EE432: Digital Signal Processing Fall 2008

Project 6: Discrete Fourier Transform

Assigned: Tues 10/7/08 Due: Wed 10/15/08

This project is intended to give you more experience in calculating the DFT and analyzing its results.

I. Calculating the DFT

1. Write a MATLAB function to compute the DFT of an input discrete signal. You can base your code on Example 3.12 (page 59) from the Etter/Ives text. That code is a MATLAB script file that computes a DFT and plots the result. This function should **NOT** plot the result, merely compute the DFT.

   Usage: *xdft = DFT*(*x*)*;*

   Here, *x* is the input signal, and *xdft* is the DFT of *x*.

2. Test your DFT function by computing the DFT of the 4-point signal that was covered in the class example: $[x_n] = \delta[n] + 0.5\delta[n-1] + 2\delta[n-2] - \delta[n-3]$. Do you get the same result provided in class? If not, troubleshoot your code.

3. Test your *DFT* function a test signal using the code below…your answer should match my answer:

```
>> n=(0:7)';       % create a column vector
>> x=n.*(-1).^n    % create signal x…alternate the sign of each element of x
>> xdft=DFT(x);    % take DFT of x
>> [x xdft]
```

   Should result in:

```
ans =

     0            -4.0000
-1.0000           -4.0000 - 1.6569i
 2.0000           -4.0000 - 4.0000i
-3.0000           -4.0000 - 9.6569i
 4.0000           28.0000
-5.0000           -4.0000 + 9.6569i
 6.0000           -4.0000 + 4.0000i
-7.0000           -4.0000 + 1.6569i
```

4. Write a MATLAB function that will take the DFT of a signal as its input (that is, the DFT is already computed), and plot the magnitude and phase of the DFT in a single figure window (a 2x1 subplot), with a frequency axes that runs from 0 up to and including $f_s/2$ Hz (this means you only plot the 1st N/2 + 1 values of the magnitude and phase). Here, f_s is sample frequency. Note that I'm asking these plots use frequency in Hz on the x-axis, vice radians/sec.

   Usage: *plotDFTt*(*xdft,T*);

   Here, *xdft* is the input DFT to plot (complex valued), and *T* is the sample period (sample time).

5. Write a MATLAB function to compute the inverse DFT.

   Usage: *y= IDFT*(*xdft*)*;*

Here, *xdft* is the input DFT and *y* is the corresponding time signal (the inverse DFT).

6. Test your *IDFT* function using the *xdft* values used in the in-class examples you used in questions 2 and 3. Do you get back the original [x_n]?

II. Analyzing the DFT Result

1. Recall that a DTMF tone is a combination of two specific sinusoids. Create a 1 sec long DTMF tone for the character "4" using a sample frequency of 8000 Hz. The table of DTMF functions for each digit on a touch-tone phone is shown below. Compute the DFT for this signal using your *DFT* function. Plot the DFT of this signal using your *plotDFT* function.

		High-Group Frequencies			
		1209Hz	1336Hz	1477Hz	1633Hz
Low-Group Frequencies	697Hz	1	ABC 2	DEF 3	A
	770Hz	GHI 4	JKL 5	MNO 6	B
	852Hz	PRS 7	TUV 8	WXY 9	C
	941Hz	*	OPER 0	#	D

2. From this plot of the DFT, using the table of DTMF tones shown above, prove that the signal is in fact the character "4" by indicating on the plot (with text boxes/arrows) the two tones present. Turn in your plot.

3. Download the file from the course website called "unknownDTMF.wav". Use *wavread* to load this file into MATLAB and determine which DTMF character it is. Do NOT assume the signal has been sampled at 8000 Hz; if you do, you are wrong and your answer will NOT make sense.

For this lab's write up, turn in your code and the required plots, and answer any questions asked.