Audio filtering

EE22BTECH11004 - Allu Lohith

I. Digital Filter

I.1 The sound file used for this code is given in this link

```
https://github.com/Lohith12321/signals−
    and−systems/blob/main/
    audio_filtering/codes/song.wav
```

I.2 Python code for removal of noise and produce the resultant audio

```
import soundfile as sf
from scipy import signal

# Read the input audio file
input_signal, fs = sf.read('song.wav')
print(input_signal)

# Check the shape of the input signal if it's
    multi−channel
# If it's multi−channel, take only the first
    channel
if len(input_signal.shape) > 1:
    input_signal = input_signal[:, 0]
print(input_signal)

# Define filter parameters
order = 6
cutoff_freq = 2000.0
Wn = 2 * cutoff_freq / fs

# Design the Butterworth low−pass filter
b, a = signal.butter(order, Wn, 'low')

# Apply the filter to the input signal
output_signal = signal.filtfilt(b, a,
    input_signal)

# Write the filtered signal to a new audio file
sf.write('reducednoise.wav', output_signal, fs
    )
```

I.3 Comparing the resultant audio file with original one in frequency domain. I obtained the spectrum analysing from this Academo portal *https : //academo.org/demos/spectrum − analyzer/*. The resulting graph is known as a spectrogram. The darker areas are those where the frequencies have very low intensities, and the orange and yellow areas represent frequencies that have high intensities in the sound.

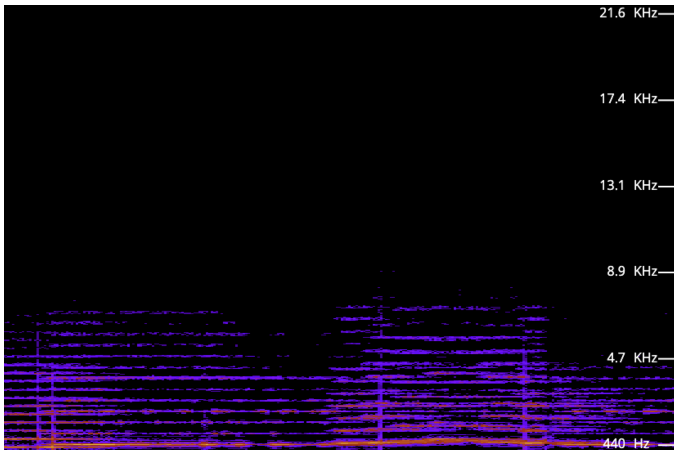

Fig. 1. Spectogram of Original audiosignal

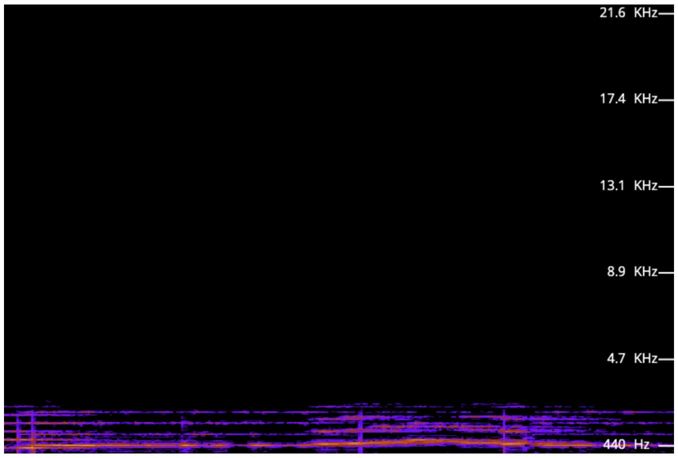

Fig. 2. Spectogram after filtering audio signal

II. Difference equation

II.1 Let

$$x(n) = \left\{\underset{\uparrow}{1}, 2, 3, 4, 2, 1\right\} \tag{1}$$

Sketch $x(n)$.

II.2 Let

$$y(n) + \frac{1}{2}y(n-1) = x(n) + x(n-2),$$

$$y(n) = 0, n < 0 \quad (2)$$

Solve ans Sketch

Solution: C code for generation of plots coordinates as text,

```
https://github.com/Lohith12321/signals−and
    −systems/blob/main/audio_filtering/
    codes/plot1.c
```

Python code for plotting the graph,

```
https://github.com/Lohith12321/signals−and
    −systems/blob/main/audio_filtering/
    codes/plot1.py
```

Plots for these text

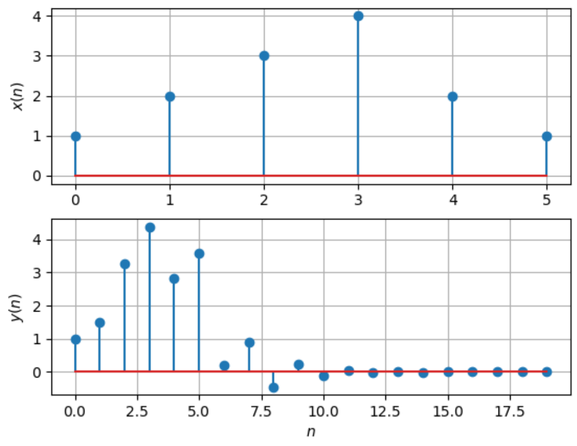

Fig. 3. Plot of x(n) and y(n)

III. Z-Transform

III.1

$$X(z) = \mathcal{Z}\{x(n)\} = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \quad (3)$$

Show that

$$\mathcal{Z}\{x(n-1)\} = z^{-1}X(z) \quad (4)$$

and find

$$\mathcal{Z}\{x(n-k)\} \quad (5)$$

Solution:

$$\mathcal{Z}\{x(n-k)\} = \sum_{n=-\infty}^{\infty} x(n-1)z^{-n} \quad (6)$$

$$\mathcal{Z}\{x(n-k)\} = x(0)z^{0} + x(1)z^{-1} + ...+ \quad (7)$$

$$\mathcal{Z}\{x(n-k)\} = z^{-1}(x(0)z^{0} + x(1)z^{-1}...) \quad (8)$$

$$\mathcal{Z}\{x(n-k)\} = z^{-1}\sum_{n=-\infty}^{\infty} x(n)z^{-n} \quad (9)$$

$$\mathcal{Z}\{x(n-k)\} = z^{-1}X(z) \quad (10)$$

Similarly we can show that

$$\mathcal{Z}\{x(n-k)\} = z^{-k}X(z) \quad (11)$$

III.2

$$H(z) = \frac{Y(z)}{X(z)} \quad (12)$$

from (2) assuming that the Z-transform is a linear operation.

Solution: Applying (11) in (2),

$$Y(z) + \frac{1}{2}z^{-1}Y(z) = X(z) + z^{-2}X(z) \quad (13)$$

$$\implies \frac{Y(z)}{X(z)} = \frac{1+z^{-2}}{1+\frac{1}{2}z^{-1}} \quad (14)$$

III.3 Find the Z transform of

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

and show that the Z-transform of

$$u(n) = \begin{cases} 1 & n \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

is

$$U(z) = \frac{1}{1-z^{-1}}, \quad |z| > 1 \quad (17)$$

Solution: It is easy to show that

$$\delta(n) \overset{\mathcal{Z}}{\longleftrightarrow} 1 \quad (18)$$

and from (16),

$$U(z) = \sum_{n=0}^{\infty} z^{-n} \quad (19)$$

$$= \frac{1}{1-z^{-1}}, \quad |z| > 1 \quad (20)$$

using the formula for the sum of an infinite geometric progression.

III.4 Show that

$$a^n u(n) \overset{\mathcal{Z}}{\longleftrightarrow} \frac{1}{1-az^{-1}} \quad |z| > |a| \tag{21}$$

Solution:

$$a^n u(n) \overset{\mathcal{Z}}{\longleftrightarrow} \sum_{n=0}^{\infty} \left(a^n u\left(n\right)\right) z^{-n} \tag{22}$$

$$\overset{\mathcal{Z}}{\longleftrightarrow} \sum_{n=0}^{\infty} \left(a^n \left(1\right)\right) z^{-n} \tag{23}$$

$$\overset{\mathcal{Z}}{\longleftrightarrow} \sum_{n=0}^{\infty} \left(az^{-1}\right)^n \tag{24}$$

$$\overset{\mathcal{Z}}{\longleftrightarrow} \frac{1}{1-az^{-1}} \quad |z| > |a| \tag{25}$$

III.5 Let

$$H\left(e^{\jmath\omega}\right) = H\left(z = e^{\jmath\omega}\right). \tag{26}$$

Plot $|H\left(e^{\jmath\omega}\right)|$. Comment. $H(e^{\jmath\omega})$ is known as the *Discret Time Fourier Transform* (DTFT) of $x(n)$.

Solution: Substituting $z = e^{j\omega}$ in (14), we get

$$\left|H\left(e^{j\omega}\right)\right| = \left|\frac{1+e^{-2j\omega}}{1+\frac{1}{2}e^{-j\omega}}\right| \tag{27}$$

$$= \sqrt{\frac{\left(1+\cos 2\omega\right)^2 + \left(\sin 2\omega\right)^2}{\left(1+\frac{1}{2}\cos\omega\right)^2 + \left(\frac{1}{2}\sin\omega\right)^2}} \tag{28}$$

$$= \frac{4|\cos\omega|}{\sqrt{5+4\cos\omega}} \tag{29}$$

$$\left|H\left(e^{j(\omega+2\pi)}\right)\right| = \frac{4|\cos\left(\omega+2\pi\right)|}{\sqrt{5+4\cos\left(\omega+2\pi\right)}} \tag{30}$$

$$= \frac{4|\cos\omega|}{\sqrt{5+4\cos\omega}} \tag{31}$$

$$= \left|H\left(e^{j\omega}\right)\right| \tag{32}$$

Therefore its fundamental period is 2π , which verifies that DTFT of a signal is always periodic.

The following code shows the plot

```
https://github.com/Lohith12321/signals
  −and−systems/blob/main/
  audio_filtering/codes/plot2.py
```

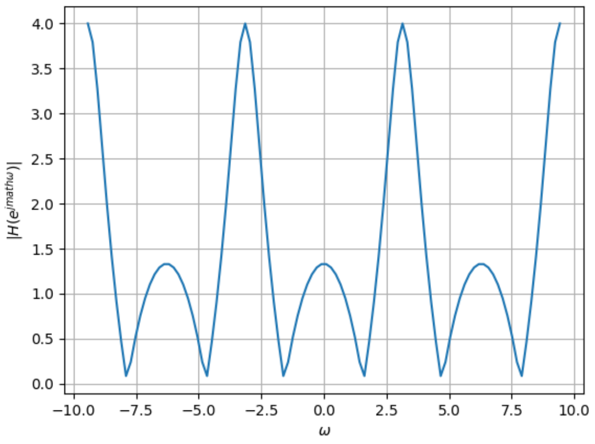

Fig. 4. $|H(e^{j\omega}|$

IV. Impulsive Response

IV.1 Find an expression for $h(n)$ using $H(z)$, given that

$$h(n) \overset{\mathcal{Z}}{\longleftrightarrow} H(z) \tag{33}$$

and there is a one to one relationship between $h(n)$ and $H(z)$. $h(n)$ is known as the *impulse response* of the system defined by (2).

Solution: From (14),

$$H(z) = \frac{1}{1+\frac{1}{2}z^{-1}} + \frac{z^{-2}}{1+\frac{1}{2}z^{-1}} \tag{34}$$

$$\implies h(n) = \left(-\frac{1}{2}\right)^n u(n) + \left(-\frac{1}{2}\right)^{n-2} u(n-2) \tag{35}$$

using (21) and (11).

IV.2 Sketch $h(n)$. Is it bounded? Convergent?

Solution: The following code plots $h\left(n\right)$

```
https://github.com/Lohith12321/
  signals−and−systems/blob/main
  /audio_filtering/codes/plot3.py
```

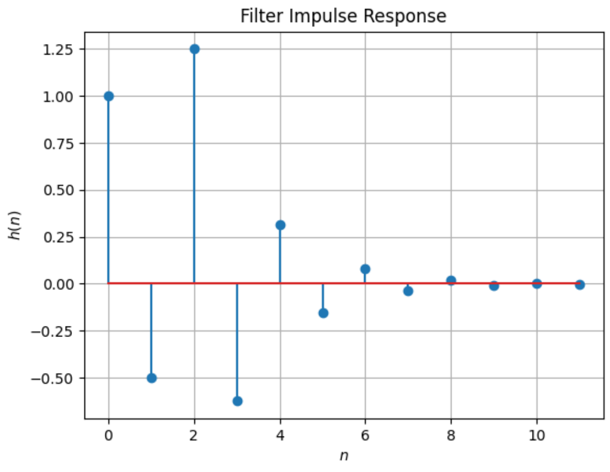

Fig. 5. $h(n)$

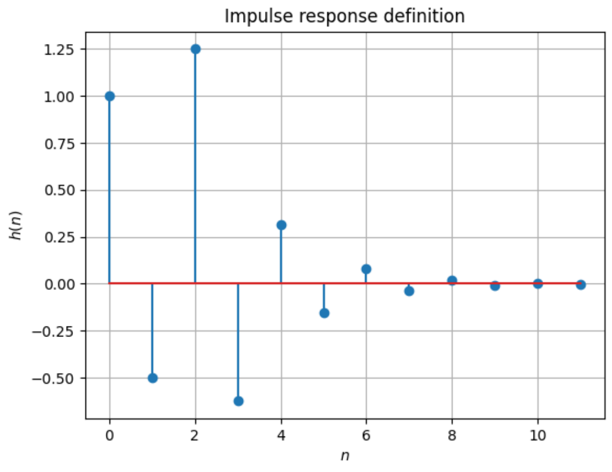

Fig. 6. $h(n)$

IV.3 The system with $h(n)$ is defined to be stable if

$$\sum_{n=-\infty}^{\infty} h(n) < \infty \tag{36}$$

Is the system defined by (2) stable for the impulse response in (33)?
Solution: For stable system (36) should converge.
From ratio test

$$\lim_{n\to\infty} \left| \frac{h(n+1)}{h(n)} \right| < 1 \tag{37}$$

As n is very large,

$$u(n) \approx (n-2) \approx 1 \tag{38}$$

$$\lim_{n\to\infty} \left(\frac{h(n+1)}{h(n)} \right) = 1/2 < 1 \tag{39}$$

Therefore it converges and stable.

IV.4 Compute and sketch $h(n)$ using

$$h(n) + \frac{1}{2}h(n-1) = \delta(n) + \delta(n-2), \tag{40}$$

This is the definition of $h(n)$.
Solution:
Definition of $h(n)$: The output of the system when $\delta(n)$ is given as input.

The following code plots Fig. V.V.3. Note that this is the same as Fig. IV.IV.3.

```
https://github.com/Lohith12321/
    signals−and−systems/blob/main
    /audio_filtering/codes/plot4.py
```

IV.5 Compute

$$y(n) = x(n) * h(n) = \sum_{n=-\infty}^{\infty} x(k)h(n-k) \tag{41}$$

Comment. The operation in (41) is known as convolution.
Solution: The following code plots Fig. IV.IV.3. Note that this is the same as $y(n)$ in Fig. II.II.3. The following code gives above plot

```
https://github.com/Lohith12321/
    signals−and−systems/blob/main
    /audio_filtering/codes/plot5.py
```

IV.6 Show that

$$y(n) = \sum_{n=-\infty}^{\infty} x(n-k)h(k) \tag{42}$$

Solution: In (41), we substitute $k = n-k$

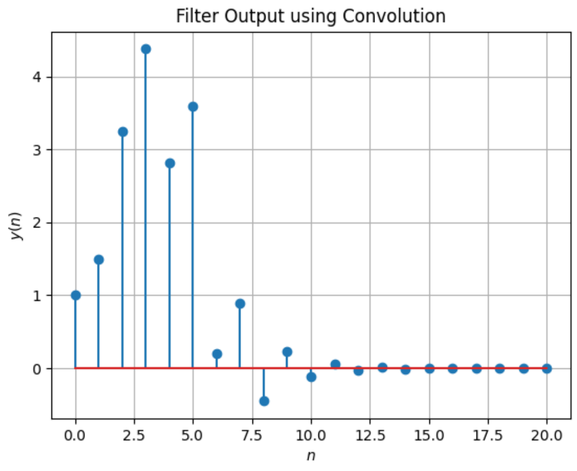

Fig. 7. $h(n)$

to get

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)\, h(n-k) \tag{43}$$
$$= \sum_{n-k=-\infty}^{\infty} x(n-k)\, h(k) \tag{44}$$
$$= \sum_{k=-\infty}^{\infty} x(n-k)\, h(k) \tag{45}$$

V. DFT and FFT

V.1 Compute

$$X(k) \overset{\triangle}{=} \sum_{n=0}^{N-1} x(n) e^{-\jmath 2\pi kn/N}, \quad k = 0, 1, \dots, N-1 \tag{46}$$

and $H(k)$ using $h(n)$.

V.2 Compute

$$Y(k) = X(k)H(k) \tag{47}$$

V.3 Compute

$$y(n) = \frac{1}{N} \sum_{k=0}^{N-1} Y(k) \cdot e^{\jmath 2\pi kn/N}, n = 0, 1, \dots, N-1 \tag{48}$$

Solution: The above three questions are solved using the code below

https://github.com/Lohith12321/signals−and−systems/blob/main/audio_filtering/codes/plot7.py

This is the plot

V.4 Repeat the previous exercise by computing $X(k), H(k)$ and $y(n)$ through FFT and IFFT.

Solution: The solution of this question can be found in the code below.

https://github.com/Lohith12321/signals−and−systems/blob/main/audio_filtering/codes/plot6.py

This code verifies the result by plotting the obtained result with the result obtained by IDFT.

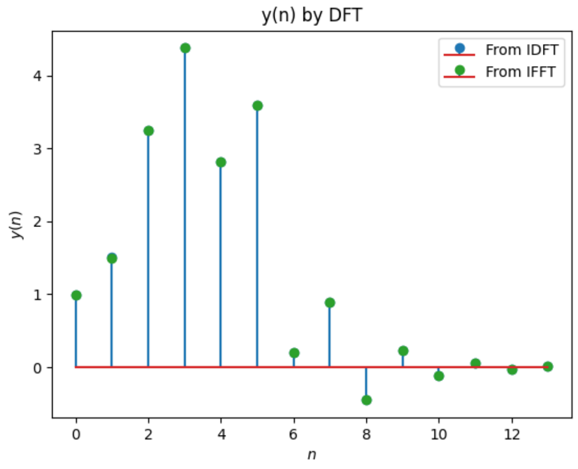

Fig. 8. $h(n)$

V.5 Wherever possible, express all the above equations as matrix equations.

Solution: The DFT matrix is defined as :

$$\mathbf{W} = \begin{pmatrix} \omega^0 & \omega^0 & \dots & \omega^0 \\ \omega^0 & \omega^1 & \dots & \omega^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \omega^0 & \omega^{N-1} & \dots & \omega^{(N-1)(N-1)} \end{pmatrix} \tag{49}$$

where $\omega = e^{-\frac{j2\pi}{N}}$. Now any DFT equation can be written as

$$\mathbf{X} = \mathbf{W}\mathbf{x} \tag{50}$$

where

$$\mathbf{x} = \begin{pmatrix} x(0) \\ x(1) \\ \vdots \\ x(n-1) \end{pmatrix} \tag{51}$$

$$\mathbf{X} = \begin{pmatrix} X(0) \\ X(1) \\ \vdots \\ X(n-1) \end{pmatrix} \tag{52}$$

Thus we can rewrite (47) as:

$$\mathbf{Y} = \mathbf{X} \odot \mathbf{H} = (\mathbf{Wx}) \odot (\mathbf{Wh}) \tag{53}$$

where the $\odot$ represents the Hadamard product which performs element-wise multiplication.

The below code computes $y(n)$ by DFT Matrix and then plots it.

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot7.py
```

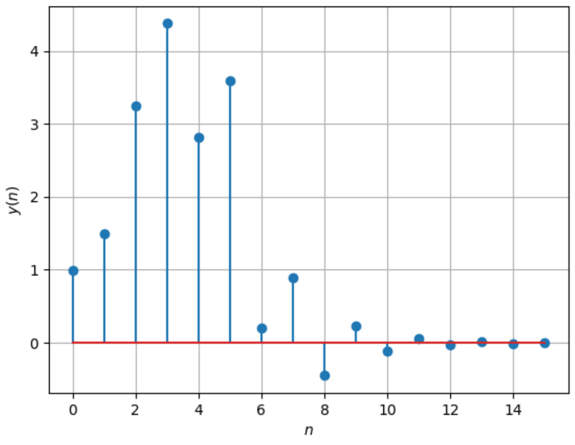

Fig. 9. $y(n)$ obtained from DFT Matrix

VI. Exercises

Answer the following questions by looking at the python code in Problem I.1.

VI.1

```
output_signal = signal.lfilter(b, a,
    input_signal)
```

in Problem I.1 is executed through the following difference equation

$$\sum_{m=0}^{M} a(m)\, y(n-m) = \sum_{k=0}^{N} b(k)\, x(n-k) \tag{54}$$

where the input signal is $x(n)$ and the output signal is $y(n)$ with initial values all 0. Replace **signal. filtfilt** with your own routine and verify.

Solution: The below code gives the output of an Audio Filter without using the built in function signal.lfilter.

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot8.py
```

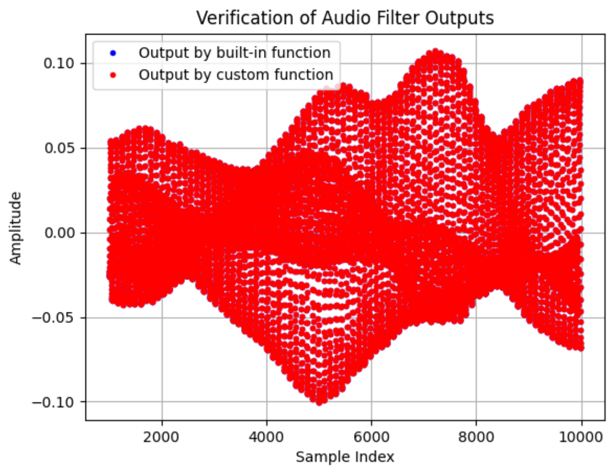

Fig. 10. Both the outputs using and without using function overlap

VI.2 Repeat all the exercises in the previous sections for the above a and b.
Solution: The code in I.1 generates the values of a and b which can be used to generate a difference equation.
And,

$$M = 5 \tag{55}$$
$$N = 5 \tag{56}$$

From 54

$$a(0)\,y(n) + a(1)\,y(n-1) + a(2)\,y(n-2) + a(3)\,y(n- \tag{57}$$

$$+\,a(4)\,y(n-4) + a(5)\,y(n-5) + a(6)\,y(n-6) \tag{58}$$

$$= b(0)\,x(n) + b(1)\,x(n-1) + b(2)\,x(n-2) + b(3)\,x(\tag{59}$$

$$+\,b(4)\,x(n-4) + b(5)\,x(n-5) + b(6)\,x(n-6) \tag{60}$$

Difference Equation is given by :

$$\begin{aligned} & y(n) - (1)\, y(n-1) + (-4.89)\, y(n-2) \\ & - (-2.23)\, y(n-3) + (0.47)\, y(n-4) \\ & = \left(29.13 \times 10^{-5}\right) x(n) + \left(116.5 \times 10^{-5}\right) x(n-1) \\ & + \left(174.8 \times 10^{-5}\right) x(n-2) + \left(116.5 \times 10^{-5}\right) x(n-3) \\ & + \left(29.1 \times 10^{-5}\right) x(n-4) \end{aligned} \tag{61}$$

From (54)

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_M z^{-N}}{a_0 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-M}} \tag{62}$$

$$H(z) = \frac{\sum_{k=0}^{N} b(k) z^{-k}}{\sum_{k=0}^{M} a(k) z^{-k}} \tag{63}$$

Partial fraction on (63) can be generalised as:

$$H(z) = \sum_i \frac{r(i)}{1 - p(i) z^{-1}} + \sum_j k(j) z^{-j} \tag{64}$$

Now,

$$a^n u(n) \overset{\mathcal{Z}}{\longleftrightarrow} \frac{1}{1 - a z^{-1}} \tag{65}$$

$$\delta(n-k) \overset{\mathcal{Z}}{\longleftrightarrow} z^{-k} \tag{66}$$

Taking inverse z transform of (64) by using (65) and (66)

$$h(n) = \sum_i r(i)[p(i)]^n u(n) + \sum_j k(j) \delta(n-j) \tag{67}$$

The below code computes the values of $r(i), p(i), k(i)$ and plots $h(n)$

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot12.py
```

$r(i)$	$p(i)$	$k(i)$
$0.0590681 - 0.14379042j$	$0.75473906 + 0.05721986j$	1.51×10^{-5}
$0.37838666 + 0.67153278j$	$0.75473906 - 0.05721986j$	–
$-0.43693011 + 0.00899083j$	$0.8005462 + 0.1658155j$	–
$-0.43693011 - 0.00899083j$	$0.8005462 - 0.1658155j$	–
$0.05853839 + 0.09906525j$	$0.89458778 + 0.25311651j$	–
$0.05853839 - 0.09906525j$	$0.89458778 - 0.25311651j$	–

//

TABLE 1
VALUES OF $r(i), p(i), k(i)$

Stability of h(n):
According to (36)

$$H(z) = \sum_{n=0}^{\infty} h(n) z^{-n} \tag{68}$$

$$H(1) = \sum_{n=0}^{\infty} h(n) = \frac{\sum_{k=0}^{N} b(k)}{\sum_{k=0}^{M} a(k)} < \infty \tag{69}$$

As both $a(k)$ and $b(k)$ are finite length sequences they converge.
The below code plots Filter frequency response

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot9.py
```

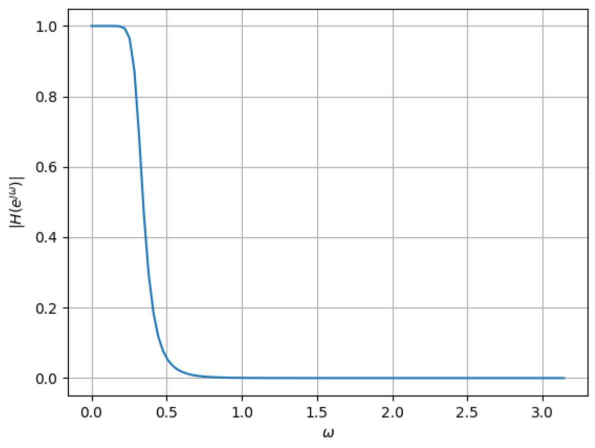

Fig. 11. Frequency Response of Audio Filter

The below code plots the Butterworth Filter in analog domain by using bilinear transform.

$$z = \frac{1 + sT/2}{1 - sT/2} \tag{70}$$

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot10.py
```

The below code plots the Pole-Zero Plot of the frequency response.

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot11.py
```

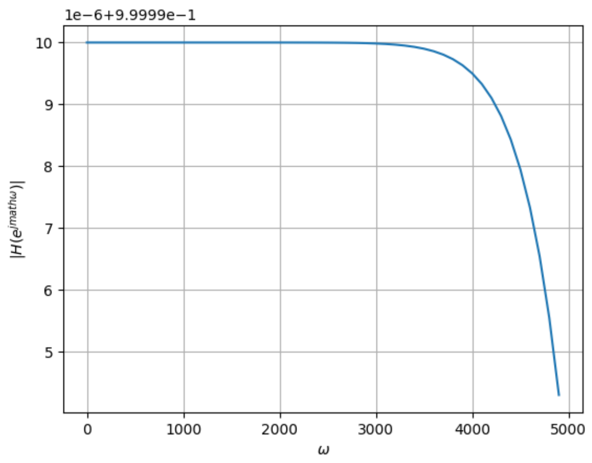

Fig. 12. Frequency Response of Audio Filter

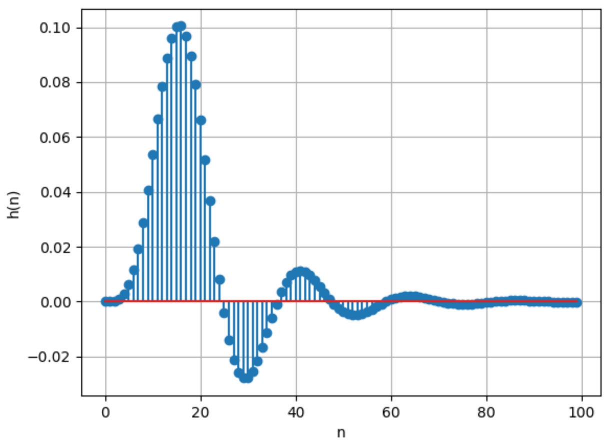

Fig. 14. h(n) of Audio Filter.It is a damped sinusoid

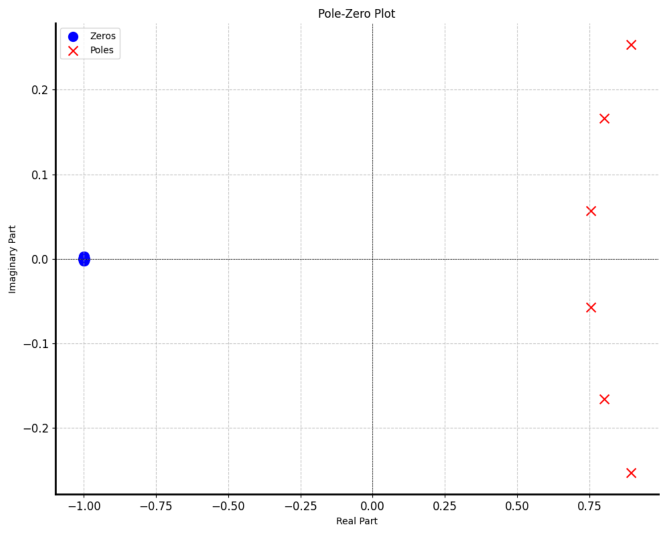

Fig. 13. As there are complex poles, so h(n) should be damped

VI.3 Implement your own fft routine in C and call this fft in python.

Solution: The below C code computes FFT of a given sequence.

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/fft.c
```

The C function involved in computing the FFT is called in the below python code and the result is computed.

Before executing the python code. Execute the following command.

```
gcc âshared âo fft.so âfPIC fft.c
```

then execute this python code

```
https://github.com/Lohith12321/
    signals−and−systems/blob/main/
    audio_filtering/codes/fft.py
```

VI.4 Find the time complexities of computing y(n) using FFT/IFFT and convolution and Compare.
Solution: The time required to compute y(n) using these two methods is calculated and the data is stored in a text file using the below C code.

```
https://github.com/Lohith12321/signals−
    and−systems/blob/main/
    audio_filtering/codes/plot13.c
```

The below python code extracts the data from these text files and plots Time vs *n* for comparison.

```
https://github.com/Lohith12321/signals
    −and−systems/blob/main/
    audio_filtering/codes/plot13.py
```

VI.5 What is the sampling frequency of the input signal?
Solution: The Sampling Frequency is 44.1KHz

VI.6 What is type, order and cutoff-frequency of the above butterworth filter
Solution: The given butterworth filter

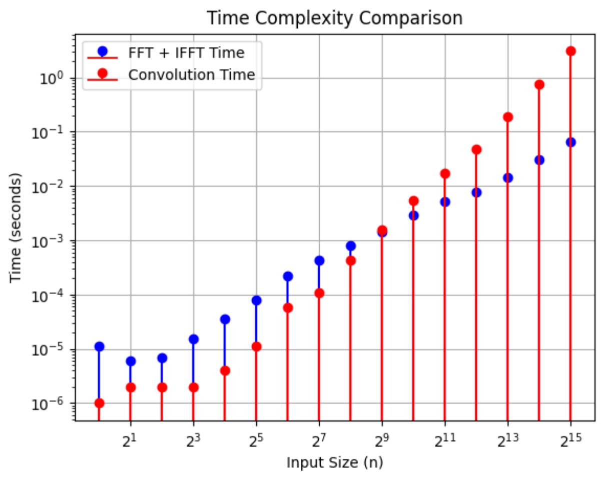

Fig. 15. The Complexity of FFT+IFFT method is $O(nlogn)$ where as by convolution is $O(n^2)$

is lowpass with order=6 and cutoff-frequency=2kHz.

VI.7 Modify the code with different input parameters and get the best possible output.
Solution: A better filtering was found on setting the order of the filter to be 5.