ECE 317 Experiment 0

The purpose of this experiment is to understand how to represent signals in MATLAB, perform the convolution of signals, and study some simple LTI systems. Please answer all questions and include all the printouts in your report!

Part A. Representing Signals

A discrete time signal can be considered as a vector, an ordered collection of real or complex numbers with two or more dimensions. In MATLAB, vectors are the fundamental date type. Vectors are 1-by-n or n-by-1 arrays, where n is the number of elements in the sequence.

1. Type the statement

```
x = [0 0 1 1 1 1 1 1 1 1]
```

in the command window and press [enter]. MATLAB will run your statement and display the running result. What is the result displayed? Then, type and run the statement

```
stem(x)
```

Print the result. Then, let

```
k = [-2:1:7]
```

and run

```
stem(k,x).
```

Print the result. Now you should have two graphics. One is u[n] and the other is the time-shift u[n-d]. In the printouts, indicate which one is u[n] and which one is u[n-d], and what is the value of d?

2. Using the statement

```
x=[zeros(1,N), ones(1,M+1)];
```

and choosing suitable vector k, display the graphics of u[n] and u[n-5] for $-10 \leqslant n \leqslant 20$, respectively. Print your codes and the output figures.

3. Let the signal x[n]=7cos(0.1n)+cos(0.95n). Print the graphics of x[n] and x[n-20] for $-40 \leqslant n \leqslant 80$, respectively.
hint: what will happen if you run cos(0.1*k) for k=[-40:80]?

Part B. Convolution and Simple LTI Systems

1. In the command window, type "help conv" and press [enter]. Read the information displayed.

2. Consider the moving average system h=[1,1,1,1,1]/5. If the input signal is x, then the output signal will be y=conv(x,h). For the following input signal x, print the graphic of

the output.

(a). x=ones(1,50)
(b). x=cos(0.1*k) for k=[-40:80]
(c). x=cos(0.95*k) for k=[-40:80]
(d). x is the signal obtained in Part A.3

3. Consider the difference system h=[1,-1]. For the following input signal x, print the graphic of the output.

(a). x=ones(1,50)
(b). x=cos(0.1*k) for k=[-40:80]
(c). x=cos(0.95*k) for k=[-40:80]
(d). x is the signal obtained in Part A.3

4. According to the results you obtained in step 2 and 3, what can you say for the average system and the difference system?

ECE 317, Digital Signal Processing I

Experiment 1

In this experiment you will investigate using a Fourier series to represent a continuous time periodic signal $x(t)$. MATLAB will be used for this. The complex exponential form for a Fourier series of $x(t)$ is given by

$$x(t) = \sum_{k=-\infty}^{k=+\infty} X_k e^{j\,k\Omega_0 t}$$

where $\Omega_0 = 2\pi/T_0$ is the fundamental frequency, T_0 is the period of $x(t)$, and the complex Fourier series coefficients are given by

$$X_k = \frac{1}{T_0} \int_{t_0}^{t_0+T_0} x(t) e^{-jk\Omega_0 t} dt$$

for any t_0. To obtain a value of $x(t)$ for some t, we must truncate the series for $x(t)$ and approximate $x(t)$ with

$$\hat{x}(t) = \sum_{k=-K}^{k=+K} X_k e^{j\,k\Omega_0 t}$$

for some K. As K is increased, we expect the approximation to improve.

1a. For some particular periodic signal a MATLAB program to calculate its Fourier series could be:

```
clf; clear; %Clear all workspace
T0 = 4.0; %The period of x(t)
W0 = 2.0*pi/T0;  %The fundamental frequency
%Let us evaluate x(t) for N time points every
%T seconds over two periods of x(t)
T = 0.001; N = 8000;
%We will get 8000 points over the range t=0.0 to t=(N-1)T secs
%Let the time points be N elements of a vector time, starting at
%time(1)=0.0, incrementing by T, until the last element time(N)
time = 0.0:T:(N-1)*T;
%Let us truncate the series for x(t) from -K to +K
%and try K=5
K = 5;
%Using a MATLAB function to set to zero the N elements
%of a vector x
x = zeros(1,N);
%For each pass through the following for-loop a term in the Fourier
```

```
%series for x(t) is evaluated for all the elements (time
% points) in the vector time
for k = -K: 1: K;
   W=k*W0;
   if k == 0;
      Xk = -2.0;%Gives the average value of x(t)
   else;
      C=1.0/W^2;
      Xk = i*(2.0*C*(sin(2.0*W)-2.0*sin(W)));
      %i is the MATLAB symbol for the square root of (-1)
   end;
   x = x + Xk*exp(i*W*time);
end;
plot(time,real(x));
grid;
xlabel('time-secs'); ylabel('volts');
title('Fourier series using a finite number of terms');
end
```

(I) Explain the purpose of the program for-loop, and thoroughly explain the activity of the program line given by: x = x + etc.
(II) Enter the program into MATLAB and run it. Print the plot. On the plot, sketch the periodic signal that you think this Fourier series approximates. What is the signal's period? What is the value of the Fourier series representation at $t = 4.0$?
(III) Add statements to the program to calculate and plot the magnitude spectrum of the signal versus frequency expressed in Hz. Use a MATLAB stem type plot for this. The frequency range must be from $-K/T_0$ Hz to $+K/T_0$ Hz. Print the program.

1b. (I) Change K to 10, and print the plots.
(II) Change K to 100, and print the plots.
(III) If K is made even larger, will the error $e(t) = x(t) - \hat{x}(t)$ between the periodic signal and its Fourier series approximation ever become zero for all t ? Explain how you know this.

1c. Will a measure of performance like the number ε^2 given by

$$\varepsilon^2 = \int_0^{T_0} e^2(t)dt$$

ever become zero if K approaches infinity? Explain your answer.

2. Define and sketch a square wave. Use the first and second nonzero digits in your SS# as its period and amplitude, respectively. Find its Fourier series coefficients. For this signal modify the program of part 1, and repeat part 1. For 2b, part (III), what is the oscillatory error behavior called?

3. Discuss the difference between the signals of parts 1 and 2 that would account for (or explain) the difference in the behavior of their Fourier series approximations as K becomes large.

ECE 317, Digital Signal Processing I

Experiment #2

Running a Digital Filter

The purpose of this experiment is to operate an FIR (Finite Impulse Response) and an IIR (Infinite Impulse Response) digital filter. By applying a sinusoidal input having different frequencies, you will see that the output amplitude and phase depend on the input frequency, and therefore such algorithms have a frequency selective behavior. MATLAB will be used to study the filters.

1. The first digital filter we will examine is defined by the 6th order FIR difference equation given by

$$y(n) = \frac{1}{21}\,[-2x(n) + 3x(n-1) + 6x(n-2) + 7x(n-3) + 6x(n-4) + 3x(n-5) - 2x(n-6)]$$

where $x(n)$ comes from sampling a continuous time signal $x(t)$ at the sampling rate f_s .

First, find the unit pulse response, which is the response $y(n)$ when the input is $x(n) = \delta(n)$. To conveniently do this in MATLAB, let B be the vector of filter coefficients, and let X be the vector that holds present and past inputs. A sample MATLAB program is:

```
clear all; clc;
B = [-2 3 6 7 6 3 -2]/21;  % the filter coefficients
X = zeros(1,7);  % initializing X to zero
N = 20;  % obtaining the unit pulse response for n=0,1,2, … N
n = 0;  % initializing the discrete time index to zero
while n <= N
        time_index(n+1) = n;  % matrix index cannot by zero
        % if n=0, input should be 1
        if n == 0
                x(n+1) = 1;
        else
                x(n+1) = 0;
        end
        X(1) = x(n+1);  % placing the present input in the first position of X
        % computing the output with the inner product of B and X
        y(n+1) = B * X';
```

```
        n = n+1;  % incrementing the time index
        % shifting content of X down to setup for processing the next input
        X = [0 X(1:6)];
end
stem(time_index,x);  % plotting the input
grid on
title('unit pulse input');
xlabel('discrete time index')
ylabel('input')
disp('depress enter to continue')
pause
stem(time_index,y);  % plotting the output
grid on
title('unit pulse response, h(n)');
xlabel('discrete time index')
ylabel('output')
disp('depress enter to continue')
pause
```

Your report should include a program listing, plots and answers to given questions.

a) In the above program, just before computing the output, give the vector X when n=0.

b) Repeat part (a) for n=3 and n=8.

c) Explain why the output is nonzero only for a finite time duration.

2. Now we will find the response of the FIR filter to a sinusoidal input. Let

$$x(t) = \cos(\omega\, t)$$

where $\omega = 2\pi f$. Let the sampling frequency be $f_s = 8000$ Hz. Find the response for $0 \leq t \leq 0.01$ sec.

a) Let $f = 100$ Hz. The program given above can be modified to start as:

```
clear all; clc;
B = [-2 3 6 7 6 3 -2]/21;  % the filter coefficients
X = zeros(1,7);  % initializing X to zero
T0 = 0.01;  % response time range
fs = 8000; T = 1/fs;  % setting the sample time increment
N = floor(T0/T);  % number of input samples to be processed
f = 100;  w = 2*pi*f;  % input frequency
n = 0;  % initializing the discrete time index to zero
while n <= N
```

```
time_index(n+1) = n;
time = n*T;
x(n+1) = cos(w*time);  % sampling input
X(1) = x(n+1);  % placing the present input in the first position of X
        .
        .
```

Complete the above program segment, and make sure that plot labels are correct. The initial behavior of the filter output, which is not sinusoidal like the input, is the transient response of the filter. Then, the output becomes a sinusoid, like the input. This is the steady state response of the filter. What is the amplitude of the sinusoidal steady state response? Provide program listing and plots.

b) Run your program to find the steady state output amplitude for each of the frequencies $f = 100 + k * 200,\ k = 0, 1, \ldots$, 18. Then give a plot of the steady state output amplitude versus input frequency. Since the input amplitude is unity, this shows the frequency response of the filter. What kind of filtering activity does this FIR filter exhibit?

c) Now let the input come from sampling

$$x(t) = \cos(\omega_1 t) + \cos(\omega_2 t)$$

where f_1 = 200 Hz and f_2 = 2400 Hz. Provide plots of the input and the output. Discuss the difference between the input and the output.

3. The second digital filter we will study is described by the 2nd order IIR difference equation described by

$$y(n) = 0.09780\, x(n) + 0.19560\, x(n-1) + 0.09780\, x(n-2) +$$

$$0.94175\, y(n-1) - 0.33296\, y(n-2)$$

First, we will find the unit pulse response. To conveniently do this with MATLAB, let the vector B hold the coefficients of the present and past values of the input, so that B is defined to be

B = [0.09780 0.19560 0.09780]

As before, let the vector X hold the present and past values of the input. At $n = 0$ we get $x(0)$, and X becomes X = [$x(0)$ 0 0]. Let the vector A hold the coefficients of the past outputs, so that A is defined to be

A = [-0.94175 33296]

Also, let the vector Y hold past outputs. Thus, initially, at $n = 0$, the vector Y is given by

Y = [y(-1) y(-2)] = [0 0]. In MATLAB, $y(0)$ is computed with: $B * X' - A * Y'$. To continue processing the input, we must shift down the content of both X and Y.

Write a MATLAB program, and provide a program listing and plots of the input and output.

a) In your program, give X and Y just before computing the output at $n = 0$.

b) Repeat part (a) for $n = 2$ and $n = 4$.

c) Is the output nonzero for a finite or an infinite time duration? Explain.

4. Now you will repeat part (2) for the IIR filter.

a) Repeat part (2a).

b) Repeat part (2b).

c) Repeat part (2c).

ECE317, Digital Signal Processing I

Experiment 03

The purpose of this experiment is to do some spectral analysis and to filter a signal $x(t)$ in the frequency domain. MATLAB will be used for this. Include in your report all of the MATLAB programs and prints of the plots. **THIS IS A TWO WEEK EXPERIMENT**

MATLAB has a built-in function to compute the DFT (discrete-time Fourier transform): fft(). For the inverse Fourier transform, the MATLAB function is ifft().

Assume $x(t)$ is a continuous-time signal. By choosing N samples with rate f_s, we can obtain a sampled discrete-time signal $\{x(k/f_s)\}$ for $k = 0, 1, 2, \cdots, N$, from the original signal in the time interval $[0, T]$, where $T = N/fs$.

1. Test the FFT and IFFT functions using signal $x(t) = 5\cos(2\pi 500t)$

a. Let the sample rate $f_s = 8$ KHz. Take $N = 10$ samples and plot the sampled signal. Is the resulting periodic signal a pure sinusoid?

Obtain the DFT by using the FFT, and plot the magnitude spectrum. Notice that the frequency resolution in your spectrum plot is $\Delta = \frac{1}{T} = \frac{f_s}{N}$. How does this result compare to the theorital spectrum of $x(t)$? If they are different, explain the reason.

Now use the IFFT function to restore the discrete time signal from its spectrum, and plot it. Does the signal go through an integer number of cycles?

b. Repeat part (a) for $f_s = 8$ kHz and $N = 16$, 25, and 32.

c. Repeat part(a) for $f_s = 8.3$ kHz and $N = 10$, 16, 25, and 32.

d. In parts (a), (b), and (c), did any spectral point occurs exactly at the frequency $f = 500$? Did any choice of f_s and N can sample the signal $x(t)$ over an integer number of cycles?

e. Set N to a power of two. Then, what is a good choice for f_s (or T) such that some integer multiple of Δ will occur at 500Hz? Using these values, obtain the DFT, and plot its magnitude. Print two copies of the magnitude spectrum.

2. Sometimes a power of two FFT must be used to obtain a DFT, while the number of data points available is not a power of two. Then, zeros are added to the available data points such that the total number of data points is a power of two. This is called zero-padding. In zero-padding, N and T are increased, and consequently, the frequency resolution $\Delta = 1/T$ of the resulting DFT is also changed.

a. Using the signal of part 1 and $f_s = 8$KHz, write a MATLAB program to obtain 50 samples of the signal. What is the resulting T? Performing FFT on the sampled signal. Is there an integer multiple of $1/T$ that occurs at the frequency 500 Hz?

b. Zero-pad the data of part (a) to obtain a total of 64 data points. What is the resulting T? Is there a resulting integer multiple of the new frequency resolution $1/T$ that occurs at the frequency 500Hz? Use the MATLAB FFT function to obtain the DFT, and plot the magnitude spectrum. At what frequency does the spectrum plot peak?

c. Repeat part (b) by performing zero-padding on the data of part 1(a) and 1(b) to $N = 64$. Compare the results. Indicate the changes after zero-padding and explain the reason.

d. Now, you can use one of the two copies of the printed spectrum in part 1(e). Without computing the DFT, sketch on this copy the expected magnitude spectrum for zero-padding with double N. After then, compute DFT to verify your sketching.

3. In this part we will see how well the DFT can be used for spectral analysis when a signal has sinusoids having frequencies that are close in value. Let us use the signal

$$x(t) = 2.0\cos(w_1 t) + 2.0\cos(w_2 t),$$

where $w_1 = 1900\pi$ and $w_2 = 2000\pi$. Set the sampling frequency to $f_s = 8$kHz again. Write a MATLAB program to compute the DFT, and plot the magnitude spectrum for $N = 16$, 64, 512 and 8192. To see two separate spectral peaks in the magnitude plot, how large must N be?

4. In view of the above results in applying the DFT for spectral analysis, give several guidelines concerned with data taking and parameter selection such that a DFT (computed with an FFT operating on a power of two number of data points) can give a good indication of the spectral content of a signal.

5. To the sampled data of part (1b) add a random noise sequence of data. MATLAB has a uniformly distributed random number generator that produces numbers in the range (0,1). Subtract 1/2 from each such number, and multiply the result by 10 to obtain numbers randomly distributed over the range (-5, +5), having an average value equal to zero. Add such a random number sequence to the sampled data. Plot the noisy signal. Is the sinusoid obvious in the noisy signal plot? Obtain the DFT, and plot the magnitude spectrum. Is the frequency of the sinusoid obvious in the spectrum plot?

6. To filter a signal:

i) Sample the signal over the range $0 < t < T$ to obtain N points $x[n] = x(nt_s)$, $n = 0, 1, \cdots, N-1$.

ii) Obtain $X(k)$, $k = 0, \cdots, N-1$, the DFT of $\{x[n]\}$.

iii) Multiply $\{X(k)\}$ by a specified filtering pattern, $H(k)$, $k = 0, \cdots, N-1$, where $H(k) = 1$ and $H(-k) = 1$ means to keep the spectral components at frequencies $f = k/T$ for that k, and $H(k) = H(-k) = 0$ if you want eliminate the corresponding frequency components.

iv) Then, take the inverse DFT of the result of step (iii).

This method of filtering is a batch processing method, unlike a real-time method using a difference equation. By batch processing we mean that all data must be available for use before any processing can occur.

Now you will do different kinds of frequency domain filtering by choosing different $H(k)$. Use samples $x[n]$ obtained from the signal:

$$x(t) = \sin(w_1 t) + \sin(w_2 t) + \sin(w_3 t),$$

where $w_1 = 400\pi$, $w_2 = 2000\pi$, and $w_3 = 4000\pi$. Set the sampling frequency to $f_s = 8$kHz, and let $N = 2^{13}$. With the given parameters, $T = 1.024$sec, and spectral lines will be obtained at integer multiples of $\Delta = 1/1.024 = 0.97656$Hz.

a. Write a MATLAB program to obtain $\{x[n]\}$ and its DFT, and plot the magnitude spectrum. Locate the spectral peaks. Are these results expected?

b. Add statements to the MATLAB program for high-pass filtering with a cut-off frequency of, for example, 500Hz. For this, the corresponding frequency domain index range must be determined. Let K be the maximum integer such that $K\Delta < 500$, resulting in $K = 512$. Set all entries in H to unity, and then let $H(k) = H(N-k) = 0$ for $k = 0, 1, \cdots, K$. Plot $H(n)$. To filter $x[n]$, do an element by element multiply of $X(n)$ and $H(n)$ to obtain $Y(n)$. Then take the inverse DFT of $Y(n)$. Plot the resulting signal. Is the result as expected?

c. Now H will be set up to a low-pass filter with a cut-off frequency at 1500Hz. Here, initialize all elements of H to zero, and then set certain values to unity to achieve the desired low-pass filter frequency response. Then, continue as in part (b) to apply the filter H on $x[n]$.

d. Repeat part (c) with a band-pass filter with a pass band from 300Hz to 1600Hz.

ECE 317, Digital Signal Processing I

Experiment 04

Effect of Pole and Zero Locations on the Frequency Response Of an LTI Discrete Time System

In this experiment you will investigate the effect of pole and zero placement in the z-plane on the frequency response of an LTI DTS IIR digital filter. MATLAB will be used for this.

The transfer function $H(z)$ of the LTI DTS that we will operate is given by

$$H(z) = \frac{(z-z_1)(z-z_2)}{(z-re^{j\theta})(z-re^{-j\theta})}$$

where the two zeros of $H(z)$ are: z_1 and z_2, and the two poles of $H(z)$ are: $re^{j\theta}$ and $re^{-j\theta}$, and where r is the magnitude of the poles and θ and $-\theta$ are the angles of the poles. The transfer function can also be written as

$$H(z) = \frac{(1-z_1 z^{-1})(1-z_2 z^{-1})}{(1-re^{j\theta} z^{-1})(1-re^{-j\theta} z^{-1})} = \frac{1-(z_1+z_2)z^{-1}+z_1 z_2 z^{-2}}{1-2r\cos(\theta) z^{-1}+r^2 z^{-2}}$$

By inspection of $H(z)$, the difference equation relating the output $y(n)$ to the input $x(n)$ is given by

$$y(n) = x(n) - (z_1+z_2)x(n-1) + z_1 z_2 x(n-2) + 2r\cos(\theta) y(n-1) - r^2 y(n-2)$$

Depending on the pole and zero locations in the z-plane, this algorithm can do different kinds of filtering. Let's investigate the possibilities.

1) Set the sampling frequency to $f_s = 8\ KHz$, and get the input $x(n)$ from sampling

$$x(t) = \cos(400\pi t) + \cos(3600\pi t) + \cos(7000\pi t)$$

(a) What are the frequencies in Hz of the sinusoids in $x(t)$? Set the poles and zeros as follows: $z_1 = -1,\ z_2 = -1,\ r = 0.95,$ and $\theta = \pi/20$. (b) In the z-plane and with respect to the unit circle, plot the poles and zeros of the transfer function. (c) Calculate and plot the magnitude frequency response over the frequency range, $0 \le f \le f_s$. (d) What kind of a filter is this? (e) Calculate

the coefficients of the difference equation, and give the difference equation. (f) Apply the input to the difference equation, and run the algorithm long enough to reach steady-state behavior, and plot the input and output. (g) In view of the frequency response, discuss the performance of the filter.

2) Repeat part (1) for $r = 0.65$. How did the pole locations change? Discuss what happened to the frequency response?

3) Repeat part (1) for $r = 0.99$. How did the pole locations change? Discuss what happened to the frequency response?

4) Repeat part (1) for: $z_1 = +1,\ z_2 = -1,\ r = 0.95,$ and $\theta = \pi / 2.1$.

5) Repeat part (1) for: $z_1 = +1,\ z_2 = +1,\ r = 0.95,$ and $\theta = \pi / 1.1$.

6) Using the coefficients of parts (1), (2), and (3), obtain and plot the unit pulse response. How does the unit pulse response change?

7) Repeat part (1) for: $z_1 = +1,\ z_2 = -1,\ r = 1.05,$ and $\theta = \pi / 2.1$. Explain what happened?

8) Overall, give a discussion about the relationship between pole magnitude and angle and the LTI DTS frequency response. How do the zero locations affect the frequency response?

9) Set the zeros to: $z_1 = 0,\ z_2 = 0,$ and set the pole magnitude to: $r = 1.0$. Three values of θ will be tried, which are: $\theta_1 = \pi / 20,\ \theta_2 = \pi / 10,$ and $\theta_3 = \pi / 5$. Set the sampling frequency to $f_s = 20\ KHz$. For each pole angle, give a pole zero plot. Describe how the poles are being moved.

(a) For θ_1, calculate the coefficients and give the difference equation. For a unit pulse input, run the algorithm just long enough to reach steady state conditions. Plot the unit pulse response. What is the frequency of the steady-state response?

b) Repeat part (a) for θ_2.

c) Repeat part (a) for θ_3.

d) This algorithm can be used as a sine wave generator. What value of θ must be used to generate a 2.5 KHz sine wave?

ECE 317, Digital Signal Processing I

Experiment #5

Signal Noise Due to Word Length

In this experiment you will investigate the effect of finite word length. Quantization error will be simulated. The resulting signal to noise ratio will be used as a measure of quantization error.

1. Below is a program that uses the function bits, which is also shown below, to change the number of bits to represent a number from the default number of bits used by MATLAB to a number of bits specified by the user of the function bits. The input to the function bits is the number x and the number of bits nF to be used for representing x. The output y is the corresponding value of x using nF bits.

```
clear all; clc;
% program terminates if the input is zero
nF = 7;
while 1
    x = input('enter a number in the range -1 to +1: ')
    if x == 0
        break
    end
    y = bits(x,nF)
end
```

```
function y = bits(x,nF)
% Input x is a signed fraction. The input is converted to output y,
% where the absolute value of y has a binary representation using nF
% fractional bits.
if x == 0 | nF <= 0
    y=0;
    return
end
z = abs(x);
if x == z
    sign=1;
else
    sign=-1;
end
if z >= 1
    y=sign;
    return
end
y = 0; half_power = 1.0;
```

```
for n=1:nF
    half_power=0.5*half_power;
    z_2 = 2*z;
    if z_2 >= 1
        y = y+half_power;
        z = z_2-1;
    else
        z = z_2;
    end
end
y = sign*y;
```

a) Enter this program and the function bits. Give a table of ten values of x and y. Repeat this for nF = 3 and 2. Are the results as expected? Give a detailed explanation.
b) If we can work only with fractions, then overflow can occur when we add fractions. If we expect to add up to sixteen fractions and maintain 7 bit, for example, accuracy, then we must actually use 11 bits, where each number to be summed is scaled to have its four leading bits set to zero. Thus, x/16 scales x, and y=bits(x/16,11) produces a number that in binary has its four leading bits set to zero. Modify the above main program to add ten fractions, while maintaining 7 bit accuracy. Provide a program listing. Describe a way to verify that your results are sensible.
c) If we multiply two fractions, then overflow cannot occur. However, using nF bits, a product requires 2*nF bits and we must truncate the result. This can also be investigated with the function bits. Consider: z = x1 * x2, using nF bits. First, let y1 = bits(x1,nF) and y2 = bits(x2,nF). Then, get w = y1 * y2. However, w must be truncated to nF bits, which is done with z1 = bits(w,nF). Write a main program to illustrate this for nF = 3. Provide a program listing. For several pairs of numbers x1 and x2, compare z and z1, and explain why results are different. Give a pair x1 and x2, where z and z1 are the same.

2. Below is a program that uses a large number N of numbers selected randomly to investigate the signal to noise ratio due to quantization error.

```
clear all; clc;
N = 100000; % using N numbers
numbers = rand(1,N); % get N random numbers uniformly distributed from
0 to 1
x = 2*numbers-1; % N numbers uniformly distributed from -1 to +1
y = zeros(1,N); % setting space for the quantized numbers
mean_x = mean(x) % ideally, x should have zero mean
x = x - mean_x; % making sure that x has zero mean
nF = 7; % represent each entry in x using nF bits
for i=1:N
    y(i)=bits(x(i),nF); % quantizing x
end
er = x-y; % get quantization error
mean_er = mean(er); % quantization error should have zero mean
er = er - mean_er; % making sure that quantization error has zero mean
er_energy = sum(er.*er); % quantization noise energy
```

```
signal_energy = sum(x.*x); % signal energy
S_N = 10*log10(signal_energy/er_energy) % signal to noise ratio
```

a) Enter the above program, and give a plot of the signal to noise ratio for nF = 1, 2, …, 10. How does the signal to noise ratio change as nF is increased? Provide a program listing. Also, try N = 1000000 for nF = 7, get the percent change in the signal to noise ratio. If for N = 1000000 the resulting signal to noise ratio is different by more than a few percentage points, then N = 100000 is not large enough for useful statistical analysis of quantization error.
b) Modify the above main program to obtain the N points for x by sampling a sinusoidal function with unity amplitude and zero phase at the sampling rate fs = 44100 samples/sec. Include MATLAB statements: sound(x,fs) and sound(y,fs), so that you can hear both the original sinusoid and the sinusoid using quantized samples. Provide a program listing. Repeat part (a) for frequencies of 100, 500, 1500 and 4000Hz. For a given nF, does the signal to noise ratio depend on frequency? At each frequency, and as you decrease nF, when is nF small enough to hear a noise component in y, and what is the signal to noise ration? Will N=100000 and N=1000000 result in different conclusions?

3. Use the operating system recording program to create a stereo WAV file of yourself speaking a sentence into a microphone. If there is an option, use 44100 Hz for the sampling rate. Instead, you could sing a song if you like. Be sure and store the WAV file in the folder where you keep your MATLAB programs (the current folder when using MATLAB). Use the multimedia player program to check your WAV file. MATLAB can read and write WAV files.

The following program reads the WAV file, quantizes each sample and plays the resulting quantized sound.

```
clear all; clc;
Title = 'zevon.wav'; % place in quotes the name of your WAV file
[my_sound,fs,n_bits] = wavread(Title); % getting WAV file
% a stereo WAV file consists of two columns having the same length.
N = length(my_sound); % each channel has N samples
left_chan = my_sound(1:N,1); % getting the left channel
x = zeros(N,1); % setting up space for the quantized sound
nF = 13;
for i=1:N
    x(i)=bits(left_chan(i),nF);
end
sound(x,fs);
```

Modify this program to include computing the signal to noise ratio. Use the program in part (2) as a model. Provide a program listing. Run this program for nF = 3, 4, …, 13, and give a plot of the signal to noise ratio vs. nF. For which nF does the quantized sound seem to be acceptable to you? What is the signal to noise ratio?

ECE317, Digital Signal Processing I

Experiment 06

The purpose of this experiment is to study the sampling process. By taking samples from a continuous-time signal, a discrete time signal will be obtained. This discrete time signal may be a good represention of the original continuous-time signal or not, depending on the singal's feature and how many samples are taken. In this lab, we will investigate what will happen during the sampling process, in both time-domain and frequency-domain. Print and put all your Matlab codes and plots in your report. Write necessary notes on each print out to explain what it is.

1. Take samples from signals

(a) Consider the cosine function of frequency 30Hz:

$$g1(t) = cos(60\pi t).$$

Take samples with a sample rate of 100Hz, i.e., $T = 0.01$ Sec. Then a discrete time signal will be obtained:

$$x1[n] = cos(0.6\pi n).$$

Plot and print three figures:

(i) $g1(t)$ for $0 < t < 0.16$. Use function "plot"

(ii) $x1(n)$ for $n = 0, \cdots, 15$. Use function "stem"

(iii) $g1(t)$ and $x1[n]$ in same figure. Maybe the "hold" function is needed.

(b) Repeat part (a) for the cosine function of frequency 70 Hz. Apply the same sampling rate.

(c) Repeat part (a) for the cosine function of frequency 130 HZ. Apply the same sampling rate.

(d) Compare the plots (ii) of parts (a), (b) and (c). Explain what have happened. Draw a figure contain all the three signals and there samples.

2. Spectral analysis

(a) Again, consider the cosine function of frequency 30Hz

$$g1(t) = cos(60\pi t), \quad t \in [0, 0.16].$$

Now, besides the sample $x1[n] = cos(0.6\pi n)$ under sample rate 100Hz, take two other sampled signals with sample rate 200Hz and 400Hz, respectively. Then we obtain $y1[n] = cos(0.3\pi n)(n = 0, \cdots, 31)$ and $z1[n] = cos(0.15\pi n)(n = 0, \cdots, 63)$. Plot them use stem function. Apply "fft" function on the signals $x1[n]$, $y1[n]$ and $z1[n]$. Plot the results.

(b) Repeat part (a) for the cosine function of frequency 70 Hz. Take samples with sampling rate 100Hz, 200Hz, and 400Hz, respectively. Then apply "fft" and plot the results.

(c) Repeat part (a) for the cosine function of frequency 130 Hz. Take samles with sampling rate 100Hz, 200Hz, and 400Hz, respectively. Then apply "fft" and plot the results.

(d) Compare all the results in parts (a), (b), and (c). In each part, you have one continuous time signal and three sampled discrete-time signals with different sample rate. The spectra are also obtained. Please find one from these three samples so that it can be the best represention of the continuous-time signal. Explain the reason of your choise.

(e) For the three consine functions with frequencies 30Hz, 70Hz, and 130Hz, respectively, what are the Nyquist sampling rates? For each of them, use Nyquist rate to obtain a sampled discrete-time signal, and apply "fft" on it. Plot the samples and their resluts of "fft".

(f) Compare the samples you have chosen in part [d] and the Nyquist samples obtained in part (e). Explain what you can find out.

ECE 317

Digital Signal Processing I

Experiment #6

Design and Operation of an FIR Digital Band-Pass Filter

In this experiment you will design and operate an FIR digital band-pass filter. The frequency selective behavior of the digital filter will be determined. MATLAB will be used for this.

The transfer function $H(z)$ is given by

$$H(z) = \sum_{n=0}^{N} h(n)\, z^{-n}$$

where the coefficients $h(n)$, $n = 0, 1, \ldots, N$ are the unit pulse response of the FIR filter, and N is the order of the filter. The response to any input $x(n)$ is given by

$$y(n) = h(n) * x(n) = \sum_{i=0}^{N} h(i)\; x(n-i)$$

1) Assume that the sampling frequency is $f_s = 16\ \ KHz$. Use the Fourier series method to design a linear phase band-pass filter with a passband from 500 Hz to 1500 Hz. Write a MATLAB program to calculate the coefficients $h(n)$ for $N = 10$, and print a table of them. Provide the table and program listing in your report. In terms of multiplication by a constant, addition and delays, plot a block diagram of the filter.

2) Write a MATLAB program to obtain and plot the magnitude and phase frequency response of the filter for $0 \le f \le f_s$. Plot enough points to show details. For the magnitude frequency response, provide both linear and dB plots. Provide the program listing and plots in your report. Does the magnitude frequency response exhibit Gibbs' oscillation, and near what frequencies? What is the slope of the phase response?

3) Now sample $x(t) = \cos(\omega t)$ to obtain $x(n)$, where $\omega = 2\pi(1000)$ rad/sec. Write a MATLAB program to run the filter and plot the input and output. Run the filter long enough for the output to reach steady-state behavior. Provide the program listing and plots in your report. From the input and output plots, what is the phase difference between the output and input? Compare this phase difference to that predicted by the phase frequency response result at 1000 Hz.

4) The transfer function $H(z)$ can be factored to find the zeros. Write a MATLAB program that uses the built in function, roots, to find the zeros. In the $z-plane$, sketch a unit circle and the zeros of the filter. Provide a program listing and the zeros plot in your report. Since this is a band-pass filter, explain why the zeros are located as you found them.

5) Repeat part (2) for a significantly larger N of your choice. Describe how Gibbs' oscillation changes as N is increased.

6) Use the Hann window to obtain $h(n)$ for the filter order you used in part (5), and repeat part (2). Describe how the magnitude frequency response changed.

7) Run the filter of part (6), and apply an input obtained from sampling

$$x(t) = \cos(\omega_1 t) + \cos(\omega_2 t + \pi/2) + \cos(\omega_3 t + \pi)$$

where $f_1 = 250\ Hz$, $f_2 = 1000\ Hz$ and $f_3 = 2000\ Hz$. Provide a program listing and plots of the input and output in your report. Describe the performance of the filter.