Devin Trejo

ECE 3522: Stochastic Processes in Signals and Systems

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122

I. PROBLEM STATEMENT

Today we look over autocorrelation and power spectral density (PSD) again. Unlike in our last analysis, we now will compare and contrast the autocorrelation and PSD of several different signals.

The first signal we will observe is a Gaussian noise signal that contains two hundred samples. For the autocorrelation plot we will generate one with 20 lags in it. Since our signal is a noise signal we expected that its power across all frequencies to be constant.

The second signal is a single impulse with one hundred samples in it. The lags in our autocorrelation will range up to 20. In similar vein to the Gaussian we expected a flat frequency response since our time domain signal is short in time. Short in time means broad in frequency. The third signal is also an impulse signal that repeats every 20 samples. Since we have a train of impulses we increase the number of samples to 200 and the number of lags to 60. The PSD should be similar to the single impulse

The fourth signal generated is a sinewave whose periods occurs every 20 samples. The frequency of said sine wave is dependent on the sample frequency. Up to know we have used a sample frequency of 8KHz therefore our sinewave has as frequency of $\frac{fs}{T} = \frac{8KHz}{20} = 400Hz$. We repeat our analysis of the autocorrelation and PSD as explained before. Also for this sine wave case we analyze the behavior if we were to change the number of samples between 14, 17, 20, 23, and 26.

The sixth signal is a sine-wave with a noise component. Given a SNR of 10db we perform the same analyses.

II. APPROACH AND RESULTS

The first signal is Gaussian white noise. We find the autocorrelation and power spectral density (PSD) and plot as seen below.

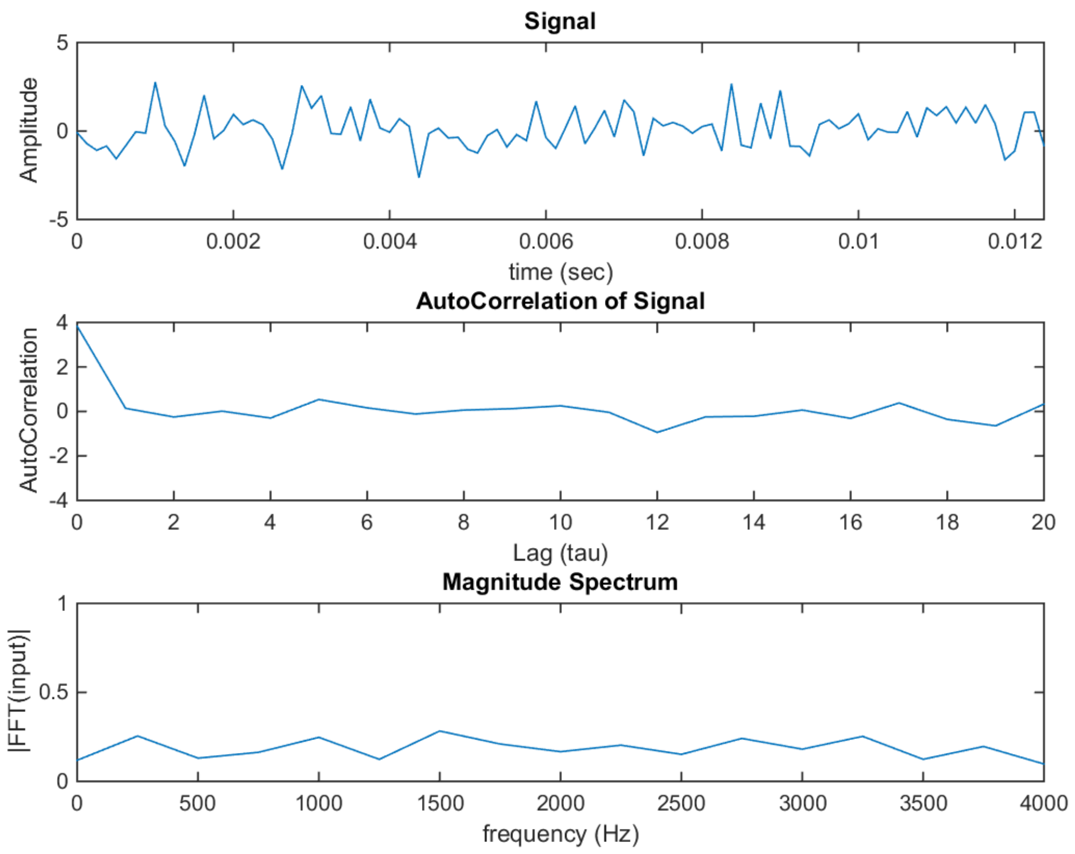


Figure 1: Gaussian white noise plotted with autocorrelation and PSD.

The Gaussian white noise is expected to have energy across all frequencies. We can find the autocorrelation function of our white noise and observe how just by chance the signal looks to be correlated a bit. In truth we know the signal is entirely random.

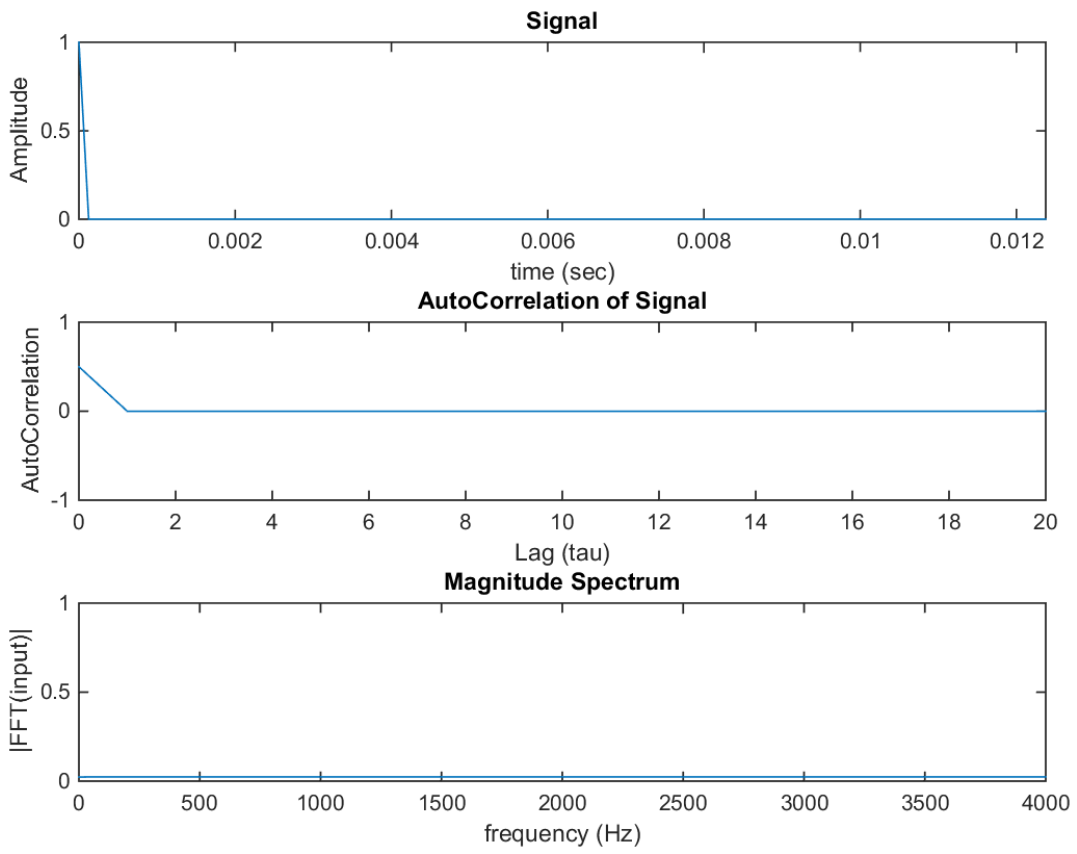


Figure 2: Impulse plotted with autocorrelation and PSD.

In our second signal we have an impulse. Recall from Signal and systems that short in the time domain correlates to wide in the frequency domain. If we find the autocorrelation of our signal we will note that the signal is never correlated to itself since it is not periodic. Our power spectral density thus is flat like our Gaussian white noise signal.

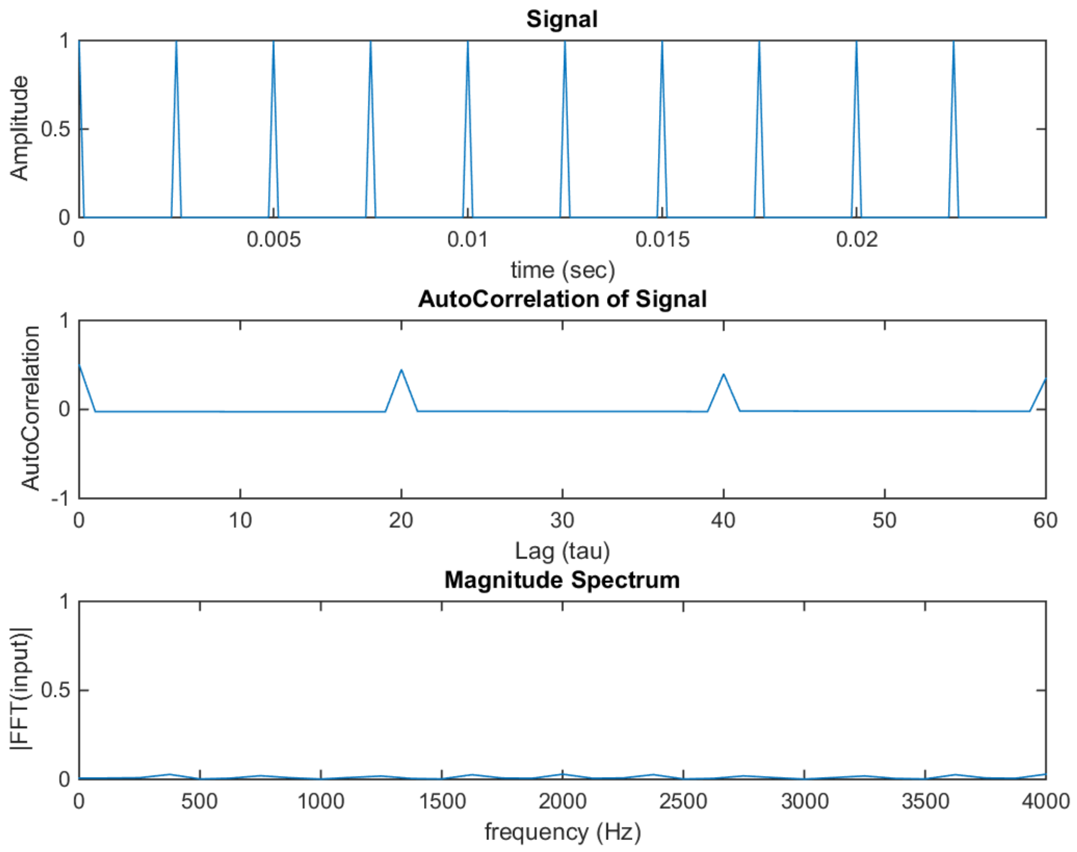
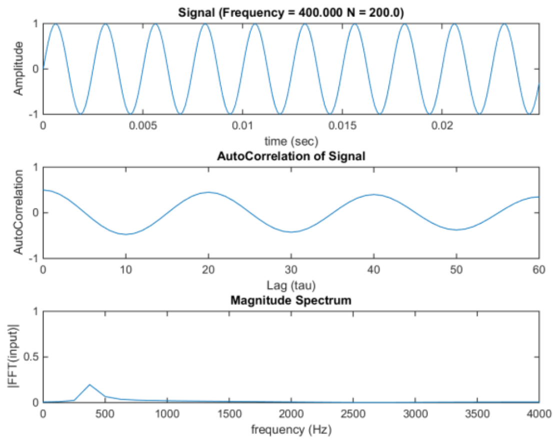
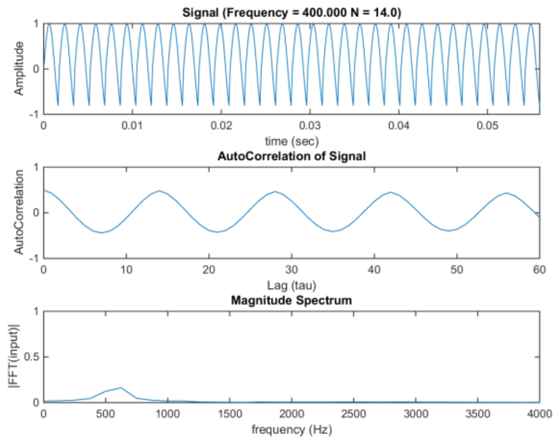


Figure 3: Impulse train plotted with autocorrelation and PSD.

Our third signal is a pulse train. Unlike before our signal is periodic every 20 samples thus our auto-correlation will have magnitude every 20 samples. In the frequency domain (when we find PSD) we can see a periodic nature to our frequency response. It appears around every 400Hz our magnitude spectrum has magnitude.





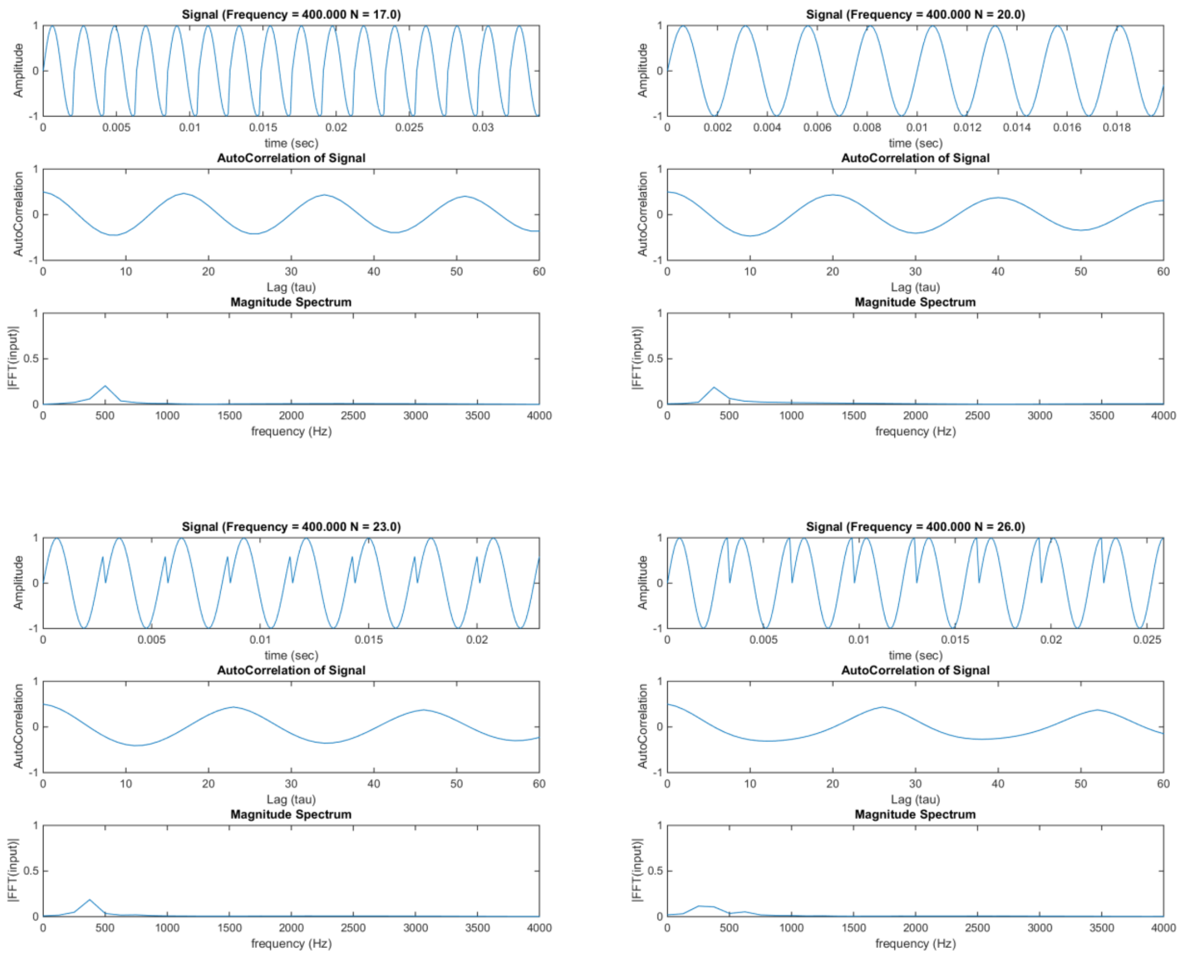


Figure 4: Truncated Sine Wave plotted with autocorrelation and PSD.

For our forth signal we had several different cases. The basis of our signal is a periodic sine wave at 400Hz. We do however truncate our signal at several different lengths. Finding the autocorrelation continues as normal expect now depending on where we truncate our signal it appears our frequency changes. In our first case we have our sine wave of length 200 samples and thus there is no truncation occurring. Our PSD corresponds correctly which a peak at 400Hz. When our sine wave is only 14 samples long however our PSD shows a frequency at around 600Hz. As we increase the amount of samples in our sine wave we move back towards our expected frequency of 400Hz.

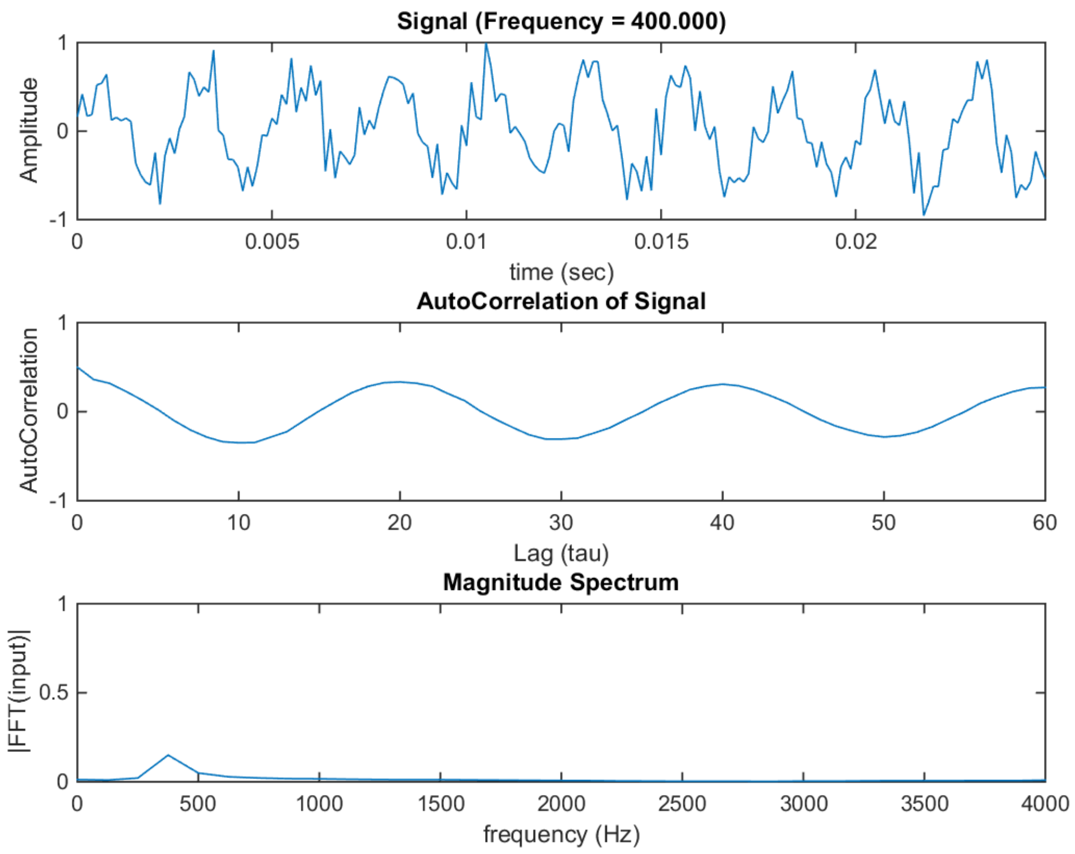


Figure 5: Sinewave with noise plotted with autocorrelation and PSD.

As you can see in our final case we have a noisy sine wave. This case is probably the most interesting since it can explain how useful it is to use have the power spectral density to find the frequency response of the signal. Although our signal is noisy we are still able to frequency response of our signal accurately.

III. MATLAB CODE

The code for this assignment is heavily based of CA 9. The challenge was to create the signals we would analyze.

```
clear; close all; clc;
stdev = 1;
mu = 0;
numSam = 100;
lags = 20;
fs = 8E3;

% Generate
wNoise = devNormDist(mu, stdev, numSam);

% Time Vector
dt = 1/fs;
t = 0:dt:(numSam-1)/fs;

% Auto-Correlation and PSD
figure('name','[ECE 3522] Class Assignment [11]');
subplot(3, 1, 1)
plot(t, wNoise);
xlim([0, max(t)]);
xlabel('time (sec)');
ylabel('Amplitude');
title('Signal');
subplot(3, 1, 2)
aCorr = devAutoCorr(wNoise, lags);
subplot(3, 1, 3)
devFFTMag(aCorr,fs);
```

1 The first signal we generate is a white noise Signal. We use our function from CA 8 to generate Gaussian distributed variables. We then find the autocorrelation using functions from 9. Then we plot the FFT of the autocorrelation.

```
%% Impulse Function
clear; close all; clc;
numSam = 100;
lags = 20;
fs = 8E3;

% Generate
imp = [1 zeros(1, numSam-1)];

% Time Vector
dt = 1/fs;
t = 0:dt:(numSam-1)/fs;

% Auto-Correlation and PSD
figure('name','[ECE 3522] Class Assignment [11]');
subplot(3, 1, 1)
plot(t, imp);
xlabel('time (sec)');
ylabel('Amplitude');
xlim([0, max(t)]);
title('Signal');
subplot(3, 1, 2)
aCorr = devAutoCorr(imp, lags);
subplot(3, 1, 3)
devFFTMag(aCorr,fs);
```

2 The impulse function is generated by setting the first sample to 1 and the rest to zeros. We perform the same analysis as in Part 1.

```matlab
%% Impulse Train
clear; close all; clc;
numSam = 200;
periodSam = 20;
lags = 60;
fs = 8E3;

% Generate
cycles = numSam/periodSam;
for n=0:cycles-1
    tempfn = [1 zeros(1, periodSam-1)];
    if n == 0
        impCycled = tempfn;
    else
        impCycled = [impCycled tempfn];
    end
end

% Time Vector
dt = 1/fs;
t = 0:dt:(numSam-1)/fs;

% Auto-Correlation and PSD
figure('name','[ECE 3522] Class Assignment [11]');
subplot(3, 1, 1)
plot(t, impCycled);
xlabel('time (sec)');
ylabel('Amplitude');
xlim([0, max(t)]);
title('Signal');
subplot(3, 1, 2)
aCorr = devAutoCorr(impCycled, lags);
subplot(3, 1, 3)
devFFTMag(aCorr,fs);
```

3 We creat the period pulse train by setting each 20th sample equal to one. We repeat according to the number of samples in our signal. We perform the same analysis as in Part 1.

```
%% SineWave
clear; close all; clc;
numSam = [200 14 17 20 23 26];
periodSam = 20;
fs = 8E3;
lags = 60;

for N = numSam
    % Time Vector
    dt = 1/fs;
    t = 0:dt:(N-1)/fs;

    % Generate
    f = fs/periodSam;
    str_sigF = sprintf('Frequency = %0.3f N = %0.1f', f, N);
    fprintf([str_sigF '\n']);
    fn = sin(2*pi*f*t);

    % Repeat the sinewave portion multiple times
    if N < lags
        for i = 1:ceil(lags/N)
            fn = [fn fn];
        end
        t = 0:dt:(length(fn)-1)/fs;
    end

    % Auto-Correlation and PSD
    figure('name','[ECE 3522] Class Assignment [11]');
    subplot(3, 1, 1)
    plot(t, fn);
    xlabel('time (sec)');
    ylabel('Amplitude');
    xlim([0, max(t)]);
    title(['Signal (' str_sigF ')']);
    subplot(3, 1, 2)
    aCorr = devAutoCorr(fn, lags);
    subplot(3, 1, 3)
    devFFTMag(aCorr,fs);
end
```

4 We start our script by defining our constants. The assignment asks a sample frequency of 8kHz, a sine of 500Hz w/ Noise, and shifts up to 16. Also on our length of time we want to observe is 0.05 secs. This time length will allow us to observe 25 cycles of our signal. Also we analyze our sine wave signal which is truncated at various samples 14, 17, 20 23, and 26. We repeat this truncated sine wave so that when we run our autocorrelation it has enough samples so our lags do not cause MatLab index errors.

```
%% Signal w/ Noise
clear;  clc;
stdev = 1;
mu = 0;
numSam = 200;
lags = 60;
fs = 8E3;

% Time Vector
dt = 1/fs;
t = 0:dt:(numSam-1)/fs;

% Generate
f = fs/20;
str_sigF = sprintf('Frequency = %0.3f', f);
fprintf([str_sigF '\n']);
noisySig = devGenerate_SineN(1, f, max(t), fs, 10);

% Auto-Correlation and PSD
figure('name','[ECE 3522] Class Assignment [11]');
subplot(3, 1, 1)
plot(t, noisySig);
xlabel('time (sec)');
ylabel('Amplitude');
xlim([0, max(t)]);
title(['Signal (' str_sigF ')']);
subplot(3, 1, 2)
aCorr = devAutoCorr(noisySig, lags);
subplot(3, 1, 3)
devFFTMag(aCorr,fs);
```

5 To generate a signal with noise we use our function 'devGenerate_SineN' from CA 9. We perform the same analysis as in Part 1.

IV. CONCLUSIONS

Throughout the different cases laid out in this computer assignment we observed the application of power spectral density in order to find the frequency response of various signals. In all cases the PSD was able to produce the expect frequency response. The application of PSD is useful when dealing with signals that contain noise. In reality all signals one analyzes has a component of noise.