Audio Filter

EE23BTECH11006 - Ameen Aazam*

I. Digital Filter

I.1 The sound file I have used in this audio filter project is available in this link

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/Tomar_%20Khola_Hawa.wav

I.2 This is the Python code used for filtering the audio signal :

```
import soundfile as sf
from scipy import signal

# Read .wav file
input_signal, fs = sf.read('Tomar_
    Khola_Hawa.wav')

# Sampling frequency of Input signal
sample_freq = fs

# Order of the filter
order = 4

# Cutoff frequency 4kHz
cutoff_freq = 4000.0

# Digital frequency
Wn = 2 * cutoff_freq / sample_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, padlen=1)

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

I.3 The mentioned audio file is analyzed with a spectrogram, using the open source platform https://academo.org/demos/spectrum-analyzer. Darker areas represent very low amplitude while the orange and yellow areas denote high intensity in the corresponding frequencies.

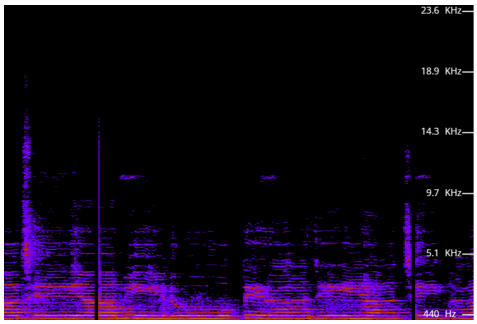

Fig. 1. Raw Audio

This is the spectrum of the original recorded audio.

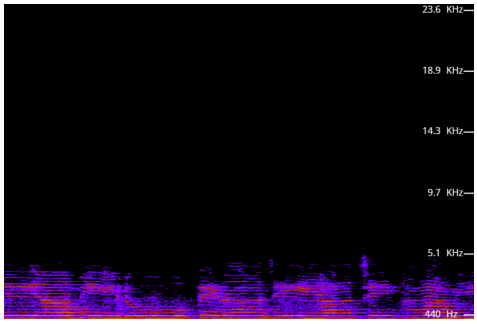

Fig. 2. Filtered Audio

And this is after the filtering. As we can see higher frequency components ($> 4kHz$) in the original signal have been significantly diminished in the filtered output.

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

$$y(n) + \frac{1}{2}y(n-1) = x(n) + x(n-2), \quad y(n) = 0, n < 0 \tag{2}$$

Sketch $y(n)$.
Solve
The following C code calculates $y(n)$ and stores the data points in a dat file.

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/y.c

Now this python code uses the dat file to plot the stem plot of $y(n)$.

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/plot.py

And this is the obtained plot of $x(n)$ and $y(n)$,

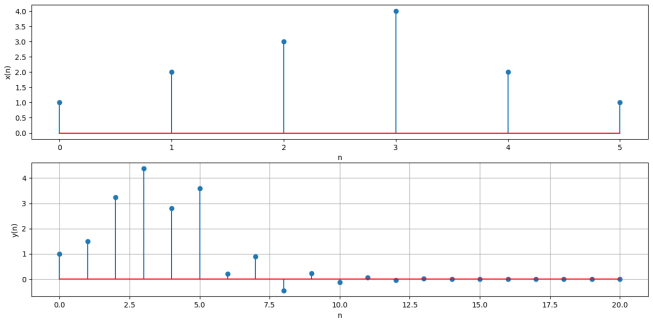

Fig. 3. Plot of $x(n)$ and $y(n)$

III. Z-Transform

III.1 The Z-transform of $x(n)$ is defined as

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

Show that

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

and find

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

Solution: From (3),

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

$$= \sum_{n=-\infty}^{\infty} x(n)z^{-n-1} = z^{-1}\sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{7}$$

resulting in (4). Similarly, it can be shown that

$$\mathcal{Z}\{x(n-k)\} = z^{-k}X(z) \tag{8}$$

III.2 Find

$$H(z) = \frac{Y(z)}{X(z)} \tag{9}$$

from (2) assuming that the Z-transform is a linear operation.
Solution: Applying (8) in (2),

$$Y(z) + \frac{1}{2}z^{-1}Y(z) = X(z) + z^{-2}X(z) \tag{10}$$

$$\implies \frac{Y(z)}{X(z)} = \frac{1+z^{-2}}{1+\frac{1}{2}z^{-1}} \tag{11}$$

III.3 Find the Z transform of

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

and show that the Z-transform of

$$u(n) = \begin{cases} 1 & n \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

is

$$U(z) = \frac{1}{1-z^{-1}}, \quad |z| > 1 \tag{14}$$

Solution: It is easy to show that

$$\delta(n) \overset{\mathcal{Z}}{\longleftrightarrow} 1 \tag{15}$$

and from (13),

$$U(z) = \sum_{n=0}^{\infty} z^{-n} \tag{16}$$

$$= \frac{1}{1-z^{-1}}, \quad |z| > 1 \tag{17}$$

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

III.4 Show that

$$a^n u(n) \overset{\mathcal{Z}}{\longleftrightarrow} \frac{1}{1-az^{-1}} \quad |z| > |a| \tag{18}$$

Solution:

$$a^n u(n) \overset{\mathcal{Z}}{\longleftrightarrow} \sum_{n=-\infty}^{+\infty} a^n u(n) z^{-n} \tag{19}$$

$$= \sum_{n=-\infty}^{0} a^n z^{-n} \tag{20}$$

$$= \frac{1}{1-az^{-1}} \quad |z| > |a| \tag{21}$$

III.5 Let

$$H\left(e^{\mathrm{J}\omega}\right) = H\left(z = e^{\mathrm{J}\omega}\right). \tag{22}$$

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

Solution: Substituting $z = e^{j\omega}$ in (11) and taking the modulus, we get,

$$\left|H\left(e^{j\omega}\right)\right| = \left|\frac{1+e^{-2j\omega}}{1+\frac{1}{2}e^{-j\omega}}\right| \tag{23}$$

$$\left|H\left(e^{j(\omega+2\pi)}\right)\right| = \left|\frac{1+e^{-2j(\omega+2\pi)}}{1+\frac{1}{2}e^{-j(\omega+2\pi)}}\right| \tag{24}$$

$$= \left|\frac{1+e^{-2j\omega}\cdot e^{-4j\pi}}{1+\frac{1}{2}e^{-j\omega}\cdot e^{-2j\pi}}\right| \tag{25}$$

$$= \left|\frac{1+e^{-2j\omega}}{1+\frac{1}{2}e^{-j\omega}}\right| \tag{26}$$

$$= \left|H\left(e^{j\omega}\right)\right| \tag{27}$$

So this verifies DTFT of any signal is always periodic with the fundamental period of 2π. Now the following Python code plots the magnitude of the transfer function.

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/3.5.py

And this is the plot,

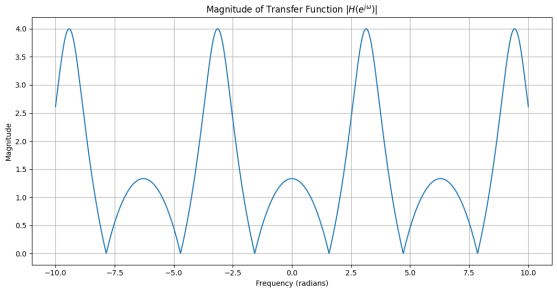

Fig. 4. $\left|H\left(e^{j\omega}\right)\right|$

IV. Impulse Response

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

$$h(n) \overset{\mathcal{Z}}{\longleftrightarrow} H(z) \tag{28}$$

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 (**??**),

$$H(z) = \frac{1}{1+\frac{1}{2}z^{-1}} + \frac{z^{-2}}{1+\frac{1}{2}z^{-1}} \tag{29}$$

$$\implies h(n) = \left(-\frac{1}{2}\right)^{n} u(n) + \left(-\frac{1}{2}\right)^{n-2} u(n-2) \tag{30}$$

using (18) and (8).

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

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

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/4.2.py

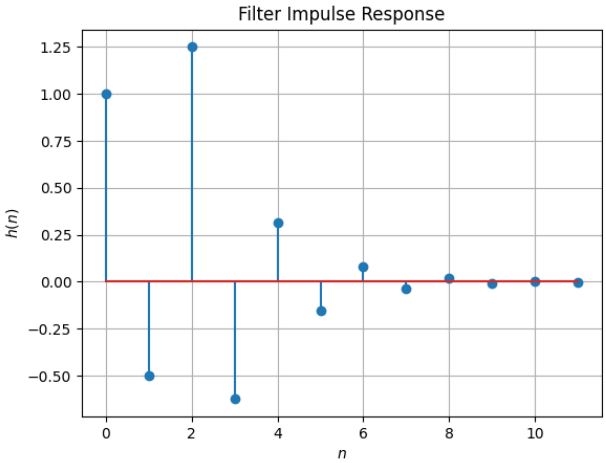

Fig. 5. $h(n)$ as the inverse of $H(z)$

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

$$\sum_{n=-\infty}^{\infty} h(n) < \infty \tag{31}$$

Is the system defined by (2) stable for the impulse response in (28)?

The convergence of (31) is obtained by the tes,

$$\lim_{n\to\infty}\left|\frac{h\left(n+1\right)}{h\left(n\right)}\right| < 1 \tag{32}$$

Now for any $n > 0$ $\delta\left(n-2\right) = \delta\left(n\right) = 0$. Now using this in (2),

$$\lim_{n\to\infty}\left|\frac{h\left(n+1\right)}{h\left(n\right)}\right| = \frac{1}{2} < 1 \tag{33}$$

So the sum is converging and the system is stable.

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

$$h(n) + \frac{1}{2}h(n-1) = \delta(n) + \delta(n-2), \qquad (34)$$

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. 6. Note that this is the same as Fig. 5.

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/hndef.py

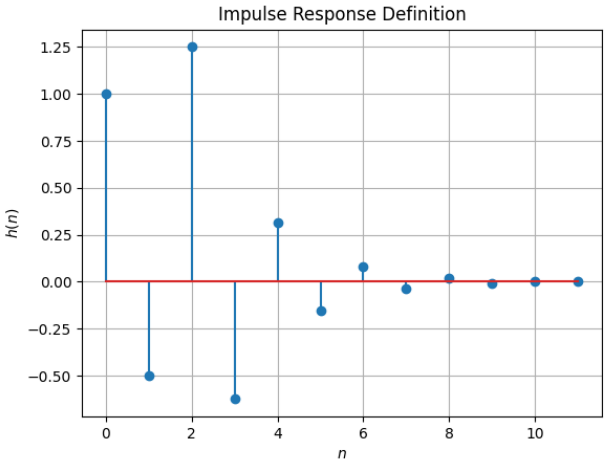

Fig. 6. $h(n)$ from the definition is same as Fig. 5

IV.5 Compute

$$y(n) = x(n) * h(n) = \sum_{n=-\infty}^{\infty} x(k)h(n-k) \qquad (35)$$

Comment. The operation in (35) is known as *convolution*.

Solution: The following code plots Fig. 7. Note that this is the same as $y(n)$ in Fig. 3.

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/4.5.py

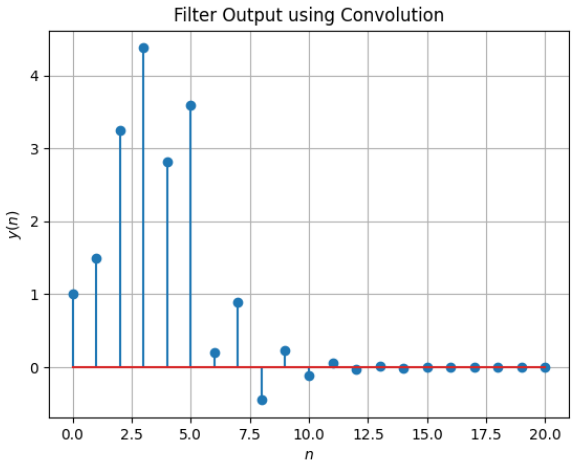

Fig. 7. $y(n)$ from the definition of convolution

IV.6 Show that

$$y(n) = \sum_{n=-\infty}^{\infty} x(n-k)h(k) \qquad (36)$$

Solution: In (35), if we substitute $k = n - k$,

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)\,h(n-k) \qquad (37)$$

$$= \sum_{n-k=-\infty}^{\infty} x(n-k)\,h(k) \qquad (38)$$

$$= \sum_{k=-\infty}^{\infty} x(n-k)\,h(k) \qquad (39)$$

V. DFT and FFT

V.1 Compute

$$X(k) \stackrel{\triangle}{=} \sum_{n=0}^{N-1} x(n)e^{-\mathrm{j}2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \qquad (40)$$

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

V.2 Compute

$$Y(k) = X(k)H(k) \qquad (41)$$

V.3 Compute

$$y(n) = \frac{1}{N}\sum_{k=0}^{N-1} Y(k)\cdot e^{\mathrm{j}2\pi kn/N}, \quad n = 0, 1, \ldots, N-1 \qquad (42)$$

Solution: The following code solves the above problems and plots the following output using DFT.

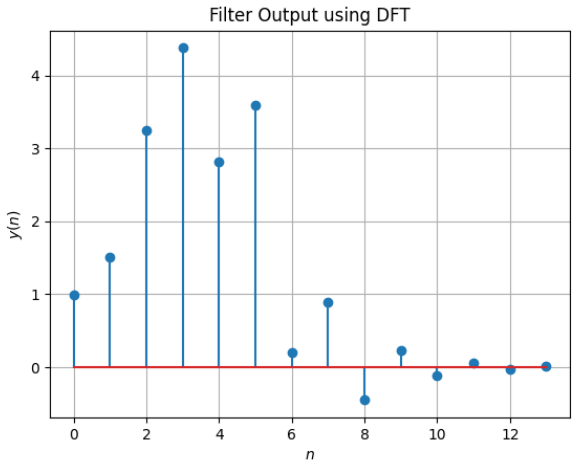

Fig. 8. $y(n)$ from DFT

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/6.py

V.4 Repeat the previous exercise by computing $X(k), H(k)$ and $y(n)$ through FFT and IFFT.
Solution: The following code uses FFT and IFFT for solving the previous question,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/5.4.py

And this is plot $y(n)$ obtained, which matches

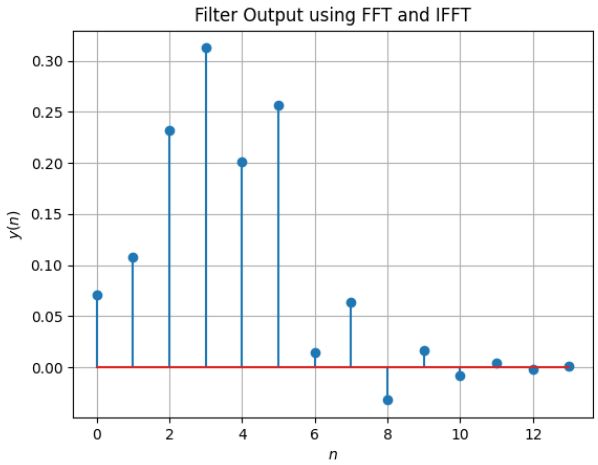

Fig. 9. $y(n)$ with FFT and IFFT

our gained $y(n)$.

V.5 Wherever possible, express all the above equations as matrix equations.
Solution: The general expression for DFT is given as,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{2\pi k}{N} n} \tag{43}$$

Here N is number of samples of the original signal. Now this equation can be expressed as a matrix equation as following,

$$\mathbf{X} = \mathbf{xW} \tag{44}$$

Where $\mathbf{x}$ is the sample points, given as,

$$\mathbf{x} = \begin{pmatrix} x(0) & x(1) & x(2) & \dots & x(N-1) \end{pmatrix} \tag{45}$$

And $\mathbf{W}$ is the DFT matrix, defined as,

$$\mathbf{W} = \begin{pmatrix} w^0 & w^0 & w^0 & \dots & w^0 \\ w^0 & w^1 & w^2 & \dots & w^{N-1} \\ w^0 & w^2 & w^4 & \dots & w^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w^0 & w^{N-1} & w^{2(N-1)} & \dots & w^{(N-1)(N-1)} \end{pmatrix} \tag{46}$$

And the $\mathbf{X}$ matrix is the output matrix defined as,

$$\mathbf{X} = \begin{pmatrix} X(0) & X(1) & X(2) & \dots & X(N-1) \end{pmatrix} \tag{47}$$

So we can express (41) as a matrix equation,

$$\mathbf{Y} = \mathbf{X} \odot \mathbf{H} \tag{48}$$

$$= (\mathbf{xW}) \odot (\mathbf{hW}) \tag{49}$$

where the $\odot$ represents the Hadamard product which performs element-wise multiplication. The following code provides the plot for $y(n)$ using the DFT matrix at first and then calculating IFFT,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/5.5.py

VI. EXERCISES

Answer the following questions by looking at the python code in Problem I.2.

VI.1 The command

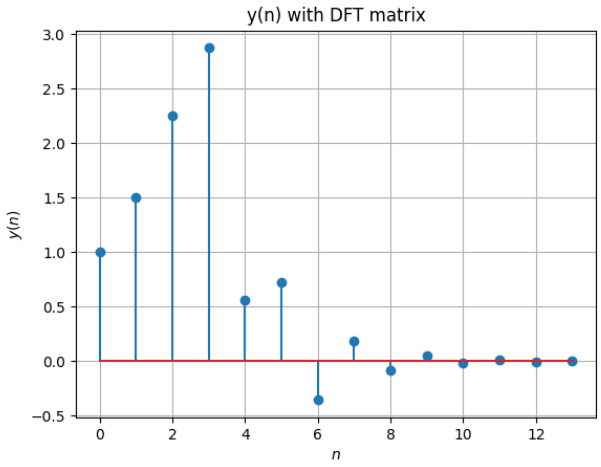

Fig. 10. $y(n)$ with DFT matrix

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

in Problem I.2 is executed through the following difference equation

$$\sum_{m=0}^{M} a(m)\, y(n-m) = \sum_{k=0}^{N} b(k)\, x(n-k) \quad (50)$$

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 following code gives the filtered output of the audio signal we are using, without using the inbuilt function.

https://github.com/Ameen−etc/Audio−Filter/blob/main/codes/6.1.py

And this is the plot we will be getting, And

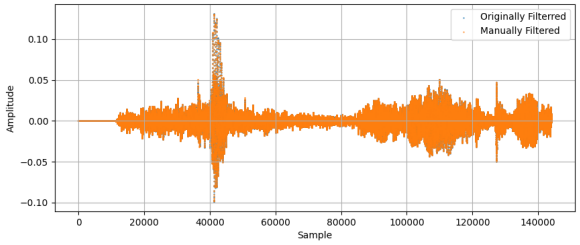

Fig. 11. Originally filtered *vs*. Manually filtered

as we can see, our output matches with the previously filtered output.

VI.2 Repeat all the exercises in the previous sections for the above a and b.
Solution: The values of a and b, we can get from the above code itself and those can be used to form the difference equation. As a 4^{th} order Butterworth filter we will be having,

$$M = 4 \quad (51)$$

$$N = 4 \quad (52)$$

And we will be getting the difference equation like,

$$\begin{aligned} &a(0)y(n) + a(1)y(n-1) + a(2)y(n-2) + \\ &a(3)y(n-3) + a(4)y(n-4) = b(0)x(n) + \\ &b(1)x(n-1) + b(2)x(n-2) + b(3)x(n-3) \\ &+ b(4)x(n-4) \end{aligned} \quad (53)$$

Now, using the values of as and bs we will have,

$$\begin{aligned} &y(n) - 2.64y(n-1) + 2.77y(n-2) - \\ &1.34y(n-3) + 0.25y(n-4) = 0.003x(n) + \\ &0.010x(n-1) + 0.015x(n-2) + 0.010x(n-3) \\ &+ 0.003x(n-4) \end{aligned} \quad (54)$$

So from (53) we can have the transfer function as,

$$H(z) = \frac{b(0) + b(1)z^{-1} + b(2)z^{-1} + b(3)z^{-3} + b(4)z^{-4}}{a(0) + a(1)z^{-1} + a(2)z^{-1} + a(3)z^{-3} + a(4)z^{-4}} \quad (55)$$

And we can split it into partial fractions,

$$H(z) = \sum_{i=1}^{4} \left(\frac{P_i}{1 - Q_i z^{-1}} \right) + R \quad (56)$$

And then using inverse Z-transform,

$$h(n) = \sum_{i=1}^{4} \left[P_i Q_i^n u(n) \right] + R\delta(n) \quad (57)$$

This code is used to get the corresponding g values of P, Q, and R,

https://github.com/Ameen−etc/Audio−Filter/blob/main/codes/residue.py

And the values are listed below,

P_i	Q_i	R
$0.21814719 - 0.35050232j$	$0.592381 + 0.13088214j$	0.00257643
$0.21814719 + 0.35050232j$	$0.592381 - 0.13088214j$	–
$-0.2095952 - 0.0102857j$	$0.72693287 + 0.3877475j$	–
$-0.2095952 + 0.0102857j$	$0.72693287 - 0.3877475j$	–

TABLE 1
RESIDUE VALUES

- Stability of $h(n)$:
  $h(n)$ will be stable if it's Z-transform contains the unit circle or $H(z)$ is defined for $z = 1$

$$H(z) = \sum_{n=0}^{\infty} h(n) z^{-n} \tag{58}$$

$$H(1) = \sum_{n=0}^{\infty} h(n) = \frac{\sum_{k=0}^{4} b(k)}{\sum_{k=4}^{M} a(k)} < \infty \tag{59}$$

  As all $a(k)$ s and $b(k)$ s are finite, the sum converges. And hence, the impulse response is stable.

Now this code provides the following discrete-time Butterworth filter frequency response,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/discrete-time-res.py

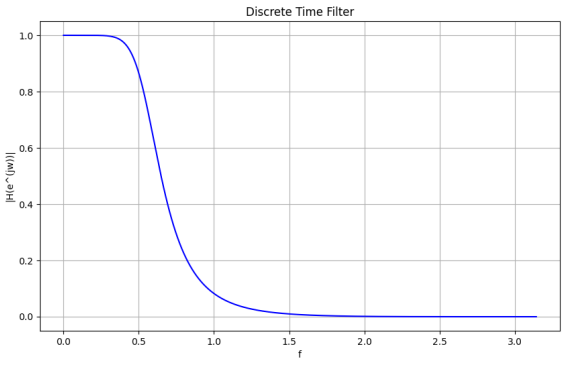

Fig. 12. Discrete-time response

Now using Bilinear Transform by replacing

$$z = \frac{1 - \frac{T}{2}s}{1 + \frac{T}{2}s} \tag{60}$$

we will be getting the continuous-time analog response. And the following code does that,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/analog-res.py

And this is the obtained frequency response, Now this code plots the pole-zero plot of the

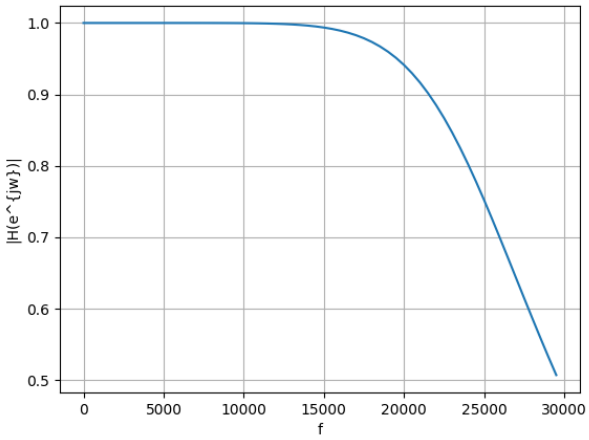

Fig. 13. Continuous-time response

filter,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/pole-zero.py

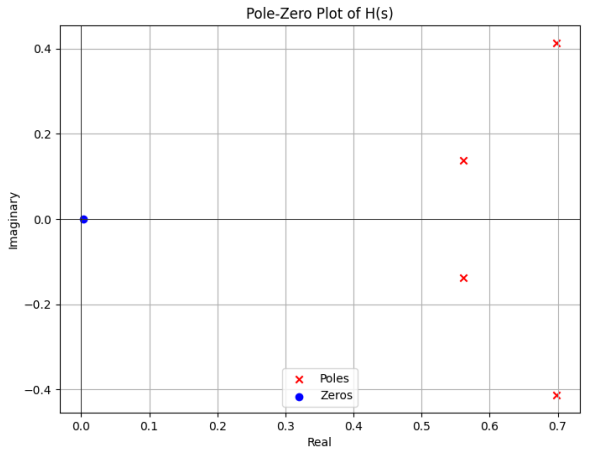

Fig. 14. Pole-Zero Plot

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

The following C code calculates the fft of any predefined array manually,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/fft.c

Now, to call this C function in the python code, we have to create a shared library by using the bash script,

```
gcc -shared -o fft.dll -fPIC fft.c
```

And this is the python code which calls the C function to perform fft,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/fft.py

VI.4 Find the time complexities of computing y(n) using FFT/IFFT and convolution and Compare. The below code plots the time taken for the calculation of $y(n)$ using convolution against FFT/IFFT method,

https://github.com/Ameen-etc/Audio-Filter/blob/main/codes/timecom.py

And this is the result,

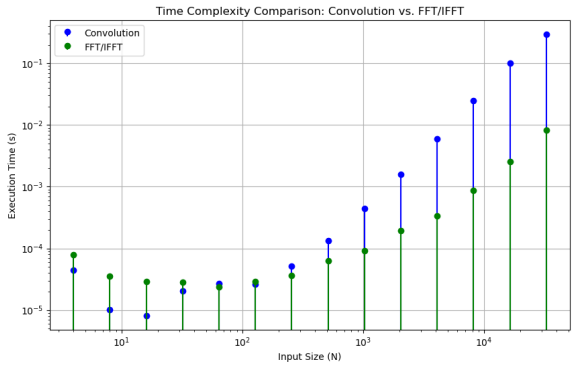

Fig. 15. Time Complexity

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 is low-pass with order=4 and cutoff-frequency=1kHz.

VI.7 Modify the code with different input parameters and get the best possible output.
Solution: A better filtering can be found by setting the order of the filter higher.