Lab 4 Report: Frequency-Domain Signal Filtering

Youssef Beltagy and Samuel Hunter
AUT21 BEE 235

December 5, 2021

1 Abstract

In this lab, we used the fourier transform of a signal to filter it. We observed how low pass, high pass, and band pass filters affect the signal. We listened to the difference in sound and observed the change in the time and frequency domain plots.

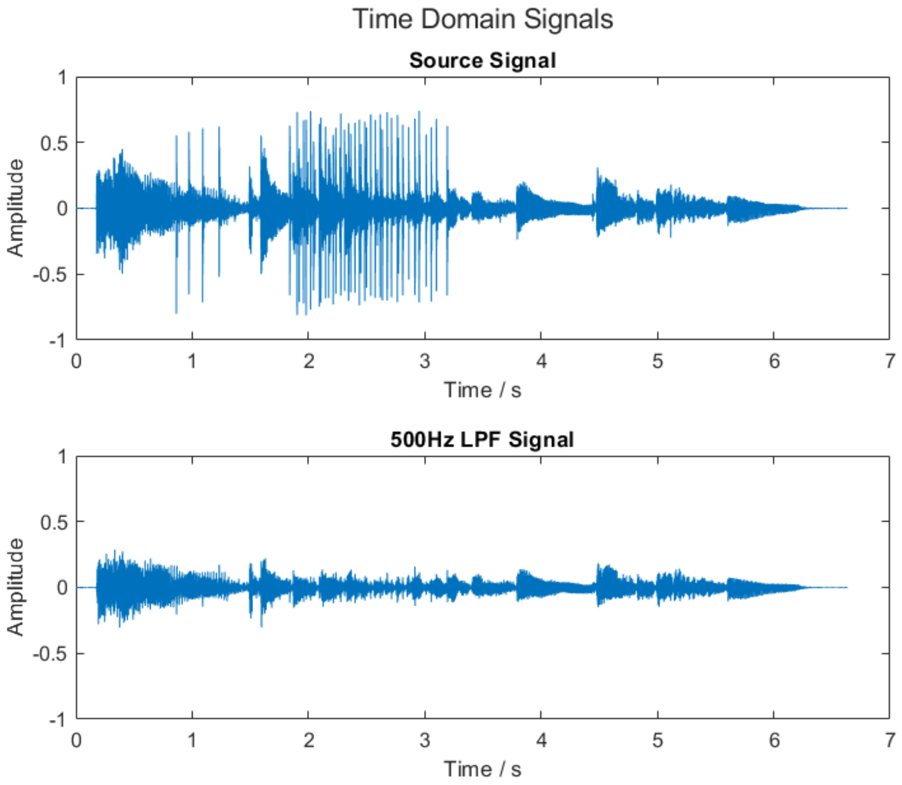
2 Exercise 1 — Low Pass Filter

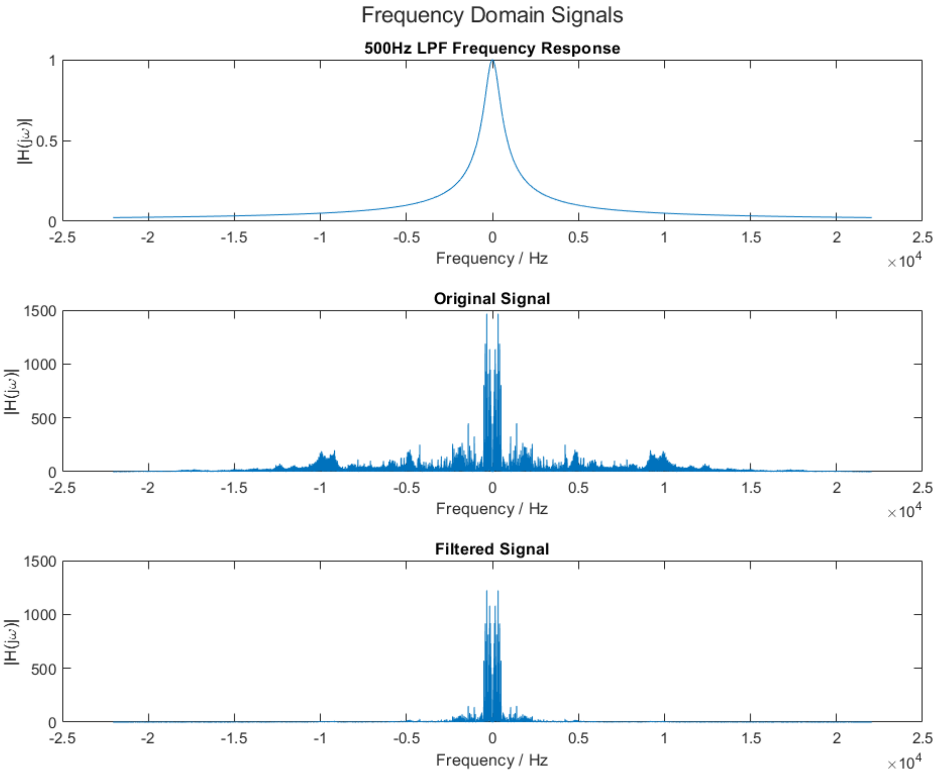
We used Fourier Transform to analyze a sound signal. We multiplied the signal spectrum with $\frac{500}{500+j\cdot\omega}$ to run it through a low pass filter (LPF). We then used the inverse fourier transform to synthesize the new signal.

The filtered signal sounds quieter and duller than the original signal. It sounds muffled, and most or all "sharp" qualities of the noise have been taken away.

Visually, the filtered time-domain signal is clearly missing some high frequency components between times 2 and 3. The filtered frequency-domain signal lost all its higher frequency values.

2.1 Output





2.2 Code

We modularized loading the data, filtering, plotting, and playing the sound. First, we load the data:

```
% Youssef Beltagy
% BEE235A, Aut 2021, Lab 4
% LoadData - loads castanets44m.wav and
% prepares time and frequency vectors


[ss, Fs] = audioread(”castanets44m.wav”);
ss = ss';
N = length(ss);
t = (0:(N-1)) ./ Fs;
f = ((-N/2 + 1):(N/2))* Fs / N;
w = f * 2 * pi;
```

Second, we run the filtering logic:

```
% Youssef Beltagy
% BEE235A, Aut 2021, Lab 4
% SignalLowPass - loads castanets44m.wav,
% Thens runs it through a low pass filter.
% The output is plotted and played.

% Load the data
LoadData;

% Prepare the filter
a = 500 * 2 * pi; % rad s^-1
H = a ./ (a + 1j .* w); % centered
Hshift = fftshift(H); % shifted

% Run the filter
FT_ss = fft(ss); % shifted
FT_os = FT_ss .* Hshift; % shifted
os = real(ifft(FT_os)); % just real components

PlotSignals(t, ss, os, f, H, FT_ss, FT_os, '500Hz LPF');
PlaySignals(ss, os, Fs);
```

Third, we plot the results:

```
% Youssef Beltagy
% BEE235A, Aut 2021, Lab 4
% PlotSignals - plots time and frequency signals

function PlotSignals(t, ss, os, f, H, FT_ss, FT_os, description)

% Plot in time domain
layout = tiledlayout(2,1);
title(layout, "Time Domain Signals");

nexttile;
plot(t,ss);
title("Source Signal");
xlabel("Time / s")
ylabel("Amplitude")
ylim([-1 1])

nexttile;
plot(t,os);
title(sprintf("%s Signal", description));
xlabel("Time / s")
ylabel("Amplitude")
ylim([-1 1])

% Plot in Frequency Domain
```

```
figure();
layout = tiledlayout(3,1);
title(layout, ”Frequency Domain Signals”);

nexttile;
plot(f, abs(H));
title(sprintf(”%s Frequency Response”, description));
xlabel(”Frequency / Hz”);
ylabel(”|H(j\omega)|”);

nexttile;
plot(f, abs(fftshift(FT_ss)));
title(”Original Signal”);
xlabel(”Frequency / Hz”);
ylabel(”|H(j\omega)|”);
ylim([0 1500])

nexttile;
plot(f, abs(fftshift(FT_os)));
title(”Filtered Signal”);
xlabel(”Frequency / Hz”);
ylabel(”|H(j\omega)|”);
ylim([0 1500])

end
```

Finally, we play the sounds:

```
% Youssef Beltagy
% BEE235A, Aut 2021, Lab 4
% PlaySignals – plays the two signals
% once with sound another with soundsc

function PlaySignals(ss, os, Fs)

% Play the sounds
sound(ss, Fs); pause(8);
sound(os, Fs); pause(8);

soundsc(ss, Fs); pause(8);
soundsc(os, Fs); pause(8);

end
```

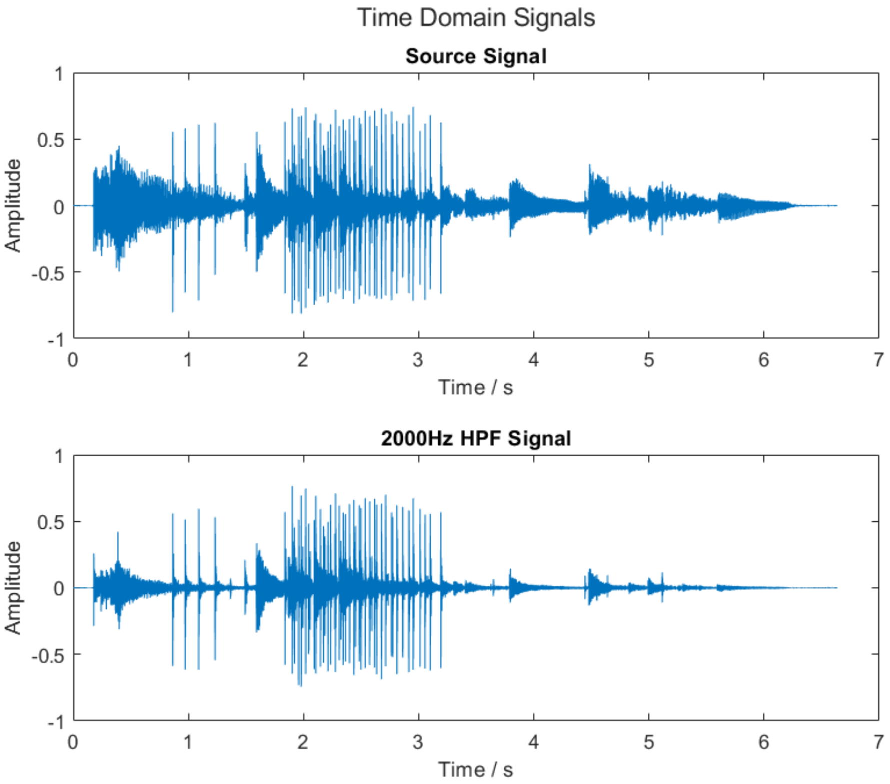
3 Exercise 2 — High Pass Filter

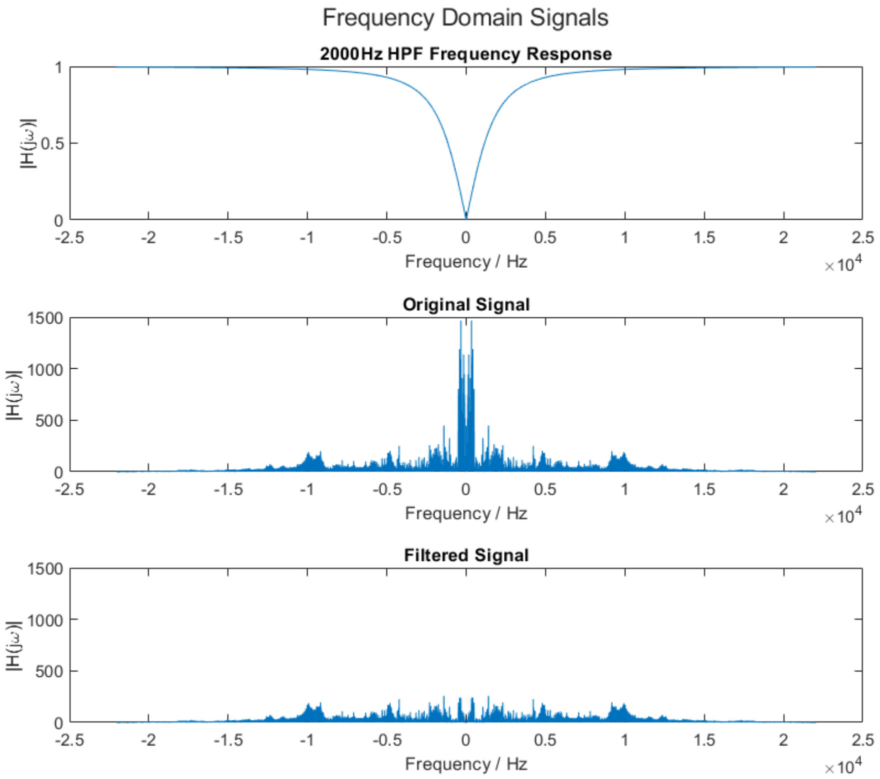
We repeated Exercise 1, but with a high pass filter (HPF). The HPF makes the signal sound sharp, but far away.

Visually, the time-domain filtered signal lost what seems like a DC component part of the signal. For example, from 0 to 1 seconds, the filtered signal has lower average amplitudes even though it seems to have the same maximum amplitude. This is likely what they call bass.

The frequency domain filtered signal clearly lost its lower frequency values. The two towers next to the 0 are gone. But the higher frequency values remain unchanged.

3.1 Output





3.2 Code

```
% Youssef Beltagy
% BEE235A, Aut 2021, Lab 4
% SignalHighPass - loads castanets44m.wav,
% Thens runs it through a high pass filter.
% The output is plotted and played.

% Load the data
LoadData;

% Prepare the filter
a = 2000 * 2 * pi; % convert to rad s^-1
H = 1 - a ./ (a + 1j .* w); % centered
Hshift = fftshift(H); % shifted

% Run the filter
FT_ss = fft(ss); % shifted
FT_os = FT_ss .* Hshift; % shifted
os = real(ifft(FT_os)); % just real components
```

```
PlotSignals(t, ss, os, f, H, FT_ss, FT_os, '2000Hz HPF');
PlaySignals(ss, os, Fs);
```

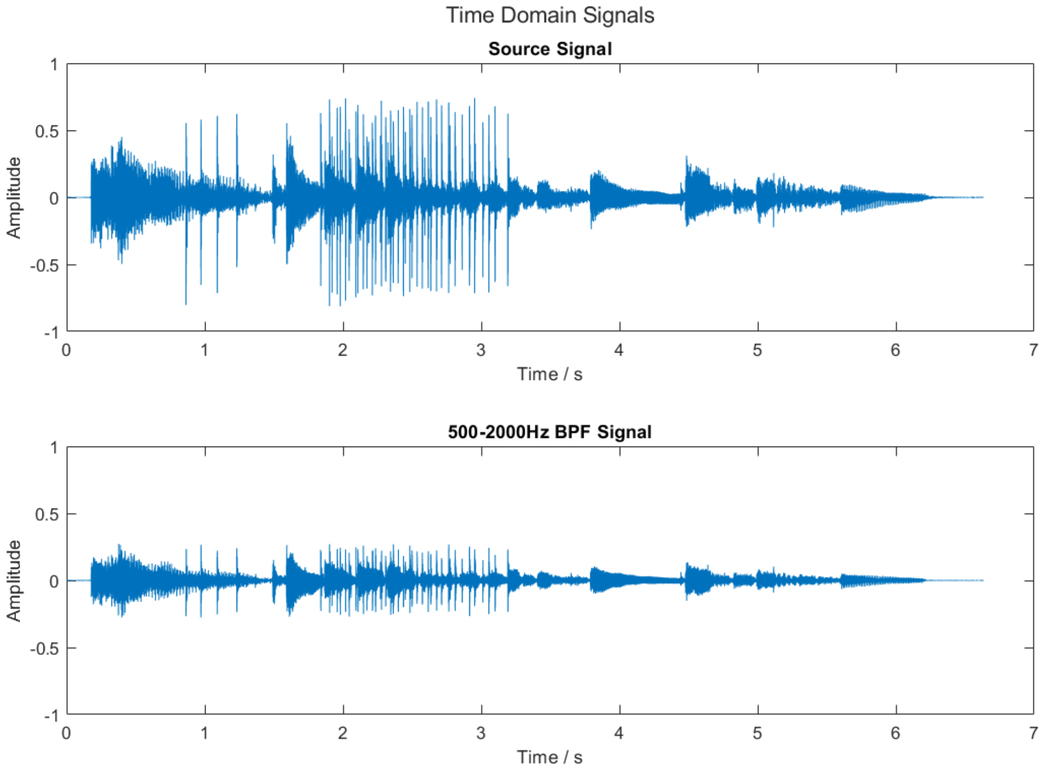
4 Exercise 3 — Band Pass Filter

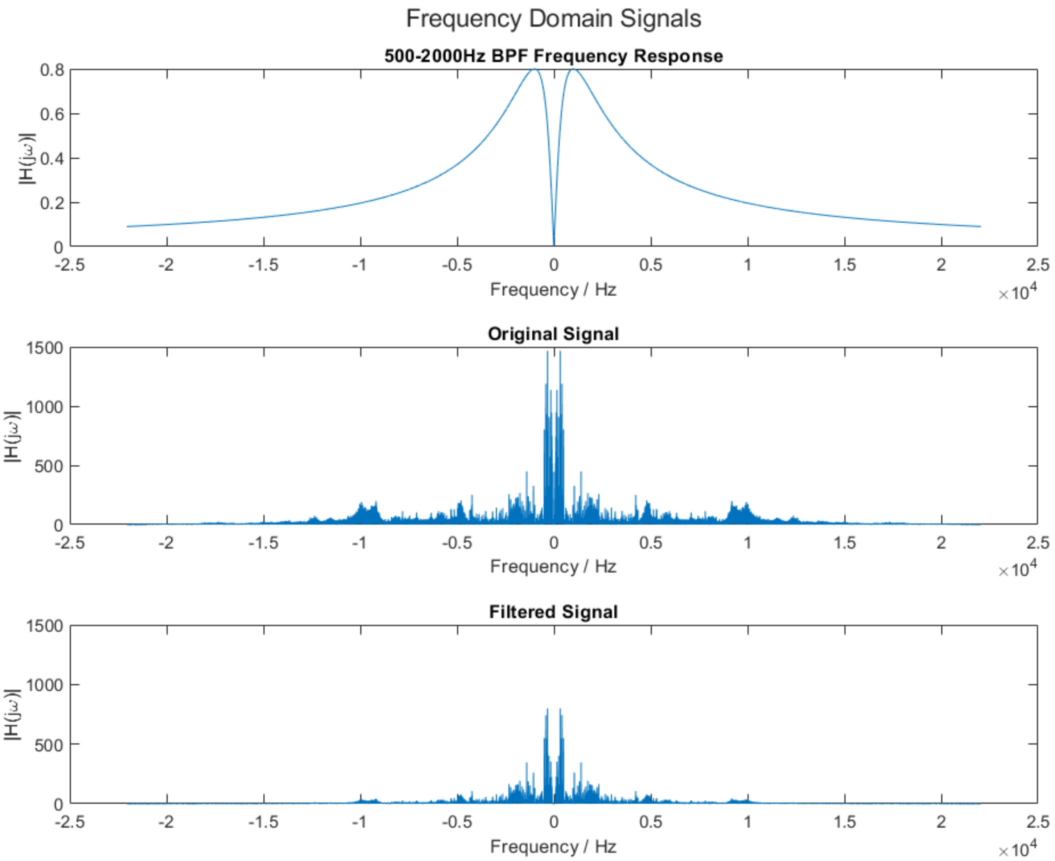
We combined exercises 1 and 2 to make a band pass filter (BPF). This signal didn't sound significantly different from the original signal.

Visually, the filtered time-domain signal retained its form, But has become weaker. It lost some of its bass components just like the HPF, but its high frequency components from seconds 1 to 2 became weaker as well. As expected, it looks like a combination of the LPF and HPF

The frequency-domain signal lost its highest frequency components. The lower frequency components (the two towers) became much weaker.

4.1 Output





4.2 Code

```
% Youssef Beltagy
% BEE235A, Aut 2021, Lab 4
% SignalBandPass - loads castanets44m.wav,
% Thens runs it through a band pass filter.
% The output is plotted and played.


% Load the data
LoadData;

% the LPF
a = 2000 * 2 * pi; % convert to rad s^-1
Hlpf = a ./ (a + 1j .* w); % centered

% the HPF
a = 500 * 2 * pi;
Hhpf = 1 - a ./ (a + 1j .* w);
```

```
% combined filter
H = Hlpf .* Hhpf;
Hshift = fftshift(H); % shifted

% Run the filter
FT_ss = fft(ss); % shifted
FT_os = FT_ss .* Hshift; % shifted
os = real(ifft(FT_os)); % just real components

PlotSignals(t, ss, os, f, H, FT_ss, FT_os, '500-2000Hz BPF');
PlaySignals(ss, os, Fs);
```

5 Conclusion

Because of the properties of the Fourier Transform, filters can be implemented by performing a reverse Fourier transform on the product of the FTT of a signal and a filter's frequency response. This has the same effect as convolving the original signal with the impulse response, and the process is straightforward and effective.

The effect of filters is easy to notice in both the time and frequency domains.