Assignment 1

Venkata Tejaswini Anagani - EE18BTECH11047

Download all the C and Python codes from

```
https://github.com/anaganitejaswini/EE3025_IDP/
    tree/main/Assignment1/codes
```

Download all the sound and data files from

```
https://github.com/anaganitejaswini/EE3025_IDP/
    tree/main/Assignment1/data
```

Run the following commands to compile and then run the C files to generate .dat files

```
gcc FILENAME.c −o FILENAME −lm
./FILENAME
```

The following code generates input data from the input sound file and stores it in a .dat file.

```
codes/generate_data.py
```

1 Digital Filter

1.1 Download the sound file from

```
wget https://raw.githubusercontent.com/
    gadepall/EE1310/master/filter/codes/
    Sound_Noise.wav
```

1.2 Write the python code for removal of out of band noise and execute the code.
Solution:

```
import soundfile as sf
from scipy import signal

#read .wav file
input_signal,fs = sf.read('Sound_Noise.wav
    ')

#sampling frequency of Input signal
sampl_freq=fs

#order of the filter
order=4

#cutoff frquency 4kHz
cutoff_freq=4000.0

#digital frequency
Wn=2*cutoff_freq/sampl_freq

# b and a are numerator and denominator
    polynomials respectively
b, a = signal.butter(order,Wn, 'low')

#filter the input signal with butterworth filter
output_signal = signal.filtfilt(b, a,
    input_signal)
#output_signal = signal.lfilter(b, a,
    input_signal)

#write the output signal into .wav file
sf.write('Sound_With_ReducedNoise.wav',
    output_signal, fs)
```

2 Difference equation

2.1 Write the difference equation of the above Digital filter obtained in problem 1.2.
Solution:

$$\sum_{m=0}^{M} a(m)\, y(n-m) = \sum_{k=0}^{N} b(k)\, x(n-k) \quad (2.0.1)$$

The resultant difference equation is given by (2.0.2)

$$\begin{aligned} y(n) - 2.52y(n-1) + 2.56y(n-2) - 1.206y(n-3) \\ +0.22013y(n-4) = 0.00345x(n) + 0.0138x(n-1)+ \\ 0.020725x(n-2) + 0.0138x(n-3) + 0.00345x(n-4) \end{aligned} \quad (2.0.2)$$

2.2 Sketch x(n) and y(n).
Solution: The following C code computes output from the difference equation and writes it into a .dat file

```
codes/yn.c
```

The following code plots Fig. 2.2 and generates Sound_de.wav file.

```
codes/xn_yn_plots.py
```

The filtered sound signal obtained through difference equation is found in

```
data/Sound_de.wav
```

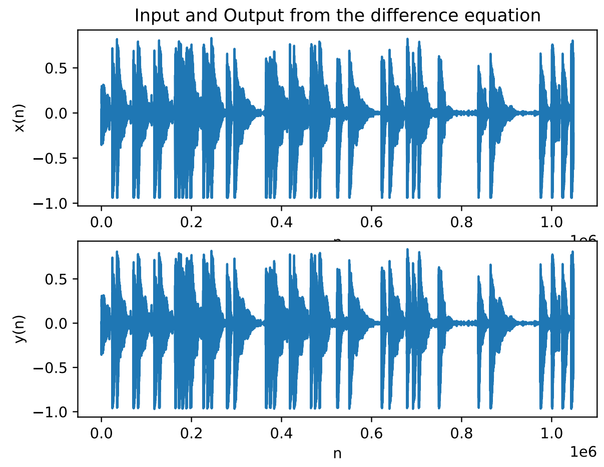

Fig. 2.2

3 Z-transform

3.1 The Z-transform of $x(n)$ is defined as

$$X(z) = \mathcal{Z}\{x(n)\} = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{3.0.1}$$

Show that

$$\mathcal{Z}\{x(n-1)\} = z^{-1}X(z) \tag{3.0.2}$$

and find

$$\mathcal{Z}\{x(n-k)\} \tag{3.0.3}$$

Solution: From (3.0.1),

$$\mathcal{Z}\{x(n-1)\} = \sum_{n=-\infty}^{\infty} x(n-1)z^{-n} \tag{3.0.4}$$

$$= \sum_{n=-\infty}^{\infty} x(n)z^{-n-1} = z^{-1}\sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{3.0.5}$$

resulting in (3.0.2). Similarly, it can be shown that

$$\mathcal{Z}\{x(n-k)\} = z^{-k}X(z) \tag{3.0.6}$$

3.2 Find

$$H(z) = \frac{Y(z)}{X(z)} \tag{3.0.7}$$

from (2.0.2) assuming that the Z-transform is a linear operation.
Solution: Applying (3.0.6) on (2.0.2) we get,

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b[0] + b[1]z^{-1} + b[2]z^{-2} + b[3]z^{-3} + b[4]z^{-4}}{a[0] + a[1]z^{-1} + a[2]z^{-2} + a[3]z^{-3} + a[4]z^{-4}} \tag{3.0.8}$$

where

$$\mathbf{a} = \begin{pmatrix} 1 & -2.52 & 2.56 & -1.206 & 0.22013 \end{pmatrix} \tag{3.0.9}$$

$$\mathbf{b} = \begin{pmatrix} 0.00345 & 0.0138 & 0.020725 & 0.0138 & 0.00345 \end{pmatrix} \tag{3.0.10}$$

3.3 Let

$$H\left(e^{\jmath w}\right) = H\left(z = e^{\jmath w}\right). \tag{3.0.11}$$

Plot $|H\left(e^{\jmath w}\right)|$.
Solution: The following code plots Fig. 3.3.

```
codes/mag_H_jw.py
```

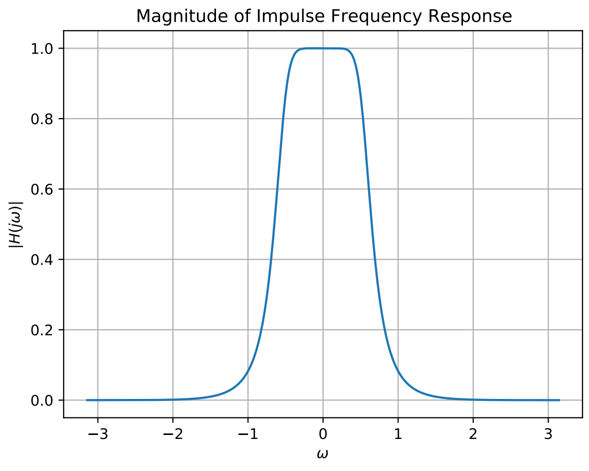

Fig. 3.3: $|H\left(e^{\jmath w}\right)|$

4 Impulse Response

4.1 Sketch h(n).
Solution: h(n) is the impulse response of the system which means for an impulse input $\delta(n)$, the output of the system is h(n). Hence, by substituting $x(n) = \delta(n)$ in Eq. (2.0.2) we can

get h(n) of the system. The following C code computes $h(n)$ from the difference equation (2.0.2) and writes it into a .dat file

```
codes/hn.c
```

The following code plots Fig. 4.1 from the .dat file.

```
codes/hn_plot.py
```

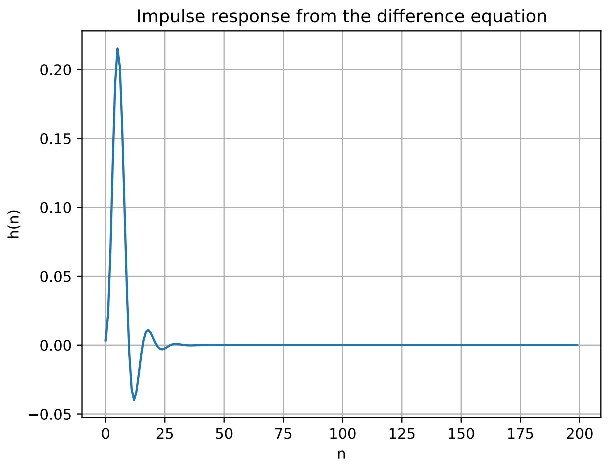

Fig. 4.1: $h(n)$

4.2 Check whether h(n) obtained is stable.
Solution: For a system to be stable, output should be bounded for any bounded input. This is known as BIBO stability. Since x(n) is bounded (the input), let B_x be some finite value

$$|y(n)| \leq B_x \sum_{-\infty}^{\infty} |x(n-k)| \tag{4.0.1}$$

Using convolution,

$$y(n) = h(n) * x(n) = x(n) * h(n) \tag{4.0.2}$$

Equation (4.0.1) implies,

$$|y(n)| = \left| \sum_{-\infty}^{\infty} h(k)x(n-k) \right| \tag{4.0.3}$$

$$|y(n)| \leq \sum_{-\infty}^{\infty} |h(k)|\,|x(n-k)| \tag{4.0.4}$$

Let B_x be the maximum value x(n-k) can take, then

$$|y(n)| \leq B_x \sum_{-\infty}^{\infty} |h(k)| \tag{4.0.5}$$

If

$$\sum_{-\infty}^{\infty} |h(k)| < \infty \tag{4.0.6}$$

Then

$$|y(n)| \leq B_y < \infty \tag{4.0.7}$$

Therefore we can say that y(n) is bounded if x(n) and h(n) are bounded. Since the audio input is bounded, the system is said to be stable if h(n) is also bounded

$$\sum_{n=-\infty}^{n=-\infty} |h(n)| < \infty \tag{4.0.8}$$

The above equation can be re written as,

$$\sum_{n=-\infty}^{n=\infty} \left| h(n)z^{-n} \right|_{|z|=1} < \infty \tag{4.0.9}$$

$$\sum_{n=-\infty}^{n=-\infty} |h(n)| \left| z^{-n} \right|_{|z|=1} < \infty \tag{4.0.10}$$

From Triangle inequality,

$$\left| \sum_{n=-\infty}^{n=-\infty} h(n)z^{-n} \right|_{|z|=1} < \infty \tag{4.0.11}$$

$$\implies |H(z)|_{|z|=1} < \infty \tag{4.0.12}$$

For the system to be stable, the Region of Convergence(ROC) should include the unit circle. Since, h(n) is right sided the ROC is outside the outer most pole. From the equation (3.0.8), poles and zeros can be found by equating denominator and numerator to zero respectively. Poles of the given transfer equation are:

$$\begin{aligned} z(approx) = 0.69786 \pm 0.4132i,\\ 0.56187 \pm 0.13779i \end{aligned} \tag{4.0.13}$$

From the above poles, we can see that that the ROC of the system is

$$|z| > \sqrt{0.69786^2 + 0.4132^2} \tag{4.0.14}$$

$$|z| > 0.811 \tag{4.0.15}$$

The following code plots the Fig. 4.2

```
codes/Reg_of_conv.py
```

From Equation (4.0.15) and Fig. 4.2 we can observe that ROC of the system includes unit circle $|z| = 1$. Therefore, the given IIR filter is stable, since h(n) is absolutely summable.

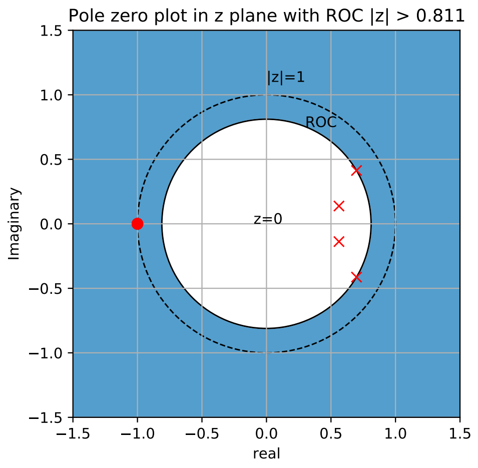

Fig. 4.2: *ROC*

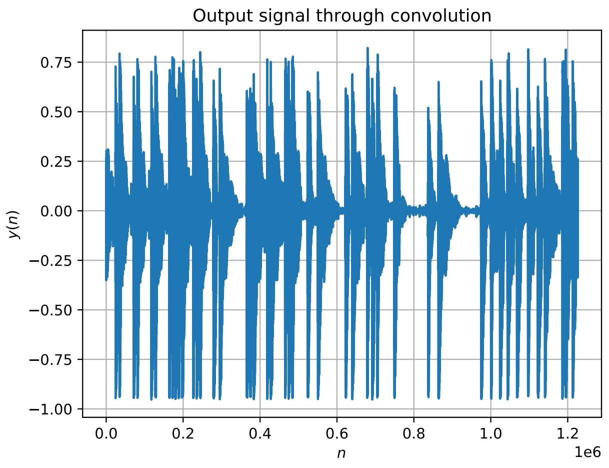

Fig. 4.3: $y(n)$ through convolution

Verification:- Given bounded input x(n) (audio sample) and system difference equation (2.0.2) From Fig. 2.2 we can see that the maximum value of x(n) is 0.8311 and minimum value is around -0.9417. Similarly from Fig. 2.2 we can also observe that the maximum value of y(n) is 0.8362 and minimum value is -0.97 and it tends to zero after the length of signal. We can conclude that for the bounded input x(n), the output y(n) is bounded. Therefore, the system is BIBO stable. Also from Fig. 4.1, we can observe that h(n) is converging to zero.

4.3 Compute filtered output using convolution formula with h(n) obtained in 4.1

$$y(n) = x(n) * h(n) = \sum_{n=-\infty}^{\infty} x(k)h(n-k) \quad (4.0.16)$$

Solution: The following code plots Fig. 4.3 where $y(n)$ is computed using convolution.

```
codes/yn_conv.py
```

We can observe that the output obtained in the Fig. 4.3 is same as y(n) obtained in Fig. 2.2. The filtered sound signal obtained through convolution method is stored in

```
data/Sound_convo.wav
```

5 FFT and IFFT

5.1 Compute

$$X(k) \triangleq \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \dots, N-1 \quad (5.0.1)$$

and $H(k)$ using h(n).
Solution: For the given IIR system with audio sample as x(n) and h(n) as impulse response where h(n) is obtained in section 4.1 DFT of the Input Signal $x(n)$ is

$$X(k) \triangleq \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \dots, N-1 \quad (5.0.2)$$

DFT of the Impulse Response $h(n)$ is

$$H(k) \triangleq \sum_{n=0}^{N-1} h(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \dots, N-1 \quad (5.0.3)$$

The following C code computes FFT of $x(n)$ and $h(n)$ using recursive sub-block approach. It also computes IFFT of Y(k) and creates a .dat file.

```
codes/fft_xn_hn_ifft_y.c
```

The following code plots FFTs of $x(n)$ and $h(n)$ from the .dat files created by the above C code.

```
codes/Xk_Hk.py
```

Magnitude plots of $|X(k)|$ and $|H(k)|$

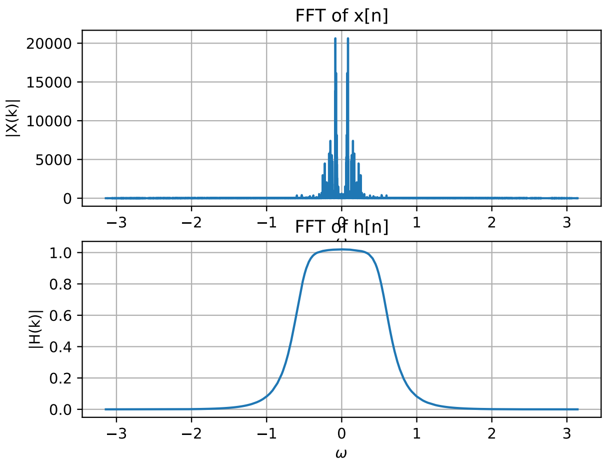

Fig. 5.1: $X(k) and H(k)$

5.2 From

$$Y(k) = X(k)H(k) \tag{5.0.4}$$

Compute

$$y(n) \triangleq \sum_{k=0}^{N-1} Y(k) e^{j2\pi kn/N}, \quad n = 0, 1, \ldots, N-1 \tag{5.0.5}$$

Solution: The following code plots Fig.5.2

```
codes/ifft_yn_plot.py
```

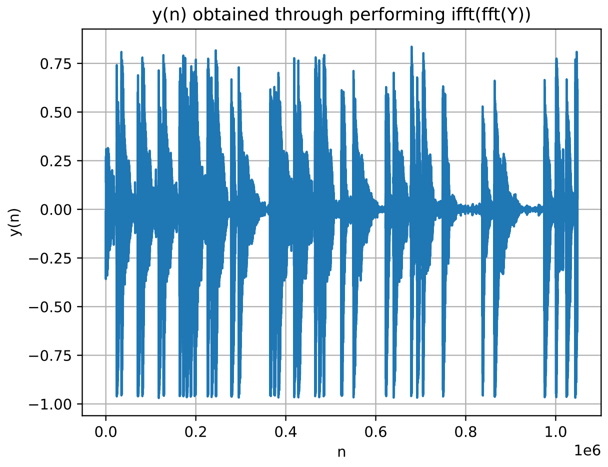

Fig. 5.2: $y(n)$

The filtered sound signal from this method is found in

```
data/Sound_ifft.wav
```

We can observe from the Fig. 5.2 that it is same as the y(n) observed in Fig.2.2