Computer Assignment (CA) No. 9:
Signal To Noise Ratio and Filtering

Truc Le

ECE 3512: Signals – Continuous and Discrete

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 1912

# Problem Statement

The overall objective of this computer assignment was to provide a function that was able to create a sine wave with a given frequency, sampling time, and add Gaussian white noise to the sine wave at various signal to noise ratios. Then, by using a Fourier Transform to observe the effect of some of the various noise ratios. Later, we would have to use a digital filter equation to filter the signal with added noise and investigate how filtering effected Fourier Transform and autocorrelation.

# Approach and Results

First, we would have to write function to create a sine wave with the frequency needed to convert into radian per second. Then, the white noise was could be created by Matlab’s wng function and summed with the sine wave. Then, by using a for loop to cycle through and gathered the autocorrelation plots for each value of signal-noise ratio. The figures below the various values for signal-noise ratio from a -30 dB to 30 dB.

From Figure 1: Autocorrelation for various signal-noise ratios, we could noticed that the autocorrelation function begins to become erroneous when at a 0 dB. And when it is at a 10 dB, we could noticed that only the first value is similar and then later values tends to fall to zero.

Figure 2 illustrated the Fourier Transform for signal noise ratio of a -30. This could be accomplished with the fft function in Matlab. We would have expected to see that the most noticeable feature will be the spike at the frequency of the sinewave as the sine wave will be far more noticeable than the noise added to the signal.

As in Figure 3, it displayed the Fourier Transform for signal noise ratio of a 30 dB. The plot is centered roughly at 4000 and the frequency at 3141 Hz away from the center including the factor of 2π. We would have to repeat this to see the same signal with a signal-noise ratio of 30 dB. This time we expect to see a Fourier Transform that looks all over the place as it should have a lot more energy at various other frequencies. From Figure 3, we noticed that there are tons of other frequencies that have similar energy levels as expected. The noise introduced has added a lot more variation to the Fourier Transform.

Figure 1: Autocorrelations for Various Signal-Noise Ratios











Figure 2: Fourier Transform for Signal-Noise Ratio of -30 dB (left) Figure 3: Fourier Transform for Signal-Noise Ratio of 30 dB (right)

Then, by using equation: y[n] = 0.5y[n-1] + x[n] to filter the data at a 30 dB signal-noise ratio to observed the effects that it would have on both the Fourier transform and the autocorrelation. After filtering the data, we would have to use Matlab’s autocorr function to get the autocorrelation plot. We could noticed that the plot for the autocorrelation approaches zero a lot more smoothly than the unfiltered would have shown in Figure 4. This could be attributed to the fact that the data is accounting for the past data and the effect of that keeps building as the signal goes on. We then had to take the Fourier Transform of this vector of filtered data is shown in Figure 5. We would noticed that this time a lot of the lower frequencies are more attenuated, whereas higher frequencies are taking over in terms of their presence. This probably mean that the filtering equation acted as a weak high pass filter and pared down some of the lower frequencies that were more dominant before.



Figure 4: Autocorrelation of the Data Filtered Sine Wave



Figure 5: Fourier Transform of Data Filtered Sine Wave

# MATLAB Code

clear; clc; clf; close all;

%function [signal] = generate\_sine(f, s, fsamp, snr\_db)

%%generate sine wave

%%Takes the inputs of Frequency (in Hz), Time duration (in secs),

%%Sampling Frequency (in Hz) and Signal-to-Noise Ratio(in dB) and

%%creates a signal of a sine wave with noise added to it.

%

%%Generate Sine:

%t = 0:(1/fsamp):s;

%omega = 2\*pi\*f;

%sine = sin(omega.\*t);

%

%%Generate Gaussian White Noise:

%noise = wgn(1, length(t), snr\_db);

%

%%Generate Signal:

%signal = zeros(1, length(t));

%

%for i = 1:1:length(t)

% signal(i) = sine(i) + noise(i);

%end

%end

% Part 2:

f = 500;

s = 1;

fsamp = 8000;

snr\_db = [-30, -20, -10, 0, 10, 20, 30];

nlags = 16;

figs = zeros(length(snr\_db), nlags+1);

for a = 1:1:length(snr\_db)

 signal = generate\_sine(f, s, fsamp, snr\_db(a));

 figs(a, :) = autocorr(signal, nlags);

end

%Part 4:

npts = 8192;

minus30 = fft(generate\_sine(f, s, fsamp, snr\_db(1)), npts);

plus30 = fft(generate\_sine(f, s, fsamp, snr\_db(7)), npts);

magneg = abs(minus30);

magpos = abs(plus30);

%Part 5:

x = generate\_sine(f, s, fsamp, snr\_db(7));

y = zeros(1, length(x));

for a = 1:1:length(x)

 if ((a-1) == 0)

 y(a) = x(a);

 end

 if ((a-1) > 0)

 y(a) = 0.5\*y(a-1) + x(a);

 end

end

ACy = autocorr(y, nlags);

FTy = abs(fft(y, npts));

%%%Plotting Section:

figure(1);

plot(figs(1, :));

title('Autocorrelation of -30 dB Signal-Noise Ratio:');

figure(2);

plot(figs(2, :));

title('Autocorrelation of -20 dB Signal-Noise Ratio:');

figure(3);

plot(figs(3, :));

title('Autocorrelation of -10 dB Signal-Noise Ratio:');

figure(4);

plot(figs(4, :));

title('Autocorrelation of 0 dB Signal-Noise Ratio:');

figure(5);

plot(figs(5, :));

title('Autocorrelation of 10 dB Signal-Noise Ratio:');

figure(6);

plot(figs(6, :));

title('Autocorrelation of 20 dB Signal-Noise Ratio:');

figure(7);

plot(figs(7, :));

title('Autocorrelation of 30 dB Signal-Noise Ratio:');

%%%Plotting Section 2:

figure(1);

plot(magneg);

xlim([(length(magneg)/2), length(magneg)]);

title('Fourier Transform for Signal-Noise Ratio of -30 dB');

figure(2);

plot(magpos);

xlim([(length(magneg)/2), length(magneg)]);

title('Fourier Transform for Signal-Noise Ratio of 30 dB');

%%%Plotting Section 3:

figure(1);

plot(ACy);

title('Autocorrelation of Data Filtered Sine with Signal-Noise Ratio of 30 dB');

figure(2);

plot(FTy);

xlim([(length(FTy)/2), length(FTy)]);

title('Fourier Transform for Data Filtered Signal-Noise Ratio of 30 dB');

function [signal] = generate\_sine(f, s, fsamp, snr\_db)

%generate sine wave

%takes the inputs of frequency (in Hz), Time duration (in seconds),

%Sampling frequency (in Hz) and Signal-to-Noise Ratio (in dB) and

%creates a signal of a sine wave with noise added to it.

%generate Sine wave:

t = 0:(1/fsamp):s;

omega = 2\*pi\*f;

sine = sin(omega.\*t);

%Generate Gaussian White Noise:

noise = wgn(1, length(t), snr\_db);

%Generate Signal:

signal = zeros(1, length(t));

for i = 1:1:length(t)

 signal(i) = sine(i) + noise(i);

end

end

# Conclusions

In conclusion, we can realized that noise can really distort a signal once the ratio goes more in favor of the noise. And Autocorrelation indicates that the noise can distort the signal to the point that it’s not identical and aperiodic with respect to itself. Furthermore, the Fourier Transform illustrated that the noise has a lot to do with what frequencies we see more or less of. To eliminate noise, filtering the data is one handy method by using a proper filtering algorithm.