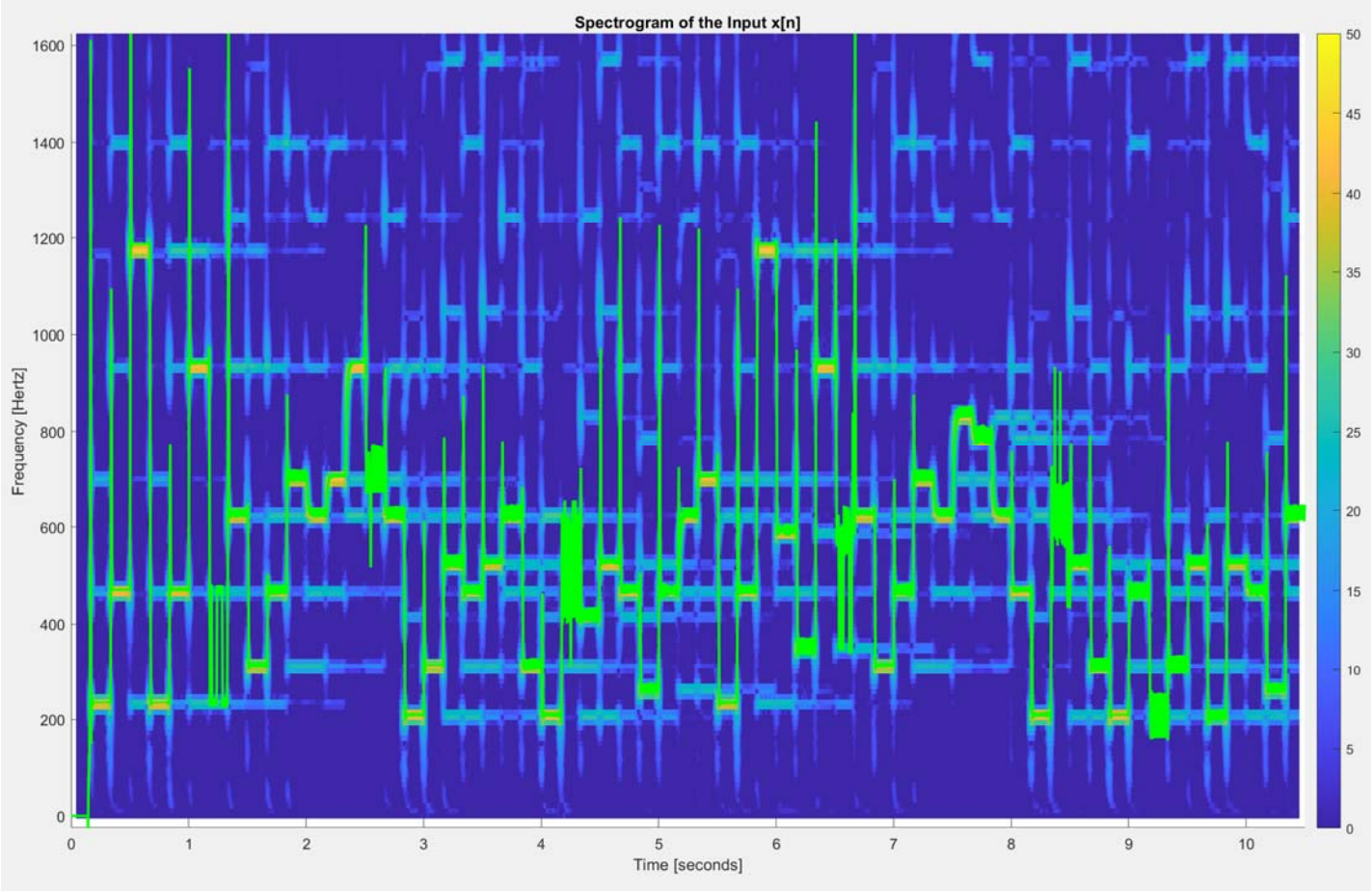
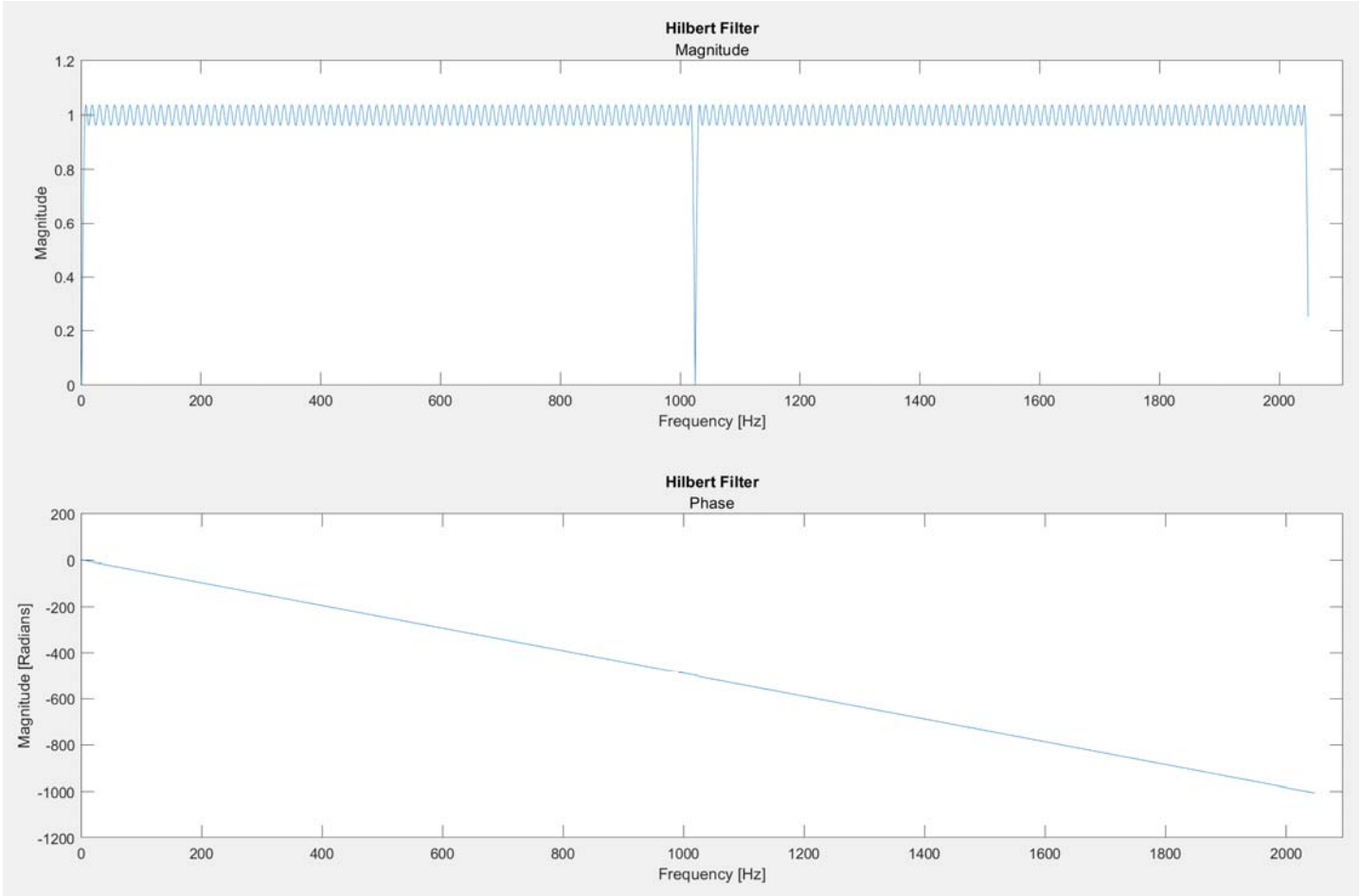
CA4 - Hilbert transform

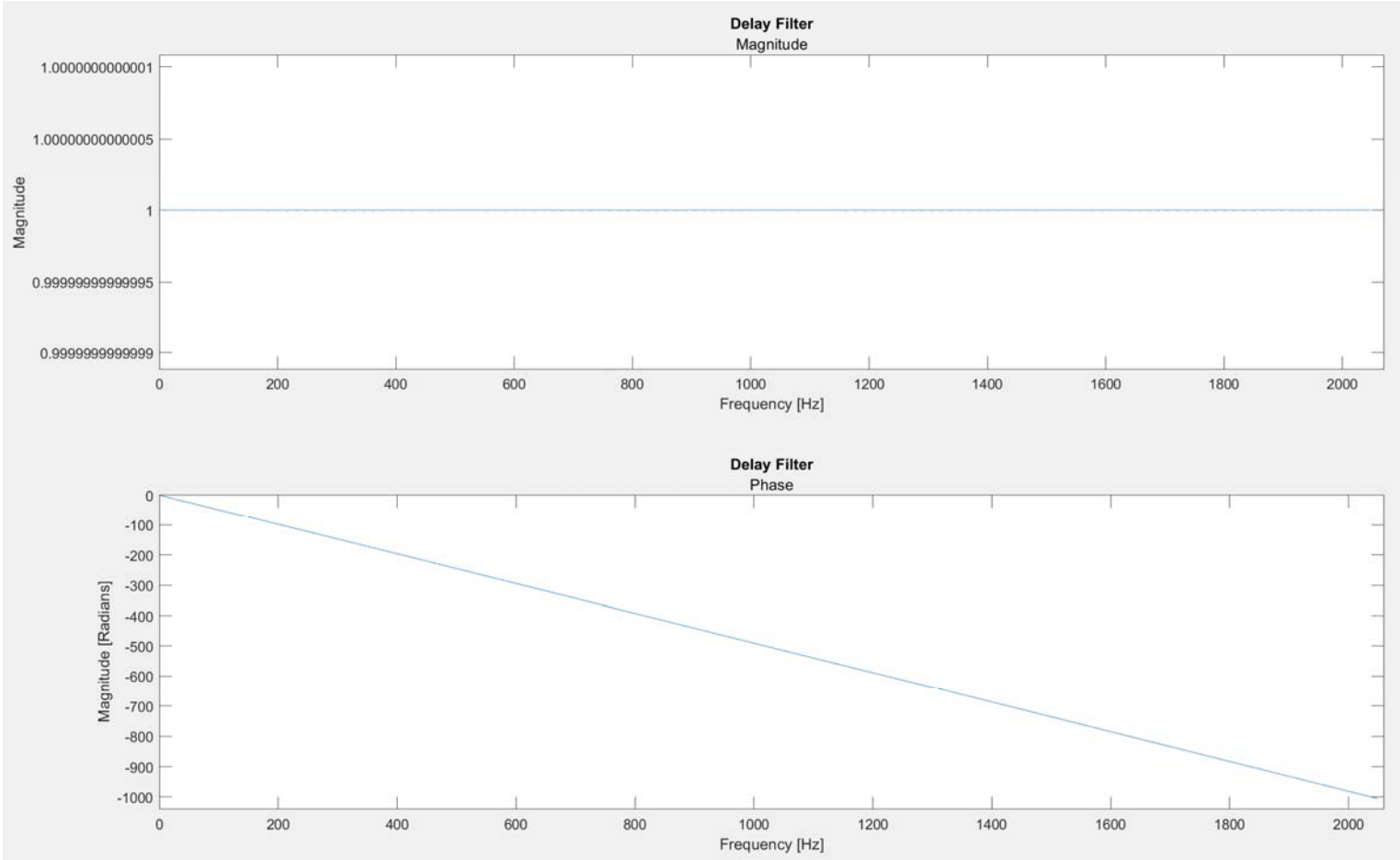
Wednesday, February 9, 2022 1:55 AM

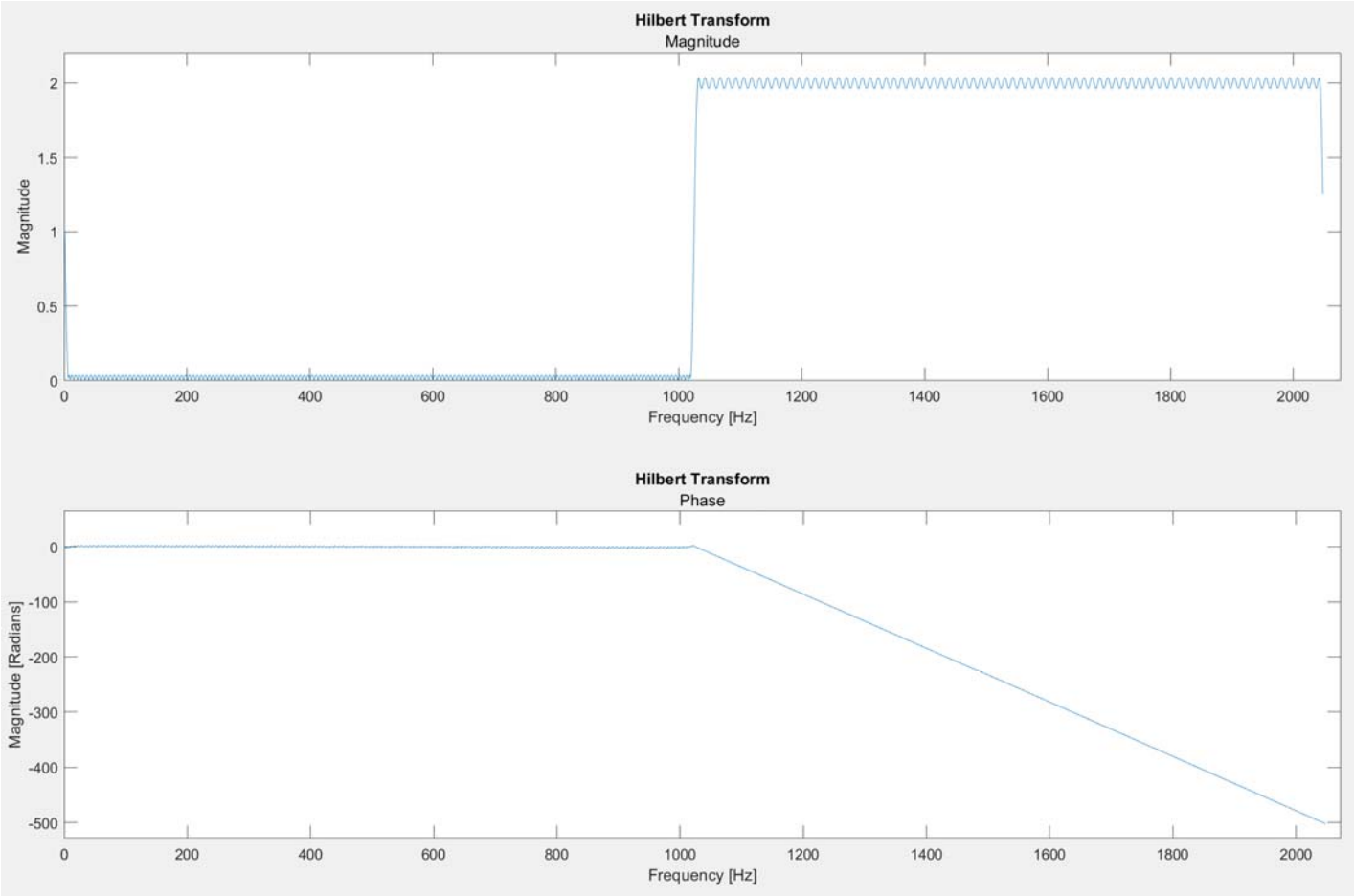


- describe the relationship between the instantaneous frequency and the spectrogram

The instantaneous frequency lines up with the frequencies with the highest amplitude on the spectrogram. As can be seen on the graph above, the instantaneous frequency can only be one single value at any time, so when there are multiple frequencies with equal amplitude on the spectrogram, the instantaneous frequency oscillates between these two frequencies quickly, creating these blocklike sections such as the one around 4.2 seconds.

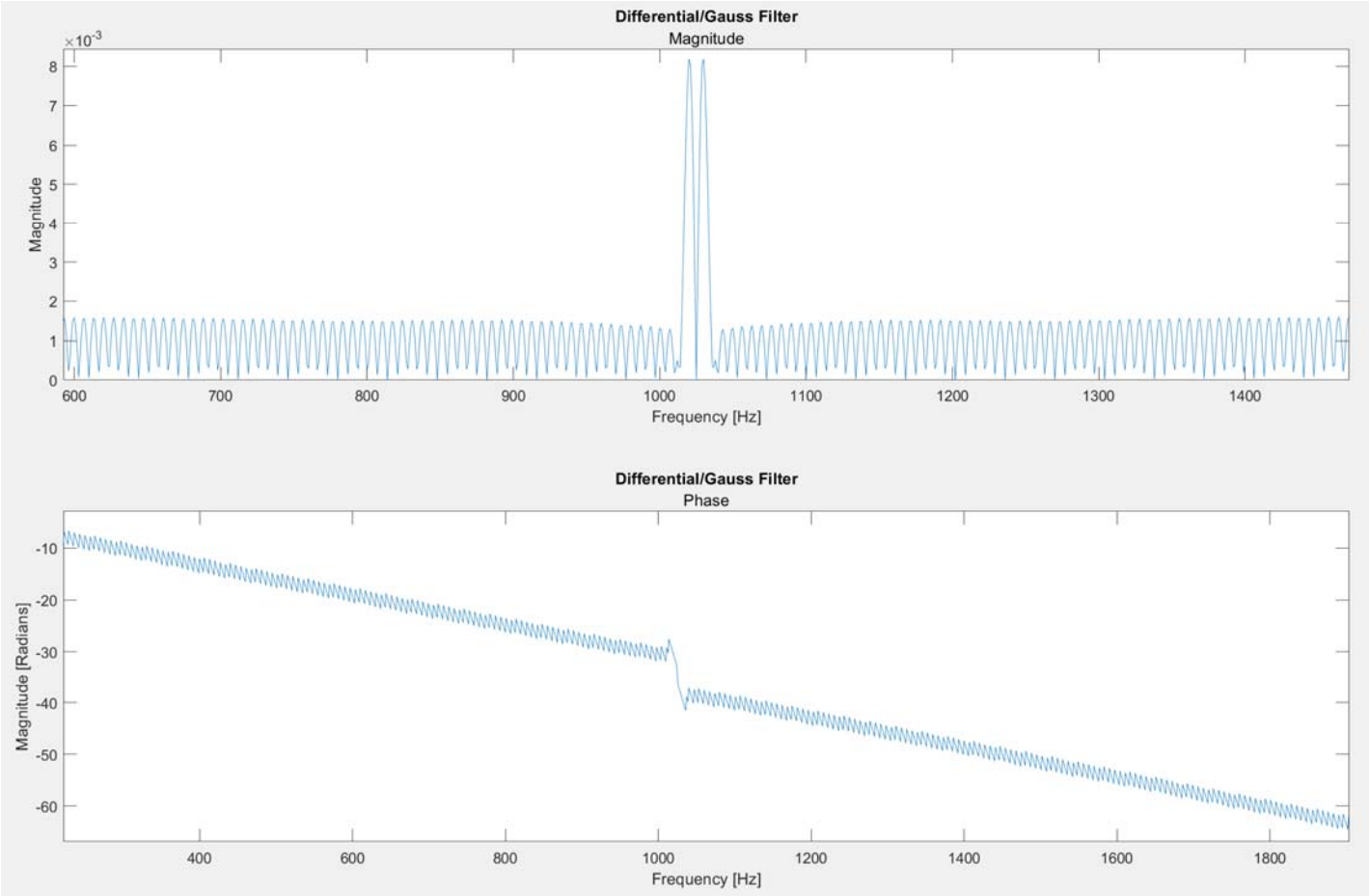

Hilbert Filter
Magnitude
Magnitude
Frequency [Hz]
Hilbert Filter
Phase
Magnitude [Radians]
Frequency [Hz]


Delay Filter
Magnitude
1.0000000000001
1.00000000000005
1
0.99999999999995
0.9999999999999
Magnitude
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Frequency [Hz]
Delay Filter
Phase
0
-100
-200
-300
-400
-500
-600
-700
-800
-900
-1000
Magnitude [Radians]
0
200
400
600
800
1000
1200
1400
1600
1800
2000
Frequency [Hz]



- describe in words the action of the Hilbert transformer based on the frequency response

The Hilbert transformer has both the delay and Hilbert filter portions, which ensure that the original signal will be passed through (delay filter) and that another 'imaginary' signal will be passed through as well. The two signals seem to have a response that decreases the total output from the input signal at lower frequencies, but as it increases past ~1000 Hz, it boosts the output a bit.




```
% Get the audio file from the Internet
filename = 'fireflyintro.wav';
[x,Fs]=audioread(filename); % Read in the file

% Design the Hilbert transformer
N = 321;
hilb = firpm(N-1,[0.005,0.995],[1,1],'Hilbert');
hilb = hilb(:); % Make sure it's a column vector
delay = zeros(N,1);
delay((N+1)/2) = 1;
hilbert_transform = delay + 1j*hilb;

% Design the derivative filter
N = 10; n = [-N:N].';
deriv = (-1).^n ./ n; % Impulse response
deriv(N+1) = 0; % Fix divide by zero
deriv = deriv.*hamming(2*N+1);

% Design the Gaussian filter
gauss = gausswin(301,2);
gauss = gauss/sum(gauss);


% Now do the processing

%do the hilbert transform, separate phase and magnitude, and unwrap
clear myFIRfilter;
y = zeros(size(x));
A = zeros(size(x));
```

```
phi = zeros(size(x));
for n = 1:length(x)
  y(n) = myFIRfilter(hilbert_transform,x(n));
  A(n) = sqrt(real(y(n))^2 + imag(y(n))^2);
  phi(n) = atan2(imag(y(n)),real(y(n)));
end
phi = unwrap(phi);

%run the derivative filter
fi = zeros(size(phi));
clear myFIRfilter;
for n = 1:length(phi)
  fi(n) = myFIRfilter(deriv,phi(n));
end

%run the gauss filter
out = zeros(size(fi));
clear myFIRfilter;
for n = 1:length(fi)
  out(n) = myFIRfilter(gauss,fi(n));
end
out = out/(2*pi) * Fs * 1.1;


%spectrograms of input and instant freq output overlay
NFFT = 2^12;
figure();
[S,F,T] = spectrogram(x,hamming(NFFT),round(0.9*NFFT),NFFT,Fs); hold
on;
imagesc(T,F,20*log10(abs(S)),[0 50]);
colorbar;
set(gca,'YDir','normal');
ylim([0 1500]);
xlabel('Time [seconds]');
ylabel('Frequency [Hz]');
title('Spectrogram of the Input x[n]');
grid on;
%plot freq overlay
t = [0:length(out)-1]/Fs;
plot(t,out,'g','LineWidth',2); hold off;


%Hilbert filter plots
figure();
H = fftshift(fft(hilb,2048));
subplot(211);
plot(abs(H));
title('Hilbert Filter','Magnitude');
xlabel('Frequency [Hz]');
ylabel('Magnitude');
subplot(212);
plot(unwrap(atan2(imag(H),real(H))));
title('Hilbert Filter','Phase');
xlabel('Frequency [Hz]');
ylabel('Magnitude [Radians]');
shg;


%delay filter plots
figure();
D = fftshift(fft(delay,2048));
subplot(211);
plot(abs(D));
title('Delay Filter','Magnitude');
```

```
xlabel('Frequency [Hz]');
ylabel('Magnitude');
subplot(212);
plot(unwrap(atan2(imag(D),real(D))));
title('Delay Filter','Phase');
xlabel('Frequency [Hz]');
ylabel('Magnitude [Radians]');
shg;


%combined hilbert transform plots
figure();
HT = fftshift(fft(hilbert_transform,2048));
subplot(211);
plot(abs(HT));
title('Hilbert Transform','Magnitude');
xlabel('Frequency [Hz]');
ylabel('Magnitude');
subplot(212);
plot(unwrap(atan2(imag(HT),real(HT))));
title('Hilbert Transform','Phase');
xlabel('Frequency [Hz]');
ylabel('Magnitude [Radians]');
shg;


%differential and Gauss filter plots
figure();
HT = fftshift(fft(conv(deriv,gauss),2048));
subplot(211);
plot(abs(HT));
title('Differential/Gauss Filter','Magnitude');
xlabel('Frequency [Hz]');
ylabel('Magnitude');
subplot(212);
plot(unwrap(atan2(imag(HT),real(HT))));
title('Differential/Gauss Filter','Phase');
xlabel('Frequency [Hz]');
ylabel('Magnitude [Radians]');
shg;
```