Signal and Systems for Comp. Eng.

BLG 354E

Assignment 1 Report

Şafak Özkan Pala
palas21@itu.edu.tr

Faculty of Computer and Informatics Engineering
Department of Computer Engineering
Date of submission: 08.05.2024

1. Solutions

1.1. Part I and II

In the first part of the assignment, we need to implement a function that takes encrypted audio file *bayrakfm.wav* and using the given encryption algorithm we need to implement a algorithm for decryption. *the_radio.py* file contains this implementation. *decrypte_file* function takes audio file path and than split the audio file into seconds. After that we perform fourier transform into every second and take the second part of every transformed second. Using inverse fourier we get the audio again and add it into our result audio file. We concatanate every second together and get the final message file. Using soundfile library we write the audio file into the Data folder.

In the second part of the assignment, I used the same *the_radio* file. We get the audio from the Data folder and perform the given function using *scipy.signal.lfiler*. After the applied function, result audio file is saved into the Data folder.

1.2. Part III

In the third part of the assignment, *_heart.py* file contains the code. All of the actions are functionalized and used in this implementation. *simulate()* function contains the main simulation. Starting with the initial point and coordinates this function keeps calling the *make_decision* function with the previous path information and coordinates. If we reach the final coordinates it stops and gives an congratulations message. In the *make_decision* function we check every possible way using *check()* function and if there is not any mine and if it is not an empty tile character moves that direction using *move()* function and returns the path and current coordinates. In *check()* function according to given direction character moves this direction and listen the audio using *listen()* function and go back to previous coordinates and returns the result of *listen()* function. In the *move()* function character moves the given direction and updates its coordinates. In the *listen()* function, using *soundcard* library. After numerous tries, sample rate 16000 gives the best result. Using first 0.001 second of the sample its wave form determined by its mean after normalization. In the start of the simulation microphone is checked and simulation starts.

1.3. Part IV

In the final part of the assignment, given transfer functions ploted using *matplotlib.pyplot*, *numpy* and *scipy.signal*. Functions are created with *scipy.signal.TransferFunction()* function and convolution of these functions are created with *numpy.convolve()* function. Using *matplotlib.pyplot.semilogx()* Magnitute and Phases are plotted. For the better visualization legends and grids are used. The Bode plot of H_1 shows a magnitude that rises until it reaches a peak and then decreases with frequency, exhibiting a slope of -40 dB/decade due to a double pole at $\omega = 10$, while its phase starts from 0 degrees and reaches its peak than starts to decrease. On the other hand, H_2 demonstrates a magnitude starting at 0 dB and decreasing with frequency at a rate of -20 dB/decade, with a phase starting at 0 degrees and decreasing to -90 degrees at $\omega = 10$. Both systems share a common pole at ω = 10, but H_1 has an additional pole, resulting in steeper slopes in magnitude and phase. Cascaded system has zero at $\omega = 1$ and 3 poles at $\omega = 10$. Resulting a faster decrease than both H_1 and H_2. You can see the graphs below.

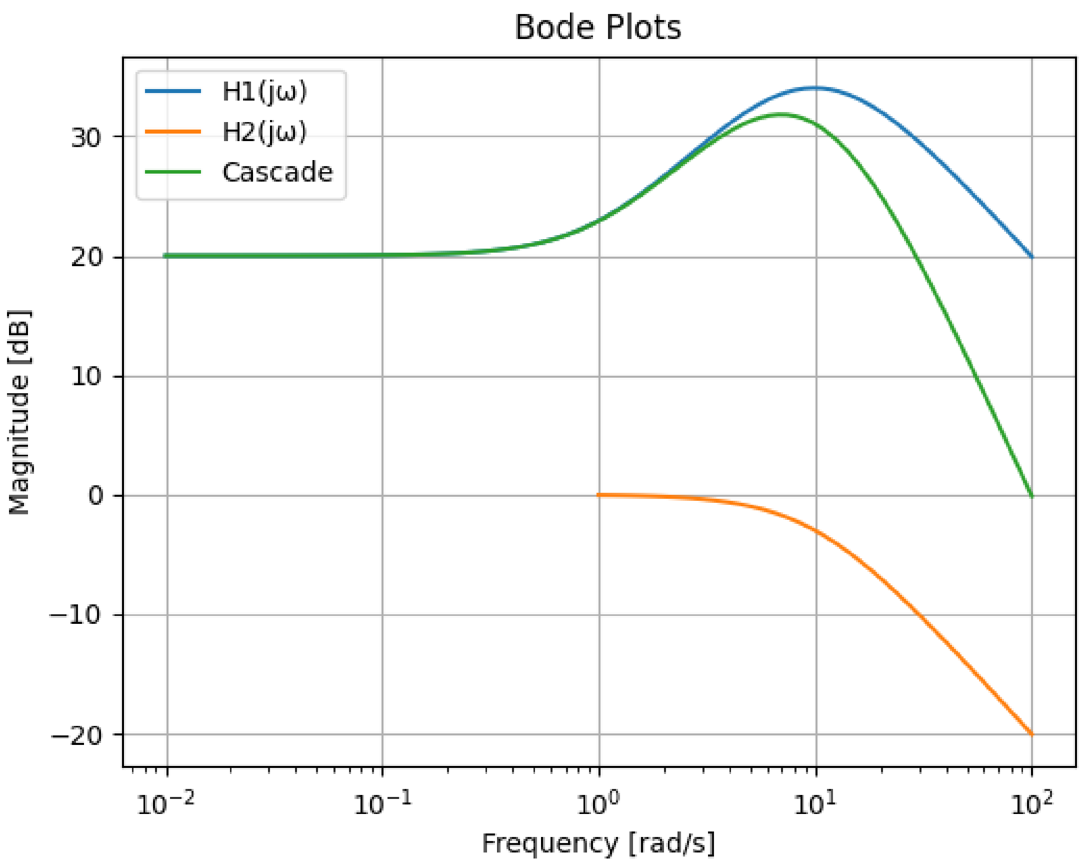

Figure 1.1: Magnitute

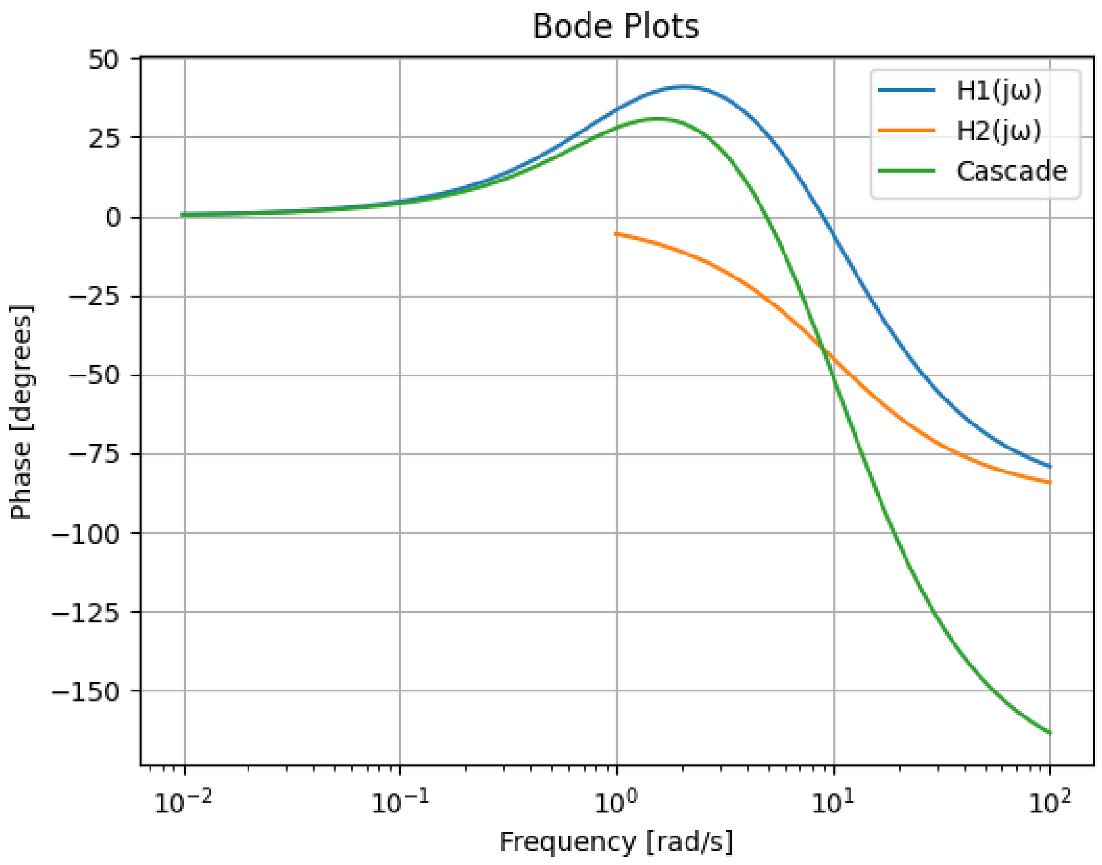

Figure 1.2: Phase