J. Reilly

Last revision Oct 5, 2015

CoE 4TL4 Lab 1
Discrete-time Fourier Analysis and Sampling

This is fundamentally a loosely–formatted project–based lab. Once you have your key access, you can go into the lab room (ITB 234) any time at your convenience to work on it. The TAs will be available during the regularly scheduled lab periods to help out and answer questions, but you are not restricted to attend only during regular lab periods.

The lab will begin **Monday Oct 5, 2015.** Your report is due 3 weeks hence on **Mon Oct 26**. Work in pairs and submit one report per pair.

Sometime during the lab interval, please complete the safety quiz form and return to the TA.

1 Overview

This lab involves various processing operations on a speech file using Matlab. The speech is typical telephone quality– the sampling frequency $f_s = 8$KHz, with 8 bits per sample, corresponding to a 64 Kbits/sec bit stream. You can play the speech through the computer's sound system by issuing the matlab command

soundsc(speech) [1]

where "speech" is the variable containing the speech samples.

Two matlab .mat speech files SPM1.mat (male speaker), and SPF1.mat (female speaker) are available for downloading from the course website (www.ece.mcmaster.ca/faculty/reilly/coe4tl4). Download the file onto your working directory, then issue the command load fname within matlab. The variable "speech" will then contain the speech samples.

There are also two other files on the website that relate to this lab. The first file is "h.mat", which contains an impulse response for a low-pass filter function. The second is "coeffs.mat", which contains the difference equation coefficients to realize another form of low-pass filter function.

This lab is open–ended, in that no one will tell you exactly what to do in the lab, nor exactly what is expected for a write-up. These details will be left up to you. The hope is that by experimenting with these various

[1] Matlab commands are indicated in blue.

signals you will learn a lot, and gain skills in the field of digital signal processing. Guidelines and suggestions for experiments are given below.

There are a few additional matlab commands that may be useful for this lab. Check out the commands conv, filter and freqz using the matlab "help" facility.

Use the matlab fft command to transform from the time domain to the frequency domain, and ifft to transform the other way. We have seen in class that if we discretize in the time domain, the corresponding spectrum becomes periodic. We must sample fast enough to avoid aliasing error. (It is interesting to note that according to the Fourier principle of duality, discreteness in *either* domain causes periodicity in the other). The discrete-time Fourier transform (DTFT) that we developed in class is discrete in time, but continuous (and periodic) in frequency.

In order to process frequency domain representations on a computer, they must be in *discrete* format. Therefore the DTFT is suitable for processing in the time domain, but not so for frequency domain representations. We therefore consider the Discrete Fourier transform (DFT), which is discrete in BOTH domains. The DFT is obtained from the DTFT by discretization in the frequency domain. Thus, the time domain also becomes periodic; i.e., the DFT produces functions which are discrete and periodic in BOTH domains. It is interesting that in order to avoid aliasing in the *time domain*, we must discretize the frequency domain sufficiently quickly (i.e., densely). This is the dual of the situation where we discretize in time. With the DFT operation, both the time and frequency sequences are represented by only *one period* of the respective (infinitely long) periodic signal. We will be covering concepts relating to the DFT in greater detail later in the course.

The matlab fft function is nothing but a fast implementation of the DFT. There is only one further point with regard to the frequency domain representation that you will need to use this function effectively. That is, the DFT operation requires that the single period of the frequency domain sequence extend from 0 radians to 2π radians. That means the positive frequency values are associated with indexes from 1 to $N/2$, where N is the number of points in the sequence, and negative frequency values are associated with indexes from $N/2+1, \ldots, N$. That is, points associated with positive frequency values are on the left side of the sequence, whereas negative values are on the right. This is in contrast to conventional graphical representations, where the negative frequency values are on the *left side* of the positive values.

2 Procedure

Here are a few suggestions. You are welcome to try out any additional experiments you can think of.

1. Filter the speech signal using both the available impulse response and the difference equation coefficients, using the matlab commands conv and filter, respectively. Play both the original speech file and the filtered versions. Evaluate and plot the frequency responses corresponding to both filter structures. Explain why the filtered speech sounds the way it does. Refer to the filtered speech variable as sf.

   Explain the relationship between the magnitude and phase responses of the filters versus the spectra of the input and output speech signals.

2. Experiment with aliasing error. You can induce aliasing error into the speech signal using a *downsampling* process. "Downsampling" means that you lower the effective sampling rate of the signal. You can downsample by a factor of 2 in matlab using the following command sequence:

```
ln = length(speech);
dsspeech = zeros(l,ln);
dsspeech(1:2:ln)= speech(1:2:ln);
```

Listen to the resulting downsampled speech. Explain the relationship between the spectra of the downsampled speech and the original speech.

3. We can experiment with downsampling and aliasing error in a different way, as follows. Try downsampling the filtered speech sf that you obtained in step 1 above. Call this speech file sfd. Play the resulting speech file and comment on how it sounds. Is it different from the original downsampled speech?

   Recall that the periodic discrete-time spectrum can be obtained from the continuous-time spectrum by convolving it with a periodic impulse train of period f_s Hz, so that the discrete-time spectrum becomes periodic with period f_s Hz. However, when downsampling by a factor of 2, the period of the impulse train effectively becomes $f_s/2$. Use this fact to explain the shape of the spectrum of the variable sfd.

4. Process the signal sfd so that it sounds the same as the original filtered speech sf you obtained from step 1.

5. Here we experiment with **group delay**. Download the file *impulsegd.mat* from the website. Evaluate the magnitude and phase responses (use the Matlab unwrap function). You will see a very rapid transition in the phase response around 2000 Hz. Filter the original speech file with this impulse response, and observe and explain the effect.

6. Specify some frequency response of your choice and implement it. The matlab function fdatool will help considerably in this respect. To export your filter coefficients, select "export" from the file menu and export to workspace. Try designing a lowpass filter, and then converting it to a bandpass filter, using procedures discussed in class.

3 Hints on the writeup

The objective in technical writing is to express your ideas as clearly as possible, using as many words as necessary, but no more. Thus, *clarity* and *brevity* are the key concepts.

Present the results of your experimental work so that someone who is knowledgable in the field, but who has not actually done the lab work, can understand and repeat what you have done. If you wish to make a point that adds to the understanding of the material, include it, but don't embellish.