DSP Laboratory
Experiment 4

Manideep Mamindlapally
(17EC10028)
Tuesday Batch

Aim

Through this experiment we aim to study and implement frequency band wise representation of a temporal audio signal and its reconstruction using just the envelopes of signal.

1 Division into frequency bands

For the purpose of frequency wise representation of the temporal signal, a frequency range of $90Hz$ to $5.76KHz$ was taken and divided logarithmically into a chosen number of sub ranges, N.
For each such frequency sub range, a fourth order butterworth filter was designed and passed with the normalised temporal audio signal, $x(t)$.

2 Envelope extraction and signal reconstruction

Real signals $x_r^{(i)}(t)$ was obtained on passing through each filter bank $l^{(i)}$. An hilbert transform is performed on the signal to obtain $\tilde{x}_r^{(i)}(t)$. The envelope was determined as $|\mathcal{H}^{(i)}(t)|$.

$$\begin{aligned} \tilde{x}_r^{(i)}(t) &= l^{(i)} \circledast x(t) \\ \mathcal{H}^{(i)}(t) &= x_r^{(i)}(t) + j\tilde{x}_r^{(i)}(t) \end{aligned} \tag{1}$$

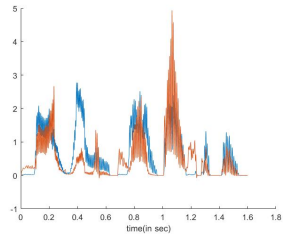
The obtained envelope was smoothened with a low pass filter and then modulated with gaussian noise, $n(t)$ to gernerate a signal $y^{(i)}(t)$. The output temporal audio signal, $y(t)$ was reconstructed as

$$y^{(i)}(t) = |\mathcal{H}^{(i)}(t)|n(t) \tag{2}$$

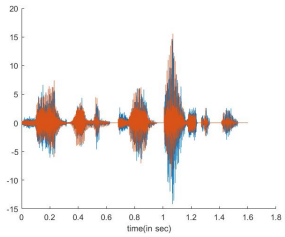
$$y(t) = \sum_i y^{(i)}(t) \circledast l^{(i)} \tag{3}$$

The output signal $y(t)$ was normalised and saved as an audio file for listening through a audio player software. Here is a plot of the normalised temporal input and output audio waveforms.

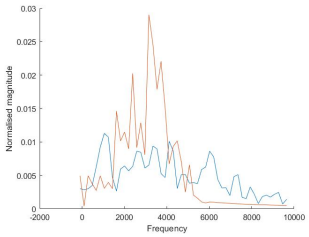
Plots



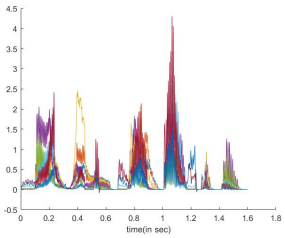
Envelopes $|\mathcal{H}^{(i)}(t)|$ for N=2



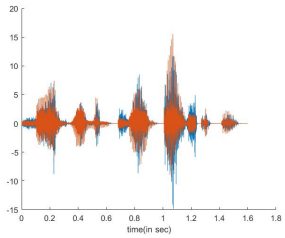
Original temporal audio signal(red) vs reconstructed signal(blue) for N=2



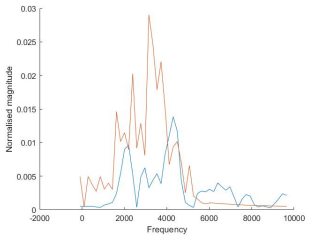
DFT of original audio signal(red) vs reconstructed signal(blue) for N=2



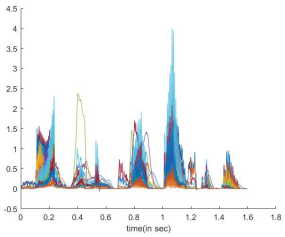
Envelopes $|\mathcal{H}^{(i)}(t)|$ for N=8



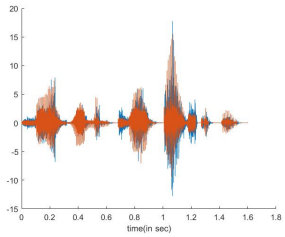
Original temporal audio signal(red) vs reconstructed signal(blue) for N=8



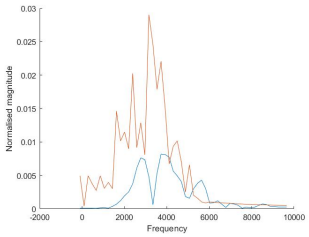
DFT of original audio signal(red) vs reconstructed signal(blue) for N=8



Envelopes $|\mathcal{H}^{(i)}(t)|$ for N=16



Original temporal audio signal(red) vs reconstructed signal(blue) for N=16



DFT of original audio signal(red) vs reconstructed signal(blue) for N=16

Discussion

- Through this experiment, we wanted to reconstruct a temporal sound signal out of a set of different bandlimited envelopes.
- This set of signals is obtained by passing through bandpass filters. The obtained set of bandlimited signals could be used for different audio recognition functions. In this experiment however, we focussed on reconstructing the original signal out of these samples.
- The theory of Hilbert transforms has been employed to arrive at envelopes of the bandlimited outputs. These outputs seemed to have a few rough spikes which were compensated by passing through a low pass filter.
- As expected, it was observed that intelligibility of the reconstructed signal improves with increased number of bands.
- The reason for the above can be attributed to the improved details of the envelopes traced for higher number of bands. This was also accompanied by lower noise component in the high frequency region of the frequency response.
- It should be noted that although the output was intelligible, it's frequency response was not same or even close to the original audio. It can be concluded from this observation that the intelligibility of the speech is largely due to its envelope and not due to the finer structures within.

MATLAB code

main.m

```
% =============== CONSTANTS =====================================
NUM_BANDS = 16;
HIGH = 5760;
LOW = 90;
FILT_ORDER = 4;
envelope_low_f = 240;

% =============== SIGNAL PREPROCESSING ==========================
[input_audio, Fs] = audioread("fivewo.wav");
t = (1:size(input_audio,1))/Fs;
alpha = rms(input_audio);
x = input_audio/alpha;
signal_size = size(x,1);

% =============== BAND DIVISION and ENVELOPE filtering ===========
r = (HIGH/LOW)^(1/NUM_BANDS);
output_audio = zeros(signal_size,1);
figure;
hold on;
for n=1:NUM_BANDS

    l_f = LOW*r^(n-1);
    h_f = LOW*r^(n);
    [b1,a1] = butter(FILT_ORDER/2, [l_f, h_f]*2/Fs);

    y = filter(b1,a1,x);
    Yr = hilbert(y);
    rough_envelope = abs(Yr);

    [b2,a2] = butter(FILT_ORDER, envelope_low_f/Fs);
    envelope = filter(b2,a2,rough_envelope);

    noise = randn(signal_size, 1);
    signal = envelope.*noise;

    bandlimited_signal = filter(b1,a1,signal);
    output_audio = output_audio + bandlimited_signal;

    %plot envelope
    plot(t,envelope);
    xlabel("time(in sec)");
end
hold off;

% ================ OUTPUT =======================================
output_audio = alpha*output_audio/rms(output_audio);
audiowrite("result.wav",output_audio,Fs);

%Plots
figure;
hold on;
plot(t,output_audio);
plot(t,input_audio);
```

```
    xlabel("time(in sec)");
    hold off;

    N=512;
    figure;
    hold on;
    dftplot(output_audio,Fs,N);
    dftplot(input_audio,Fs,N);
    hold off;

%==========================================================================
%==========================================================================
dftplot.m

function []= dftplot(x,Fs,N)
    f=linspace(-Fs/2,Fs/2,N);
    y=fft(x,N)/N;
    y_spect=fftshift(y);
    m=abs(y_spect);
    plot(f(N/2:N/2+N/10),m(N/2:N/2+N/10));
    xlabel("Frequency");
    ylabel("Normalised magnitude");
end
```