

## ECE 320 Signals and Systems II

### Matlab Project 2: DFT Analysis of Sinusoidal Signals

**Issued:** Wednesday, November 15, 2000

**Due:** Monday, November 27, 2000

---

The purpose of this exercise is to explore the problem of analyzing the frequency content of a signal using the DFT. Specifically, it uses sinusoids as test signals to examine the effects of windowing and spectral sampling on frequency resolution. The project culminates in a short design problem that involves estimating the vibration frequencies of a piece of machinery from measured data.

#### I. Window Functions

There are a number of standard window functions that can be used in DFT analysis. Two of the more common windows are the rectangular (or boxcar) window and the Hamming window. The Matlab functions `boxcar` and `hamming` can be used to generate these windows. The argument to each function is the window length (number of points).

Investigate the characteristics of these two windows. First plot the window as a function of  $n$  for 3 different lengths: 16, 64, 256. Then plot the corresponding frequency response magnitudes for these 2 windows and 3 different lengths. Be sure to use a sufficient number of points in the FFT calculation; 1024 should be adequate to show the structure of the frequency response. Plot the log of the frequency response, i.e.,  $10 * \log_{10} |W(e^{j\omega})|$ . Note: Please use `subplot` to consolidate these plots for easy comparison.

Briefly describe your plots and discuss the relative merits of each window.

#### II. Simulating a Sampled Sinusoid

Sampled sinusoids will be used as test signals in the remainder of this project. Write a short Matlab function that returns samples of the signal  $\sin(2\pi f_0 t)$  over the period 0 to  $T_w$  seconds taken with a sample rate  $f_s$ . The inputs to the function should be  $f_0$  (frequency of sinusoid in Hz),  $T_w$  (the length of the time interval in seconds), and  $f_s$  (the sample rate in Hz). The function should return the samples along with a time vector for plotting against. You may use the template below for writing your function. Make sure that you test the function, e.g., use it to compute 1 second of a 10 Hz sinusoid sampled at 1000 Hz. Verify that your samples have a spacing of .001 seconds and that the sinusoid goes through 10 cycles in 1 second.

```
function [x,t]=samp_sine(f0,Tw,fs)
%% SAMP_SINE  Function to produce samples of a sinewave
%%           The user specifies the sinusoid frequency, the length of
%%           the time interval, and the sample frequency.
%%
%% Usage:
%%   [x,t]=samp_sine(f0,Tw,fs)
%%
%% Variables:
%%   f0 = frequency of the sinusoid (Hz)
%%   Tw = length of time interval (seconds)
%%   fs = sampling frequency (Hz)
%%
%%   x = samples of the sinusoidal signal
%%   t = vector of sample times (for plotting)
```

### III. DFT Analysis Design Tradeoffs

Note: Your writeup for this part should include plots, a brief description of the plots and answers to the questions posed in each section. Include your Matlab code as an appendix at the end.

#### A. Impact of DFT Length on Resolution

Consider the following test signal:

$$x(t) = \sin(2\pi(1000)t) + \sin(2\pi(1100)t) + \sin(2\pi(3000)t) \quad (1)$$

Generate an 0.0024 second sample of the signal  $x(t)$  using a sample rate of 10,000 kilohertz. (The resulting signal  $x[n]$  should contain 25 samples). Using both types of windows (rectangular and Hamming) compute windowed DFT's of three different lengths: 25, 128, 1024. In other words, multiply  $x[n]$  by  $w[n]$ , where  $w[n]$  is the appropriate 25-point window (rectangular or Hamming) and then use Matlab's `fft` to compute the discrete Fourier transform. Note that the 128-point and 1024-point transforms will be zero-padded. Plot the magnitude of your results as a function of continuous-time frequency. See the Hints section for help in calculating the vector of frequencies to plot against.

Answer the following questions:

- How do the rectangular window results compare to the Hamming window results?
- Are you able to see from the frequency response magnitude that the signal contains three separate sinusoids? If so, for which DFT length can you distinguish the sinusoids?
- What does increasing the DFT length accomplish?

#### B. Impact of Window (Data) Length on Resolution

Now consider keeping the DFT length fixed while using longer intervals of samples of  $x(t)$  (Equation 1). Compare results for the following data window lengths: 0.0024 seconds, 0.0049 seconds, and 0.099 seconds. (These should result in 25-point, 50-point, and 100-point signal lengths, respectively.) Compute 1024-point DFT's for each and plot the frequency response magnitudes. Again compare results for both the rectangular and Hamming windows.

Answer the following questions:

- How do the rectangular window results compare to the Hamming window results?
- Are you able to see from the frequency response magnitude that the signal contains three separate sinusoids? If so, for which DFT length can you distinguish the sinusoids?
- What does increasing the length of the data window accomplish?

#### C. Window Type: Sidelobe Levels and Leakage

In this part you will consider how a high-amplitude sinusoid might mask a low-amplitude sinusoid due to leakage effects (caused by the data window). Consider 2 sinusoidal signals,  $x_1$  and  $x_2$ , with different amplitudes:

$$x_1(t) = \sin(2\pi(50)t) \quad (2)$$

$$x_2(t) = 0.031 \sin(2\pi(77.5)t) \quad (3)$$

Sample these signals at a rate of 1,000 Hz over an interval of 0.127 seconds, to obtain the 128-point signals  $x_1[n]$  and  $x_2[n]$ . Let  $x[n] = x_1[n] + x_2[n]$ .

Use a DFT length of 1024-points to compute the windowed transforms of the total signal  $x[n]$  and the signal  $x_1[n]$  by itself. Compare the results for two different windows: 128-point rectangular vs. 128-point Hamming. Use plots of  $10 \log_{10} |X|$  and  $10 \log_{10} |X_1|$  to make your comparisons.

Answer the following questions:

- Describe what you observe in these plots.
- How do the sidelobe levels of the windows affect the results?
- Which window(s) allow you to detect the presence of both sinusoids? Why?

#### IV. Design Problem

You have recently been hired as a consultant for Elections-R-Us, Inc, a company that makes punchcard ballot counting machines. They are under pressure to ship a large quantity of new machines in the next week, and they need your help to fix a problem. These machines cycle through the ballots rapidly and count them extremely accurately. The problem is that occasionally one of the mechanical rollers becomes stuck, and this can cause severe damage to the machine if it goes undetected for a significant period of time. The company engineers have determined that it is possible to detect this stuck condition by looking at the output of a vibration sensor located on the case. During normal operation the machine vibrates at 60 Hz, but when a roller becomes stuck, an additional harmonic vibration at 59.25 Hz is also present. This error-mode vibration has an amplitude that is 20 dB less than the 60 Hz vibration.

The engineers inform you that the vibration sensor is sampled at 1,000 Hz. They plan to implement a DFT-based processor (as discussed above) that will display a spectrum on a screen so that the operator can tell when there is a jam and turn the machine off.

Your task is to determine how long a sample of the signal is required to determine if the error-mode vibration is present and to decide all the necessary parameter of the DFT-based processing. The goal is to be able to detect the error vibration in as short a time as possible.

Your writeup for this part be a short memo, written for a president of the company who needs the information but does not have much time to read it (because she is constantly being interview by CNN). It should be concise, but complete. Include a few plots to confirm your design.

Feel free to use other standard windows available in Matlab if you wish. Some of these are: `bartlett`, `hanning`, `blackman`, `kaiser`.

### Hints on plotting the frequency responses

Plotting the frequency response requires obtaining a vector of frequencies to plot against. As discussed in class, the discrete-time frequency vector may be defined as:

```
w=(2*pi/N)*[ 0:(N-1) ]'; % Set up vector of omegas: 0 to 2pi
```

In this case the samples range from 0 to  $2\pi$ . For plotting it is sometimes useful to have the samples go from  $-\pi$  to  $+\pi$ . The `fftshift` command can be used to rearrange the samples of the spectrum for plotting this way, e.g., `Xnew=fftshift(X)`. You will need a new vector of frequency samples to plot `Xnew` against. The following Matlab commands compute the new frequency vector and store back in the variable `w`:

```
mid=ceil(N/2)+1;  
w(mid:N)=w(mid:N)-2*pi; % move [pi,2pi) to [-pi,0)
```

Note: For this project, you need to convert your discrete-time frequency to continuous-time frequency (by the usual transformation) and then divide out  $2\pi$  to get a frequency vector in Hz.