EE3025 - Assignment 1

Ch Pranay Prakash - EE18BTECH11009

Download all python codes from

```
https://github.com/pranayEE11009/EE3025_IDP/tree/main/Assignment_1/codes
```

and latex-tikz codes from

```
https://github.com/pranayEE11009/EE3025_IDP/tree/main/Assignment_1
```

1 Problem

1.1. Let

$$x(n) = \left\{ \underset{\uparrow}{1}, 2, 3, 4, 2, 1 \right\} \tag{1.1.1}$$

and

$$y(n) + \frac{1}{2}y(n-1) = x(n) + x(n-2)$$
$$y(n) = 0, n < 0 \tag{1.1.2}$$

Compute

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \tag{1.1.3}$$

and H(k) using h(n).

2 Solution

2.1. Since, We know that the Impulse response of an LTI system is the output of the system when an Unit Impulse signal is given as the input to the system.

Now, from equation (1.1.2) the impulse response of the system can be defined as,

$$h(n) + \frac{1}{2}h(n-1) = \delta(n) + \delta(n-2) \tag{2.1.1}$$

$$\implies h(n) = \delta(n) + \delta(n-2) - \frac{1}{2}h(n-1) \tag{2.1.2}$$

2.2. **Computing the DFT manually:**
The DFT of the Input Signal x(n) is given by :

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \tag{2.2.1}$$

Similarly, the DFT of the Impulse Response h(n) is given by :

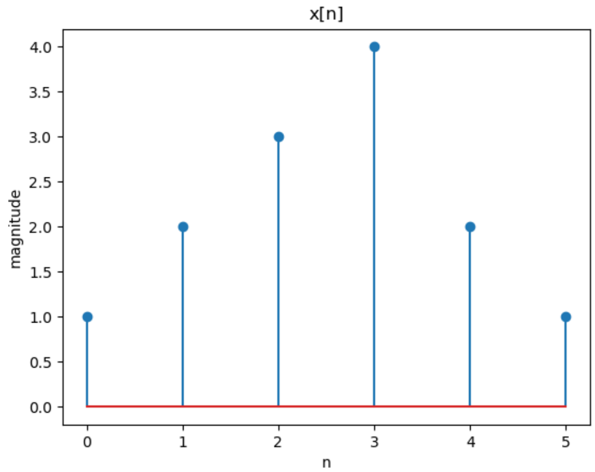
$$H(k) = \sum_{n=0}^{N-1} h(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \tag{2.2.2}$$

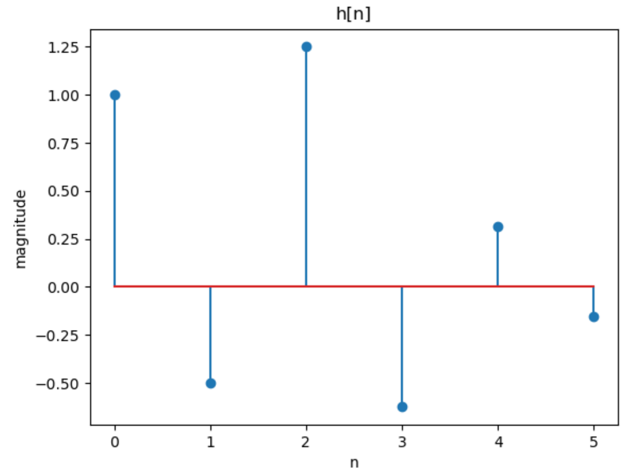
The following python code computes the DFT of x(n) and h(n).

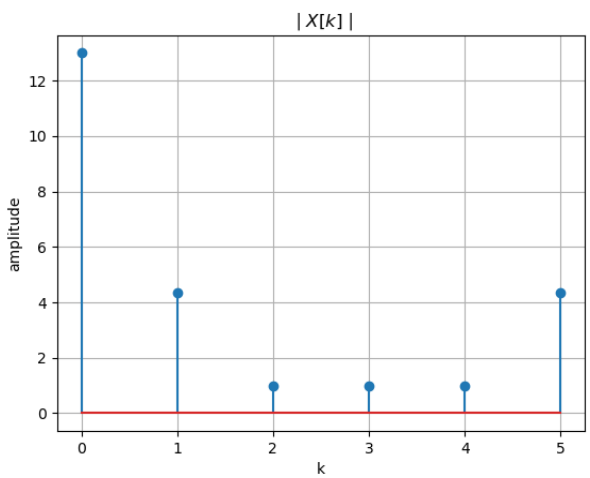
```
https://github.com/pranayEE11009/EE3025_IDP/blob/main/Assignment_1/codes/ee18btech11009.py
```

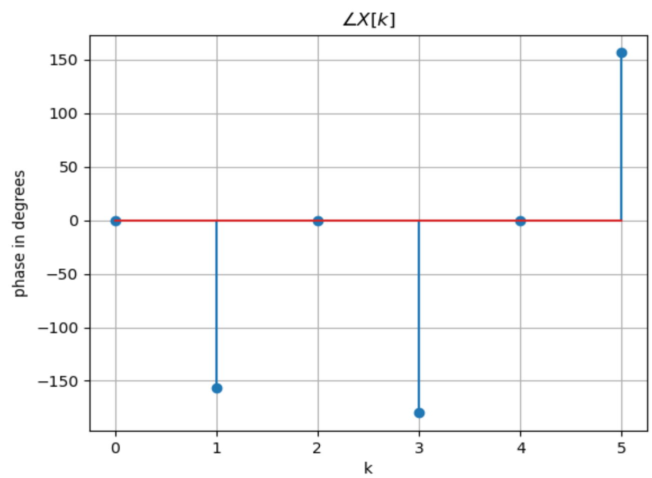
The plots are in

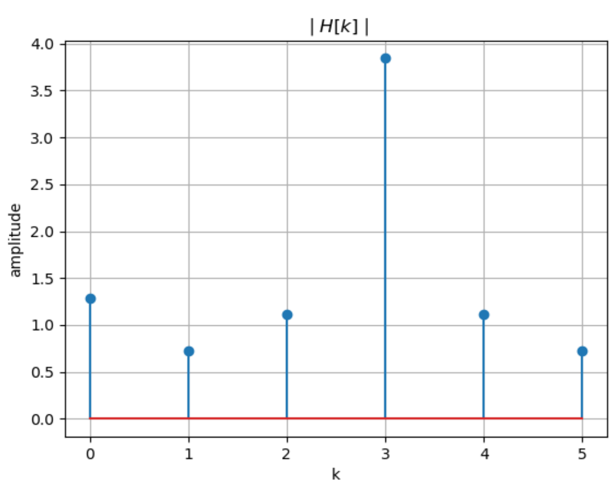
```
https://github.com/pranayEE11009/EE3025_IDP/tree/main/Assignment_1/figs
```

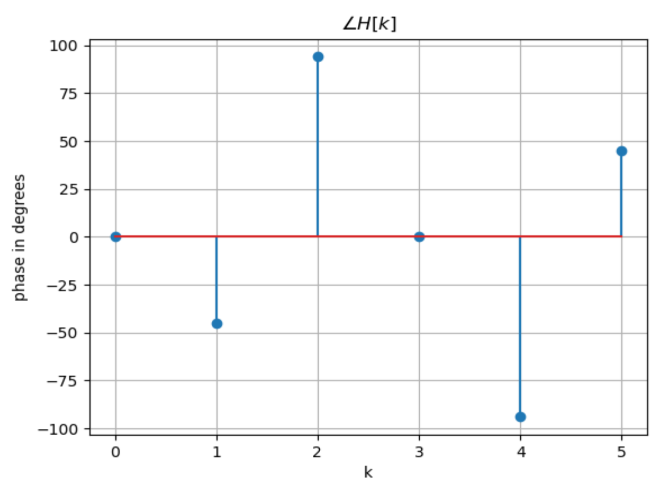

x[n]
magnitude
n
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0
0 1 2 3 4 5


h[n]
magnitude
n
−0.50 −0.25 0.00 0.25 0.50 0.75 1.00 1.25
0 1 2 3 4 5


$|X[k]|$
amplitude
k
0 2 4 6 8 10 12
0 1 2 3 4 5


$\angle X[k]$
phase in degrees
k
−150 −100 −50 0 50 100 150
0 1 2 3 4 5


$|H[k]|$
amplitude
k
0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0
0 1 2 3 4 5


$\angle H[k]$
phase in degrees
k
−100 −75 −50 −25 0 25 50 75 100
0 1 2 3 4 5

2.3. **Computing the DFT manually using matrix multiplication:**

We know that DFT of x(n) is,

$$X(k) \triangleq \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \tag{2.3.1}$$

Let us define a matrix "W", a NxN matrix, such that,

$$DFT(x) = X = W.x \tag{2.3.2}$$

Where (.) represents the matrix multiplication of W and x.
Each point of the matrix W is defined as,

$$W_N^{nk} = e^{-j2\pi kn/N} \tag{2.3.3}$$
$$\tag{2.3.4}$$

where n and k are column and row indices of the matrix respectively.

Now from equation: (2.3.2), DFT(x) using the DFT matrix is given by,

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ \vdots \\ X(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ \vdots \\ x(N-1) \end{bmatrix} \tag{2.3.5}$$

Given, x(n) = $\left\{\underset{\uparrow}{1}, 2, 3, 4, 2, 1\right\}$ and N=6
we have,

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \end{bmatrix} = \begin{bmatrix} 1+2+3+4+2+1 \\ 1+(2)e^{-j\pi/3}+...+(1)e^{-j5\pi/3} \\ 1+(2)e^{-2j\pi/3}+...+(1)(e^{-2j5\pi/3} \\ 1+(2)e^{-3j\pi/3}+...+(1)e^{-3j5\pi/3} \\ 1+(2)e^{-4j\pi/3}+...+(1)e^{-4j5\pi/3} \\ 1+(2)e^{-5j\pi/3}+...+(1)e^{-5j5\pi/3} \end{bmatrix} \tag{2.3.6}$$

On Solving we get,

$$X(0) = 13 + 0j \tag{2.3.7}$$
$$X(1) = -4 - 1.732j \tag{2.3.8}$$
$$X(2) = 1 + 0j \tag{2.3.9}$$
$$X(3) = -1 + 0j \tag{2.3.10}$$
$$X(4) = 1 + 0j \tag{2.3.11}$$
$$X(5) = -4 + 1.732j \tag{2.3.12}$$

We can observe that these values are same as the values that we obtained from the plots.

Similarly, the DFT of h(n) can be defined as,

$$DFT(h) = H = W.h \tag{2.3.13}$$

So, we have,

$$\begin{bmatrix} H(0) \\ H(1) \\ H(2) \\ \vdots \\ H(N-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{N-1} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} h(0) \\ h(1) \\ h(2) \\ \vdots \\ h(N-1) \end{bmatrix} \tag{2.3.14}$$

$$\begin{bmatrix} H(0) \\ H(1) \\ H(2) \\ H(3) \\ H(4) \\ H(5) \end{bmatrix} = \begin{bmatrix} h(0)+h(1)+h(2)+h(3)+h(4)+h(5) \\ h(0)+h(1)e^{-j\pi/3}+...+h(5)e^{-j5\pi/3} \\ h(0)+h(1)e^{-2j\pi/3}+...+h(5)e^{-2j5\pi/3} \\ h(0)+h(1)e^{-3j\pi/3}+...+h(5)e^{-3j5\pi/3} \\ h(0)+h(1)e^{-4j\pi/3}+...+h(5)e^{-4j5\pi/3} \\ h(0)+h(1)e^{-5j\pi/3}+...+h(5)e^{-5j5\pi/3} \end{bmatrix} \tag{2.3.15}$$

Now, from equation: (2.1.2) we know that,

$$h(n) = \delta(n) + \delta(n-2) - \frac{1}{2}h(n-1) \tag{2.3.16}$$

for N=6,

$$h(n) = \left\{\underset{\uparrow}{1}, -0.5, 1.25, -0.625, 0.3125, -0.15625\right\} \tag{2.3.17}$$

Therefore, on solving we get,

$$H(0) = 1.28125 + 0j, \quad (2.3.18)$$
$$H(1) = 0.515625 - 0.5142026j, \quad (2.3.19)$$
$$H(2) = -0.078125 + 1.109595j, \quad (2.3.20)$$
$$H(3) = 3.84375 + 0j, \quad (2.3.21)$$
$$H(4) = -0.078125 - 1.109595j. \quad (2.3.22)$$
$$H(5) = 0.515625 + 0.51420256j \quad (2.3.23)$$

We can observe that these values are same as the values that we obtained from the plots.

3 8-point FFT

3.1. **N-point FFT Algorithm:**
The N-point DFT of a signal $x(n)$ is,

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \quad (3.1.1)$$

Let F_N be a N-point DFT matrix, then the DFT of $x(n)$ in the matrix form is given by,

$$X(k) = F_N.x(n) \quad (3.1.2)$$

For the ease of calculation eq(3.1.1) can be broken down into two terms. Even-indexed n term and odd-indexed n term.

$$X(k) = \sum_{n=0(even-n)}^{N-1} x(n)W_N^{kn} + \sum_{n=0(odd-n)}^{N-1} x(n)W_N^{kn} \quad (3.1.3)$$

$$= \sum_{m=0}^{N/2-1} x(2m)W_N^{2mk} + \sum_{m=0}^{N/2-1} x(2m+1)W_N^{(2m+1)k} \quad (3.1.4)$$

$$= \sum_{m=0}^{N/2-1} x(2m)W_N^{2mk} + W_N^k \sum_{m=0}^{N/2-1} x(2m+1)W_N^{2mk} \quad (3.1.5)$$

$\because$ complex exponential, W_N^{nk} exhibits the following property,

$$(i) \quad W_N^2 = W_{N/2} \quad (3.1.6)$$

$$[\because (e^{-j2\pi kn/N})^2 = e^{-j2\pi kn/(N/2)}] \quad (3.1.7)$$

using this property eq(3.1.5) can be written as,

$$X(k) = \sum_{m=0}^{N/2-1} x(2m)W_{N/2}^{km} + W_N^k \sum_{m=0}^{N/2-1} x(2m+1)W_{N/2}^{km} \quad (3.1.8)$$

On comparing the above equation with the DFT equation (3.1.1), we can observe that the first term is a N/2-point DFT of $x(2m)$ and second term is a N/2-point DFT of $x(2m+1)$. Let,

$$X_e(k) = \sum_{m=0}^{N/2-1} x(2m)W_{N/2}^{km} \quad (3.1.9)$$
$$= F_{N/2}.x(2m) \quad (3.1.10)$$
$$X_o(k) = \sum_{m=0}^{N/2-1} x(2m+1)W_{N/2}^{km} \quad (3.1.11)$$
$$= F_{N/2}.x(2m+1) \quad (3.1.12)$$

So, $X(k)$ can be written as,

$$X(k) = X_e(k) + W_N^k X_o(k) \quad (3.1.13)$$

$$F_N.x(n) = F_{N/2}.x(2m) + F_{N/2}.x(2m+1) \quad (3.1.14)$$

From the above equation we can say see that a N-point DFT is calculated from the sum of two N/2-point DFT.

Due to the periodicity of complex exponential function, $W_{N/2}^{nk}$ satisfies the following properties,

$$(ii) \quad W_{N/2}^{k+N/2} = W_{N/2}^k \quad (3.1.15)$$

$$\because e^{-j2\pi n(k+N/2)/(N/2)} = e^{-j2\pi nk/(N/2)}.e^{-j2\pi n} \quad (3.1.16)$$
$$= e^{-j2\pi nk/(N/2)}.1 \quad (3.1.17)$$
$$= e^{-j2\pi nk/(N/2)} \quad (3.1.18)$$

$$(iii) \quad W_N^{k+N/2} = -W_N^k \quad (3.1.19)$$

$$\because e^{-j2\pi n(k+N/2)/N} = e^{-j2\pi nk/N}.e^{-j\pi n} \quad (3.1.20)$$
$$= e^{-j2\pi nk/N}.(-1) \quad (3.1.21)$$
$$= -e^{-j2\pi nk/N} \quad (3.1.22)$$

From eq((3.1.8)), .

$$X(k+N/2) = \sum_{m=0}^{N/2-1} x(2m)W_{N/2}^{m(k+N/2)} + W_N^{k+N/2} \sum_{m=0}^{N/2-1} x(2m+1)W_{N/2}^{m(k+N/2)} \quad (3.1.23)$$

Using properties (ii) and (iii), we have,

$$X(k+N/2) = \sum_{m=0}^{N/2-1} x(2m)W_{N/2}^{mk} - W_N^{k} \sum_{m=0}^{N/2-1} x(2m+1)W_{N/2}^{mk} \quad (3.1.24)$$

$$X(k+N/2) = X_e(k) - W_N^k X_o(k) \quad (3.1.25)$$

Now, from equations (3.1.13) and (3.1.25) i.e,

$$X(k) = X_e(k) + W_N^k X_o(k) \quad (3.1.26)$$

$$X(k+N/2) = X_e(k) - W_N^k X_o(k) \quad (3.1.27)$$

we have,

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ \vdots \\ X(N/2-1) \end{bmatrix} = \begin{bmatrix} X_e(0) \\ X_e(1) \\ X_e(2) \\ \vdots \\ X_e(N/2-1) \end{bmatrix} + \begin{bmatrix} W_N^0 & 0 & 0 & . & . \\ 0 & W_N^1 & 0 & 0 & . \\ 0 & 0 & W_N^2 & 0 & . \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & . & . & W_N^{N/2-1} \end{bmatrix} \begin{bmatrix} X_o(0) \\ X_o(1) \\ X_o(2) \\ \vdots \\ X_o(N/2-1) \end{bmatrix} \quad (3.1.28)$$

$$\begin{bmatrix} X(N/2) \\ X(N/2+1) \\ X(N/2+2) \\ \vdots \\ X(N) \end{bmatrix} = \begin{bmatrix} X_e(0) \\ X_e(1) \\ X_e(2) \\ \vdots \\ X_e(N/2-1) \end{bmatrix} - \begin{bmatrix} W_N^0 & 0 & 0 & . & . \\ 0 & W_N^1 & 0 & 0 & . \\ 0 & 0 & W_N^2 & 0 & . \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & . & . & W_N^{N/2-1} \end{bmatrix} \begin{bmatrix} X_o(0) \\ X_o(1) \\ X_o(2) \\ \vdots \\ X_o(N/2-1) \end{bmatrix} \quad (3.1.29)$$

In this way, by dividing the input signal into even and odd indices of n and using the properties of complex exponential we can calculate a N-point DFT by the sum of two N/2-point DFT's. Recursively we can further divide each N/2-point DFT into two N/4-point DFT's and so on.

The **FFT algorithm** decreases the time complexity from $O(N^2)$ to $O(NlogN)$, i.e, the DFT is computed faster.

Note: This algorithm is valid only when $N = 2^m$ *for* $m \in \mathbb{Z}^+$ *is satisfied*

3.2. **FFT algorithm for 8-point DFT:**
For a signal x(n) and $N = 8(= 2^3)$,
The 8-point DFT is given by,

$$X(k) = F_8.x(n) \tag{3.2.1}$$

Using the FFT algorithm we can write 8-point DFT in 4-point DFT's,

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} X_e(0) \\ X_e(1) \\ X_e(2) \\ X_e(3) \end{bmatrix} + \begin{bmatrix} W_8^0 & 0 & 0 & 0 \\ 0 & W_8^1 & 0 & 0 \\ 0 & 0 & W_8^2 & 0 \\ 0 & 0 & 0 & W_8^3 \end{bmatrix} \begin{bmatrix} X_o(0) \\ X_o(1) \\ X_o(2) \\ X_o(3) \end{bmatrix} \tag{3.2.2}$$

$$\begin{bmatrix} X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = \begin{bmatrix} X_e(0) \\ X_e(1) \\ X_e(2) \\ X_e(3) \end{bmatrix} - \begin{bmatrix} W_8^0 & 0 & 0 & 0 \\ 0 & W_8^1 & 0 & 0 \\ 0 & 0 & W_8^2 & 0 \\ 0 & 0 & 0 & W_8^3 \end{bmatrix} \begin{bmatrix} X_o(0) \\ X_o(1) \\ X_o(2) \\ X_o(3) \end{bmatrix} \tag{3.2.3}$$

Here,

$$X_e = F_4.[x(0), x(2), x(4), x(6)] \tag{3.2.4}$$
$$X_o = F_4.[x(1), x(3), x(5), x(7)] \tag{3.2.5}$$

Now, from 4-point DFT's we get 2-point DFT's recursively,

$$\begin{bmatrix} X_e(0) \\ X_e(1) \end{bmatrix} = \begin{bmatrix} X_{ee}(0) \\ X_{ee}(1) \end{bmatrix} + \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} X_{eo}(0) \\ X_{eo}(1) \end{bmatrix} \tag{3.2.6}$$

$$\begin{bmatrix} X_e(2) \\ X_e(3) \end{bmatrix} = \begin{bmatrix} X_{ee}(0) \\ X_{ee}(1) \end{bmatrix} - \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} X_{eo}(0) \\ X_{eo}(1) \end{bmatrix} \tag{3.2.7}$$

$$\begin{bmatrix} X_o(0) \\ X_o(1) \end{bmatrix} = \begin{bmatrix} X_{oe}(0) \\ X_{oe}(1) \end{bmatrix} + \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} X_{oo}(0) \\ X_{oo}(1) \end{bmatrix} \tag{3.2.8}$$

$$\begin{bmatrix} X_o(2) \\ X_o(3) \end{bmatrix} = \begin{bmatrix} X_{oe}(0) \\ X_{oe}(1) \end{bmatrix} - \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} X_{oo}(0) \\ X_{oo}(1) \end{bmatrix} \tag{3.2.9}$$

Here,

$$X_{ee} = [X_e(0), X_e(2)] \tag{3.2.10}$$
$$= F_2.[x(0), x(4)] \tag{3.2.11}$$
$$X_{eo} = [X_e(1), X_e(3)] \tag{3.2.12}$$
$$= F_2.[x(2), x(6)] \tag{3.2.13}$$
$$X_{oe} = [X_o(0), X_o(2)] \tag{3.2.14}$$
$$= F_2.[x(1), x(5)] \tag{3.2.15}$$
$$X_{oo} = [X_o(1), X_o(3)] \tag{3.2.16}$$
$$= F_2.[x(3), x(7)] \tag{3.2.17}$$

Now, 2-point DFT Matrix is given by,

$$\because F_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{3.2.18}$$

$\implies$ we get,

$$\begin{bmatrix} X_{ee}(0) \\ X_{ee}(1) \end{bmatrix} = F_2 \begin{bmatrix} x(0) \\ x(4) \end{bmatrix} = \begin{bmatrix} x(0) + x(4) \\ x(0) - x(4) \end{bmatrix} \tag{3.2.19}$$

$$\begin{bmatrix} X_{eo}(0) \\ X_{eo}(1) \end{bmatrix} = F_2 \begin{bmatrix} x(2) \\ x(6) \end{bmatrix} = \begin{bmatrix} x(2) + x(6) \\ x(2) - x(6) \end{bmatrix} \tag{3.2.20}$$

$$\begin{bmatrix} X_{oe}(0) \\ X_{oe}(1) \end{bmatrix} = F_2 \begin{bmatrix} x(1) \\ x(5) \end{bmatrix} = \begin{bmatrix} x(1) + x(5) \\ x(1) - x(5) \end{bmatrix} \tag{3.2.21}$$

$$\begin{bmatrix} X_{oo}(0) \\ X_{oo}(1) \end{bmatrix} = F_2 \begin{bmatrix} x(3) \\ x(7) \end{bmatrix} = \begin{bmatrix} x(3) + x(7) \\ x(3) - x(7) \end{bmatrix} \tag{3.2.22}$$

3.3. **Example problem:**
Let x(n) be a input signal,

$$x(n) = \left\{ \underset{\uparrow}{1}, 2, 3, 4, 2, 1 \right\} \tag{3.3.1}$$

Now to compute 8-point DFT, N should be 8.
$\because$ For input signal N=6.
So for N=8 we zero-pad the input signal with 2 zeros.

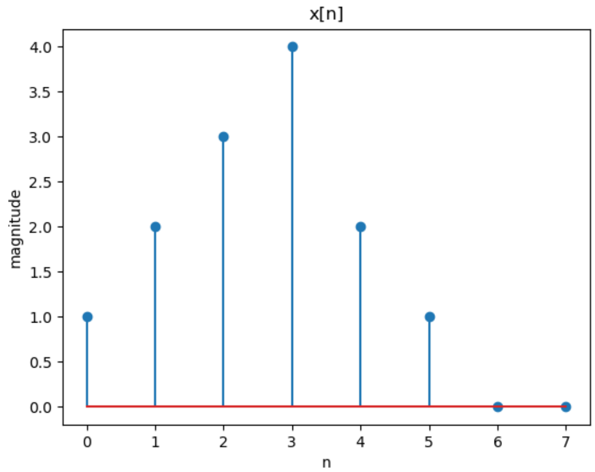
$$\implies x(n) = \left\{ \underset{\uparrow}{1}, 2, 3, 4, 2, 1, 0, 0 \right\} \tag{3.3.2}$$

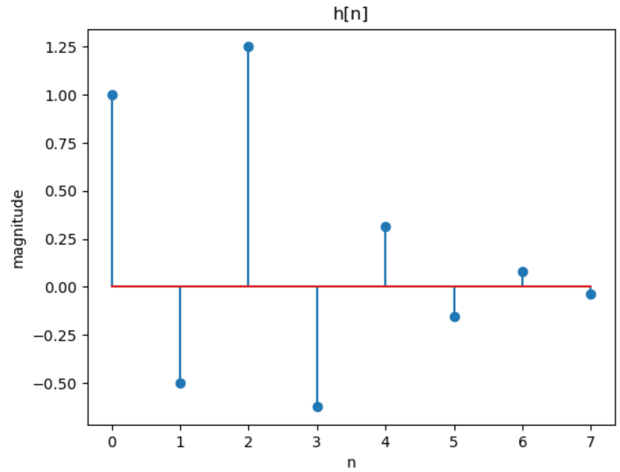
The 8-point FFT algorithm to compute the fourier transform of x(n) and h(n)is in the following python code.

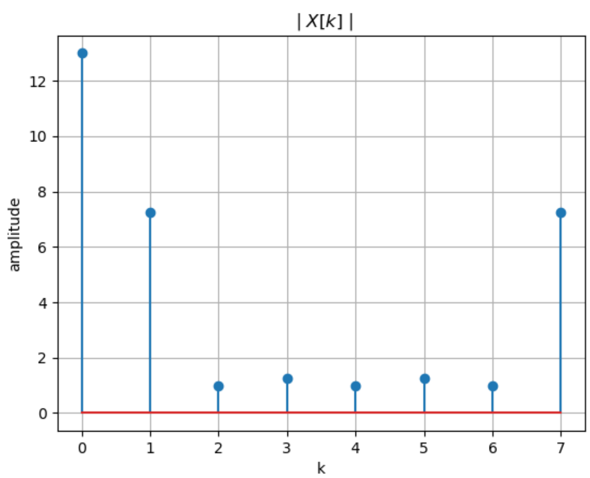
https://github.com/pranayEE11009/EE3025_IDP/blob/main/Assignment_1/codes/ee18btech11009_fft.py

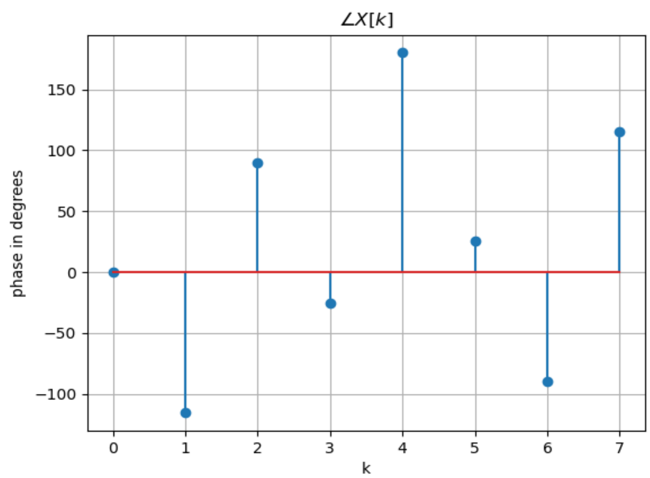
The following C program computes the 8-point FFT.

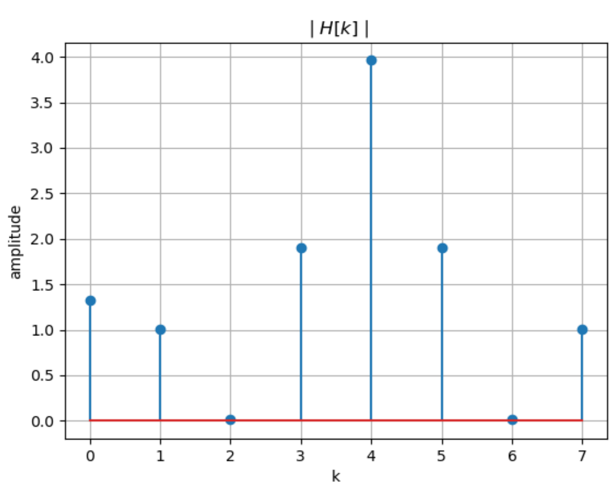
https://github.com/pranayEE11009/EE3025_IDP/blob/main/Assignment_1/codes/ee18btech11009_fft.c

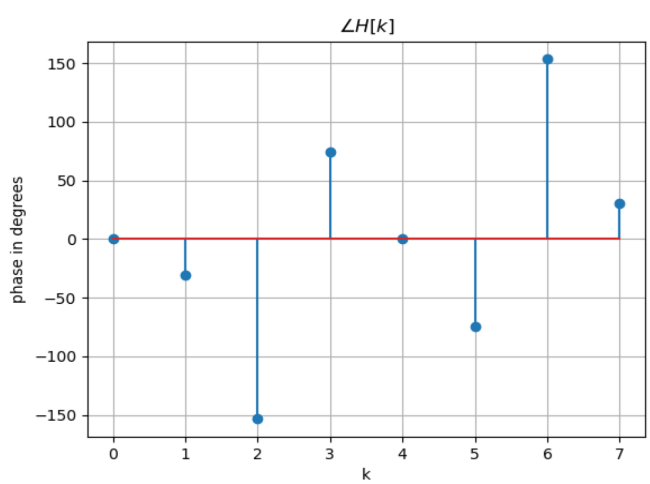

x[n]
magnitude
n


h[n]
magnitude
n


$|X[k]|$
amplitude
k


$\angle X[k]$
phase in degrees
k


$|H[k]|$
amplitude
k


$\angle H[k]$
phase in degrees
k