Digital Signal Processing

Sathvika Marri

Contents

***Abstract*—This manual provides a simple introduction to digital signal processing.**

1 Software Installation

Run the following commands (commands may change depending on Linux distro)

```
$ sudo apt update && sudo apt upgrade
$ sudo apt install libffi−dev libsndfile1 python3−
    scipy python3−pip python3−numpy python3−
    matplotlib
$ pip3 install cffi pysoundfile
```

2 Digital s

2.1 Download the sound file using

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/Sound_Noise.wav
```

2.2 You will find a spectrogram at https://academo.org/demos/spectrum-analyzer. Upload the sound file that you downloaded in Problem 2.1 in the spectrogram and play. Observe the spectrogram. What do you find?
Solution: There are a lot of yellow lines between 440 Hz to 5.1 KHz. These represent the synthesizer key tones. Also, the key strokes are audible along with background noise.

2.3 Write the python code for removal of out of band noise and execute the code.
Solution: Download the source code using

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/2_3.py
```

and execute it using

```
$ python3 2_3.py
```

2.4 The output of the python script in Problem 2.3 is the audio file `Sound_With_ReducedNoise.wav`. Play the file in the spectrogram in Problem 2.2. What do you observe?
Solution: The key strokes as well as background noise is subdued in the audio. Also, the signal is blank for frequencies above 5.1 kHz.

3 Difference Equation

3.1 Let

$$x(n) = \left\{\underset{\uparrow}{1}, 2, 3, 4, 2, 1\right\} \tag{3.1}$$

Sketch $x(n)$.

3.2 Let

$$y(n) + \frac{1}{2}y(n-1) = x(n) + x(n-2),$$
$$y(n) = 0, n < 0 \tag{3.2}$$

Sketch $y(n)$.
Solution: The following C code calculates $y(n)$.

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/3_2.c
```

Run it using

```
$ gcc −lm −Wall −g −O2 3_2.c
$ ./a.out
```

The following code plots Fig. (3.1).

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/3_2.py
```

Execute it using

```
$ python3 3_2.py
```

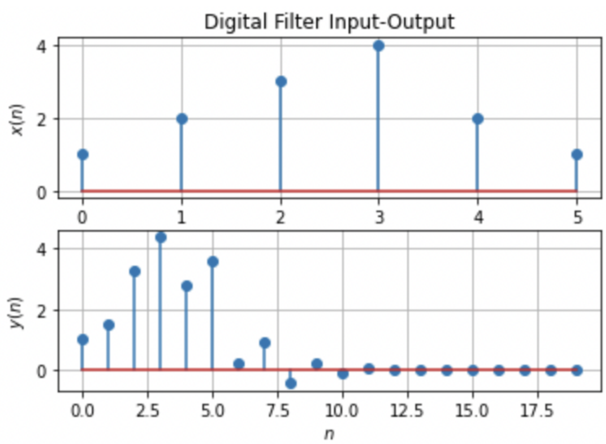

Fig. 3.1: Plot of $x(n)$ and $y(n)$

4 Z-transform

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

$$X(z) = \mathcal{Z}\{x(n)\} = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{4.1}$$

Show that

$$\mathcal{Z}\{x(n-1)\} = z^{-1}X(z) \tag{4.2}$$

and find

$$\mathcal{Z}\{x(n-k)\} \tag{4.3}$$

Solution: From (4.1),

$$\mathcal{Z}\{x(n-k)\} = \sum_{n=-\infty}^{\infty} x(n-k)z^{-n} \tag{4.4}$$

$$= \sum_{n=-\infty}^{\infty} x(n)z^{-n-k} \tag{4.5}$$

$$= z^{-k} \sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{4.6}$$

$$= z^{-k}X(z) \tag{4.7}$$

Putting $k = 1$ gives (4.2). For the given $x(n)$, we have

$$X(z) = 1 + 2z^{-1} + 3z^{-2} + 4z^{-3} + 2z^{-4} + z^{-5} \tag{4.8}$$

$$\implies \mathcal{Z}\{x(n-1)\} = z^{-1} + 2z^{-2} + 3z^{-3} + 4z^{-4} + 2z^{-5} + z^{-6} \tag{4.9}$$

$$= z^{-1}X(z) \tag{4.10}$$

4.2 Find

$$H(z) = \frac{Y(z)}{X(z)} \tag{4.11}$$

from (3.2) assuming that the Z-transform is a linear operation.
Solution: Applying (4.7) in (3.2),

$$Y(z) + \frac{1}{2}z^{-1}Y(z) = X(z) + z^{-2}X(z) \tag{4.12}$$

$$\implies \frac{Y(z)}{X(z)} = \frac{1 + z^{-2}}{1 + \frac{1}{2}z^{-1}} \tag{4.13}$$

4.3 Find the Z transform of

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.14}$$

and show that the Z-transform of

$$u(n) = \begin{cases} 1 & n \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.15}$$

is

$$U(z) = \frac{1}{1 - z^{-1}}, \quad |z| > 1 \tag{4.16}$$

Solution: We see using (4.14) that

$$\mathcal{Z}\{\delta(n)\} = \delta(0) = 1 \tag{4.17}$$

and from (4.15),

$$U(z) = \sum_{n=0}^{\infty} z^{-n} \tag{4.18}$$

$$= \frac{1}{1 - z^{-1}}, \quad |z| > 1 \tag{4.19}$$

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

4.4 Show that

$$a^n u(n) \overset{\mathcal{Z}}{\rightleftharpoons} \frac{1}{1 - az^{-1}} \quad |z| > |a| \tag{4.20}$$

Solution:

$$a^n u(n) \overset{\mathcal{Z}}{\rightleftharpoons} \sum_{n=0}^{\infty} \left(az^{-1}\right)^n \tag{4.21}$$

$$= \frac{1}{1 - az^{-1}} \quad |z| > |a| \tag{4.22}$$

4.5 Let

$$H\left(e^{\jmath\omega}\right) = H\left(z = e^{\jmath\omega}\right). \tag{4.23}$$

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

Solution: The following code plots Fig. (4.1).

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/4_5.py
```

The figure can be generated using

```
$ python3 4_5.py
```

Using (4.13), we observe that $|H\left(e^{\jmath\omega}\right)|$ is given by

$$|H\left(e^{\jmath\omega}\right)| = \left|\frac{1 + e^{-2\jmath\omega}}{1 + \frac{1}{2}e^{-\jmath\omega}}\right| \tag{4.24}$$

$$= \sqrt{\frac{(1 + \cos 2\omega)^2 + (\sin 2\omega)^2}{\left(1 + \frac{1}{2}\cos\omega\right)^2 + \left(\frac{1}{2}\sin\omega\right)^2}} \tag{4.25}$$

$$= \sqrt{\frac{2\left(1 + \cos 2\omega\right)}{\frac{5}{4} + \cos\omega}} \tag{4.26}$$

$$= \sqrt{\frac{2\left(2\cos^2\omega\right)}{\frac{5}{4} + \cos\omega}} \tag{4.27}$$

$$= \frac{4|\cos\omega|}{\sqrt{5 + 4\cos\omega}} \tag{4.28}$$

Thus,

$$\left|H\left(e^{\jmath(\omega+2\pi)}\right)\right| = \frac{4|\cos\left(\omega + 2\pi\right)|}{\sqrt{5 + 4\cos\left(\omega + 2\pi\right)}} \tag{4.29}$$

$$= \frac{4|\cos\omega|}{\sqrt{5 + 4\cos\omega}} \tag{4.30}$$

$$= |H\left(e^{\jmath\omega}\right)| \tag{4.31}$$

and so its fundamental period is 2π.

4.6 Express $h(n)$ in terms of $H(e^{\jmath\omega})$.

Solution: We have,

$$H(e^{\jmath\omega}) = \sum_{k=-\infty}^{\infty} h(k)e^{-\jmath\omega k} \tag{4.32}$$

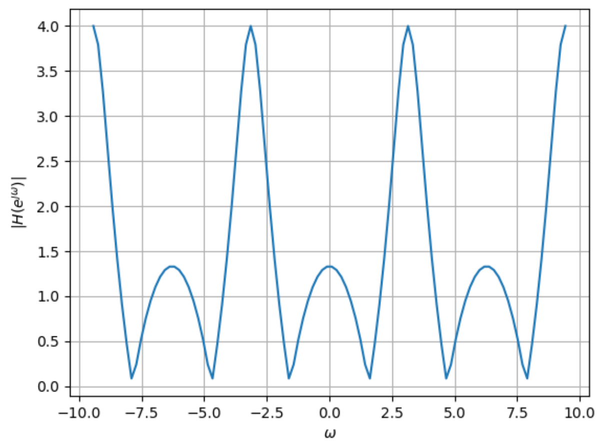

Fig. 4.1: Plot of $|H\left(e^{\jmath\omega}\right)|$ against ω

However,

$$\int_{-\pi}^{\pi} e^{\jmath\omega(n-k)} d\omega = \begin{cases} 2\pi & n = k \\ 0 & \text{otherwise} \end{cases} \tag{4.33}$$

and so,

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{\jmath\omega})e^{j\omega n} d\omega \tag{4.34}$$

$$= \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\int_{-\pi}^{\pi} h(k)e^{\jmath\omega(n-k)} d\omega \tag{4.35}$$

$$= \frac{1}{2\pi}2\pi h(n) = h(n) \tag{4.36}$$

which is known as the Inverse Discrete Fourier Transform. Thus,

$$h(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H(e^{\jmath\omega})e^{\jmath\omega n} d\omega \tag{4.37}$$

$$= \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{1 + e^{-2\jmath\omega}}{1 + \frac{1}{2}e^{-\jmath\omega}} e^{\jmath\omega n} d\omega \tag{4.38}$$

5 Impulse Response

5.1 Using long division, compute $h(n)$ for $n < 5$ from $H(z)$.

Solution: We substitute $x := z^{-1}$, and perform the long division.

$$\begin{array}{r@{\,}r} & 2x - 4 \\ \cline{2-2} \frac{1}{2}x + 1\big) & x^2 \qquad + 1 \\ & -x^2 - 2x \qquad \\ \cline{2-2} & -2x + 1 \\ & 2x + 4 \\ \cline{2-2} & 5 \end{array}$$

Thus,

$$H(z) = -4 + 2z^{-1} + \frac{5}{1+\frac{1}{2}z^{-1}} \tag{5.1}$$

$$= -4 + 2z^{-1} + 5\sum_{n=0}^{\infty}\left(-\frac{1}{2}\right)^n z^{-n} \tag{5.2}$$

$$= 1 - \frac{1}{2}z^{-1} + 5\sum_{n=2}^{\infty}\left(-\frac{1}{2}\right)^n z^{-n} \tag{5.3}$$

$$= \sum_{n=0}^{\infty}\left(-\frac{1}{2}\right)^n z^{-n} + 4\sum_{n=2}^{\infty}\left(-\frac{1}{2}\right)^n z^{-n} \tag{5.4}$$

$$= \sum_{n=-\infty}^{\infty} u(n)\left(-\frac{1}{2}\right)^n z^{-n} + \sum_{n=-\infty}^{\infty} u(n-2)\left(-\frac{1}{2}\right)^{n-2} z^{-n} \tag{5.5}$$

Therefore, from (4.1),

$$h(n) = \left(-\frac{1}{2}\right)^n u(n) + \left(-\frac{1}{2}\right)^{n-2} u(n-2) \tag{5.6}$$

5.2 Find an expression for $h(n)$ using $H(z)$, given that

$$h(n) \overset{\mathcal{Z}}{\rightleftharpoons} H(z) \tag{5.7}$$

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 (3.2).
Solution: From (4.13),

$$H(z) = \frac{1}{1+\frac{1}{2}z^{-1}} + \frac{z^{-2}}{1+\frac{1}{2}z^{-1}} \tag{5.8}$$

$$\implies h(n) = \left(-\frac{1}{2}\right)^n u(n) + \left(-\frac{1}{2}\right)^{n-2} u(n-2) \tag{5.9}$$

using (4.20) and (4.7).

5.3 Sketch $h(n)$. Is it bounded? Convergent?
Solution: The following code plots Fig. (5.1).

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/5_2.py
```

and execute it using

```
$ python3 5_2.py
```

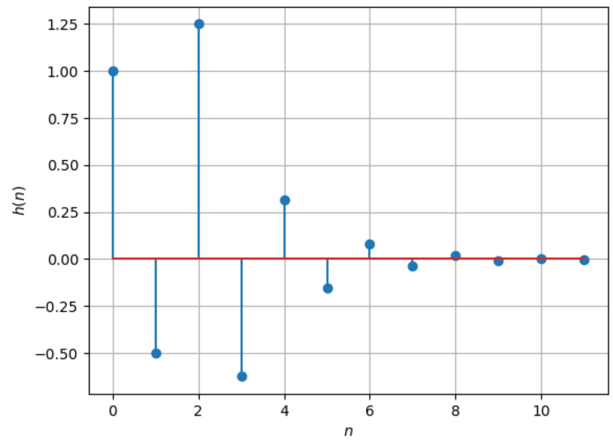

Fig. 5.1: $h(n)$ as the inverse of $H(z)$

We see that $h(n)$ is bounded. For large n,

$$h(n) = \left(-\frac{1}{2}\right)^n + \left(-\frac{1}{2}\right)^{n-2} \tag{5.10}$$

$$= \left(-\frac{1}{2}\right)^n (4+1) = 5\left(-\frac{1}{2}\right)^n \tag{5.11}$$

$$\implies \left|\frac{h(n+1)}{h(n)}\right| = \frac{1}{2} \tag{5.12}$$

and therefore, $\lim_{n\to\infty}\left|\frac{h(n+1)}{h(n)}\right| = \frac{1}{2} < 1$. Hence, we see that $h(n)$ converges.

5.4 The system with $h(n)$ is defined to be stable if

$$\sum_{n=-\infty}^{\infty} h(n) < \infty \tag{5.13}$$

Is the system defined by (3.2) stable for the impulse response in (5.7)?
Solution: Note that

$$\sum_{n=-\infty}^{\infty} h(n) = \sum_{n=-\infty}^{\infty}\left(-\frac{1}{2}\right)^n u(n) + \left(-\frac{1}{2}\right)^{n-2} u(n-2) \tag{5.14}$$

$$= 2\left(\frac{1}{1+\frac{1}{2}}\right) = \frac{4}{3} \tag{5.15}$$

Thus, the given system is stable. The limit is verified at

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/5_3.py
```

and the code can be run using

```
$ python3 5_3.py
```

5.5 Compute and sketch $h(n)$ using

$$h(n) + \frac{1}{2}h(n-1) = \delta(n) + \delta(n-2), \tag{5.16}$$

This is the definition of $h(n)$.
Solution: The following code plots Fig. (5.2). Note that this is the same as Fig. (5.1).

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/5_4.py
```

and executed using

```
$ python3 5_4.py
```

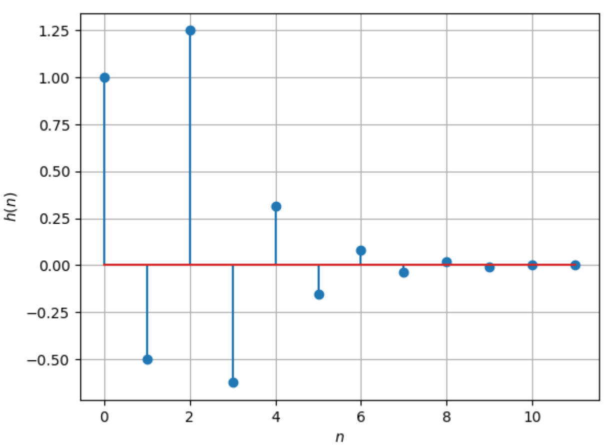

Fig. 5.2: $h(n)$ as the inverse of $H(z)$

5.6 Compute

$$y(n) = x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) \tag{5.17}$$

Comment. The operation in (5.17) is known as *convolution*.
Solution: The following code plots Fig. (5.3). Note that this is the same as $y(n)$ in Fig. (3.1).

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/5_5.py
```

and executed using

```
$ python3 5_5.py
```

We use Toeplitz matrices for convolution

$$\mathbf{y} = \mathbf{x} \circledast \mathbf{h} \tag{5.18}$$

$$\mathbf{y} = \begin{pmatrix} h_1 & 0 & . & . & . & 0 \\ h_2 & h_1 & . & . & . & 0 \\ h_3 & h_2 & h_1 & . & . & 0 \\ . & . & . & . & . & . \\ 0 & . & . & h_3 & h_2 & h_1 \\ 0 & . & . & . & h_2 & h_1 \\ 0 & . & . & . & 0 & h_1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \tag{5.19}$$

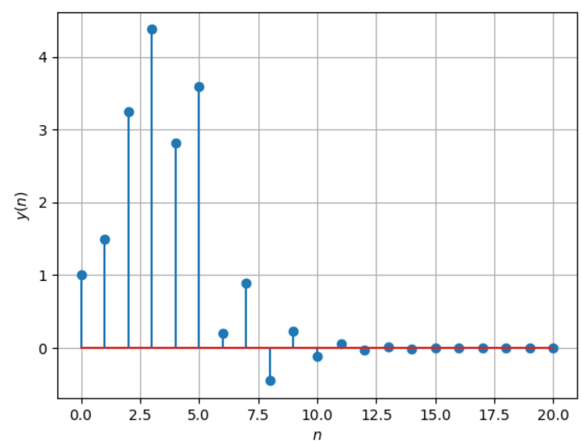

Fig. 5.3: $y(n)$ from the definition

5.7 Show that

$$y(n) = \sum_{k=-\infty}^{\infty} x(n-k)h(k) \tag{5.20}$$

Solution: From (5.17), we substitute $k := n-k$ to get

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)\, h(n-k) \tag{5.21}$$

$$= \sum_{n-k=-\infty}^{\infty} x(n-k)\, h(k) \tag{5.22}$$

$$= \sum_{k=-\infty}^{\infty} x(n-k)\, h(k) \tag{5.23}$$

6 DFT and FFT

6.1 Compute

$$X(k) \triangleq \sum_{n=0}^{N-1} x(n) e^{-\jmath 2\pi kn/N}, \quad k = 0, 1, \dots, N-1 \tag{6.1}$$

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

6.2 Compute

$$Y(k) = X(k)H(k) \tag{6.2}$$

6.3 Compute

$$y(n) = \frac{1}{N}\sum_{k=0}^{N-1} Y(k)\cdot e^{\jmath 2\pi kn/N}, \quad n = 0, 1, \ldots, N-1 \tag{6.3}$$

Solution: The following code plots Fig. (6.1) and computes $X(k)$ and $Y(k)$. Note that this is the same as $y(n)$ in Fig. (3.1). Download the code using

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_3.py
```

and execute it using

```
$ python3 6_3.py
```

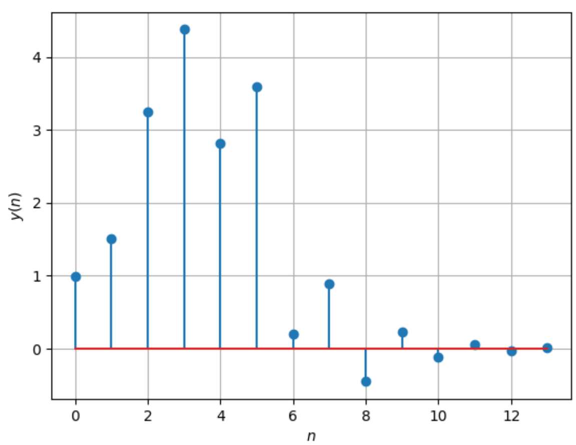

Fig. 6.1: $y(n)$ from the DFT

6.4 Repeat the previous exercise by computing $X(k), H(k)$ and $y(n)$ through FFT and IFFT.
Solution: Download the code from

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_4.py
```

and execute it using

```
$ python3 6_4.py
```

The values of $y(n)$ using all the three methods have been plotted on one stem plot for convenience. Note that there is very little difference in the values of $y(n)$.

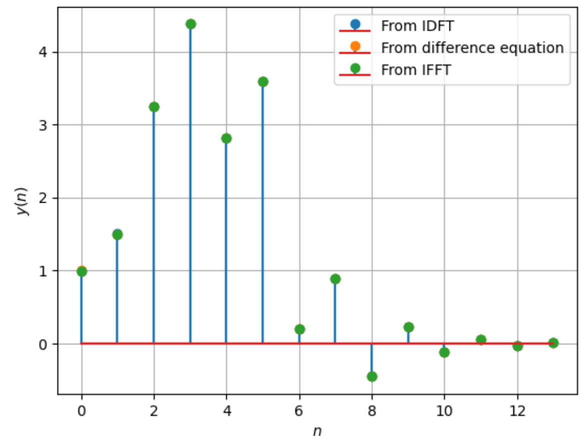

Fig. 6.2: $y(n)$ using FFT and IFFT

6.5 Wherever possible, express all the above equations as matrix equations.
Solution: We use the DFT Matrix, where $\omega = e^{-\frac{j2k\pi}{N}}$, which is given by

$$\mathbf{W} = \begin{pmatrix} \omega^0 & \omega^0 & \ldots & \omega^0 \\ \omega^0 & \omega^1 & \ldots & \omega^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \omega^0 & \omega^{N-1} & \ldots & \omega^{(N-1)(N-1)} \end{pmatrix} \tag{6.4}$$

i.e. $W_{jk} = \omega^{jk}$, $0 \le j, k < N$. Hence, we can write any DFT equation as

$$\mathbf{X} = \mathbf{W}\mathbf{x} = \mathbf{x}\mathbf{W} \tag{6.5}$$

where

$$\mathbf{x} = \begin{pmatrix} x(0) \\ x(1) \\ \vdots \\ x(n-1) \end{pmatrix} \tag{6.6}$$

Using (6.3), the inverse Fourier Transform is given by

$$\mathbf{x} = \mathcal{F}^{-1}(\mathbf{X}) = \mathbf{W}^{-1}\mathbf{X} = \frac{1}{N}\mathbf{W^H}\mathbf{X} = \frac{1}{N}\mathbf{X}\mathbf{W^H} \tag{6.7}$$

$$\implies \mathbf{W}^{-1} = \frac{1}{N}\mathbf{W^H} \tag{6.8}$$

where H denotes hermitian operator. We can rewrite (6.2) using the element-wise multiplication operator as

$$\mathbf{Y} = \mathbf{H}\cdot\mathbf{X} = (\mathbf{W}\mathbf{h})\cdot(\mathbf{W}\mathbf{x}) \tag{6.9}$$

The plot of $y(n)$ using the DFT matrix in Fig. (6.3) is the same as $y(n)$ in Fig. (3.1).

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_5.py
```

and run it using

```
$ python3 6_5.py
```

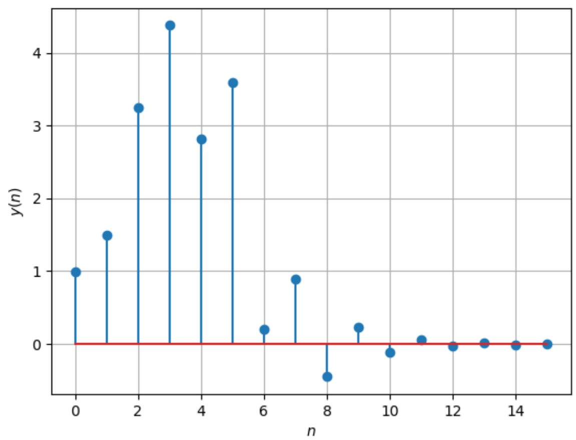

Fig. 6.3: $y(n)$ using the DFT matrix

6.6 Implement your own FFT and IFFT routines and verify your routine by plotting $y(n)$.
Solution: The code can be downloaded using

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_6.py
```

and can be run using

```
$ python3 6_6.py
```

The plot is shown in Fig. (6.4)

6.7 Find the time complexities of computing $y(n)$ using FFT/IFFT and convolution.
Solution: The C code for finding the running times of these three algorithms can be downloaded from

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_7.c
```

The C code is compiled and run using

```
$ gcc -lm -O2 -Wall -g 6_7.c
$ ./a.out
```

This code generates three text files that are used to plot the runtimes of the algorithms in the following Python codes

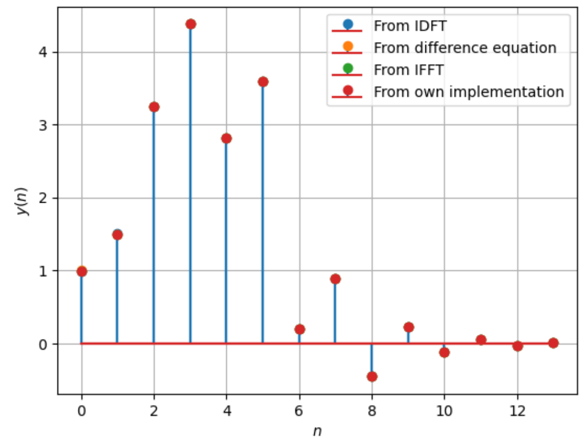

Fig. 6.4: $y(n)$ using own implementation of FFT and IFFT

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_7_1.py
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/6_7_2.py
```

Figs. (6.5) and (6.6) are generated by executing the codes.

```
$ python3 6_7_1.py
$ python3 6_7_2.py
```

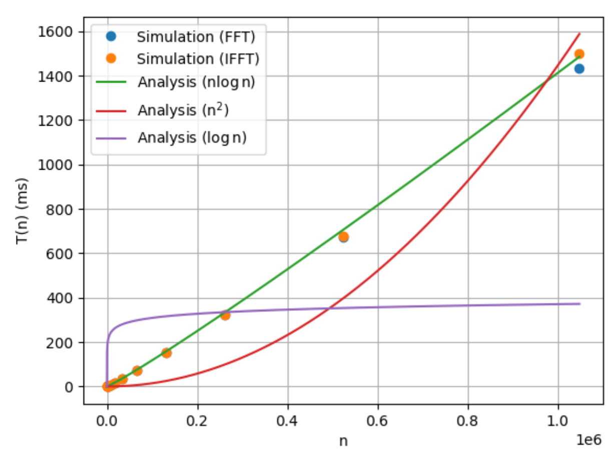

Fig. 6.5: The worst-case complexity of FFT/IFFT is $O(n \log n)$

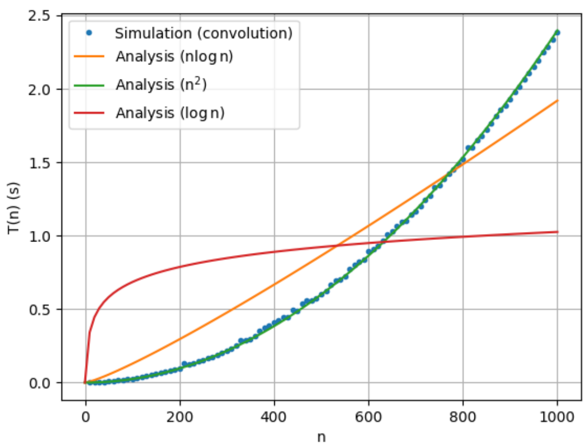

Fig. 6.6: The worst case complexity of convolution is $O(n^2)$

7 FFT

7.1. The DFT of $x(n)$ is given by

$$X(k) \triangleq \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \tag{7.1}$$

7.2. Let

$$W_N = e^{-j2\pi/N} \tag{7.2}$$

Then the N-point *DFT matrix* is defined as

$$\mathbf{F}_N = \left[W_N^{mn}\right], \quad 0 \le m, n \le N-1 \tag{7.3}$$

where W_N^{mn} are the elements of $\mathbf{F}_N$.

7.3. Let

$$\mathbf{I}_4 = \begin{pmatrix} \mathbf{e}_4^1 & \mathbf{e}_4^2 & \mathbf{e}_4^3 & \mathbf{e}_4^4 \end{pmatrix} \tag{7.4}$$

be the 4×4 identity matrix. Then the 4 point *DFT permutation matrix* is defined as

$$\mathbf{P}_4 = \begin{pmatrix} \mathbf{e}_4^1 & \mathbf{e}_4^3 & \mathbf{e}_4^2 & \mathbf{e}_4^4 \end{pmatrix} \tag{7.5}$$

7.4. The 4 point *DFT diagonal matrix* is defined as

$$\mathbf{D}_4 = diag\begin{pmatrix} W_8^0 & W_8^1 & W_8^2 & W_8^3 \end{pmatrix} \tag{7.6}$$

7.5. Show that

$$W_N^2 = W_{N/2} \tag{7.7}$$

Solution: We write

$$W_N^2 = \left(e^{-\frac{j2\pi}{N}}\right)^2 = e^{-\frac{j2\pi}{N/2}} = W_{N/2} \tag{7.8}$$

7.6. Show that

$$\mathbf{F}_4 = \begin{bmatrix} \mathbf{I}_2 & \mathbf{D}_2 \\ \mathbf{I}_2 & -\mathbf{D}_2 \end{bmatrix} \begin{bmatrix} \mathbf{F}_2 & 0 \\ 0 & \mathbf{F}_2 \end{bmatrix} \mathbf{P}_4 \tag{7.9}$$

Solution: Observe that for $n \in \mathbb{N}$, $W_4^{4n} = 1$ and $W_4^{4n+2} = -1$. Using (7.7),

$$\mathbf{D}_2\mathbf{F}_2 = \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} W_2^0 & W_2^0 \\ W_2^0 & W_2^1 \end{bmatrix} \tag{7.10}$$

$$= \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} W_4^0 & W_4^0 \\ W_4^0 & W_4^2 \end{bmatrix} \tag{7.11}$$

$$= \begin{bmatrix} W_4^0 & W_4^0 \\ W_4^1 & W_4^3 \end{bmatrix} \tag{7.12}$$

$$\implies -\mathbf{D}_2\mathbf{F}_2 = \begin{bmatrix} W_4^2 & W_4^6 \\ W_4^3 & W_4^9 \end{bmatrix} \tag{7.13}$$

and

$$\mathbf{F}_2 = \begin{pmatrix} W_2^0 & W_2^0 \\ W_2^0 & W_2^1 \end{pmatrix} \tag{7.14}$$

$$= \begin{pmatrix} W_4^0 & W_4^0 \\ W_4^0 & W_4^2 \end{pmatrix} \tag{7.15}$$

Hence,

$$\mathbf{W}_4 = \begin{pmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^2 & W_4^1 & W_4^3 \\ W_4^0 & W_4^4 & W_4^2 & W_4^6 \\ W_4^0 & W_4^6 & W_4^3 & W_4^9 \end{pmatrix} \tag{7.16}$$

$$= \begin{bmatrix} \mathbf{I}_2\mathbf{F}_2 & \mathbf{D}_2 F_2 \\ \mathbf{I}_2\mathbf{F}_2 & -\mathbf{D}_2 F_2 \end{bmatrix} \tag{7.17}$$

$$= \begin{bmatrix} \mathbf{I}_2 & \mathbf{D}_2 \\ \mathbf{I}_2 & \mathbf{D}_2 \end{bmatrix} \begin{bmatrix} \mathbf{F}_2 & 0 \\ 0 & \mathbf{F}_2 \end{bmatrix} \tag{7.18}$$

Multiplying (7.18) by $\mathbf{P}_4$ on both sides, and noting that $\mathbf{W}_4\mathbf{P}_4 = \mathbf{F}_4$ gives us (7.9).

7.7. Show that

$$\mathbf{F}_N = \begin{bmatrix} \mathbf{I}_{N/2} & \mathbf{D}_{N/2} \\ \mathbf{I}_{N/2} & -\mathbf{D}_{N/2} \end{bmatrix} \begin{bmatrix} \mathbf{F}_{N/2} & 0 \\ 0 & \mathbf{F}_{N/2} \end{bmatrix} \mathbf{P}_N \tag{7.19}$$

Solution: Observe that for even N and letting $\mathbf{f}_N^i$ denote the i^{th} column of $\mathbf{F}_N$, from (7.12) and (7.13),

$$\begin{pmatrix} \mathbf{D}_{N/2}\mathbf{F}_{N/2} \\ -\mathbf{D}_{N/2}\mathbf{F}_{N/2} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_N^2 & \mathbf{f}_N^4 & \ldots & \mathbf{f}_N^N \end{pmatrix} \tag{7.20}$$

and

$$\begin{pmatrix} \mathbf{I}_{N/2}\mathbf{F}_{N/2} \\ \mathbf{I}_{N/2}\mathbf{F}_{N/2} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_N^1 & \mathbf{f}_N^3 & \ldots & \mathbf{f}_N^{N-1} \end{pmatrix} \tag{7.21}$$

Thus,

$$\begin{bmatrix}\mathbf{I}_2\mathbf{F}_2 & \mathbf{D}_2\mathbf{F}_2\\ \mathbf{I}_2\mathbf{F}_2 & -\mathbf{D}_2\mathbf{F}_2\end{bmatrix} = \begin{bmatrix}\mathbf{I}_{N/2} & \mathbf{D}_{N/2}\\ \mathbf{I}_{N/2} & -\mathbf{D}_{N/2}\end{bmatrix}\begin{bmatrix}\mathbf{F}_{N/2} & 0\\ 0 & \mathbf{F}_{N/2}\end{bmatrix}$$
$$= \begin{pmatrix}\mathbf{f}_N^1 & \dots & \mathbf{f}_N^{N-1} & \mathbf{f}_N^2 & \dots & \mathbf{f}_N^N\end{pmatrix} \tag{7.22}$$

and so,

$$\begin{bmatrix}\mathbf{I}_{N/2} & \mathbf{D}_{N/2}\\ \mathbf{I}_{N/2} & -\mathbf{D}_{N/2}\end{bmatrix}\begin{bmatrix}\mathbf{F}_{N/2} & 0\\ 0 & \mathbf{F}_{N/2}\end{bmatrix}\mathbf{P}_N$$
$$= \begin{pmatrix}\mathbf{f}_N^1 & \mathbf{f}_N^2 & \dots & \mathbf{f}_N^N\end{pmatrix} = \mathbf{F}_N \tag{7.23}$$

7.8. Find

$$\mathbf{P}_4\mathbf{x} \tag{7.24}$$

Solution: We have,

$$\mathbf{P}_4\mathbf{x} = \begin{pmatrix}\mathbf{e}_4^1 & \mathbf{e}_4^3 & \mathbf{e}_4^2 & \mathbf{e}_4^4\end{pmatrix}\begin{pmatrix}x(0)\\x(1)\\x(2)\\x(3)\end{pmatrix} = \begin{pmatrix}x(0)\\x(2)\\x(1)\\x(3)\end{pmatrix} \tag{7.25}$$

7.9. Show that

$$\mathbf{X} = \mathbf{F}_N\mathbf{x} \tag{7.26}$$

where $\mathbf{x}, \mathbf{X}$ are the vector representations of $x(n), X(k)$ respectively.

Solution: Writing the terms of X,

$$X(0) = x(0) + x(1) + \ldots + x(N-1) \tag{7.27}$$

$$X(1) = x(0) + x(1)e^{-\frac{\jmath 2\pi}{N}} + \ldots + \\ + x(N-1)e^{-\frac{\jmath 2(N-1)\pi}{N}} \tag{7.28}$$

$$\vdots$$

$$X(N-1) = x(0) + x(1)e^{-\frac{\jmath 2(N-1)\pi}{N}} + \ldots + \\ + x(N-1)e^{-\frac{\jmath 2(N-1)(N-1)\pi}{N}} \tag{7.29}$$

Clearly, the term in the m^{th} row and n^{th} column is given by ($0 \le m \le N-1$ and $0 \le n \le N-1$)

$$T_{mn} = x(n)e^{-\frac{\jmath 2mn\pi}{N}} \tag{7.30}$$

and so, we can represent each of these terms as a matrix product

$$\mathbf{X} = \mathbf{F}_N\mathbf{x} \tag{7.31}$$

where $\mathbf{F}_N = \left[e^{-\frac{-\jmath 2mn\pi}{N}}\right]_{mn}$ for $0 \le m \le N-1$ and $0 \le n \le N-1$.

7.10. Derive the following Step-by-step visualisation of 8-point FFTs into 4-point FFTs and so on

$$\begin{bmatrix}X(0)\\X(1)\\X(2)\\X(3)\end{bmatrix} = \begin{bmatrix}X_1(0)\\X_1(1)\\X_1(2)\\X_1(3)\end{bmatrix} + \begin{bmatrix}W_8^0 & 0 & 0 & 0\\ 0 & W_8^1 & 0 & 0\\ 0 & 0 & W_8^2 & 0\\ 0 & 0 & 0 & W_8^3\end{bmatrix}\begin{bmatrix}X_2(0)\\X_2(1)\\X_2(2)\\X_2(3)\end{bmatrix} \tag{7.32}$$

$$\begin{bmatrix}X(4)\\X(5)\\X(6)\\X(7)\end{bmatrix} = \begin{bmatrix}X_1(0)\\X_1(1)\\X_1(2)\\X_1(3)\end{bmatrix} - \begin{bmatrix}W_8^0 & 0 & 0 & 0\\ 0 & W_8^1 & 0 & 0\\ 0 & 0 & W_8^2 & 0\\ 0 & 0 & 0 & W_8^3\end{bmatrix}\begin{bmatrix}X_2(0)\\X_2(1)\\X_2(2)\\X_2(3)\end{bmatrix} \tag{7.33}$$

4-point FFTs into 2-point FFTs

$$\begin{bmatrix}X_1(0)\\X_1(1)\end{bmatrix} = \begin{bmatrix}X_3(0)\\X_3(1)\end{bmatrix} + \begin{bmatrix}W_4^0 & 0\\ 0 & W_4^1\end{bmatrix}\begin{bmatrix}X_4(0)\\X_4(1)\end{bmatrix} \tag{7.34}$$

$$\begin{bmatrix}X_1(2)\\X_1(3)\end{bmatrix} = \begin{bmatrix}X_3(0)\\X_3(1)\end{bmatrix} - \begin{bmatrix}W_4^0 & 0\\ 0 & W_4^1\end{bmatrix}\begin{bmatrix}X_4(0)\\X_4(1)\end{bmatrix} \tag{7.35}$$

$$\begin{bmatrix}X_2(0)\\X_2(1)\end{bmatrix} = \begin{bmatrix}X_5(0)\\X_5(1)\end{bmatrix} + \begin{bmatrix}W_4^0 & 0\\ 0 & W_4^1\end{bmatrix}\begin{bmatrix}X_6(0)\\X_6(1)\end{bmatrix} \tag{7.36}$$

$$\begin{bmatrix}X_2(2)\\X_2(3)\end{bmatrix} = \begin{bmatrix}X_5(0)\\X_5(1)\end{bmatrix} - \begin{bmatrix}W_4^0 & 0\\ 0 & W_4^1\end{bmatrix}\begin{bmatrix}X_6(0)\\X_6(1)\end{bmatrix} \tag{7.37}$$

$$P_8\begin{bmatrix}x(0)\\x(1)\\x(2)\\x(3)\\x(4)\\x(5)\\x(6)\\x(7)\end{bmatrix} = \begin{bmatrix}x(0)\\x(2)\\x(4)\\x(6)\\x(1)\\x(3)\\x(5)\\x(7)\end{bmatrix} \tag{7.38}$$

$$P_4\begin{bmatrix}x(0)\\x(2)\\x(4)\\x(6)\end{bmatrix} = \begin{bmatrix}x(0)\\x(4)\\x(2)\\x(6)\end{bmatrix} \tag{7.39}$$

$$P_4\begin{bmatrix}x(1)\\x(3)\\x(5)\\x(7)\end{bmatrix} = \begin{bmatrix}x(1)\\x(5)\\x(3)\\x(7)\end{bmatrix} \tag{7.40}$$

Therefore,

$$\begin{bmatrix}X_3(0)\\X_3(1)\end{bmatrix} = F_2\begin{bmatrix}x(0)\\x(4)\end{bmatrix} \tag{7.41}$$

$$\begin{bmatrix}X_4(0)\\X_4(1)\end{bmatrix} = F_2\begin{bmatrix}x(2)\\x(6)\end{bmatrix} \tag{7.42}$$

$$\begin{bmatrix}X_5(0)\\X_5(1)\end{bmatrix} = F_2\begin{bmatrix}x(1)\\x(5)\end{bmatrix} \tag{7.43}$$

$$\begin{bmatrix} X_6(0) \\ X_6(1) \end{bmatrix} = F_2 \begin{bmatrix} x(3) \\ x(7) \end{bmatrix} \tag{7.44}$$

Solution: We write out the values of performing an 8-point FFT on $\mathbf{x}$ as follows.

$$X(k) = \sum_{n=0}^{7} x(n) e^{-\frac{j2kn\pi}{8}} \tag{7.45}$$

$$= \sum_{n=0}^{3} \left(x(2n) e^{-\frac{j2kn\pi}{4}} + e^{-\frac{j2k\pi}{8}} x(2n+1) e^{-\frac{j2kn\pi}{4}} \right) \tag{7.46}$$

$$= X_1(k) + e^{-\frac{j2k\pi}{4}} X_2(k) \tag{7.47}$$

where $\mathbf{X}_1$ is the 4-point FFT of the even-numbered terms and $\mathbf{X}_2$ is the 4-point FFT of the odd numbered terms. Noticing that for $k \geq 4$,

$$X_1(k) = X_1(k-4) \tag{7.48}$$

$$e^{-\frac{j2k\pi}{8}} = -e^{-\frac{j2(k-4)\pi}{8}} \tag{7.49}$$

we can now write out $X(k)$ in matrix form as in (7.32) and (7.33). We also need to solve the two 4-point FFT terms so formed.

$$X_1(k) = \sum_{n=0}^{3} x_1(n) e^{-\frac{j2kn\pi}{8}} \tag{7.50}$$

$$= \sum_{n=0}^{1} \left(x_1(2n) e^{-\frac{j2kn\pi}{4}} + e^{-\frac{j2k\pi}{8}} x_2(2n+1) e^{-\frac{j2kn\pi}{4}} \right) \tag{7.51}$$

$$= X_3(k) + e^{-\frac{j2k\pi}{4}} X_4(k) \tag{7.52}$$

using $x_1(n) = x(2n)$ and $x_2(n) = x(2n+1)$. Thus we can write the 2-point FFTs

$$\begin{bmatrix} X_3(0) \\ X_3(1) \end{bmatrix} = F_2 \begin{bmatrix} x(0) \\ x(4) \end{bmatrix} \tag{7.53}$$

$$\begin{bmatrix} X_4(0) \\ X_4(1) \end{bmatrix} = F_2 \begin{bmatrix} x(2) \\ x(6) \end{bmatrix} \tag{7.54}$$

Using a similar idea for the terms X_2,

$$\begin{bmatrix} X_5(0) \\ X_5(1) \end{bmatrix} = F_2 \begin{bmatrix} x(1) \\ x(5) \end{bmatrix} \tag{7.55}$$

$$\begin{bmatrix} X_6(0) \\ X_6(1) \end{bmatrix} = F_2 \begin{bmatrix} x(3) \\ x(7) \end{bmatrix} \tag{7.56}$$

But observe that from (7.25),

$$\mathbf{P}_8 \mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \tag{7.57}$$

$$\mathbf{P}_4 \mathbf{x}_1 = \begin{pmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \end{pmatrix} \tag{7.58}$$

$$\mathbf{P}_4 \mathbf{x}_2 = \begin{pmatrix} \mathbf{x}_5 \\ \mathbf{x}_6 \end{pmatrix} \tag{7.59}$$

where we define $x_3(k) = x(4k)$, $x_4(k) = x(4k+2)$, $x_5(k) = x(4k+1)$, and $x_6(k) = x(4k+3)$ for $k = 0, 1$.

7.11. For

$$\mathbf{x} = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 2 \\ 1 \end{pmatrix} \tag{7.60}$$

compte the DFT using (7.26)
Solution: Download the Python code from

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/7_11.py
```

and run it using

```
$ python3 7_11.py
```

7.12. Repeat the above exercise using the FFT after zero padding $\mathbf{x}$.

7.13. Write a C program to compute the 8-point FFT.
Solution: The C code for the above two problems can be downloaded from

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/7_13.c
```

Compile and run the code using

```
$ gcc −lm −Wall −O2 7_13.c
$ ./a.out
```

8 Exercises

Answer the following questions by looking at the python code in Problem 2.3.

8.1. The command

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

in Problem 2.3 is executed through the following difference equation

$$\sum_{m=0}^{M} a(m)\,y(n-m) = \sum_{k=0}^{N} b(k)\,x(n-k) \quad (8.1)$$

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: Download the source code by typing the next command

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/8_1.py
```

and run it using

```
$ python3 8_1.py
```

8.2. Repeat all the exercises in the previous sections for the above a and b.
Solution: For the given values, the difference equation is

$$\begin{aligned} &y(n) - (2.52)\,y(n-1) + (2.56)\,y(n-2) \\ &- (1.21)\,y(n-3) + (0.22)\,y(n-4) \\ &= \left(3.45\times10^{-3}\right)x(n) + \left(1.38\times10^{-2}\right)x(n-1) \\ &+ \left(2.07\times10^{-2}\right)x(n-2) + \left(1.38\times10^{-2}\right)x(n-3) \\ &+ \left(3.45\times10^{-3}\right)x(n-4) \end{aligned} \quad (8.2)$$

From (8.1), we see that the transfer function can be written as follows

$$H(z) = \frac{\sum_{k=0}^{N} b(k)z^{-k}}{\sum_{k=0}^{M} a(k)z^{-k}} \quad (8.3)$$

$$= \sum_i \frac{r(i)}{1-p(i)z^{-1}} + \sum_j k(j)z^{-j} \quad (8.4)$$

where $r(i)$, $p(i)$, are called residues and poles respectively of the partial fraction expansion of $H(z)$. $k(i)$ are the coefficients of the direct polynomial terms that might be left over. We can now take the inverse z-transform of (8.4) and get using (4.20),

$$h(n) = \sum_i r(i)[p(i)]^n u(n) + \sum_j k(j)\delta(n-j) \quad (8.5)$$

Substituting the values,

$$\begin{aligned} h(n) &= [(-0.24-0.71\jmath)(0.56+0.14\jmath)^n \\ &+ (-0.24+0.71\jmath)(0.56-0.14\jmath)^n \\ &+ (-0.25+0.12\jmath)(0.70+0.41\jmath)^n \\ &+ (-0.25-0.12\jmath)(0.70-0.41\jmath)^n]u(n) \\ &+ \left(1.6\times10^{-2}\right)\delta(n) \end{aligned} \quad (8.6)$$

$$\begin{aligned} \implies h(n) &= (1.5)(0.58)^n \cos(n\alpha_1 + \beta_1) \\ &+ (0.55)(0.81)^n \cos(n\alpha_2 + \beta_2) \\ &+ \left(1.6\times10^{-2}\right)\delta(n) \end{aligned} \quad (8.7)$$

where

$$\tan\alpha_1 = 0.25 \quad (8.8)$$
$$\tan\beta_1 = 2.96 \quad (8.9)$$
$$\tan\alpha_2 = 0.59 \quad (8.10)$$
$$\tan\beta_2 = -0.48 \quad (8.11)$$

The values $r(i)$, $p(i)$, $k(i)$ and thus the impulse response function are computed and plotted at

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/8_2_1.py
```

The filter frequency response is plotted at

```
$ wget https://github.com/Sathvikaaa/EE3900/
    blob/main/codes/8_2_2.py
```

Observe that for a series $t_n = r^n$, $\frac{t_{n+1}}{t_n} = r$. By the ratio test, t_n converges if $|r| < 1$. We observe that for all i, $|p(i)| < 1$ and so, as $h(n)$ is the sum of many convergent series, we see that $h(n)$ converges and is bounded. From (4.1),

$$\sum_{n=0}^{\infty} h(n) = H(1) = \frac{\sum_{k=0}^{N} b(k)}{\sum_{k=0}^{M} a(k)} = 1 < \infty \quad (8.12)$$

Therefore, the system is stable. From Fig. (8.1), $h(n)$ is negligible after $n \geq 64$, and we can apply a 64-bit FFT to get y(n). The following code uses the DFT matrix to generate $y(n)$ in Fig. (8.3).

```
t$ wget https://github.com/Sathvikaaa/EE3900
    /blob/main/codes/8_2_3.py
```

The codes can be run all at once by typing a small shell script

```
$ for file in 8_2_*.py; do python ${file};
    done
```

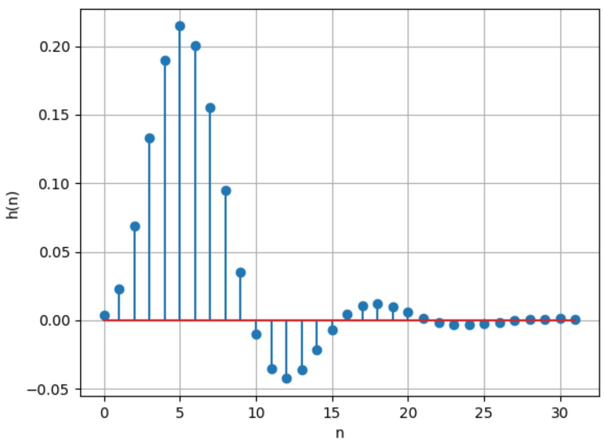

Fig. 8.1: Plot of $h(n)$

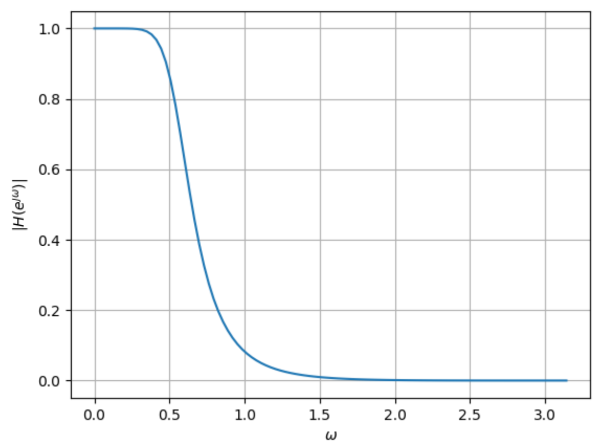

Fig. 8.2: Filter frequency response

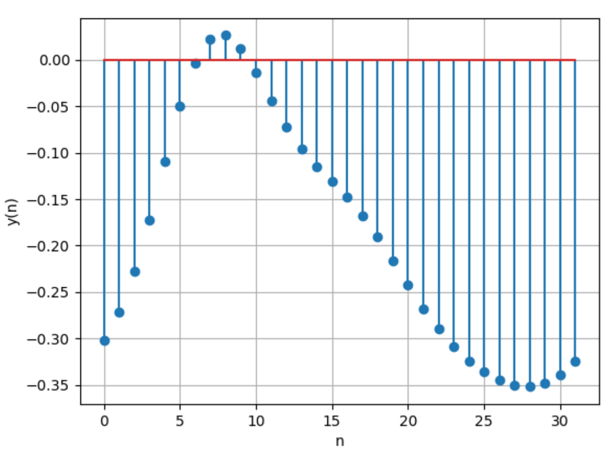

Fig. 8.3: Plot of $y(n)$

8.3. What is the sampling frequency of the input signal?
Solution: Sampling frequency f_s = 44.1 kHZ.

8.4. 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 4 kHz.

8.5. 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 7.