Introduction to DSP
ECE-S352

Lab 2

The DFT and Spectrum Analysis

Introduction

The objective of this lab is to use computers to analyze the spectra of real-world signals. The Discrete Fourier Transform (DFT), which is the discrete-time version of the Fourier series, is the main tool we will be using. Once a signal is sampled, the spectrum can be analyzed by applying the DFT to the samples. The DFT can be rapidly computed using an algorithm known as the Fast Fourier Transform (FFT). Matlab has a built-in function, fft, that implements this algorithm. The FFT algorithm was discovered by Gauss and later rediscovered and popularized by Cooley and Tukey in the late 1960's. The ability to perform spectrum analysis on a computer revolutionized science and engineering, and the FFT is one of the most important tools in signal processing and computational science. In fact, you have all encountered the FFT before; every time you listen to an MP3 file or view a JPEG image you are using an FFT-based algorithm.

Overview

The DFT is the Fourier series representation for discrete-time signals. Let x[n], n = 0, . . . , N – 1, denote a signal of duration N samples. The DFT coefficients of x[n] are computed by

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j2\pi \frac{k}{N} n}, \quad k = 0, \ldots, N-1$$

The signal x[n] can then be represented or synthesized by the following weighted sum of complex exponential functions

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k]\, e^{j2\pi \frac{k}{N} n}$$

The DFT coefficient X[k] appropriately weights the corresponding complex exponential function,

$$e^{j2\pi\frac{k}{N}n} = \cos\left(2\pi\frac{k}{N}n\right) + j\sin\left(2\pi\frac{k}{N}n\right)$$

The digital frequency of this function is f' = k/N cycles/sample. To relate the digital frequency to the corresponding frequency of the original sampled signal, simply covert from cycles per sample to samples per second. If x[n] resulted from sampling a continuous-time signal x(t) at a rate of fs samples/second, then the digital frequency f corresponds to

f' (cycles/sample) × fs (samples/sec) = f (cycles/second or Hz)

MATLAB has a built in function to calculate the DFT. Try searching the MATLAB documentation for fft.

Lab

Consider the continuous-time sinusoid

$$x(t) \;=\; A\cos(2\pi f_0 t + \phi)$$

Sampling this sinusoid at a rate of fs (i.e., one sample every Ts = 1/fs seconds) results in the discrete-time signal

$$x[n] \;=\; A\cos(2\pi f_0 T_s n + \phi)$$

If we sample the sinusoid over the time period 0 ≤ t ≤ T seconds, then we will acquire N = T × fs samples. The following Matlab script generates and plots a sampled signal:

```
A = 10;
f0 = 100;
phi = pi/4;
fs = 400;
Ts = 1/fs;
T = 1;
N = T*fs;
tn = (0:N-1)/fs;
x = A*cos(2*pi*f0*Ts*(0:N-1)+phi);
figure(1)
plot(tn,x)
```

What is the frequency of the sinusoid that was sampled? Was it adequately sampled, at or above the Nyquist rate?
How many samples were acquired per cycle of the sinusoid?

The DFT of this signal can be computed by simply implementing the mathematical formula above. For example, this function computes the DFT coefficients:

```
function X = mydft(x)
N = length(x);
for k=1:N
   X(k) = 0;
   for n=1:N
      X(k) = X(k)+x(n)*exp(-j*2*pi*(k-1)/N*(n-1));
   end
end
```

The output X is a vector of the DFT coefficients.

Create another function, **my_idft**, that takes X and synthesizes according to

$$x[n] \;=\; \tfrac{1}{N}\sum_{k=0}^{N-1} X[k]\, e^{j2\pi\frac{k}{N}n}.$$

Compare then your functions to the corresponding built-in Matlab functions **fft** and **ifft .** Does your functions generates the same results? Compare the speed of your functions to the built-in Matlab functions using this script:

```
tic;
myX = mydft(x);
mytime = toc;
tic;
matX = fft(x);
mattime = toc;
```

How does **mytime** compare to **mattime**? Is the **FFT** fast?

Discuss the results of the above exercises in your lab report.

Lab Exercise

Mathematically explain why the spectrum of the 150Hz sinusoid in the warm-up did not consist of two perfect peaks. This effect is often called “spectral leakage”.