ECE 4250
Assignment 2

Due February 28th 11:59 pm

General Instructions

The assignment submissions are via Canvas. There are two parts: a problem set and a coding (programming) component. Your answers to the problems can be written up any way you want (e.g., using Latex or scanning your handwritten answers). As long as we can read it, we will grade it. For the programming part, you will code in Python 3, using Jupyter Notebook. Make sure that all cell outputs are clearly visible before saving the notebook. Thus, we recommend that you re-run your code after you are done editing. You should then print this file into a pdf. For the coding component, please submit both the ".pdf" and ".ipynb" files. Please zip up all your work into a single file and submit that.

Any acknowledgments of collaboration, answers to questions, comments to code or other notes should all be included in your problem set answer sheet and/or Jupyter Notebook. General guidelines regarding assignments apply. In other words, you can collaborate at the ideation/brainstorming stage, but whatever you submit should represent your own individual work. You are not allowed to copy/paste others' code/answers or work on same code with someone else. You are not allowed to use functions outside of the Python Standard Library, unless specified otherwise or you have received permission to do so. Your code will be graded on readability, executability, and accuracy. You are strongly advised to use detailed commenting that explains the algorithmic steps in your code. Explain every function and loop. Use piazza to ask questions. Take advantage of TA and Instructor Office Hours to seek help.

Problem Set

Question 1. Derive the analytic form for the *causal* signal that has the following z-transform:

$$X(z) = \frac{2 - \frac{1}{2}z^{-1} + \frac{3}{5}z^{-2}}{1 - \frac{1}{2}z^{-1} + \frac{3}{5}z^{-2}}$$

Note that you are supposed to get an expression that is as compact as possible.

Question 2. A linear interpolator constructs a continuous signal by connecting successive samples. For example:

$$\hat{x}(t) = x(nT_s - T_s) + \frac{x(nT_s) - x(nT_s - T_s)}{T_s}(t - nT_s), nT_s \leq t \leq (n+1)T_s$$

This interpolator has a delay of T_s seconds. I.e. $\hat{x}(nT_s) = x(nT_s - T_s)$.

1. Derive the impulse response of the linear filter that yields the interpolation function described above when applied to an impulse train $\sum_{k=-\infty}^{\infty} x(kT_s)\delta(t - kT_s)$.
2. Is this a causal filter?
3. Derive the corresponding frequency response. How does this correspond to the ideal interpolator's frequency response?

Question 3. Consider a finite duration sequence $x(n)$, with non-zero values for $0 \leq n \leq 7$. Its z-transform is $X(z)$. Consider following points on the z-plane:

$$z_k = 0.8e^{j[2\pi k/8 + \pi/8]}, 0 \leq k \leq 7$$

1. Sketch these points in the complex z-plane.
2. Determine a sequence $s(n)$ such that its DFT will give the samples of $X(z)$ at these points. Note we are looking for an expression for $s(n)$ in terms of $x(n)$.

Question 4. Consider a band-limited analog signal $x_a(t)$, where $X_a(\omega) = 0$, for all $|\omega| > 2\pi B$. Derive the Nyquist rate for:

1. $\frac{dx_a(t)}{dt}$
2. $x_a(t) \cos 6\pi Bt$
3. $x_a(3t)$

Question 5. If $X(k)$ is the N-point DFT of $x(n)$, what is the N-point DFT of:

$$x_c(n) = x(n) \cos \frac{2\pi k_o n}{N}, 0 \leq n \leq N - 1$$

in terms of $X(k)$?

Question 6. Consider $x(n)$ such that $x(-2) = 1, x(-1) = 2, x(0) = 3, x(1) = 2, x(2) = 1, x(3) = 0$ and all else is 0. Determine its Fourier transform $X(\omega)$. Now, compute the 6-point DFT $Y(k)$ of $y(n) = [3, 2, 1, 0, 1, 2]$. How is $X(\omega)$ and $Y(k)$ related?

Programming Questions

1 Convolutions

Define the following signals

$$x = \left[3, 4, 1, 2, 5, 6, 7, 8, 2, 4\right]$$
$$h = \left[\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right]$$

Calculate the convolution between these two signals, $x * h$, via the frequency domain. Confirm your answer by computing the convolution directly in the time-domain (you did this in assignment 1). You can use numpy's `fft` function. Note that `fft` is a fast algorithm that computes the Discrete Fourier Transform and stands for Fast Fourier Transform.
Hint: remember to pad the signals so that they are the same size, so their fft's can be multiplied.

2 Fourier Transforms

a) Load the "Corcovado.wav" file from the previous assignment. Extract the signal from 2:02 - 2:05 second time window. Recall the frequency sampling rate is included in the file, and this needs to be read also. Take the `fft` of the extracted sound clip.

b) Where is the DC component (zero frequency component)? Now do an `fftshift`, where does the DC component move to?

c) What is the frequency of the saxophone, the dominant instrument in this clip? Show how you get from the `fft` to a frequency. Generate a sound wave of that frequency and compare it to the original sound clip and play both to listen to if there is a good match to check your answer.

Note: These audio files are stereo. For simplicity, just use the left channel - i.e, channel #1, and not #0. Also, feel free to use the pyaudio python package for generating sounds.

3 Implement the FFT

(a) Consider the following signal with $f = 1$ Hz:

$$x(t) = \cos(2\pi f t).$$

Sample $x(t)$ at 128 equally spaced time-points between 0 and 1 seconds, and store this signal as $x(n)$. Is this sampling rate above the Nyquist rate?

(b) Using the original time-domain definition of DFT, write a function that takes the discrete signal $x(n)$ as input, together with a positive integer N (which can default to the length of x) and computes $X(k)$, the N-point DFT of $x(n)$.

(c) Now, you will implement your own FFT function using the Radix-2 (Cooley-Tukey) algorithm (as described below).

i Write a function called *separate* that separates the even and odd indices of an input signal. The inputs are a discrete signal $x(n)$ and a positive integer N, where N is a factor of 2. The outputs are $x_1(n) = x(2n)$ and $x_2(n) = x(2n+1)$, for $n = 0, \ldots, N/2 - 1$.

ii Write a recursive function (*MyFFT*) that takes an input signal $x(n)$ and a positive integer N and outputs the N-point DFT.

If $N < 2$, the output is simply equal to the input.

Otherwise, split the input signal using the *separate* function (into x_1 and x_2) and recursively call $MyFFT(x_i, N/2)$, for $i = 1, 2$. We can denote the output FFTs as X_1 and X_2. The next step is to combine the separate FFTs using the butterfly operation:

$$\begin{aligned} X(k) &= X_1(k) + W_N^k X_2(k) \\ X(k + N/2) &= X_1(k) - W_N^k X_2(k)\,, \end{aligned}$$

for $k = 0, \ldots, N/2 - 1$ and where $W_n^k = e^{-j2\pi k/N}$ is the phase factor (aka "twiddle factor").

For more information refer to: `https://en.wikipedia.org/wiki/Cooley%E2%80%93Tukey_FFT_algorithm`

(d) Using *MyFFT* compute the DFT of $x(n)$ from (a). Compare this result to your previous DFT result from (b).

(e) Read "clip.wav" file and take the first 8192 points from the first channel. Calculate the DFT X for the clipped sound using the DFT and FFT functions you wrote and compare their run-times.