2020 ASAS Homework 1: Basic Audio Signal Manipulations and Fourier Transform

Prof. Yi-Wen Liu

March 5, 2020

1.

(a) In MATLAB, create an array of length N=16000, and fill the array with the following values: $y[n] = 0.5g[n-8000]\sin(2\pi f_0 nT)$, where T = 1/16000 sec, n = 0:15999, f_0 (Hz) is a frequency of your choice, and

$$g[n] = \begin{cases} \cos^2\left(\frac{\pi}{2}\frac{n}{M}\right), -M \le n \le M \\ 0, \text{elsewhere} \end{cases}$$

So $g[n-8000]$ should be a bell-shaped function centered around time n=8000 and its width is $2M$ samples.

(b) Set the sampling rate $f_s = 16000$, M=8000, and play y[n]. By appropriately choosing f_0, you should hear a pip tone. Listen to it and record the range of f_0 that you can hear something. #discussion

(c) Now, set $f_0 = 1000$ Hz. Shorten M and record the range of M at which you can still hear a definite pitch.

(d) Change $\sin(2\pi f_0 nT)$ to $\cos(2\pi f_0 nT)$. Can you hear any difference? This is essentially asking the question of whether the *phase* of an audio signal is important. #discussion

(e) Change the definition of $g[n]$ to $g[n] = 1, -M \le n \le M$, and $g[n] = 0$ elsewhere. Can you hear a difference in $y[n]$? #discussion

2. The discrete Fourier transform is defined as follows,

$$Y(k) = \sum_{n=0}^{N-1} y[n]e^{-jk(2\pi)n/N},$$

where $k = 0,1,2, \ldots, N-1$.

(a) Create an array of length N=16000 and calculate $Y(k)$ for all k. Your result should be identical as MATLAB's fast Fourier transform (FFT) function: Y = `fft(y)`; but here the purpose is to verify that indeed this is the case. If so, you can feel free to use FFT from now on.

(b) Plot $|Y(k)|$ against frequency $f = \frac{k}{N}f_s$. The result is called the magnitude spectrum of the signal. Notes: Sometimes, we prefer to view $|Y(k)|$ in dB by taking $20\log_{10}|Y(k)|$.

(c) Adjusting M and f_0 and observe how changes occur in the magnitude spectrum.

3. The Short time Fourier transform $Y(n, \omega)$ is defined as

$$Y(n, \omega) = \sum_{m=-\infty}^{\infty} y[m] w[m-n] \, e^{-j\omega n},$$

where $w[n]$ is a (usually positively-valued) *window function*.

(a) Choose an appropriate window function w[n] and calculate $Y(n, \omega)$ for $\omega \in [0, \pi]$. Verify whether your implementation produces the same result as MATLAB's `spectrogram()` function.

(b) In the audio field, it is a common practice to view the magnitude of STFT $|Y(n, \omega)|$. Read some sound files into MATLAB by `audioread()`, and view the spectrogram to see if you can make sense of it. Are there any patterns that correspond to audible properties of the sound?

(c) Is it possible to reconstruct the signal y[n] from the magnitude of STFT, $|Y(n, \omega)|$? Why or why not? Note that by taking the absolute value of complex numbers we have ignored the phase information. #discussion

Preview of the next homework: We will *filter* audio signals in the sense that each frequency receives a different gain (and delay).