BLG 354E Signals & Systems Homework 1

Due: Wednesday, May 8, 2024

Lecturer: Assoc. Prof. Ömer Melih Gül CRN: 21351

Fatih Baskın
150210710

Contents

The Radio

The file `bayrakfm.wav` has a sampling rate of 44100 Hz, and it's duration is a little more than 3 minutes 37 seconds. There had been some modifications to this file:

- Both the song (x) and the message (y) were divided into parts ($x_0, x_1, ..., x_N$ and $y_0, y_1, ..., y_M$) for each second.
- For every x_i & y_i, Fourier transform is done and arrays for frequency domain are obtained ($f(x_i), f(y_i)$).
- Second half of the $f(x_i)$ was replaced with first half of $f(y_i)$.
- The resulting $f(x_i)$ is inversely transformed to time space.

To reveal the hidden message, first, I have divided the song into 1 second long (44100 samples each) segments, there were 218 segments and each having 44100 samples except the last segment which had around 8900 samples.

Then, using the `numy.fft` I have taken the Fourier transform of each segment. It resulted in a `numpy` array consisting of 44100 numbers, corresponding to coefficients in the Fourier transform. First half of each segment's FFT belongs to the song and second half of the segment's FFT belongs to the message. In the **Figure 1**, the FFT of the first segment is given.

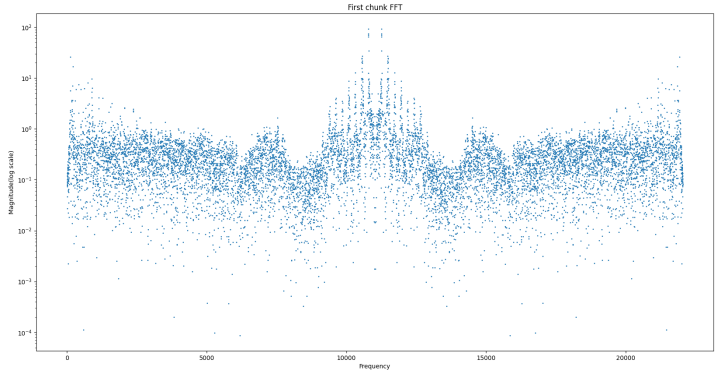

Figure 1: FFT of the first segment

Then for each segment first half and the second half of the FFTs are separated, and they are padded back with original FFT's length. For a segment with 44100 samples, FFT would have 44100 values and if we separate that into two from the middle, then we would have 22050 values left. I have added padding to restore it back to 44100 Hz.

After that, I have taken the inverse FFT for each segment and merged them together. In the end there were two files: `part1_message.wav` and `part1_song.wav`. Message can be clearly heard with very little resemblance of song.

The code is provided as `part1.py`. Packages inside the `requirements.txt` must be installed first before running the Python file.

System Simulation

The equation for DT LTI system is given:
$H(z) = \frac{1-\frac{7}{4}z^{-1}-\frac{1}{2}z^{-2}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}}$
With a little help of algebra, we can find a program to simulate $H(z)$.
$H(z) = \frac{Y(z)}{X(z)} = \frac{1-\frac{7}{4}z^{-1}-\frac{1}{2}z^{-2}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}} = \frac{1-\frac{7}{4}z^{-1}-\frac{1}{2}z^{-2}}{1+\frac{1}{4}z^{-1}-\frac{1}{8}z^{-2}} \frac{A(z)}{A(z)}$
$Y(z) = (1 - \frac{7}{4}z^{-1} - \frac{1}{2}z^{-2}) \times A(z) = A(z) - \frac{7}{4}A(z)z^{-1} - \frac{1}{2}A(z)z^{-2}$
$X(z) = (1 + \frac{1}{4}z^{-1} - \frac{1}{8}z^{-2}) \times A(z) = A(z) + \frac{1}{4}A(z)z^{-1} - \frac{1}{8}A(z)z^{-2}$
$A(z) = X(z) - \frac{1}{4}A(z)z^{-1} + \frac{1}{8}A(z)z^{-2}$
From these equations, if we say $B = A(z)z^{-1}$ and $C = A(z)z^{-2}$ we can infer this pseudocode:

- for i in range 0, end, do:
- Load Xi
- A = Xi - 1/4 * B + 1/8 * C
- Yi = A - 7/4 * B - 1/2 * C
- C = B
- B = A
- save Yi

With the operations above, the message signal will be affected as shown in the **Figure 2**.

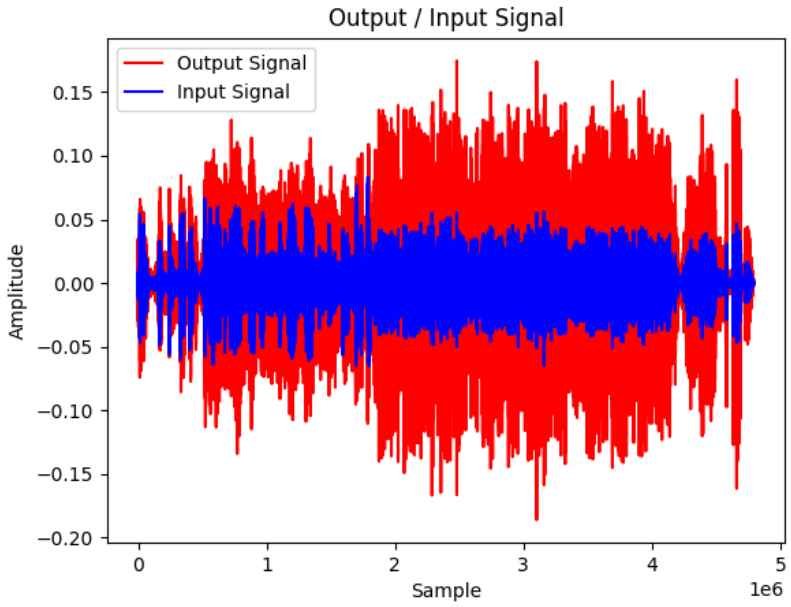

Figure 2: Before & after system simulation

Listen to Your Heart, Closely

In this part, there were some key observations about the square and triangular signals. The standard deviations of the raw audio amplitude signal is for square wave is higher than the triangular wave. To observe this, I have written a basic script to test it out several times. In used `part3_plot.py` for this task. My sampling rate was $148,000\ Hz$ and my number of samples was $1,000,000$ At the end, I got these two standard deviations:

- Standard deviation for triangular wave: around 0.57
- Standard deviation for square wave: around 0.83

To determine whether a signal is triangular or square, I have taken the sample of it as the settings above, then I have used the threshold of 0.7 to differentiate it. If the standard deviation of the signal is larger than 0.7 then it is a square signal, otherwise it is a triangular signal. The triangular and square signals are shown in **Figure 5** and **Figure 6**.

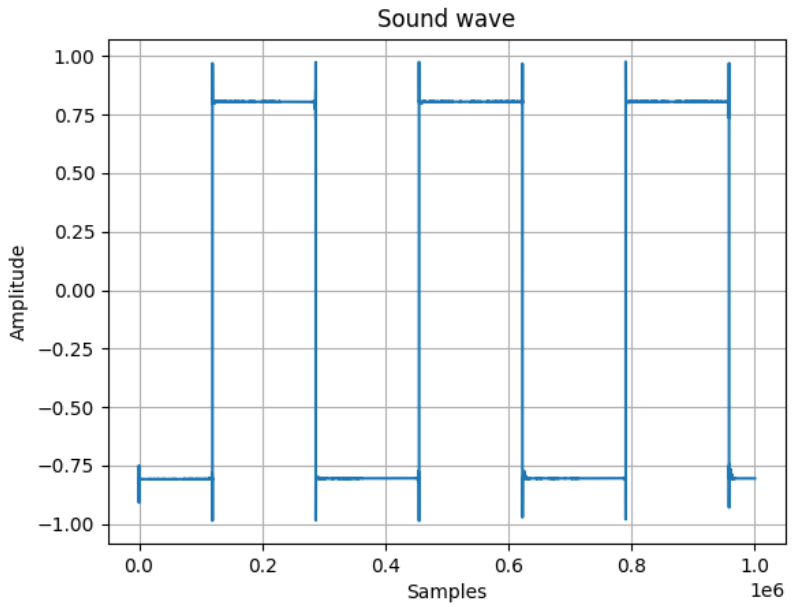

Figure 3: Square Wave

The maze solving logic was actually quite simple, I used DFS algorithm to solve the maze. I have saved the grid size of 11×7 to represent the maze, where obstacles are marked as 1. As I progress, I mark mines and previous tiles as 1 or obstacle. Eventually algorithm reaches its destination.

One problem is with `keyDown` and `keyUp` timings, since accurately getting values for those is quite hard, in some cases monster can't get to the `1/6` strip of area to listen to the mines. Therefore it might crash into a mine unintentionally.

There is a small delay to start the program. That delay is used for letting the user click on the browser page where the game is running, otherwise the Python is unable to listen to the output device (loopback

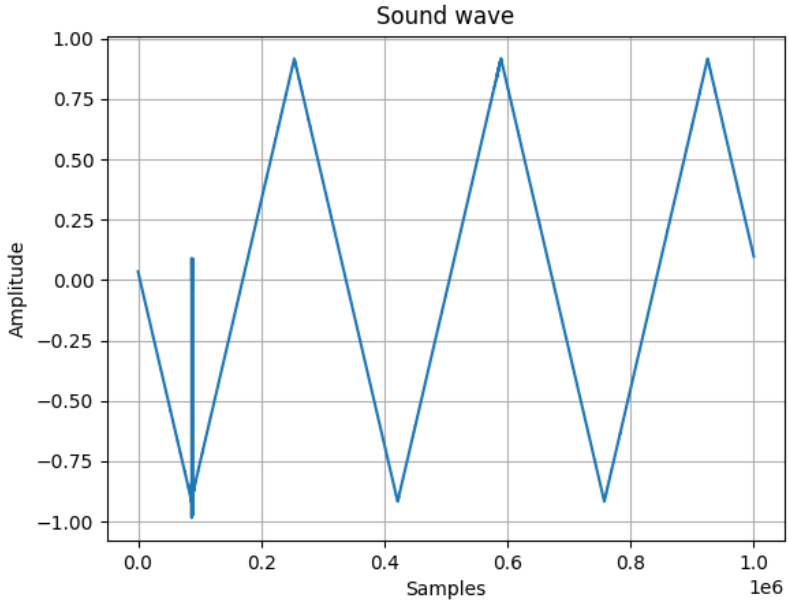

Figure 4: Triangular Wave

microphone).

The main exacutable is `part3.py` and the values (constants) might needed to be changed to run those codes in different machines. Note that this code works with Bluetooth headphones for me. In Windows, my soundcard was clipping the amplitudes, making it quite harder for me to differentiate signals.

Also one problem observed is in Ubuntu OS with Gnome desktop environment, `pyautogui` library does not work. For security reasons the `Wayland` window manager will block the inputs coming from other programs. I had to switch my OS to Windows to fix this issue.

Bode Plots

For both H1 and H2, functions that simulate the frequency response is written by hand. To cascade those systems, frequency responses are multiplied, dual of convolution in time domain.

H1 has a zero at $\omega = 1$ and it has two poles at $\omega = 10$. It is expected that function will have $-20dB/decade$ slope at the end.

H2 has only one pose at $\omega = 10$. It is expected that function will have $-20dB/decade$ slope at the end.

Their cascade will have $-20dB/decade$ slope at the end. Bode plots are given in **Figure 5** (magnitude response) and **Figure 6** (phase response).

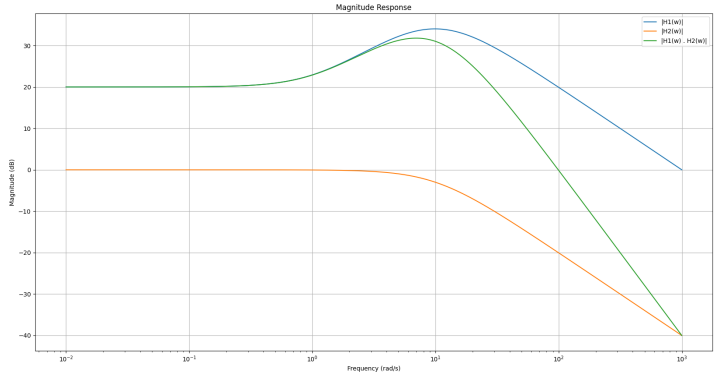

Figure 5: Magnitude response of the systems

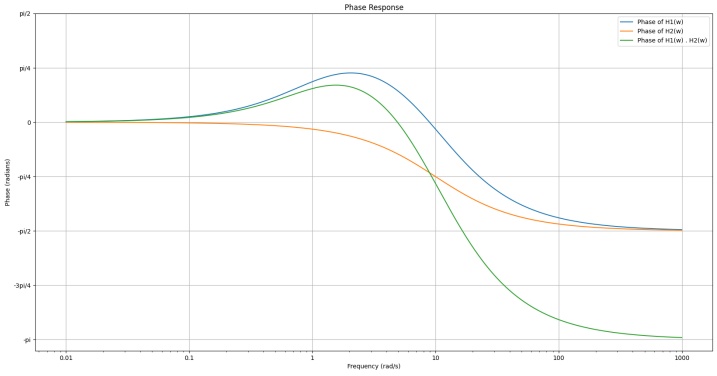

Figure 6: Phase response of the systems