Assignment 1

Sudeep Veggalam - EE18BTECH11045

Download all the codes from

```
https://github.com/sudeepv/EE3025_IDP/
    Assignment1/codes
```

and latex-tikz codes from

```
https://github.com/sudeepv/EE3025_IDP/
    Assignment1
```

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')
print(b)
print(a)

#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: From 1.2, we get,

$$a[n] = \left\{\underset{\uparrow}{0.003}, 0.014, 0.021, 0.0138, 0.003\right\} \tag{2.0.1}$$

$$b[n] = \left\{\underset{\uparrow}{1}, -2.519, 2.561, -1.206, 0.220\right\} \tag{2.0.2}$$

From

$$\sum_{m=0}^{M} a(m)\, y(n-m) = \sum_{k=0}^{N} b(k)\, x(n-k) \tag{2.0.3}$$

The resultant difference equation is given by (2.0.4)

$$\begin{aligned} y(n) - 2.52y(n-1) + 2.56y(n-2) - 1.206y(n-3) \\ +0.22013y(n-4) = 0.00345x(n) + 0.0138x(n-1)+ \\ 0.020725x(n-2) + 0.0138x(n-3) + 0.00345x(n-4) \end{aligned} \tag{2.0.4}$$

2.2 Sketch x(n) and y(n).
Solution: The following code writes the .wav file into a .dat file.

```
codes/generate_x.c
```

The following C code computes $x(n)$ and $y(n)$ from the difference equation

```
codes/plot_xnyn.c
```

The following code plots Fig. 2.2

codes/plot_xnyn.py

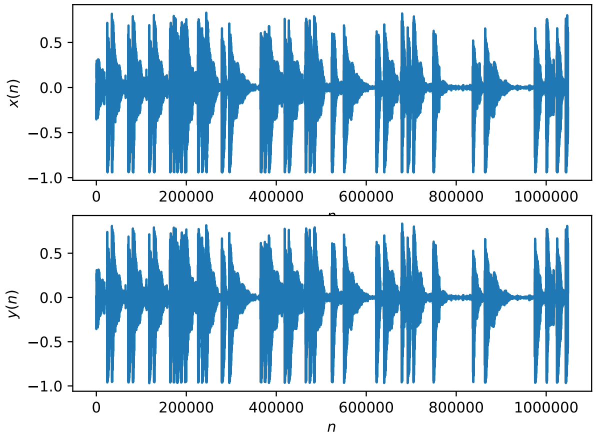

Fig. 2.2: $x(n)$ and $y(n)$ obtained from difference equations

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 (2.0.4) assuming that the Z-transform is a linear operation.
Solution: From (3.0.6) and (2.0.4) we get,

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

where a and b are given by (2.0.1) and (2.0.2)

3.3 Let

$$H\left(e^{\jmath w}\right) = H\left(z = e^{\jmath w}\right). \tag{3.0.9}$$

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

codes/plot_H_jw.py

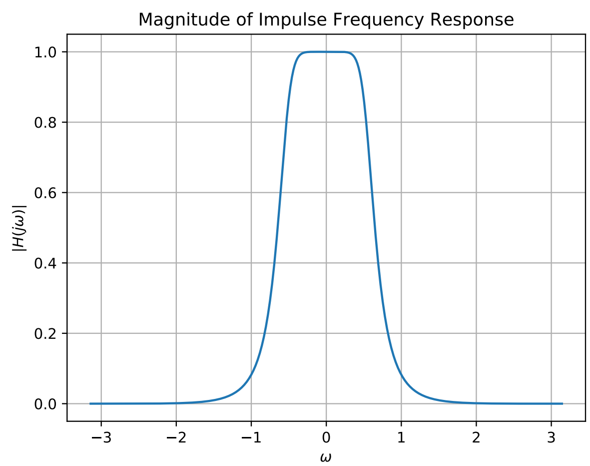

Fig. 3.3: $\left|H\left(e^{\jmath w}\right)\right|$

4 Impulse Response

4.1 Sketch h(n).
Solution: $h(n)$ (impulse response) is the output of the system if the unit impulse $\delta(n)$ is given as the input. Substituting $x(n) = \delta(n)$ in Eq. (2.0.4), we get h(n) of the system.
The following C code computes $h(n)$ from the difference equation (2.0.4)

codes/hn_compute.c

The following code plots Fig. 4.1

codes/hn_plot.py

4.2 Check whether h(n) obtained is stable.
Solution: From BIBO stability criterion - for a

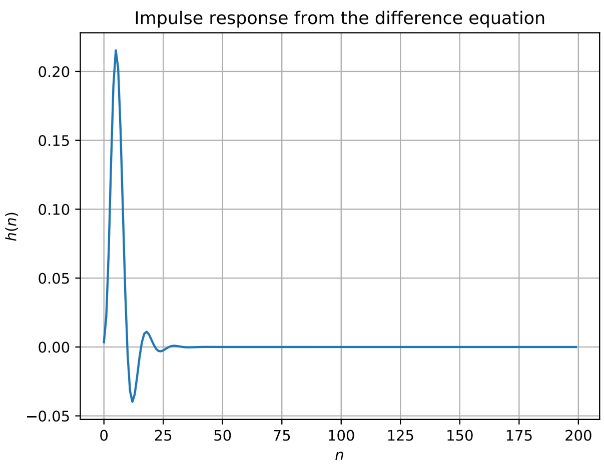

Fig. 4.1: Impulse Response ($h(n)$)

system to be stable, output should be bounded for any bounded input Since the input x(n) is bounded, if B_x is some finite value

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

As y(n) is the convolved output of x(n) and h(n)

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

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. The following code computes the poles of the transfer function given by (3.0.8)

```
codes/compute_poles.py
```

Poles of the given transfer equation are:

$$\begin{aligned} z(approx) = 0.69786079 \pm 0.41316978j, \\ 0.56187146 \pm 0.13779107j \end{aligned} \tag{4.0.13}$$

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

$$|z| > max(\sqrt{0.69^2 + 0.41^2}, \sqrt{0.56^2 + 0.13^2}) \tag{4.0.14}$$

$$|z| > max(0.811, 0.578) \tag{4.0.15}$$

$$|z| > 0.811 \tag{4.0.16}$$

From (4.0.16), ROC of the system includes unit circle $|z| = 1$. Therefore, the given IIR filter is stable, since h(n) is absolutely summable.

Verification:- Given bounded input x(n) (audio sample) and system difference equation (2.0.4) 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.

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

$$y(n) = x(n) * h(n) = \sum_{n=-\infty}^{\infty} x(k)h(n-k) \quad (4.0.17)$$

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

```
codes/conv_plot.py
```

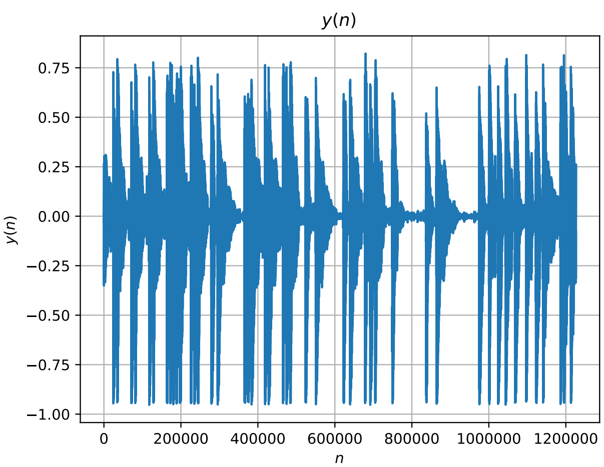

Fig. 4.3: $y(n)$ through convolution

We can observe that the output obtained in the Fig. 4.3 is same as y(n) obtained in Fig. 2.2.

5 FFT and IFFT

5.1 Compute

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

and $H(k)$ using h(n).
Solution: The audio sample x(n) has been obtained in 2.2 and the impulse response h(n) has been obtained in 4.1 DFT of the Input Signal $x(n)$ is

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

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

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

The following C code computes DFT of $x(n)$ and $h(n)$ efficiently using fft algorithm. It also computes IFFT of Y(k) and creates a .dat file.

```
codes/fft.c
```

The following code plots FFTs of $x(n)$ and $h(n)$.

```
codes/fft_plot.py
```

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

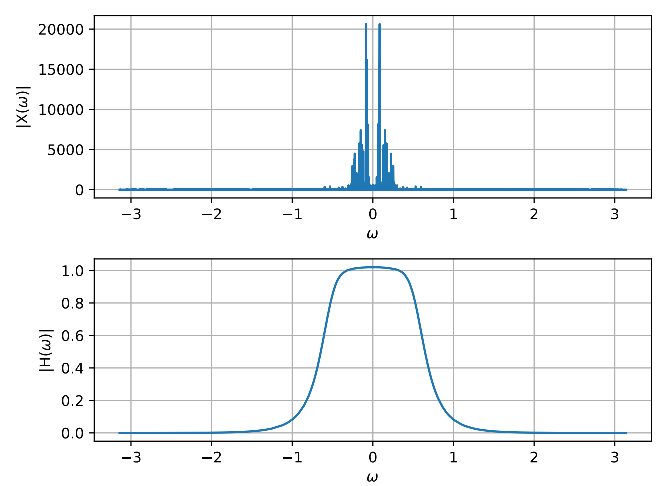

Fig. 5.1: $X(k) and H(k)$

5.2 From

$$Y(k) = X(k)H(k) \quad (5.0.4)$$

Compute

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

Solution: The following code computes the DFT of $y(n)$ by multiplying $X(\omega)$ and $H(\omega)$ and subsequently computes inverse DFT of $Y(\omega)$ to get $y(n)$.

codes/ifft.py

The following code plots $y(n)$.

codes/ifft_plot.py

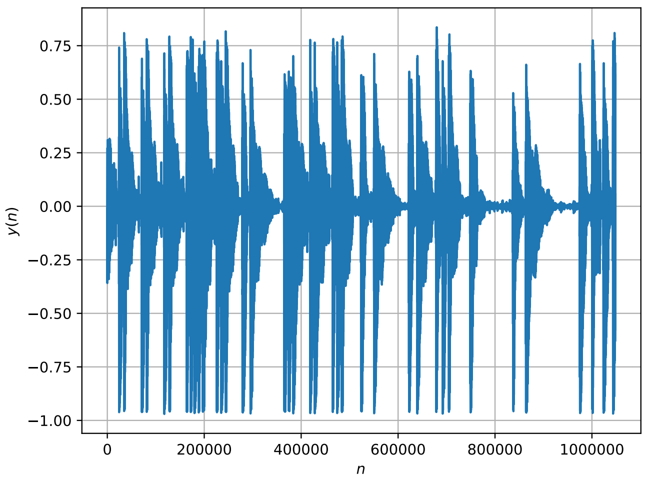

Fig. 5.2: $y(n)$ through ifft

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