MIDDLE EAST TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

CENG 384

Spring 2023

Homework 4

Regulations:

- **Grouping:** You are strongly encouraged to work in pairs.
- **Submission:** You need to submit a pdf file named 'hw4.pdf' to the odtuclass page of the course. You need to use the given template 'hw4.tex' to generate your pdf files. Otherwise you will receive zero.
- **Deadline:** 23:55, 6 June, 2023 (Tuesday).
- **Late Submission:** Not allowed.

1. (20 pts) Consider the following LTI system defined by the frequency response below:

$$H(j\omega) = \frac{j\omega - 1}{j\omega + 1}.$$

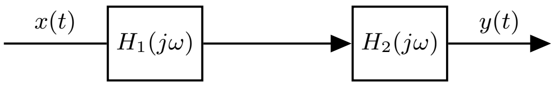
   (a) (5 pts) Find the differential equation which represents this system.
   (b) (5 pts) Find the impulse response of this system.
   (c) (5 pts) Find and plot the output $y(t)$ for the input $x(t) = e^{-2t}u(t)$.
   (d) (5 pts) Draw a block diagram of this system using integrators, adders and multiplication factors.

2. (15 pts) Consider the LTI system described by the following difference equation:

$$y[n+1] - \frac{1}{2}y[n] = x[n+1].$$

   (a) (5 pts) Find the frequency response $H(e^{j\omega})$.
   (b) (5 pts) Find the impulse response $h[n]$.
   (c) (5 pts) Find the output $y[n]$ for the input $x[n] = (\frac{3}{4})^n u[n]$.

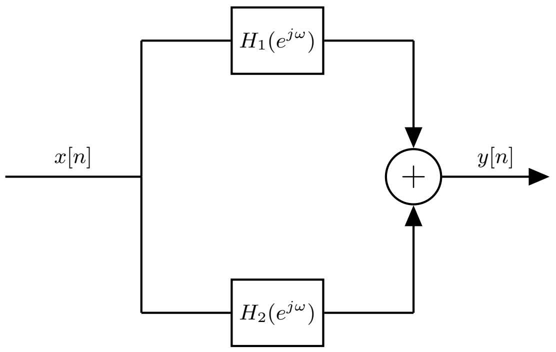
3. (15 pts) Consider the following continuous-time LTI system:



   where $H_1(j\omega) = \frac{1}{j\omega+1}$ and $H_2(j\omega) = \frac{1}{j\omega+2}$.

   (a) (5 pts) Determine the differential equation describing the overall system.
   (b) (5 pts) Find the impulse response of this system from the frequency response.
   (c) (5 pts) Find the output $y(t)$ when $x(t)$ is a signal with the Fourier Transform $X(j\omega) = j\omega$. Use the frequency response.

4. (15 pts) Consider the following block diagram:



   where $H_1(e^{j\omega}) = \frac{3}{3+e^{-j\omega}}$ and $H_2(e^{j\omega}) = \frac{2}{2+e^{-j\omega}}$.

   (a) (5 pts) Find the difference equation describing this system.
   (b) (5 pts) Find the frequency response of this system.
   (c) (5 pts) Determine the system's impulse response.

5. (35 pts) Programming - Frequency Domain Encoding

- **Introduction**
  We are not using text messages anymore. They are very boring. Now, almost all mainstream messaging apps support voice messages. Therefore, as of this moment, you and I will communicate with voice messages, but I have a problem. I am paranoid about privacy. I do not trust any Big Tech company, so I encoded my voice message with a special encoding that only you and I know. Your task is to decode and write my message. Don't worry. I will give you the decoding recipe.
- **Decoding Recipe**(Encoding is also same, so you can send me encoded messages using the same recipe)
  (a) Transform the given voice signal to the Fourier domain using Fast Fourier Transform.
  (b) Split the Fourier domain representation into two parts, positive and negative frequencies, reverse both parts and concatenate them again. For example, if Fast Fourier Transform of x is $X[jw] = [a, b, c, d, e, f, g, h]$, then the encoded list must be $X'[jw] = [d, c, b, a, h, g, f, e]$
  (c) Return to the time domain using Inverse Fast Fourier Transform and listen the message.
- **Fast Fourier Transform(FFT)**
  DTFT and CTFT are great tools for theoretical purposes and filter designs, but they are not so practical for digital signals because they are defined in infinite domain. On the other hand, there is a better tool to analyze the finite sampled signals in a more elegant way, called Discrete Fourier Transform(DFT). The method is the bridge between continous time signals and discrete time signals. Also, it is not possible to catch some frequencies in a sampled signals.(see Nyquist-Shannon Sampling Theorem).
  DFT is one of the fundamental tools for analyzing signals in frequency domain. DFT, X[k], of a signal x[n] is defined as follows:

$$Discrete\,Fourier\,Transform: \; X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi\frac{k}{N}n}, \; k = [0, .., N-1]$$

$$Inverse\,Discrete\,Fourier\,Transform: \; x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j2\pi\frac{k}{N}n}, \; n = [0, .., N-1]$$

  The complexity of naive DFT algorithm is O(n^2). Therefore, a lot effort was spent to improve the efficiency of DFT algorithm family. The result is elegant divide and conquer Fast Fourier Transfrom(FFT) Algorithm, which is chosen as one of the most important 10 algorithms in the 20th century by Science. Although the algorithm was invented by Carl Friedrich Gauss in 1805 when he needed it to interpolate the orbit of asteroids Pallas and Juno from sample observations, it is reinvented and popularized during 60s. The complexity of the algorithm is O(N logN). After that point, lots of FFT variants was proposed.
  You will implement the best-known FFT algorithm. The main idea is to divide DFT algorithm into odd and even parts. It first computes the DFTs of the even-indexed inputs ($x_{2m} = x_0, x_2, \dots, x_{N-2}$) and of the odd-indexed inputs ($x_{2m+1} = x_1, x_3, \dots, x_{N-1}$), and then combines those two results to produce the DFT of the whole sequence. This idea can then be performed recursively to reduce the overall runtime to O(N log N).

$$X[k] = \sum_{n=0}^{N/2-1} x[2n]e^{\frac{-j2\pi}{N}(2n)k} + \sum_{n=0}^{N/2-1} x[2n+1]e^{\frac{-j2\pi}{N}(2n+1)k}$$

  We can simplify the procedure of the formula as follows:

$$X[k] = O[k] + E[k]e^{\frac{-j2\pi}{N}(k-1)}$$

  where O[k] and E[k] are the discrete Fourier Transforms of elements with odd and even indices, respectively. Moreover, since we know that the discrete Fourier Transform of a signal is periodic, we do not have to calculate two periods in the summations, we can calculate only the first period and then concatenate the result with itself. The only concern is that we are multiplying E[k] with $e^{\frac{-j2\pi}{N}}$. However, it has a nice property that:

$$e^{\frac{-j2\pi}{N}(k-1+N/2)} = e^{\frac{-j2\pi}{N}(k-1)}$$

  Therefore, we can write this equation as:

$$X[k] = \begin{cases} O[k] + E[k]e^{\frac{-j2\pi}{N}(k-1)}, & \text{if } k \le N/2 \\ O[k-N/2] - E[k-N/2]e^{\frac{-j2\pi}{N}(k-1-N/2)}, & \text{if } k > N/2 \end{cases}$$

  You can implement ifft() function by using fft() function. Think about that.
- **Hints**
  (a) You can check your fft() function by comparing numpy.fft.fft(). If the individual differences is below 10^{-7} for our input, your function works correctly.
  (b) For simplicity, you can assume that the length of the input file is 2^n, where n $\in \mathbb{N}$
  (c) Complexity of Fast Fourier Transform algorithm is **O(N logN)**. Please be careful about the complexity.
  (d) To read the sound data, you can use **scipy.io**. It also returns the sample rate of the audio file. It is very useful to determine the frequency bins.
  (e) Please be careful about the frequency bins of your implementation even if they are not required to complete the homework. They may be out of order.

- **Regulations**
  (a) You should add the plot of frequency domain magnitude of encoded and decoded signal and the time domain plots to your reports to see the difference between two signals. That means, your report must contain **4** different plots.
  (b) You should write the secret message to your reports. You can find the encoded message in your homework file. The name of the message is **encoded.wav**
  (c) You should use **Python3** during homework.
  (d) You are not allowed to use any library other than **numpy**, **matplotlib.pyplot** and **scipy.io**
  (e) You are not allowed to use **numpy.fft** in the homework, you should implement your own **fft()** and **ifft()** function.