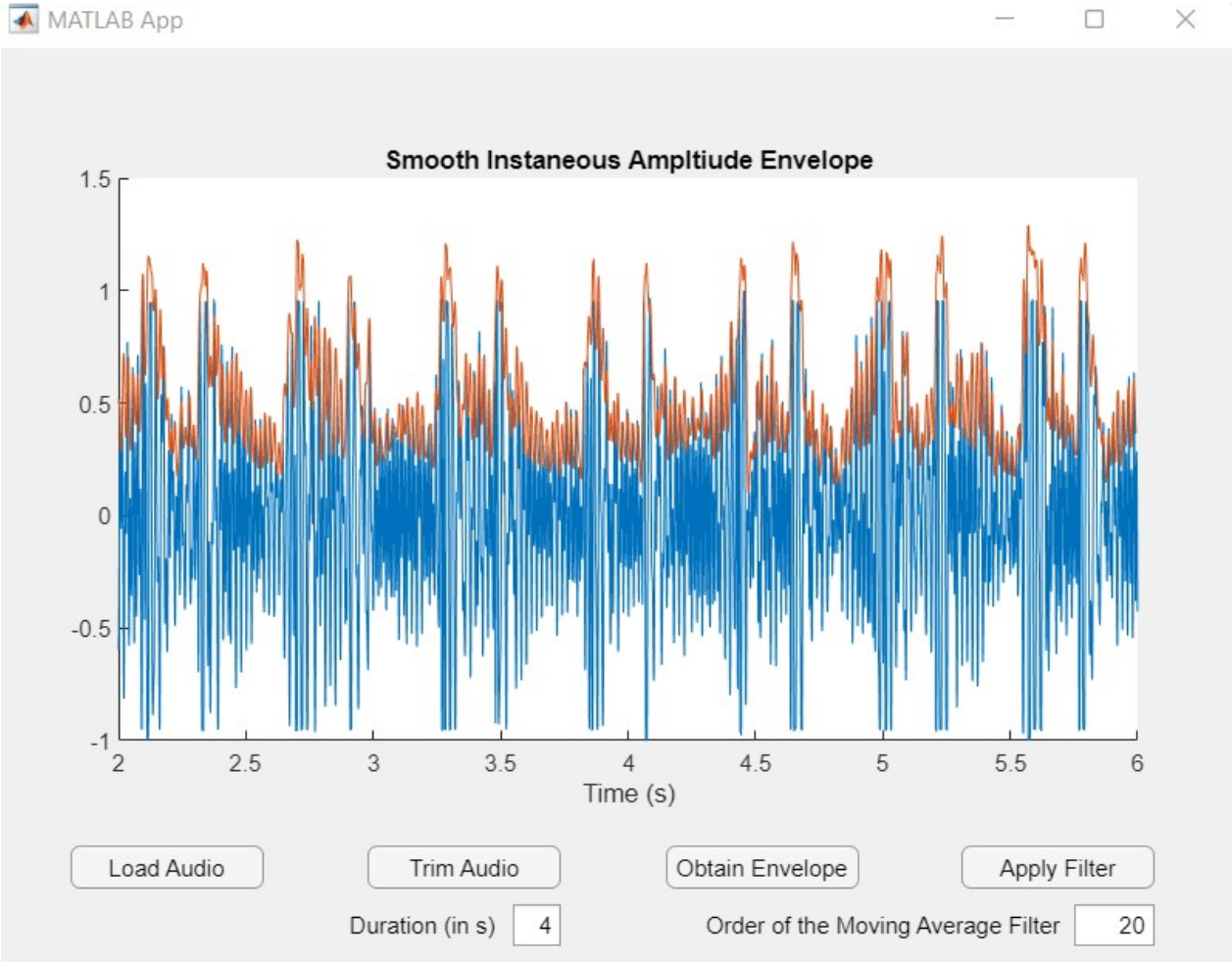
When *Apply Filter* is clicked:



When *Trim Audio* is clicked after changing the duration:



When *Apply Filter* is clicked after changing the order:



4 Challenges

The following challenges were encountered while doing the project,

1. Due to the lack of proper recording apparatus, there was a presence of significant amount of noise in the signal. So I used the noise reduction effect in audacity software to tackle this issue.
2. Due to the presence of noise, even after pre-processing in audacity, the obtained instantaneous amplitude envelope also had significant noise in it. To resolve this issue, I have used a MA filter to smoothen the envelope.

5 Observations

While doing the project, I have made the following observations,

1. The Heart Sound Signal, is roughly composed of two peaks at periodic intervals, most probably corresponding to the systolic and diastolic phases of the heart.
2. And the smoothing operation of a MA filter is simply brilliant. As it can be seen in the report, significant amounts of unwanted noise was removed from the envelope just by passing it through a simple MA filter.
3. Since the obtained envelope contains the information regarding the period and intensity of heart beat, one can determine the presence of abnormalities like murmurs, by examining the envelope.

6 Conclusion

It can be concluded from this report that, Estimating the Instantaneous Amplitude Envelope of the analytical signal of the sampled Heart Sound signal can be done with ease using the DFT based Hilbert Transformer approach. The envelope obtained can be processed further to diagnose the health of the heart.