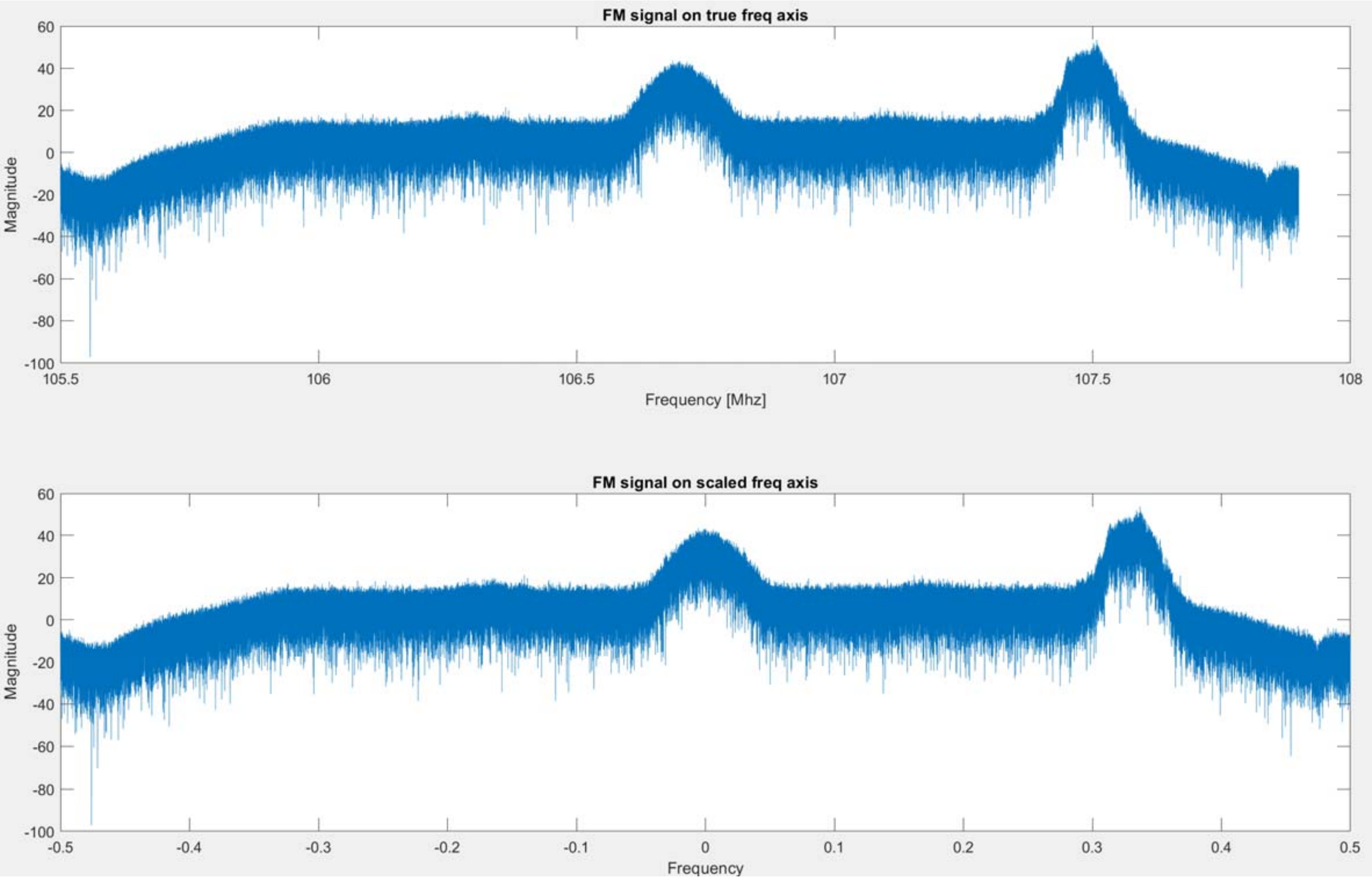
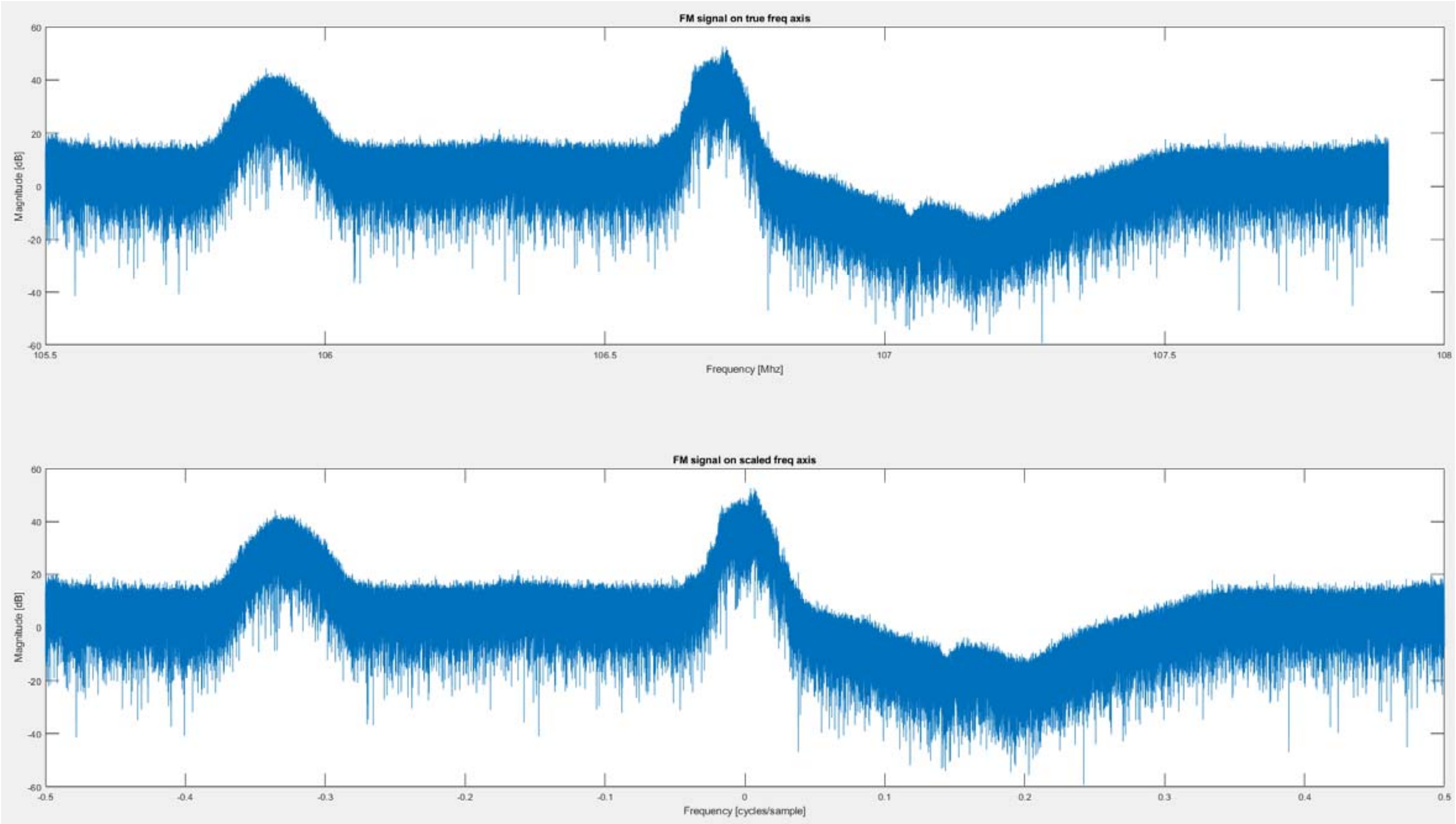
CA7 - FM Demodulating

Thursday, April 7, 2022 1:10 PM

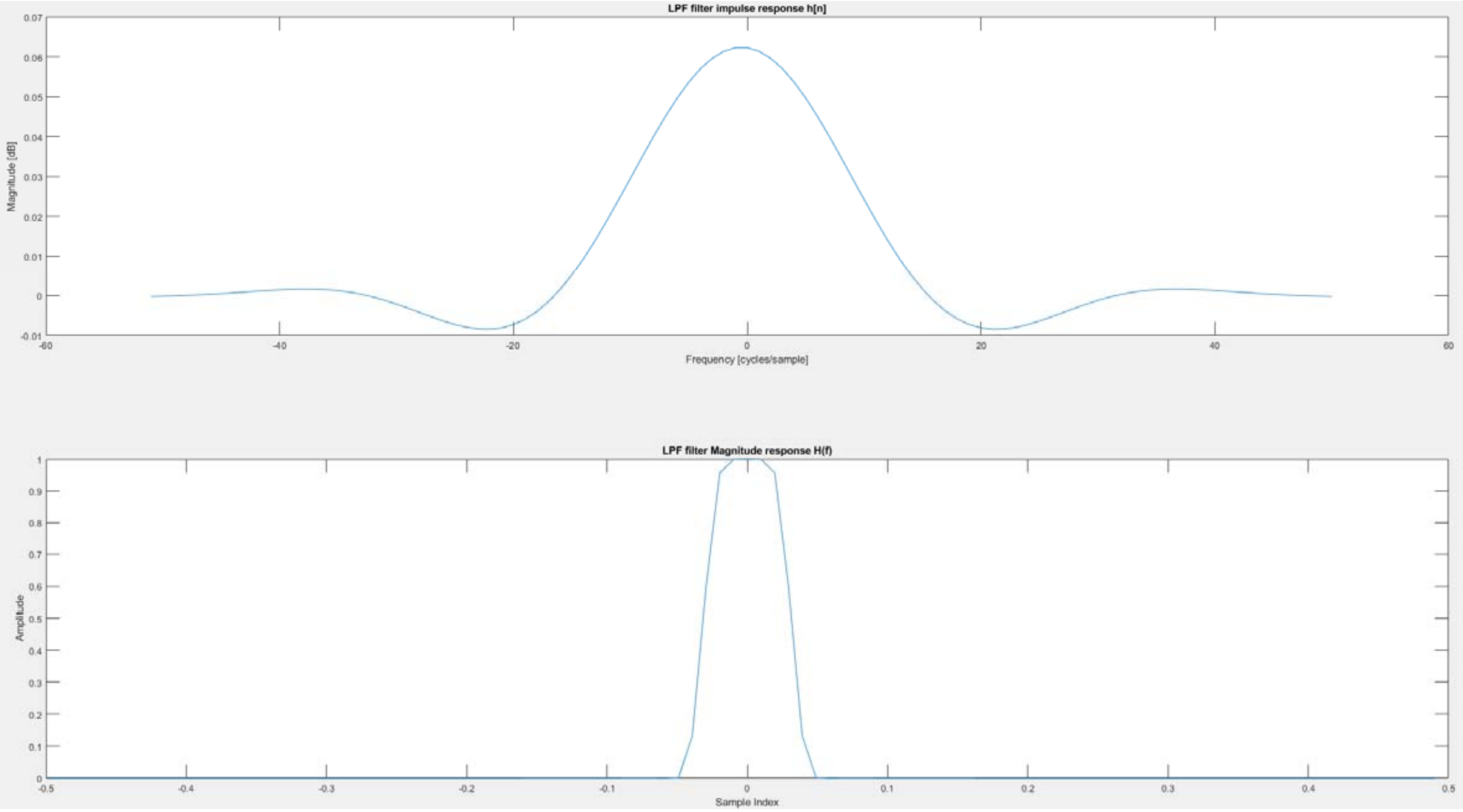
Input Signal:



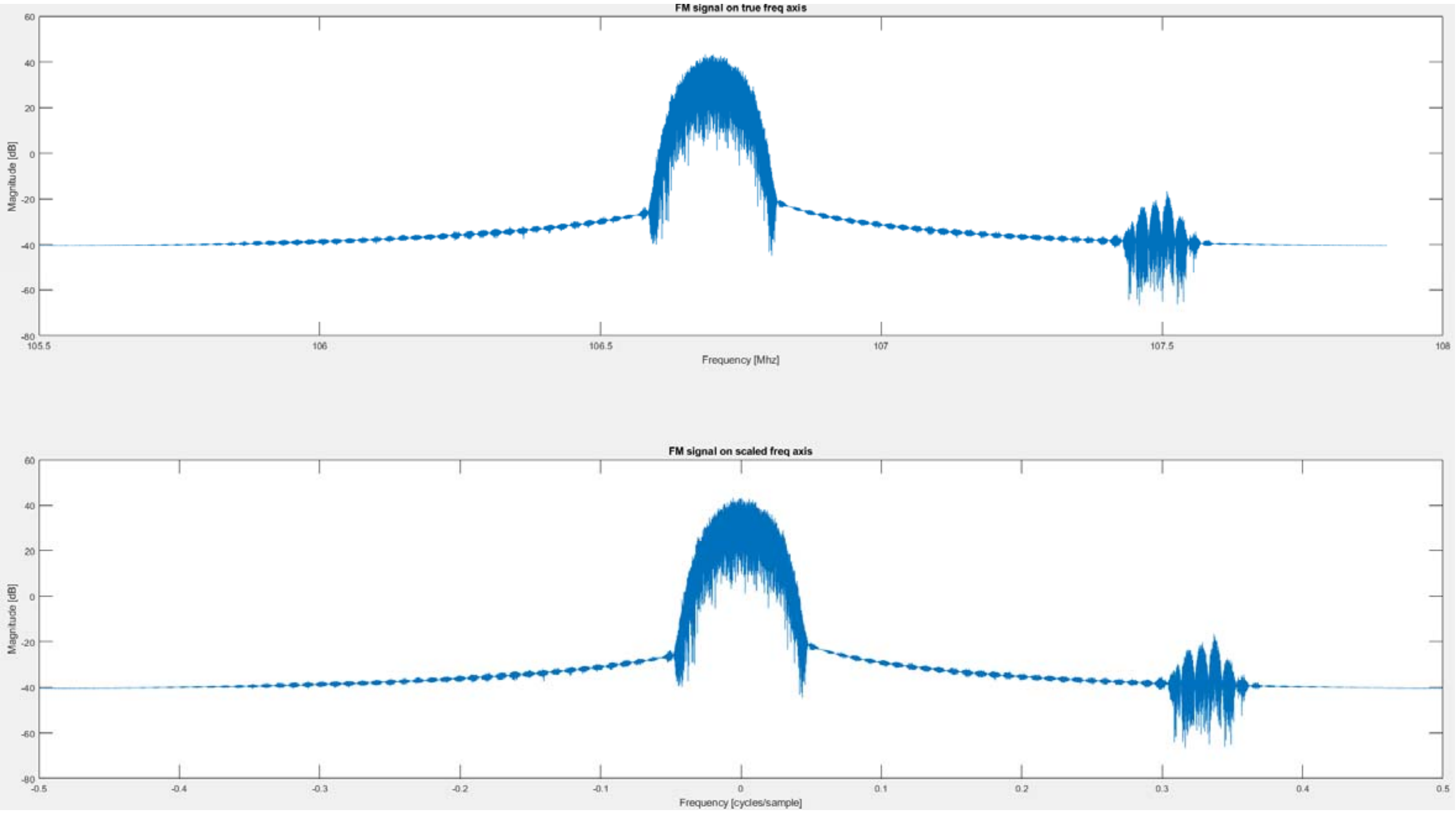
Shifted input signal:



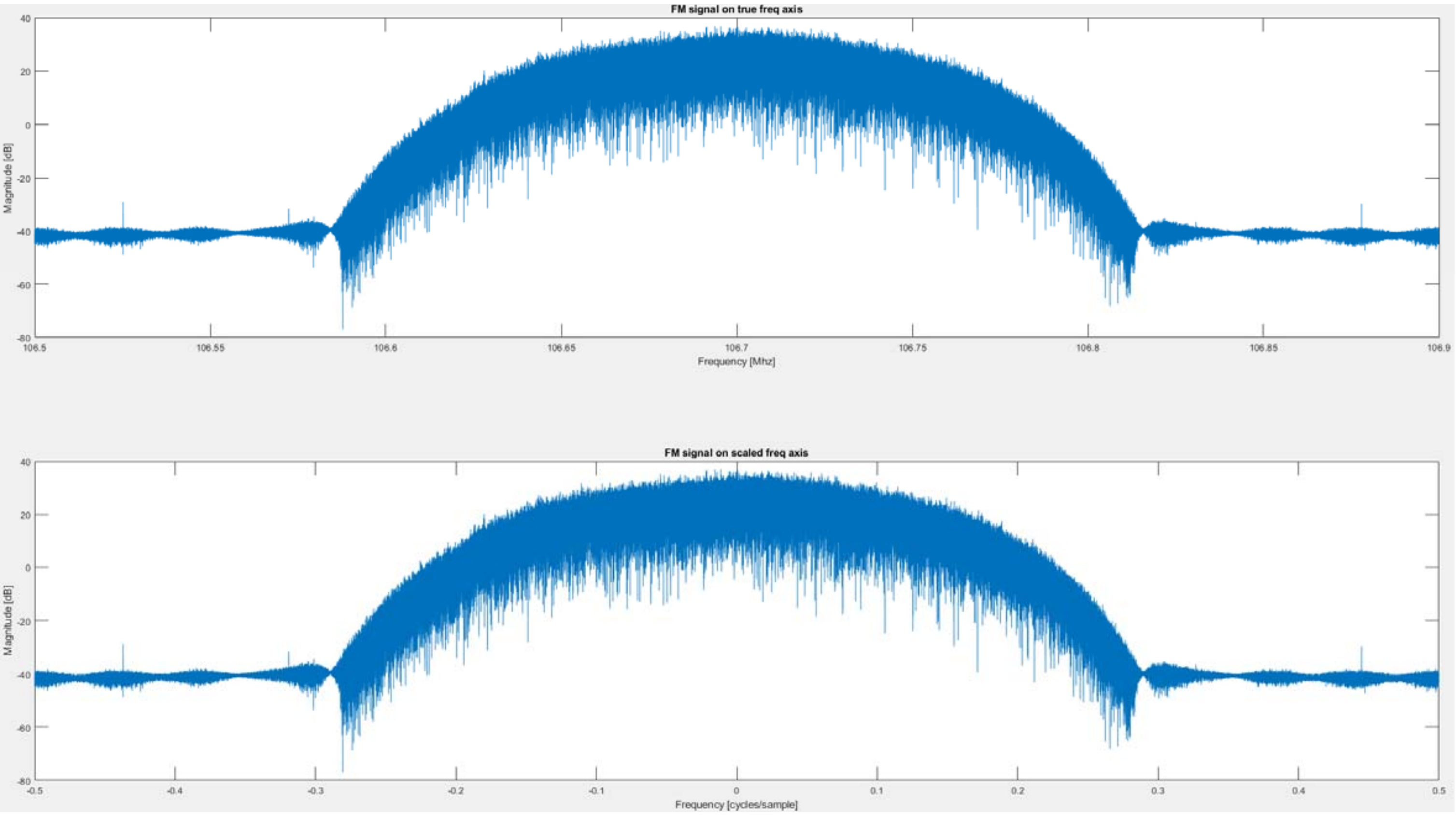
1st LPF impulse and magnitude response:

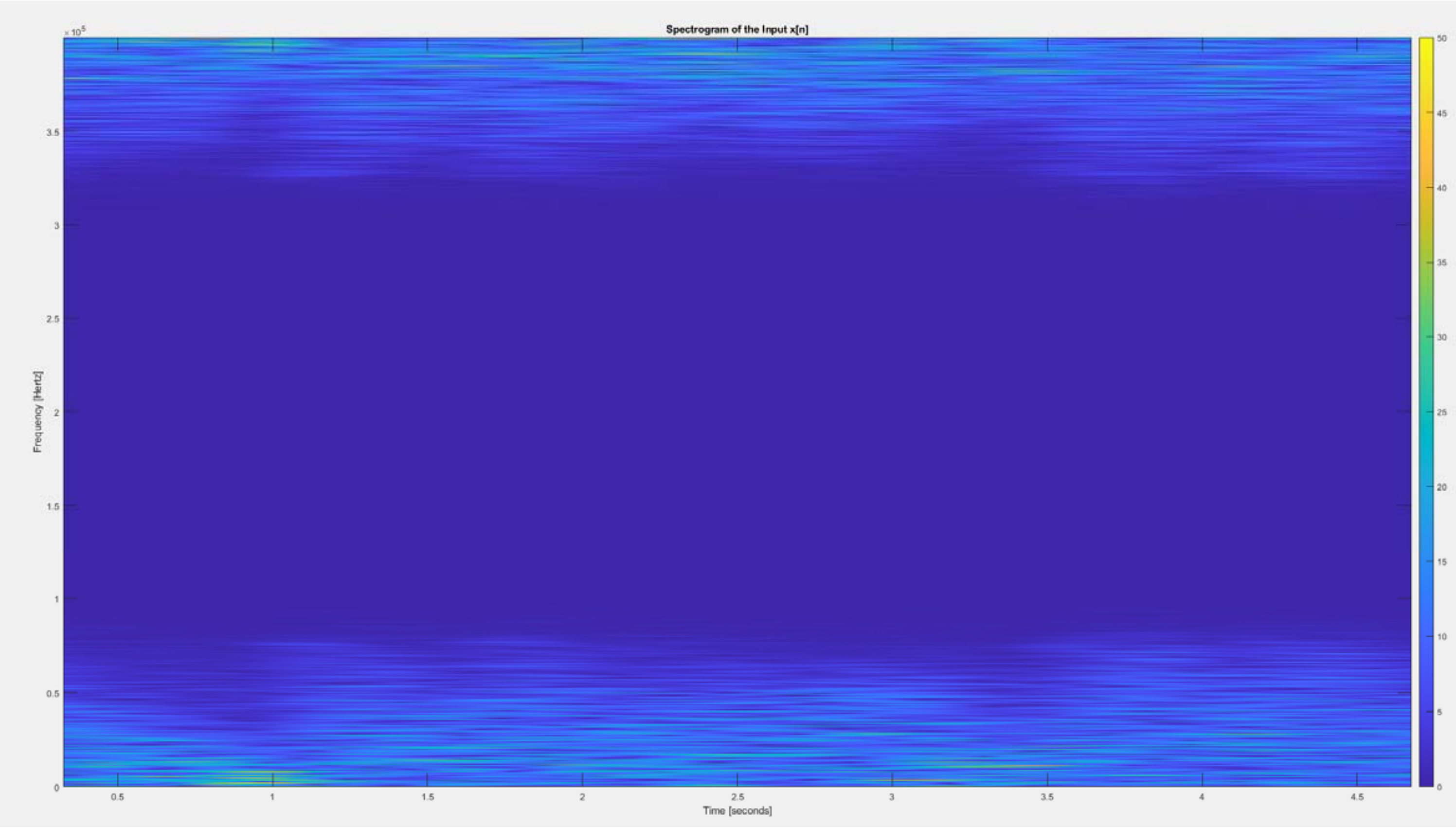


After 1st LPF:



After 6x decimation:





Impulse response derivation:

$$h[n] = \mathcal{F}^{-1}\left[H(f)\right] = \mathcal{F}^{-1}\left[j2\pi f\right]$$

$$h[n] = \int_{-\frac{1}{2}}^{\frac{1}{2}} j2\pi f e^{j2\pi f n} \, d2\pi f$$

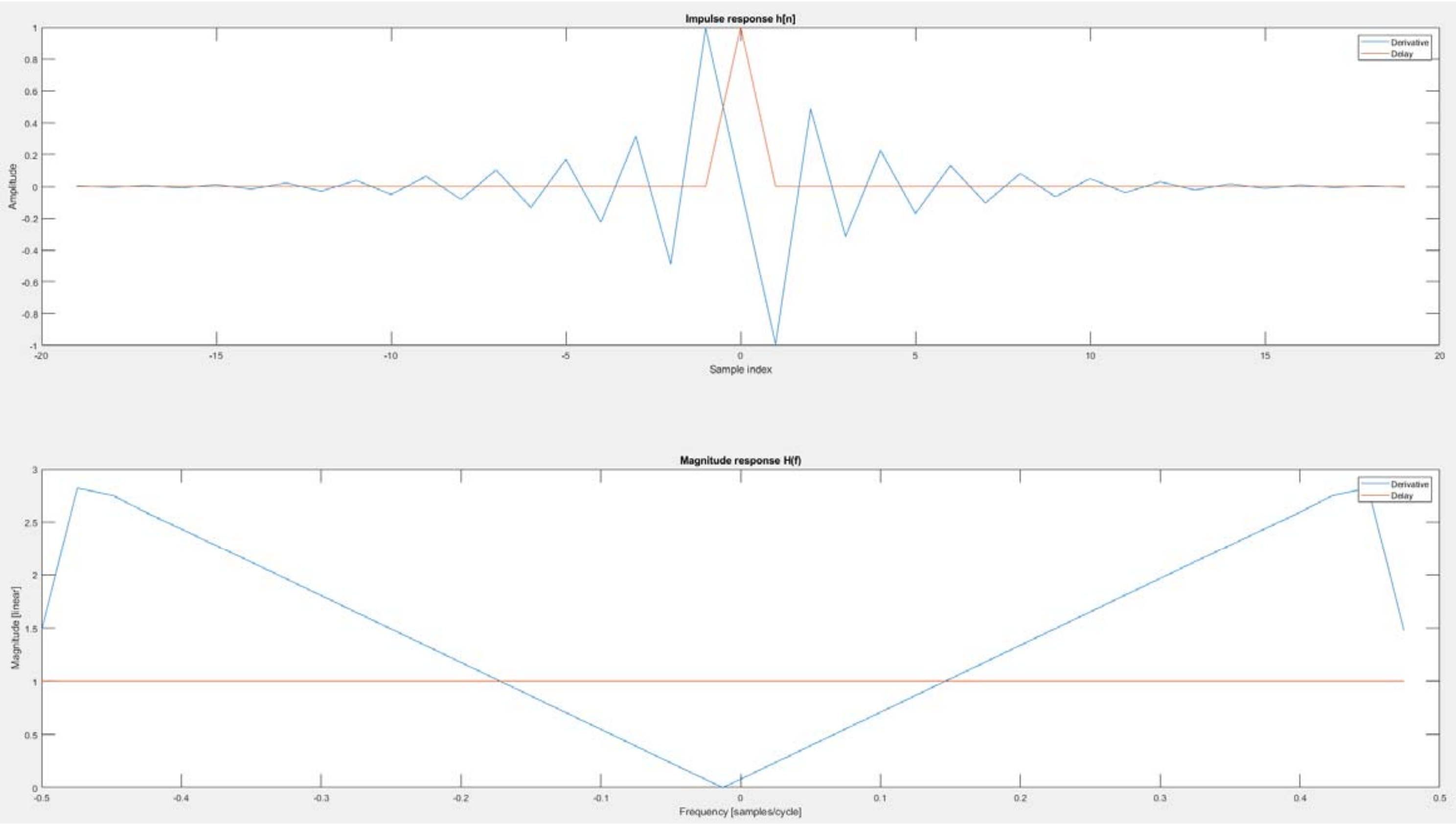
$$= j\left(\frac{2\pi f}{jn} e^{j2\pi f n}\Big|_{-1/2}^{1/2} - \frac{1}{n}\int_{-1/2}^{1/2} e^{j2\pi f n} \, d2\pi f\right.$$

$$= \frac{1}{n}\cos(\pi n)$$

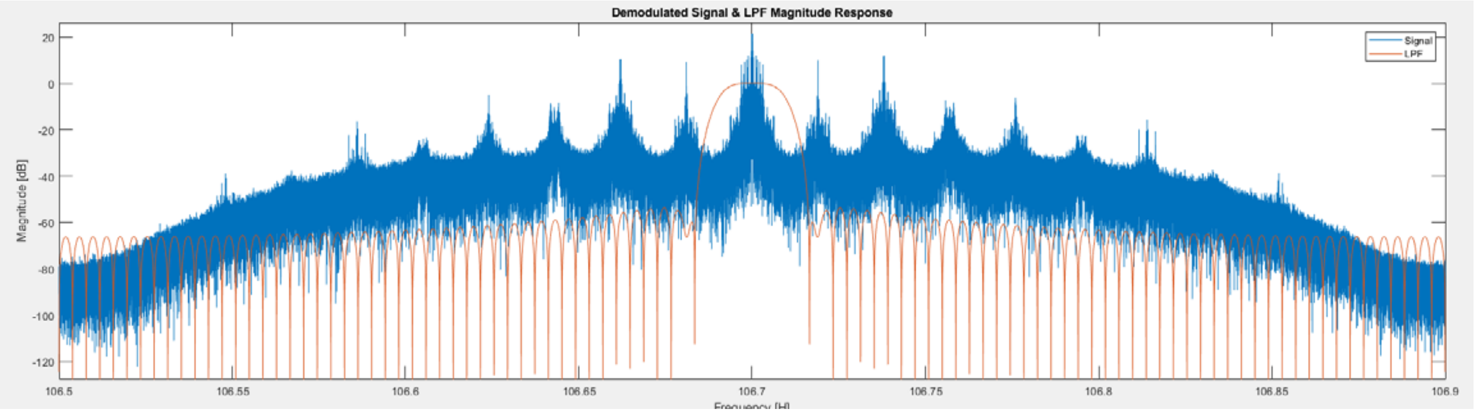
$\cos(\pi n) = (-1)^n$

$\therefore \quad \dfrac{(-1)^n}{n}$ for all $n \neq 0$

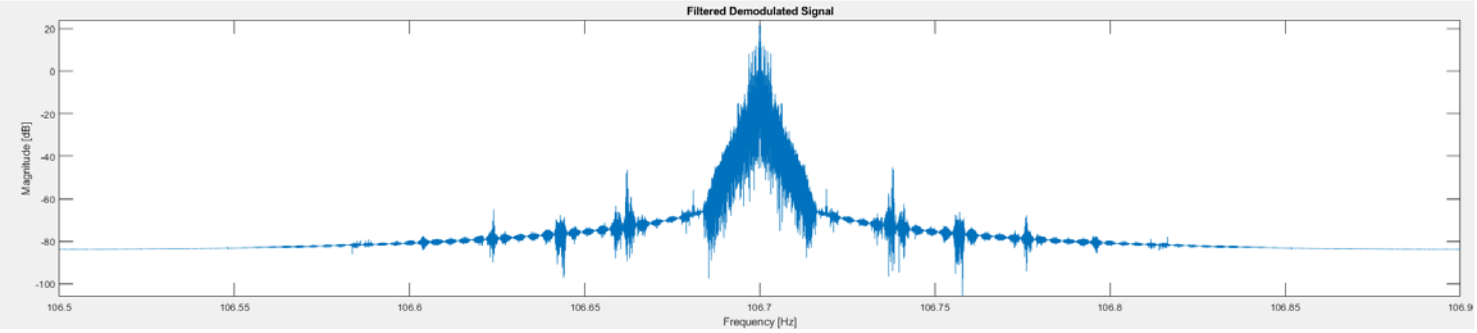
Derivative and Delay Filters:



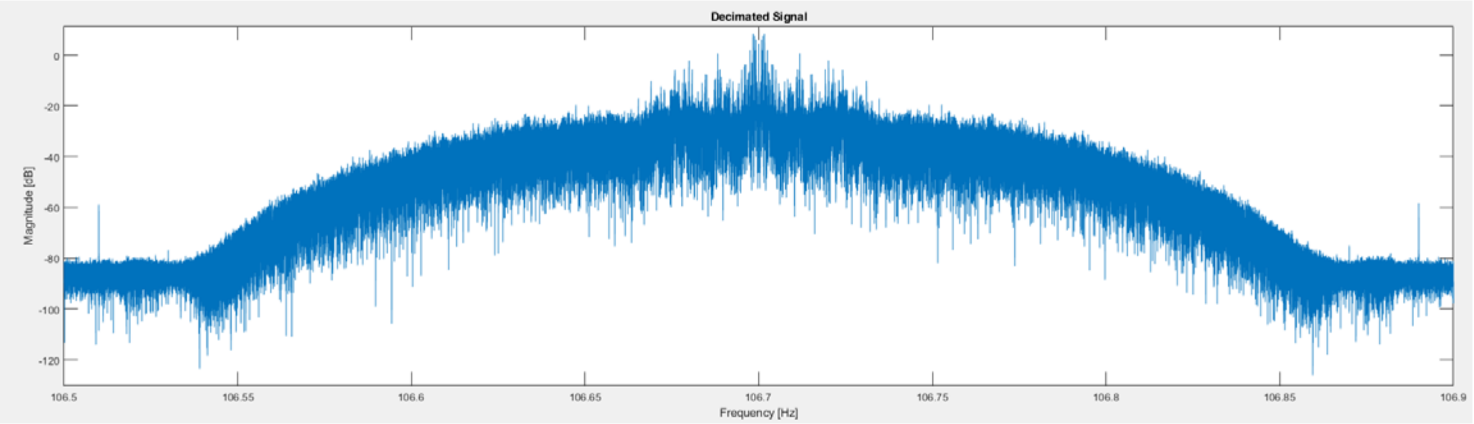
Demodulated signal and LPF2 Magnitude response:



After filtering:

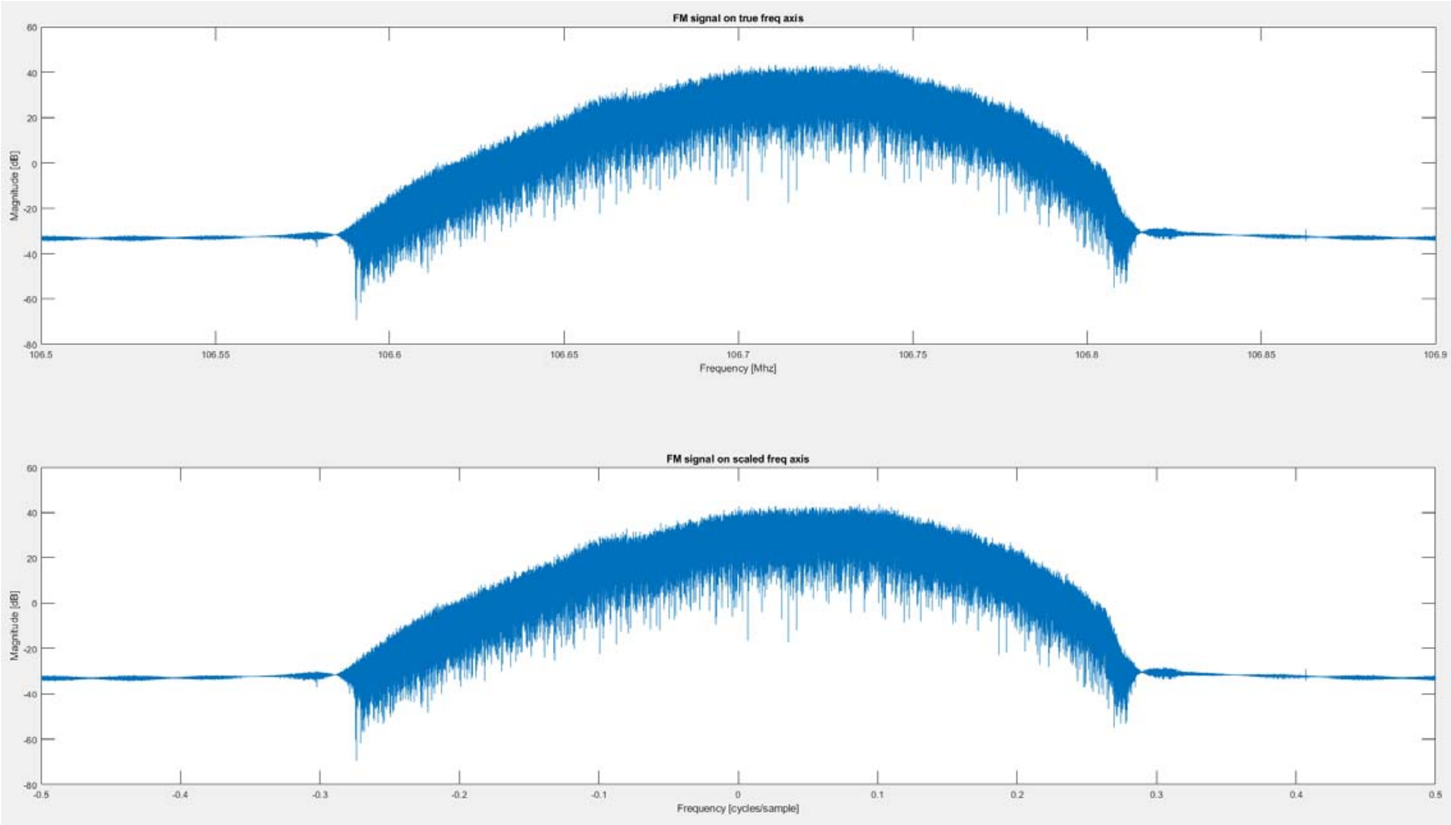


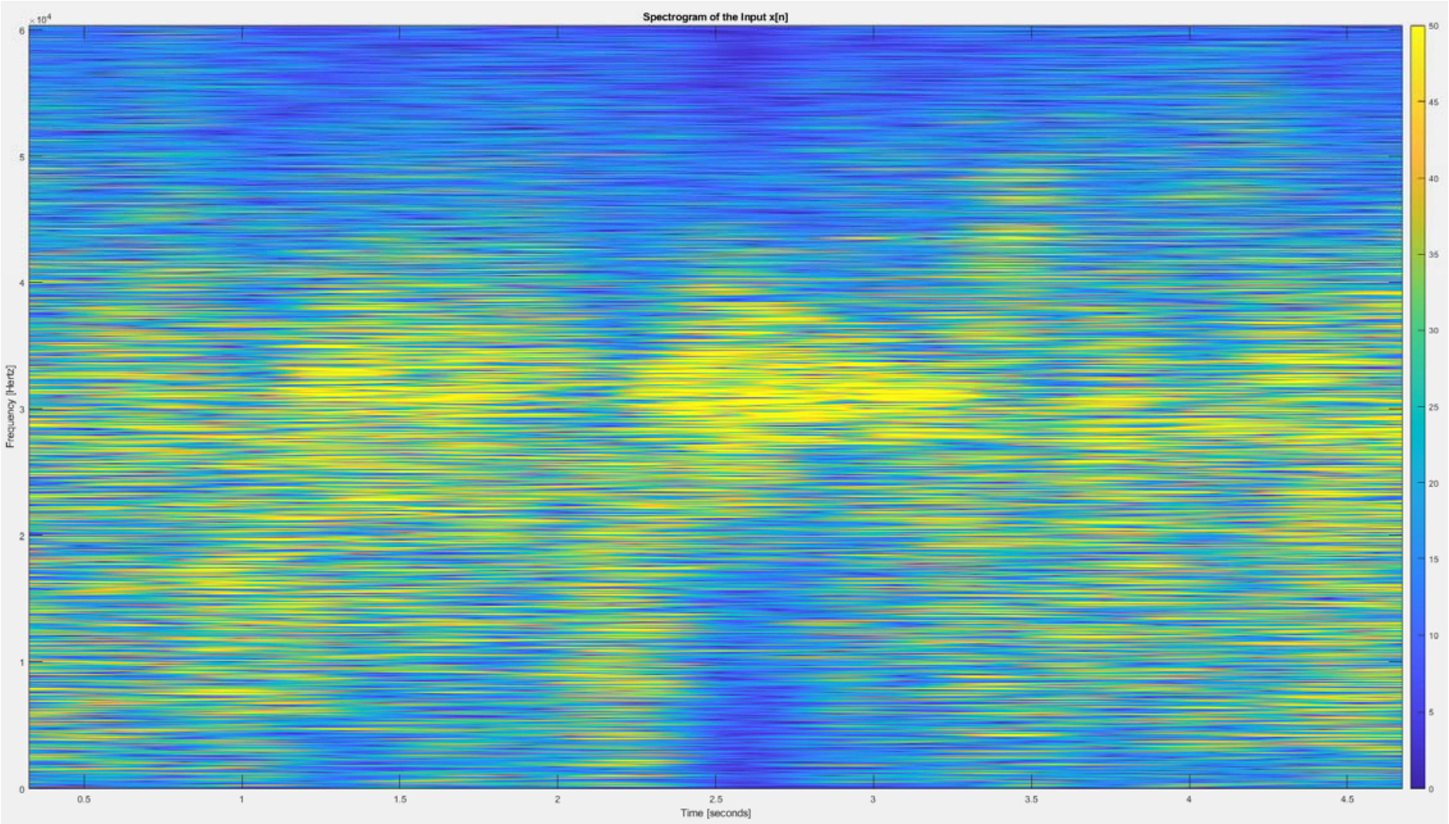
After decimation:

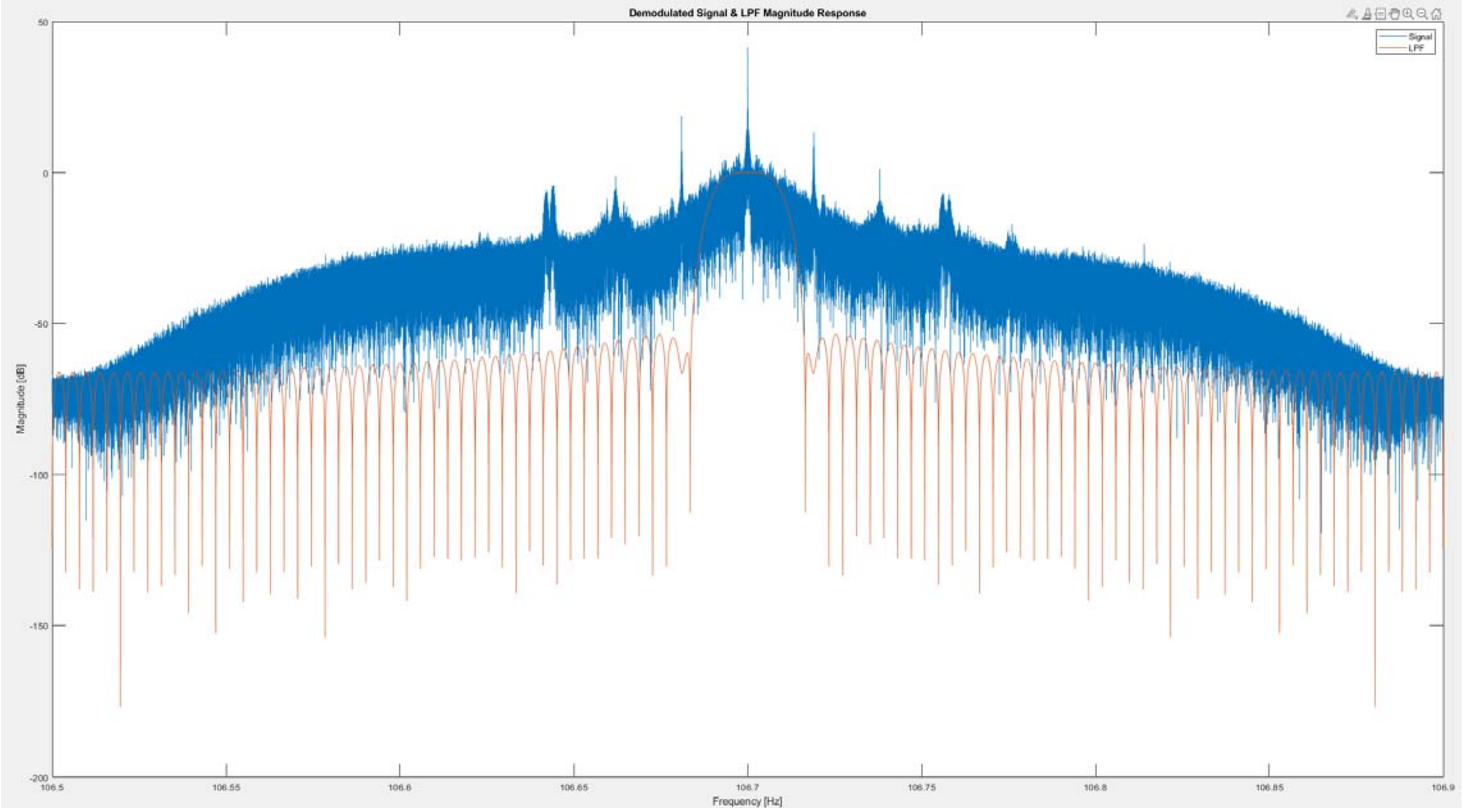


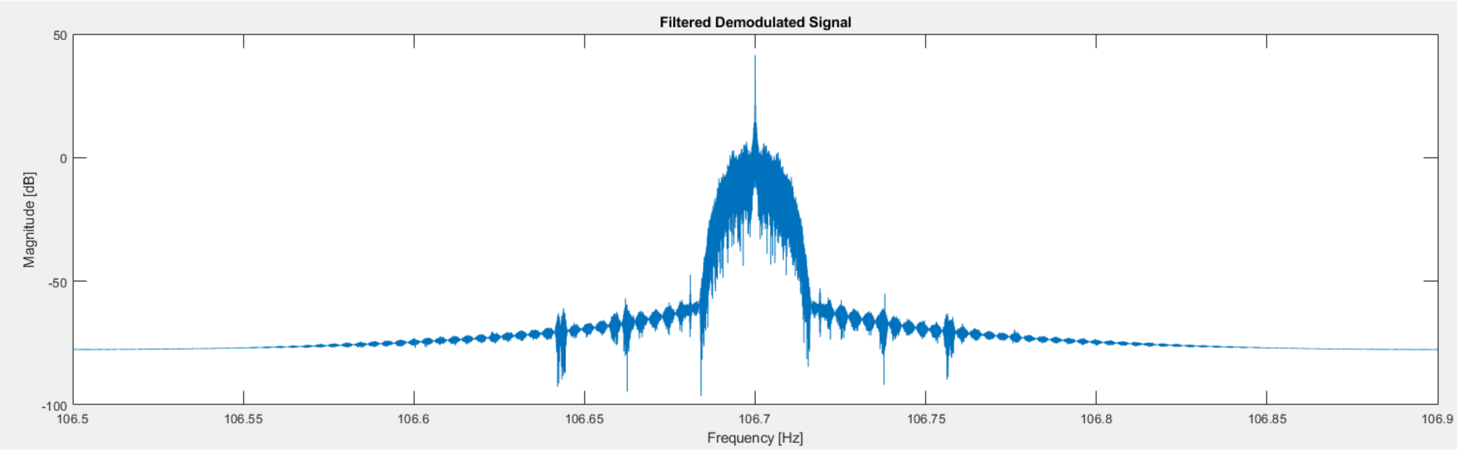
The audio signal was very quiet, but after some amplification a clip of guitar music could be clearly heard.

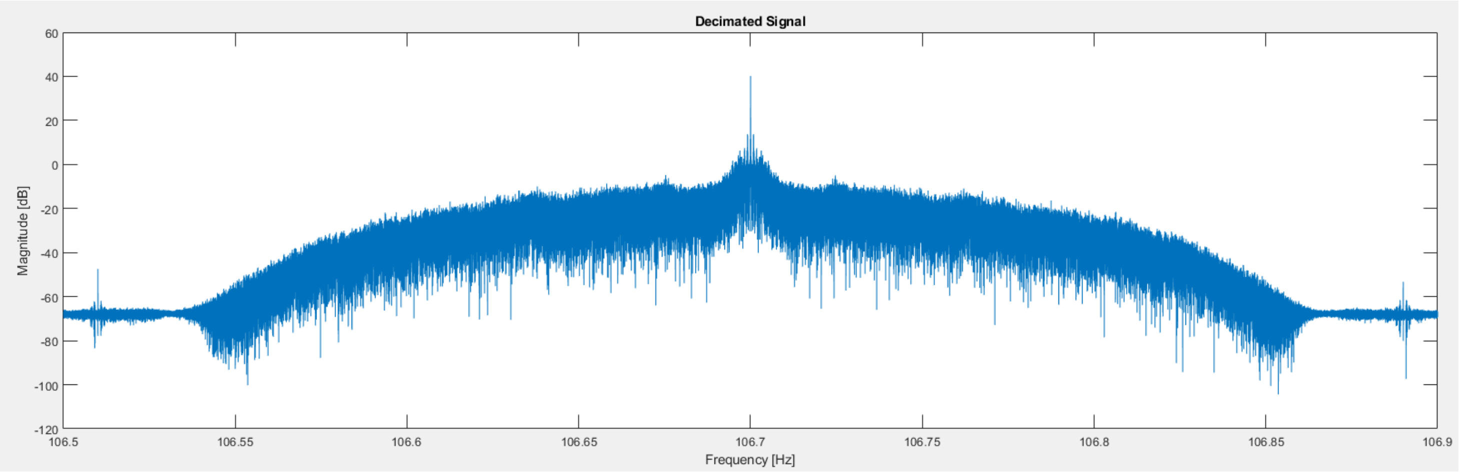
AFTER SHIFT:


FM signal on true freq axis
Magnitude [dB]
60
40
20
0
-20
-40
-60
-80
106.5
106.55
106.6
106.65
106.7
106.75
106.8
106.85
106.9
Frequency [Mhz]
FM signal on scaled freq axis
Magnitude [dB]
60
40
20
0
-20
-40
-60
-80
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5
Frequency [cycles/sample]


Spectrogram of the Input x[n]
×10^4
6
5
4
3
2
1
0
Frequency [Hertz]
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Time [seconds]
50
45
40
35
30
25
20
15
10
5
0

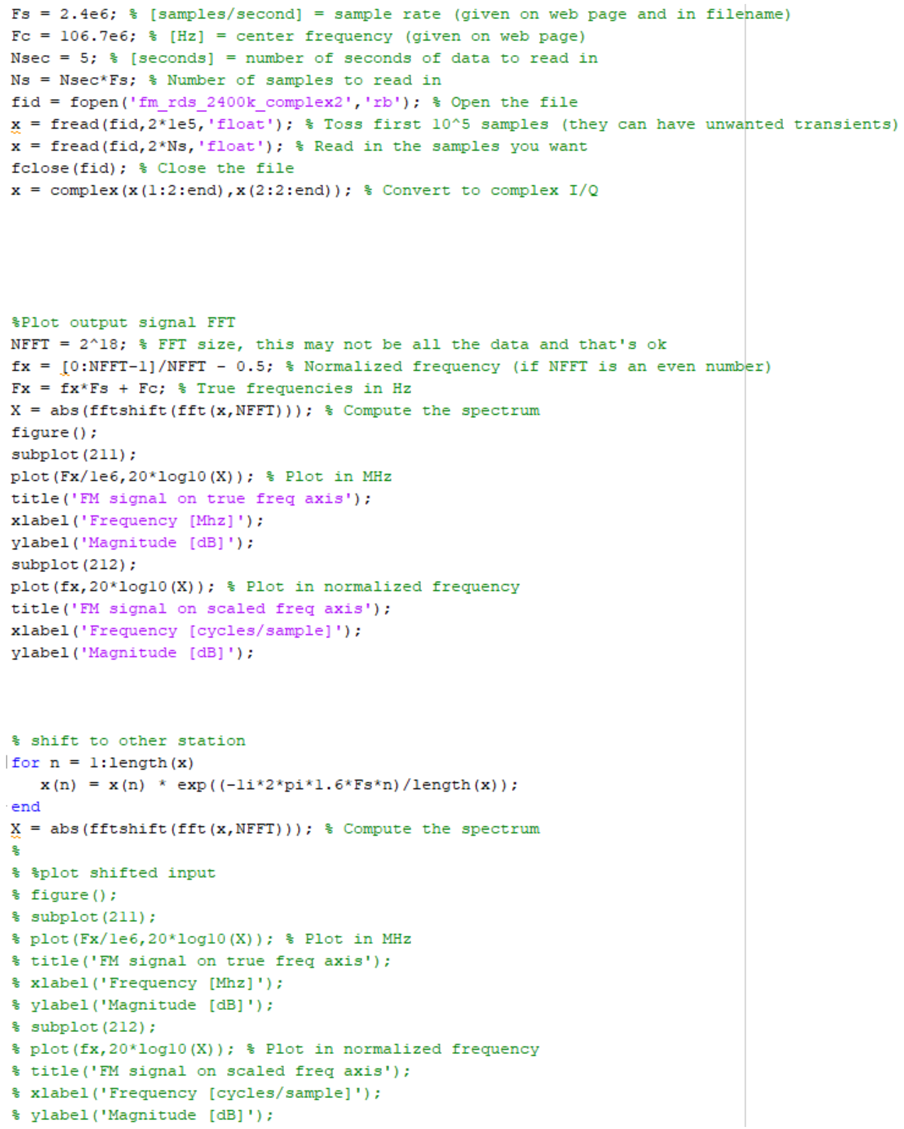

Demodulated Signal & LPF Magnitude Response
Signal
LPF
50
0
-50
-100
-150
-200
Magnitude [dB]
106.5
106.55
106.6
106.65
106.7
106.75
106.8
106.85
106.9
Frequency [Hz]



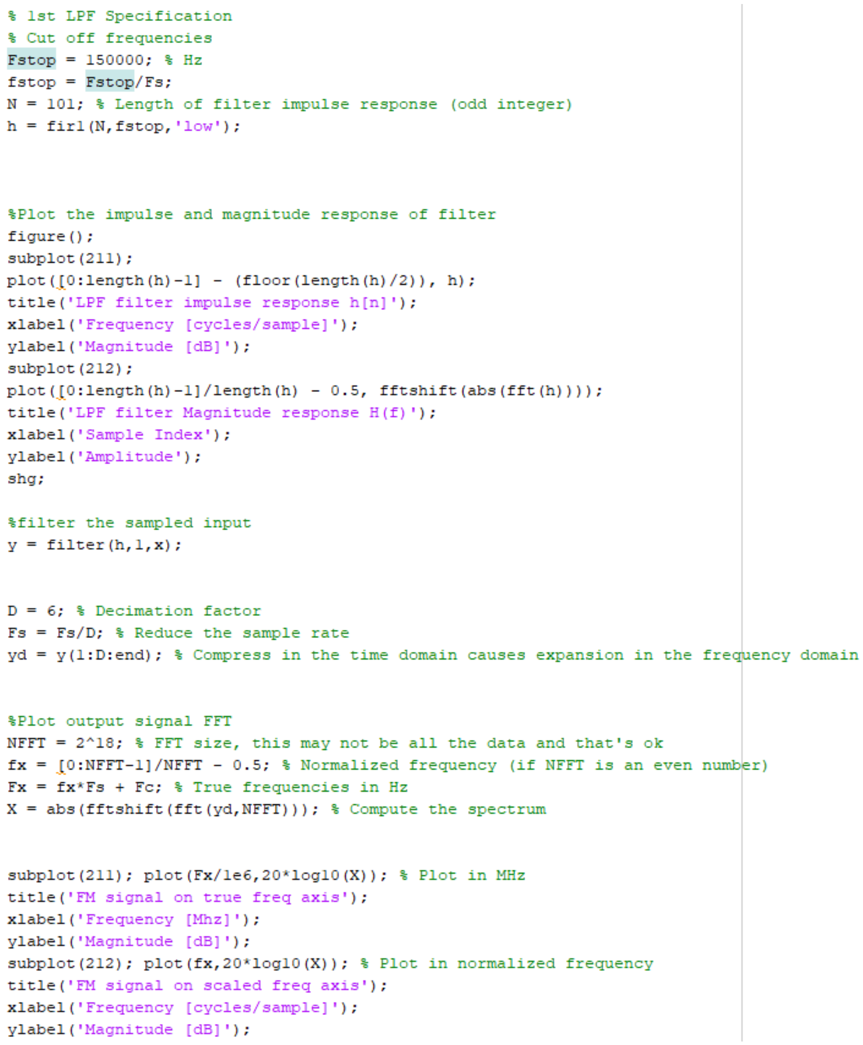


"..Washington monument or at least rename it, to take down the Jefferson memorial.."

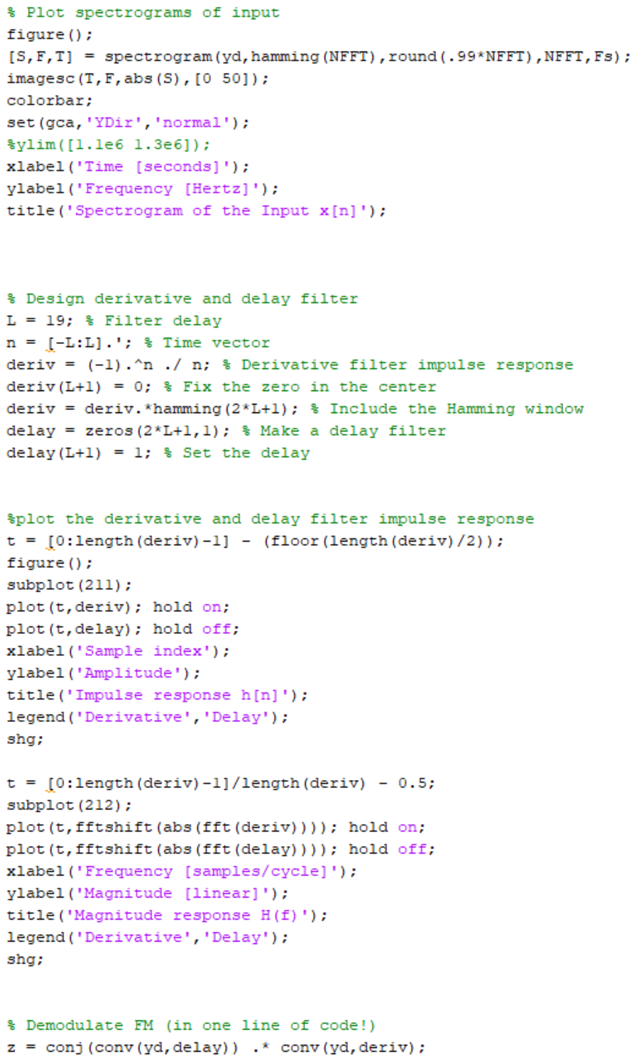
Code:



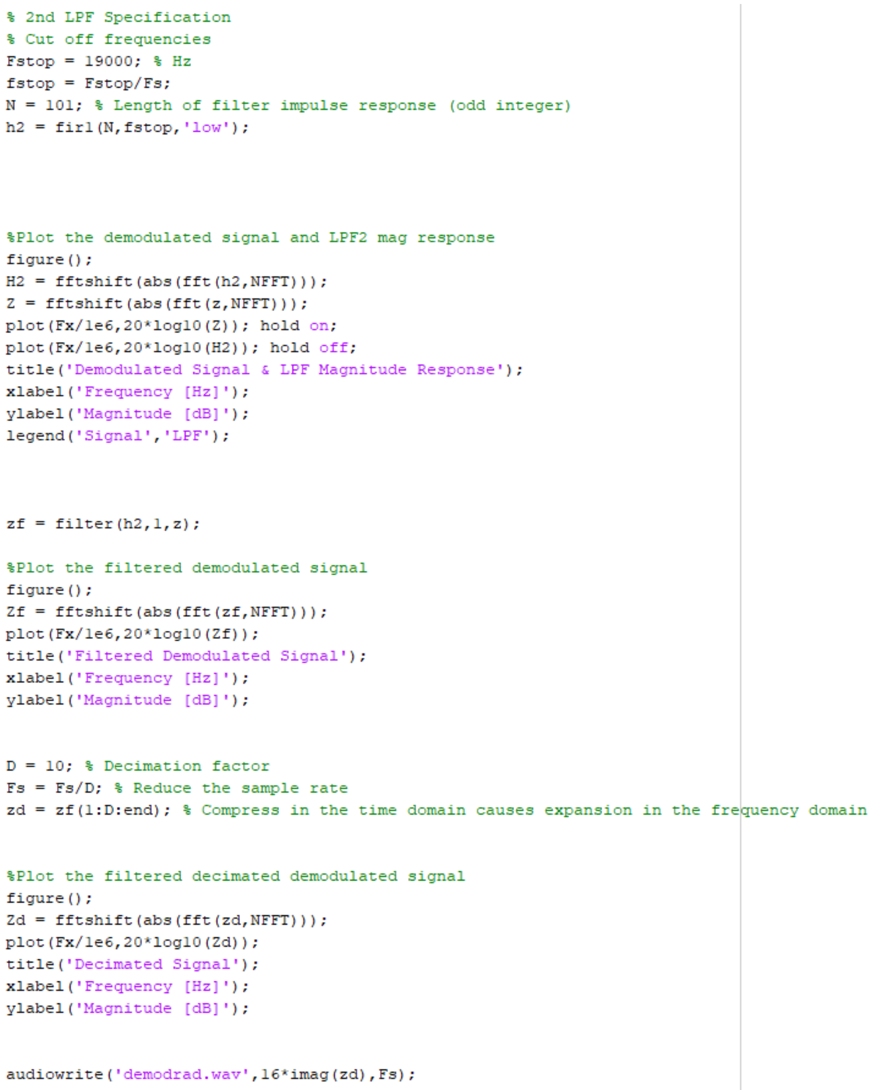
```
Fs = 2.4e6; % [samples/second] = sample rate (given on web page and in filename)
Fc = 106.7e6; % [Hz] = center frequency (given on web page)
Nsec = 5; % [seconds] = number of seconds of data to read in
Ns = Nsec*Fs; % Number of samples to read in
fid = fopen('fm_rds_2400k_complex2','rb'); % Open the file
x = fread(fid,2*1e5,'float'); % Toss first 10^5 samples (they can have unwanted transients)
x = fread(fid,2*Ns,'float'); % Read in the samples you want
fclose(fid); % Close the file
x = complex(x(1:2:end),x(2:2:end)); % Convert to complex I/Q





%Plot output signal FFT
NFFT = 2^18; % FFT size, this may not be all the data and that's ok
fx = [0:NFFT-1]/NFFT - 0.5; % Normalized frequency (if NFFT is an even number)
Fx = fx*Fs + Fc; % True frequencies in Hz
X = abs(fftshift(fft(x,NFFT))); % Compute the spectrum
figure();
subplot(211);
plot(Fx/1e6,20*log10(X)); % Plot in MHz
title('FM signal on true freq axis');
xlabel('Frequency [Mhz]');
ylabel('Magnitude [dB]');
subplot(212);
plot(fx,20*log10(X)); % Plot in normalized frequency
title('FM signal on scaled freq axis');
xlabel('Frequency [cycles/sample]');
ylabel('Magnitude [dB]');



% shift to other station
for n = 1:length(x)
   x(n) = x(n) * exp((-1i*2*pi*1.6*Fs*n)/length(x));
end
X = abs(fftshift(fft(x,NFFT))); % Compute the spectrum
%
% %plot shifted input
% figure();
% subplot(211);
% plot(Fx/1e6,20*log10(X)); % Plot in MHz
% title('FM signal on true freq axis');
% xlabel('Frequency [Mhz]');
% ylabel('Magnitude [dB]');
% subplot(212);
% plot(fx,20*log10(X)); % Plot in normalized frequency
% title('FM signal on scaled freq axis');
% xlabel('Frequency [cycles/sample]');
% ylabel('Magnitude [dB]');
```



```
% 1st LPF Specification
% Cut off frequencies
Fstop = 150000; % Hz
fstop = Fstop/Fs;
N = 101; % Length of filter impulse response (odd integer)
h = fir1(N,fstop,'low');



%Plot the impulse and magnitude response of filter
figure();
subplot(211);
plot([0:length(h)-1] - (floor(length(h)/2)), h);
title('LPF filter impulse response h[n]');
xlabel('Frequency [cycles/sample]');
ylabel('Magnitude [dB]');
subplot(212);
plot([0:length(h)-1]/length(h) - 0.5, fftshift(abs(fft(h))));
title('LPF filter Magnitude response H(f)');
xlabel('Sample Index');
ylabel('Amplitude');
shg;

%filter the sampled input
y = filter(h,1,x);


D = 6; % Decimation factor
Fs = Fs/D; % Reduce the sample rate
yd = y(1:D:end); % Compress in the time domain causes expansion in the frequency domain


%Plot output signal FFT
NFFT = 2^18; % FFT size, this may not be all the data and that's ok
fx = [0:NFFT-1]/NFFT - 0.5; % Normalized frequency (if NFFT is an even number)
Fx = fx*Fs + Fc; % True frequencies in Hz
X = abs(fftshift(fft(yd,NFFT))); % Compute the spectrum


subplot(211); plot(Fx/1e6,20*log10(X)); % Plot in MHz
title('FM signal on true freq axis');
xlabel('Frequency [Mhz]');
ylabel('Magnitude [dB]');
subplot(212); plot(fx,20*log10(X)); % Plot in normalized frequency
title('FM signal on scaled freq axis');
xlabel('Frequency [cycles/sample]');
ylabel('Magnitude [dB]');
```



```
% Plot spectrograms of input
figure();
[S,F,T] = spectrogram(yd,hamming(NFFT),round(.99*NFFT),NFFT,Fs);
imagesc(T,F,abs(S),[0 50]);
colorbar;
set(gca,'YDir','normal');
%ylim([1.1e6 1.3e6]);
xlabel('Time [seconds]');
ylabel('Frequency [Hertz]');
title('Spectrogram of the Input x[n]');



% Design derivative and delay filter
L = 19; % Filter delay
n = [-L:L].'; % Time vector
deriv = (-1).^n ./ n; % Derivative filter impulse response
deriv(L+1) = 0; % Fix the zero in the center
deriv = deriv.*hamming(2*L+1); % Include the Hamming window
delay = zeros(2*L+1,1); % Make a delay filter
delay(L+1) = 1; % Set the delay


%plot the derivative and delay filter impulse response
t = [0:length(deriv)-1] - (floor(length(deriv)/2));
figure();
subplot(211);
plot(t,deriv); hold on;
plot(t,delay); hold off;
xlabel('Sample index');
ylabel('Amplitude');
title('Impulse response h[n]');
legend('Derivative','Delay');
shg;

t = [0:length(deriv)-1]/length(deriv) - 0.5;
subplot(212);
plot(t,fftshift(abs(fft(deriv)))); hold on;
plot(t,fftshift(abs(fft(delay)))); hold off;
xlabel('Frequency [samples/cycle]');
ylabel('Magnitude [linear]');
title('Magnitude response H(f)');
legend('Derivative','Delay');
shg;


% Demodulate FM (in one line of code!)
z = conj(conv(yd,delay)) .* conv(yd,deriv);
```



```
% 2nd LPF Specification
% Cut off frequencies
Fstop = 19000; % Hz
fstop = Fstop/Fs;
N = 101; % Length of filter impulse response (odd integer)
h2 = fir1(N,fstop,'low');




%Plot the demodulated signal and LPF2 mag response
figure();
H2 = fftshift(abs(fft(h2,NFFT)));
Z = fftshift(abs(fft(z,NFFT)));
plot(Fx/1e6,20*log10(Z)); hold on;
plot(Fx/1e6,20*log10(H2)); hold off;
title('Demodulated Signal & LPF Magnitude Response');
xlabel('Frequency [Hz]');
ylabel('Magnitude [dB]');
legend('Signal','LPF');



zf = filter(h2,1,z);

%Plot the filtered demodulated signal
figure();
Zf = fftshift(abs(fft(zf,NFFT)));
plot(Fx/1e6,20*log10(Zf));
title('Filtered Demodulated Signal');
xlabel('Frequency [Hz]');
ylabel('Magnitude [dB]');


D = 10; % Decimation factor
Fs = Fs/D; % Reduce the sample rate
zd = zf(1:D:end); % Compress in the time domain causes expansion in the frequency domain


%Plot the filtered decimated demodulated signal
figure();
Zd = fftshift(abs(fft(zd,NFFT)));
plot(Fx/1e6,20*log10(Zd));
title('Decimated Signal');
xlabel('Frequency [Hz]');
ylabel('Magnitude [dB]');


audiowrite('demodrad.wav',16*imag(zd),Fs);
```