ECE130B January 23, 2017

Home Work 3

Due date: January 31, 2017 by 8p.m. at the home work box.

Reading assignment. Chapter 3 of the text book.

Exercise 1. On the class web-site you will find three audio files labelled `guitar1.wav`, `guitar2.wav` and `guitar3.wav`. Each of the files is an audio recording at 8000Hz of three different notes of a guitar. For each of these audio files do the following tasks:

i. Take 1000 contiguous samples from the file and pretend that it is a discrete signal of period 1000. Use your program from Home work 2 Exercise 8 to find its 1000 Fourier coefficients. In this task it better to skip the first 1000 samples or so in the recording. Look at the recording (say in Audacity). Can you see why it is a good idea to skip the starting part of the signal?

ii. Plot the magnitudes of the 1000 Fourier coefficients and submit it.

iii. Find the discrete frequency (a number between 0 [included] and 1000 [excluded]) of the largest peak in your graph of the magnitudes of the Fourier coefficients. To what continuous frequency does it correspond? Can you identify the note that was being played (for example the A above middle C is at 440Hz)? (You can look up the standard frequencies for the other notes in Western music.)

Exercise 2. The definitions of square wave, triangular wave and parabolic wave are given at the beginning of home work 2. Pick an identical period T for each of the 3 wave-forms. Pick an identical number of samples (should be fairly large, at least in the hundreds) in one single period of each of the 3 wave-forms. Compute the corresponding discrete Fourier series coefficients for each of 3 wave-forms using your program from home work 2 Exercise 8.

i. On a *single* graph plot the **magnitude** of the Fourier series coefficients of all 3 wave-forms. What differences do you observe? Can you tell the 3 wave-forms apart just by looking at the magnitude of their Fourier coefficients?

ii. On a *single* graph plot the **phase** of the Fourier series coefficients of all 3 wave-forms. What differences do you observe? Can you tell the 3 wave-forms apart just by looking at the phase of their Fourier coefficients?

Exercise 3. Let z denote a complex number. Establish the following sums:

a)

$$\sum_{n=1}^{N} n = \frac{N(N+1)}{2}.$$

Hint: Sum the equality $2n = n(n+1) - (n-1)n$ from 1 to N, and notice that almost all the terms in the right-hand side sum cancel each other out. (It helps to write out the terms explicitly for a small value of N, say $N = 5$.) This is called a *telescoping* sum. (You can also do it by induction.)

b)

$$\sum_{n=1}^{N} n^2 = \frac{N(N+1)(2N+1)}{6}.$$

Hint: $6n^2 = n(n+1)(2n+1) - (n-1)n(2n-1)$. When summed this will give rise to a telescoping sum on the right-hand side. Or, use induction.

c)

$$\sum_{n=0}^{N-1} z^n = \begin{cases} \frac{1-z^N}{1-z}, & z \neq 1, \\ N, & z = 1. \end{cases}$$

d) If $|z|<1$,

$$\sum_{n=0}^{\infty} z^n = \frac{1}{1-z}.$$

e) Summation by parts:

$$\sum_{n=1}^{N} a_{n+1}(b_{n+1}-b_n) = a_{N+1}b_{N+1} - a_1 b_1 - \sum_{n=1}^{N} b_n(a_{n+1}-a_n).$$

Hint: Expand both sides and compare terms.

f)

$$\sum_{n=1}^{N} n\, z^n = \begin{cases} \frac{z(1-z^N)}{(1-z)^2} - \frac{N z^{N+1}}{1-z}, & z \neq 1, \\ \frac{N(N+1)}{2}, & z = 1. \end{cases}$$

Hint: Use summation by parts. Or, differentiate the formula in part c) with respect to z.

g) If $|z|<1$,

$$\sum_{n=0}^{\infty} n\, z^n = \frac{z}{(1-z)^2}.$$

h) If $|z|<1$,

$$\sum_{n=N+1}^{\infty} n\, z^n = \frac{z^{N+1}(N+1-Nz)}{(1-z)^2}.$$

Hint: Subtract the previous two sums.

Put the above formulas in your cheat sheet for the mid-term and final exams.

Ideal band-pass filter. A signal $h[n]$ of length N with Fourier series coefficients

$$a_k = \begin{cases} 1, & k_1 \leqslant k \leqslant k_2, \\ 0, & 0 \leqslant k < k_1, \\ 0, & k_2 < k < N, \end{cases}$$

is called an ideal band-pass filter with $[k_1, k_2]$ as the associated frequency band.

Exercise 4. Find explicit mathematical expressions for the ideal band-pass filter $h[n]$. Next, take $N = 160$, $k_1 = 20$, and $k_2 = 40$, plot the real and imaginary parts of the corresponding $h[n]$, and submit the plots along with the rest of your answer.

FFT. The **F**ast **F**ourier **T**ransform is an algorithm for computing the discrete Fourier series coefficients rapidly. We briefly describe one version when the length N of the signal is an exact power of 2; that is $N = 2^M$ for some integer M. Let $x[n]$ be the given signal for $n = 0, \ldots, N-1$. The k-th Fourier series coefficient of this signal is given by the formula:

$$a_k = \frac{1}{2^M} \sum_{n=0}^{2^M-1} x[n] \exp\left(-\frac{2\pi j\, k\, n}{2^M}\right).$$

We split this sum into two parts. The first part will contain the terms for which n is even and the second will contain the terms for which n is odd:

$$a_k = \frac{1}{2^M} \sum_{r=0}^{2^{M-1}-1} x[2r] \exp\left(-\frac{2\pi\, j\, k\, 2r}{2^M}\right) + \frac{1}{2^M} \sum_{r=0}^{2^{M-1}-1} x[2r+1] \exp\left(-\frac{2\pi\, j\, k\, (2r+1)}{2^M}\right). \quad (1)$$

Exercise 5. Show that the above splitting is correct. In particular check that the upper bounds on the two sums are correct.

These two new sums can be simplified a bit to obtain:

$$\begin{aligned} a_k \;=\; & \frac{1}{2}\frac{1}{2^{M-1}}\sum_{r=0}^{2^{M-1}-1} x[2r]\exp\left(-\frac{2\pi j k r}{2^{M-1}}\right)+ \\ & \frac{1}{2}\exp\left(-\frac{2\pi j k}{2^M}\right)\frac{1}{2^{M-1}}\sum_{r=0}^{2^{M-1}-1} x[2r+1]\exp\left(-\frac{2\pi j k r}{2^{M-1}}\right). \end{aligned} \tag{2}$$

Exercise 6. Show that indeed Equation (1) can be simplified into the above form.

We now define two signals $x_e[n]$ and $x_o[n]$ of length $N/2 = 2^{M-1}$, which contain just the even and odd samples of $x[n]$ respectively. That is $x_e[n] = x[2n]$ and $x_o[n] = x[2n+1]$. Now, let b_k denote the discrete Fourier series coefficients of $x_e[n]$, and let c_k denote the Fourier series coefficients of $x_o[n]$.

Exercise 7. Show that, if $0 \leqslant k < 2^{M-1}$, then Equation (2) can be written in terms of b_k and c_k as:

$$a_k = \frac{1}{2}b_k + \frac{1}{2}\exp\left(-\frac{2\pi j k}{2^M}\right)c_k, \qquad 0 \leqslant k < N/2.$$

Exercise 8. Given b_k and c_k, how many arithmetic operations do you require to compute the first $N/2 = 2^{M-1}$ Fourier series coefficients a_k?

We still need to find a_k when $N/2 \leqslant k < N$.

Exercise 9. When $N/2 \leqslant k < N$, let $k = s + N/2 = s + 2^{M-1}$. Substitute this into Equation (2) and show that:

$$\begin{aligned} a_{s+N/2} \;=\; & \frac{1}{2}\frac{1}{2^{M-1}}\sum_{r=0}^{2^{M-1}-1} x[2r]\exp\left(-\frac{2\pi j s r}{2^{M-1}}\right)- \\ & \frac{1}{2}\exp\left(-\frac{2\pi j s}{2^M}\right)\frac{1}{2^{M-1}}\sum_{r=0}^{2^{M-1}-1} x[2r+1]\exp\left(-\frac{2\pi j s r}{2^{M-1}}\right). \end{aligned}$$

Note, the minus sign between the two terms. Conclude (show) that

$$a_{s+N/2} = \frac{1}{2}b_s - \frac{1}{2}\exp\left(-\frac{2\pi j s}{2^M}\right)c_s, \qquad 0 \leqslant s < N/2.$$

Exercise 10. Given b_k and c_k how many arithmetic operations are required to find the last $N/2$ Fourier series coefficients a_k?

Exercise 11. Give a recursive implementation of the FFT in your favorite programming language, structured according to the following pseudo-code:

```
FFT (x, N) -> a = {
    if N > 1 then {
      x_e = even_samples (x, N)
      x_o = odd_samples (x, N)
      b = FFT (x_e, N/2) // recursive call
      c = FFT (x_o, N/2) // recursive call
      for k = 0 to (N/2)-1 do
        a[k] = ? * b[k] + ? * c[k]
      for k = N/2 to N-1 do
        a[k] = ? * b[k-(N/2)] - ? * c[k-(N/2)]
      return (a)
    } else { // N = 1 -- base case
      return (?)
    }
}
```

In the above code I am assuming that array (e.g., `x`, `a`, `x_e`, `x_o`, `b` and `c`) indices start at 0; that may not be the case for your language (e.g. Matlab). Make suitable adjustments in the code. You also need to implement the two functions `even_samples` and `odd_samples`. We are also assuming that N is an exact power of 2. Submit a printed copy of your working code.

Exercise 12. Check your implementation of FFT by computing the discrete Fourier series coefficents of a **non-zero** signal of length 32 of your choice in two ways. First, compute it using your FFT code. Second, compute it using your code from `Home work 2, Exercise 8`. Report the maximum error in the coefficients computed by the two codes. It will *not* be zero! Why? Is *your* error small enough that you can be confident that there is no bug in your code? Why?

Exercise 13. For signals of length 2^{12}, 2^{13} and 2^{14} find the time taken by *your* FFT code. Compare it with the timings reported in `Home work 2, Exercise 11`. How much faster (if at all) was your FFT code?