Experiment 1
Application of DFT in Signal Detection and Interpolation

a) Application of DFT in Signal Detection

Objective: To detect specific harmonic frequencies from a noise corrupted discrete time signal

Procedure:

a) Download the noise corrupted signal xn.mat from the course website and load it into Matlab workspace.
b) Construct subsets of the noise corrupted signal consisting of the first 128, 256, 512, 1024 and 1792 samples. Name them as s1, s2, s3, s4, s5 in MATLAB.
c) Apply DFT to each subset of samples and display magnitude of resulting DFT sequences. Comment on the results. (What frequency vector will you choose for each subset? Hint: signal is real valued! Do peaks become prominent as length of the subset increase? If yes, why?) Helpful MATLAB functions: fft, abs, plot, linspace and the : operator.
d) Apply the DFT averaging method with K = 128 and L = 14.
    1. Using a for loop create L subsets of xn each of length K
    2. Compute the DFT of each of these vectors
    3. Add these and divide by L to compute the average.
    4. Plot the magnitude of the spectrum
    5. Identify the harmonics and report their values (magnitude and frequency)
    6. Identify the smallest value of L for which the peaks are clearly visible.
    7. Comment on the results: (In your comments explain which method of signal detection is better and why?)

b) Application of DFT in Signal Interpolation

Objective: To study different approximations of the original signal from the information contained in its samples taken at different instants.

Procedure:

a) Load the music signal handel and generate the signals from it as follows

```
N = 20000;
x = y(1:N);
x2 = x(1:2:N); % produces a sequence with even length
x3 = x(1:3:N); % produces a sequence with odd length
x4 = x(1:4:N); % produces another sequence with even length
```

b) Compute the DFTs of the above generated signals and store them as X2,X3,X4. Also compute the length of DFT sequences and store them as N2,N3 and N4 respectively.

c) Apply DFT based method for interpolation (x2 with K = 1, x3 with K = 2, x4 with K = 3)

1. Perform zero insertion

```
If Nd is odd
No = (Nd+1)/2;
Xo = [Xd(1:No); zeros(K*Nd,1); Xd((No+1):Nd)];
If Nd is even
Ne = Nd/2;
  Xe = [Xd(1:Ne); Xd(Ne+1)/2; zeros(K*Nd-1, 1); Xd(Ne+1)/2; Xd((Ne+2):Nd)];
```

2. Take inverse discrete fourier transform (Hint : Use matlab function ifft) which will give the time domain representation of the interpolated signal.
3. Rescale the amplitude of the time domain signal by multiplying with K+1
4. Take its first (K+1)(N-1) + 1 samples

d) Calculate the difference between original signal and interpolated signal using 2-norm (length of the two signals might be different, you might have to pad zeros at the end. So calculate the difference in length ‘p’ between sequence and original signal and pad p zeros to the sequence)

e) Take first 50 samples of the original and interpolated signal and plot them together on the same graph.

Eg:

```
plot(x(1:50),'k');%plot first 50 samples of x
hold on;
plot(xo(1:50),'--k');%plot first 50 samples of xo
legend('original','interpolated')
xlabel('time in seconds')
ylabel('signal amplitude')
title('Comparision between interpolated signal x with k = ‘’and original signal')
```

f) Observe and comment

How does the difference between the original signal and approximations vary with K?