K. J. Somaiya College of Engineering, Mumbai-77

Batch: A-4 **Roll No.:** 16010122151

Experiment No.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title: Write a program to Compute linear and circular convolution of two discrete time signal sequences using Matlab.

Objective: To familiarize the beginner to MATLAB by introducing the basic features and commands of the program.

Expected Outcome of Experiment:

CO	Outcome
CO3	To understand the concept of convolution and perform different convolution operations on the given input signals.

Books/ Journals/ Websites referred:

1. http://www.mathworks.com/support/
2. www.math.mtu.edu/~msgocken/intro/intro.html
3. www.mccormick.northwestern.edu/docs/efirst/matlab.pdf
4. A.Nagoor Kani "Digital Signal Processing", 2nd Edition, TMH Education.

Pre Lab/ Prior Concepts:

Convolution

Discrete time convolution is a method of finding response of linear time invariant system. It is based on the concepts of linearity and time invariance and assumes that the

system information
is known in terms of its impulse response h[n].

Convolution is defined as

$$Y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] = h[n]*x[n]$$

Convolution consists of folding, shifting, Multiplication and summation operations.

Circular Convolution

Circular convolution between two length N sequences can be carried out as shown by the expression below:

$$y_C[n] = \sum_{m=0}^{N-1} g[m]h[\langle n-m \rangle_N]$$

Since the above operation involves two length-N sequences it is referred to as the N-point circular convolution and denoted by:

$$y_C[n] = g[n] \textcircled{N} h[n]$$

As in linear convolution circular convolution is commutative.
i.e.

$$g[n] \textcircled{N} h[n] \equiv h[n] \textcircled{N} g[n]$$

Example Of Linear Convolution:

Convolution

$$y(n) = x(n) * h(n) = \sum_{m=-\infty}^{\infty} x(m)\, h(n-m)$$

Properties of linear convolution
→ Commutative
→ Associative
→ Distributive

$$x(n) = \{\overset{x_l}{1}, 2, 1, \overset{x_h}{2}\}$$
$$h(n) = \{\underset{h_l}{1}, 1, \underset{h_h}{1}\}$$

$$y(n) = \sum_{n=-\infty}^{\infty} x(k)\, h(n-k) \quad \text{— (1)}$$

$$x(k) = \{\underset{\uparrow 0}{1}, \underset{1}{2}, \underset{2}{1}, \underset{3}{2}\}$$

$$h(k) = \{\underset{0}{1}, \underset{1}{1}, \underset{2}{1}\}$$

Range of $y(n)$ = total no. of sample of x & h - 1
$= 4 + 3 - 1$
$= 6$

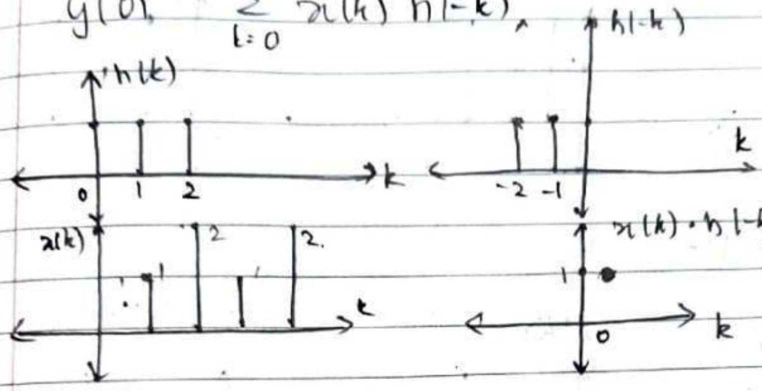
Starting range of $y(n)$
$y_l = x_l + h_l$
$y_l = 0 + 0 = 0$
$y_h = x_h + h_h$
$= 3 + 2 = 5$.

$n \rightarrow$ 0 to 5
$k \rightarrow$ 0 to 3

i) Graphical Method

$n = 0$

$$y(0) = \sum_{k=0}^{3} x(k)\, h(-k)$$

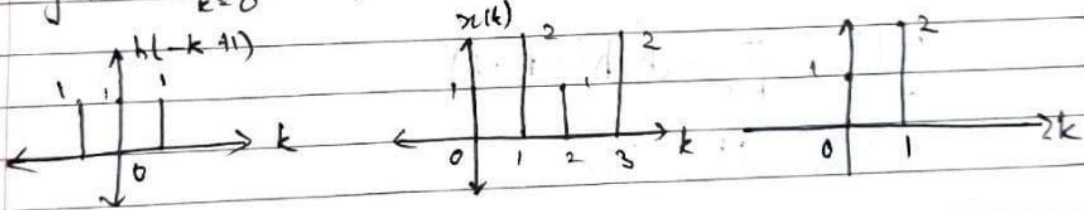


$y(0) = 1$

$n = 1$

$$y(1) = \sum_{k=0}^{3} x(k)\, h(1-k)$$

x(k)h(1-k)

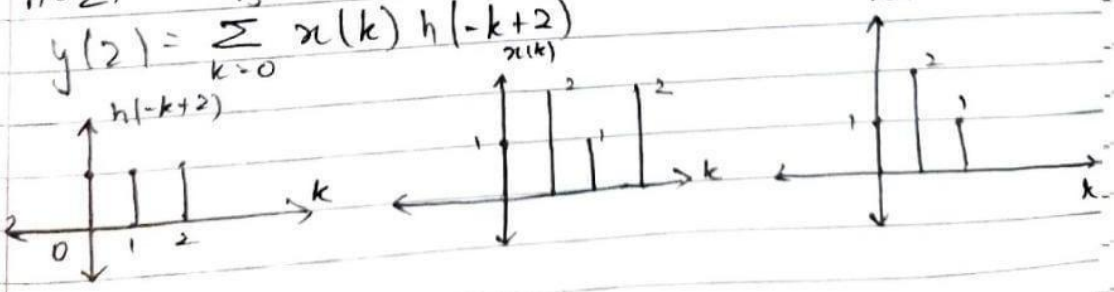


$\therefore y(1) = 1+2 = 3.$

$n = 2.$

$$y(2) = \sum_{k=0}^{3} x(k)\, h(-k+2)$$

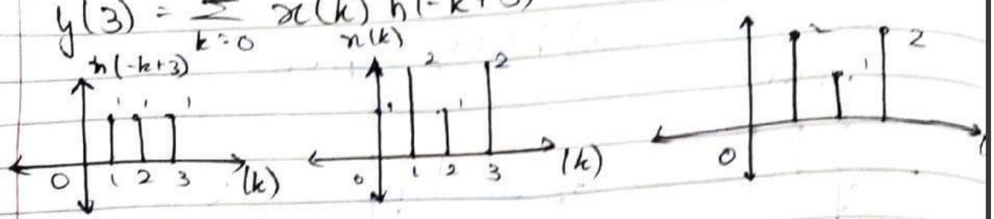
x(k).h(-k+2)



$\therefore y(2) = 1+2+1 = 4$

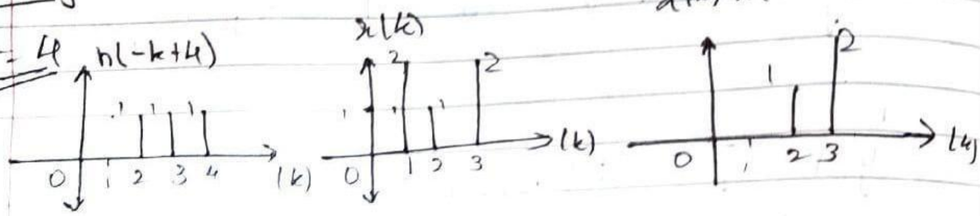
$n = 3$

$$y(3) = \sum_{k=0}^{3} x(k)\, h(-k+3)$$



$$y(3) = 2+1+2 = 5$$

$n = 4$



$$y(4) = \sum_{k=0}^{3} x(k)\, h(-k+4)$$

$$y(4) = 1+2 = 3$$

$n = 5$

$y(5) = 2$

$y(n) = \{1, 3, 4, 5, 3, 2\}$

Example Of Circular Convolution:

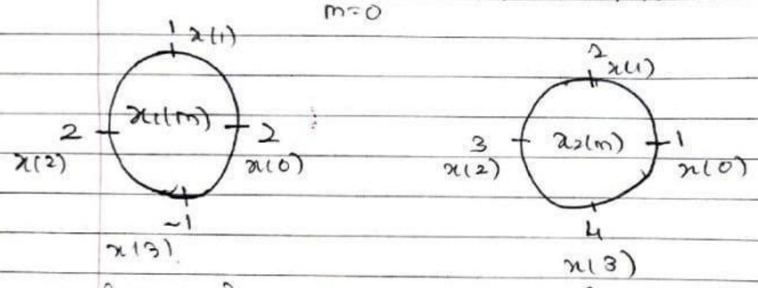
$H(n-1) \rightarrow$ Anticlockwise shift 1 sample.
$H(n+1) \rightarrow$ Clockwise shift one sample
$H(-n) \rightarrow$ Clockwise
$H(n) \rightarrow$ Anticlockwise
$H(-n+1) \rightarrow$ —"— shift 1 sample.

Q) $x_1(n) = \{2, 1, 2, -1\}$
$x_2(n) = \{1, 2, 3, 4\}$

Circular Convolution of $x_1(n)$ & $x_2(n) \rightarrow$

$$x_3(n) = \sum_{m=0}^{N-1} x_1(m)\, x_2((n-m))_N$$

$$= \sum_{m=0}^{N-1} x_1(m)\, x_2(n-m)$$

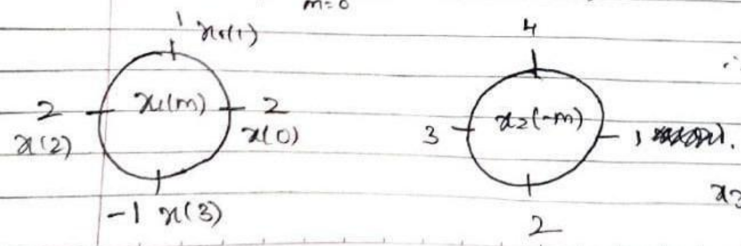


$x_1 = \{2, 1, 2, -1\}$ $x_2 = \{1, 2, 3, 4\}$

$n = 0$ to 3
$m = 0$ to 3

for $n=0$

$$x_3(0) = \sum_{m=0}^{3} x_1(m)\, x_2(0-m)$$

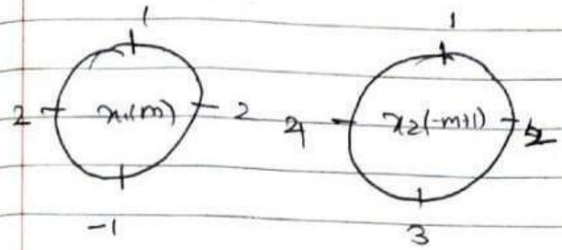


$\therefore x_3(0) = (1\times2) + (1\times4) + (3\times2) + (-1\times2)$

$x_3(0) = 10$

$n = 1$

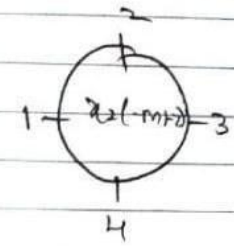
$x_3(1) = \sum_{m=0}^{3} x_1(m)\, x_2(1-m)$



$x_3(1) = 4 + 1 + 8 - 3$
$= 10$

$n = 2$

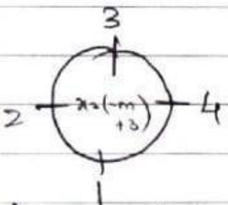
$x_3(2) = \sum_{m=0}^{3} x_1(m)\, x_2(2-m)$



$x_3(2) = 6 + 2 + 2 - 4$
$= 6$

$n = 3$

$x_3(3) = \sum_{m=0}^{3} x_1(m)\, x_2(3-m)$



$x_3(3) = 8 + 3 + 4 - 1 = 14$

* Circular Convolution using Tabular

m	-3	-2	-1	0	1	2	3
$x_1(m)$				2	1	2	-1
$x_2(m)$				1	2	3	4
$x_2((-m)) = x_{2,0}(m)$	4	3	2	1			
$x_2((1-m)) = x_{2,1}(m)$		4	3	2	1	4	3
$x_2((2-m)) = x_{2,2}(m)$			4	3	2	1	4
$x_2((3-m)) = x_{2,3}(m)$				4	3	2	1

Implementation details along with screenshots:

Linear convolution:

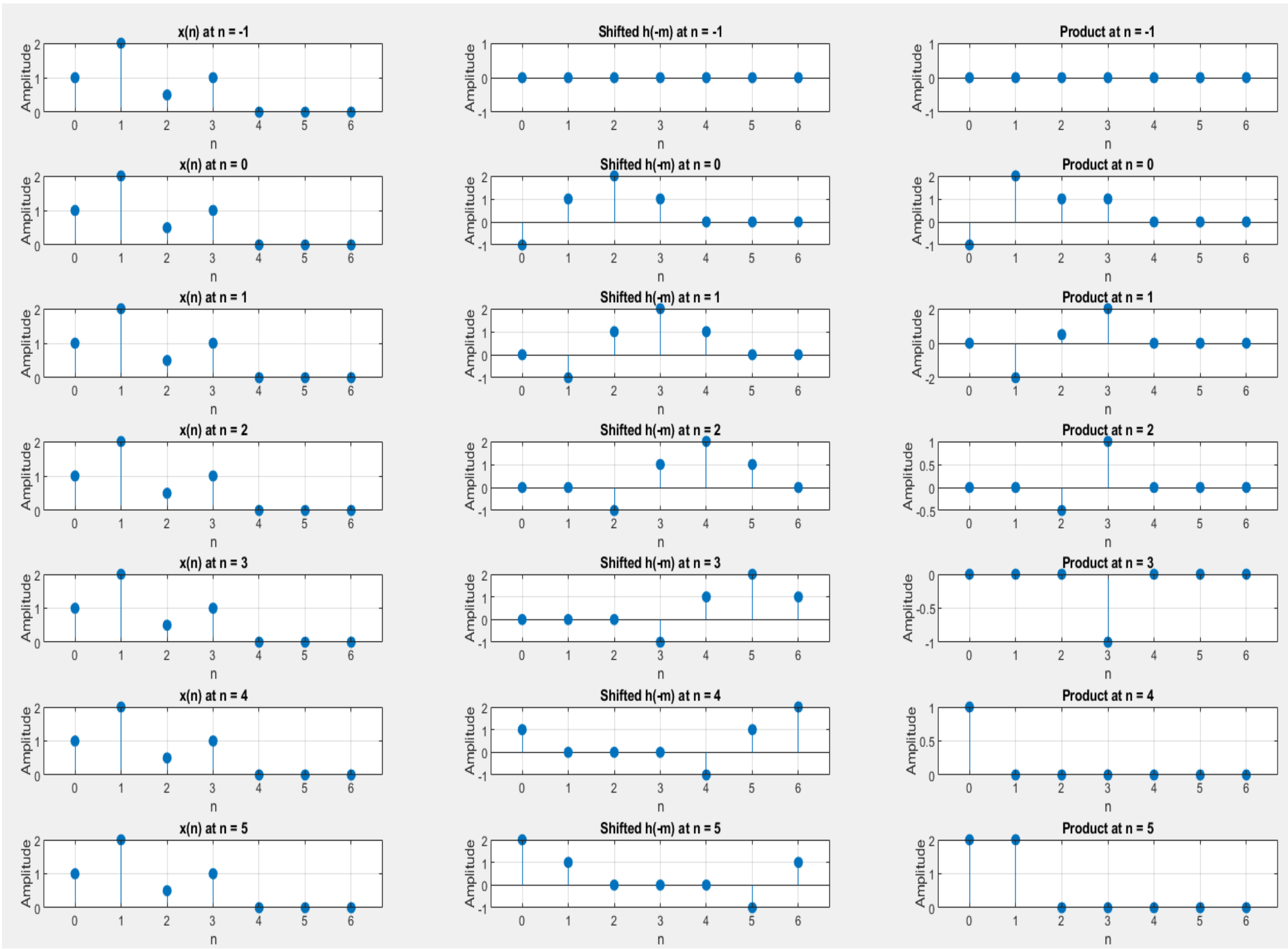
CODE:

```
x = [1, 2, 0.5, 1];
h = [1, 2, 1, -1];
Nx = length(x);
Nh = length(h);
Ny = Nx + Nh - 1;
h_flipped = fliplr(h);
x_padded = [x, zeros(1, Ny - Nx)];
h_padded = [h_flipped, zeros(1, Ny - Nh)];
y = zeros(1, Ny);
figure;
n_range = -1:5;
num_steps = length(n_range);
for k = 1:num_steps
    n = n_range(k);
    if n < 0

        h_shifted = zeros(1, Ny);
    else

        h_shifted = circshift(h_padded, [0, n]);
    end

    product = x_padded .* h_shifted;


    if n >= 0 && n < Ny
        y(n + 1) = sum(product);
    end


    subplot(num_steps, 3, (k-1)*3 + 1);
    stem(0:Ny-1, x_padded, 'filled');
    title(['x(n) at n = ', num2str(n)]);
    xlabel('n');
    ylabel('Amplitude');
    grid on;

    subplot(num_steps, 3, (k-1)*3 + 2);
    stem(0:Ny-1, h_shifted, 'filled');
    title(['Shifted h(-m) at n = ', num2str(n)]);
    xlabel('n');
    ylabel('Amplitude');
    grid on;

    subplot(num_steps, 3, (k-1)*3 + 3);
    stem(0:Ny-1, product, 'filled');
    title(['Product at n = ', num2str(n)]);
    xlabel('n');
```

```
    ylabel('Amplitude');
    grid on;
end

disp('The output sequence y(n) is:');
disp(y);
```

OUTPUT:

```
>> untitled2
The output sequence y(n) is:
    3.0000    0.5000    0.5000   -1.0000    1.0000    4.0000         0
```



Circular convolution:

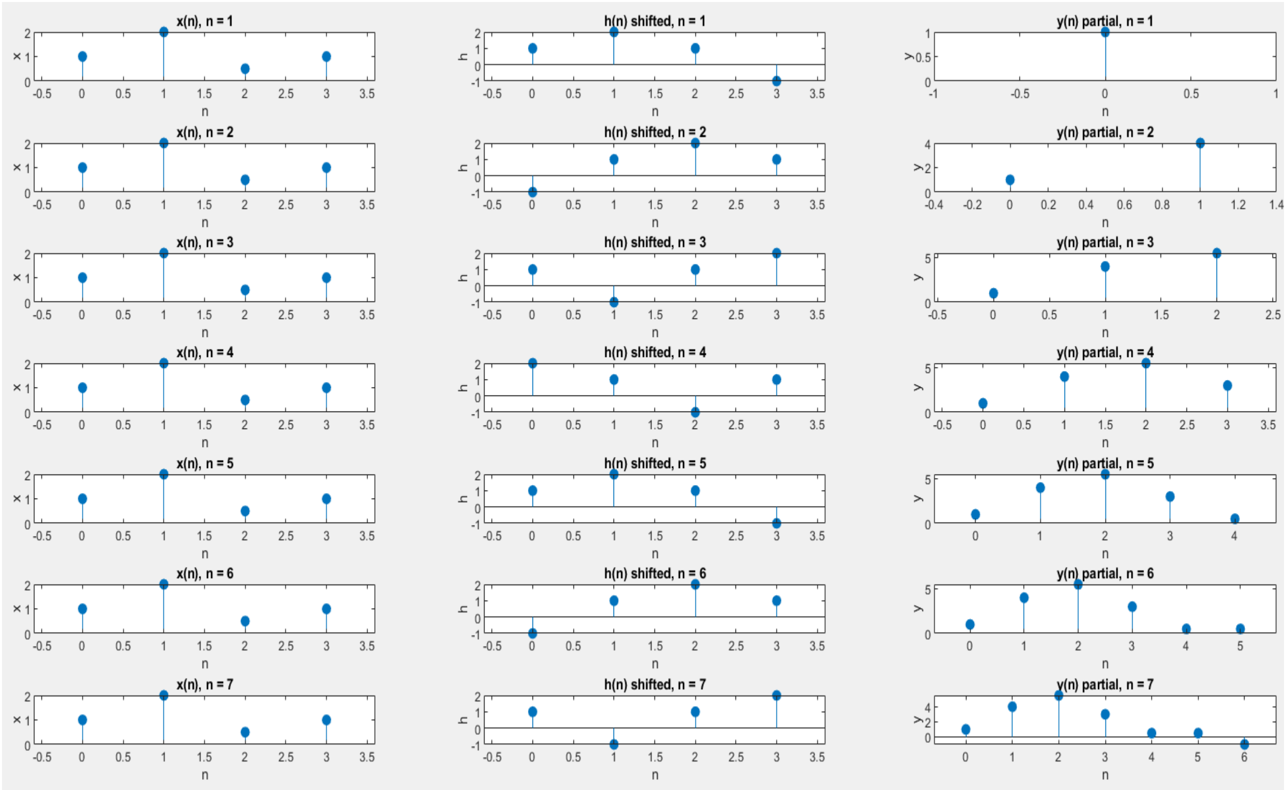
CODE:

```
x = [1, 2, 0.5, 1];
h = [1, 2, 1, -1];
```

```
linearConvolutionSteps(x, h);
function linearConvolutionSteps(x, h)
    length_of_x = length(x);
    length_of_h = length(h);
    length_rez = length_of_x + length_of_h - 1;
    x_pad = [x, zeros(1, length_rez - length_of_x)];
    h_pad = [h, zeros(1, length_rez - length_of_h)];
    rez = zeros(1, length_rez);

    figure;
    for n = 1:length_rez
        for j = 1:length_of_x
            if (n - j + 1 > 0 && n - j + 1 <= length_of_h)
                rez(n) = rez(n) + x_pad(j) * h_pad(n - j + 1);
            end
        end

        subplot(8, 3, (n-1)*3 + 1); stem(0:length_of_x-1, x, 'filled');
        title(['x(n), n = ', num2str(n)]);
        xlabel('n'); ylabel('x');

        subplot(8, 3, (n-1)*3 + 2); stem(0:length_of_h-1, circshift(h, n-1),
'filled');
        title(['h(n) shifted, n = ', num2str(n)]);
        xlabel('n'); ylabel('h');

        subplot(8, 3, (n-1)*3 + 3); stem(0:n-1, rez(1:n), 'filled');
        title(['y(n) partial, n = ', num2str(n)]);
        xlabel('n'); ylabel('y');
    end
    disp('Input sequence x:');
    disp(x);
    disp('Input sequence h:');
    disp(h);
    disp('Output sequence y:');
    disp(rez);
end
```

OUTPUT:

```
>> untitled3
Input sequence x:
    1.0000    2.0000    0.5000    1.0000

Input sequence h:
     1     2     1    -1

Output sequence y:
    1.0000    4.0000    5.5000    3.0000    0.5000    0.5000   -1.0000
```



Conclusion:- Thus, in this experiment, we have learnt about convolution, its types and Mathematical calculations and formulae based on it. Then, we implemented separate programs computing convolution and correlation using MATLAB.

Date: ______________ **Signature of faculty in-charge**

Post Lab Descriptive Questions

1. **Explain the role of convolution in signal processing.**

Convolution is a fundamental mathematical operation in signal processing that helps analyze and modify signals. It is used to determine the output of a linear time-invariant (LTI) system when an input signal is applied. Mathematically, the convolution of two signals x(t) and h(t) is given by:

$$y(t) = (x * h)(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

For discrete signals:

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

Key Roles of Convolution in Signal Processing:

System Response Calculation: It helps determine the output of an LTI system when the impulse response and input signal are known.

Filtering: Many filters, including low-pass, high-pass, and band-pass filters, are implemented using convolution to modify signals.

Smoothing and Edge Detection: In image and audio processing, convolution is used for blurring, sharpening, and feature extraction.

Feature Extraction in Machine Learning: Convolutional Neural Networks (CNNs) use convolution to detect edges, patterns, and textures in images.

Echo and Reverb Effects in Audio Processing: Convolution simulates room acoustics and applies sound effects.

2. Explain the difference between linear and circular convolution?

Feature	Linear Convolution	Circular Convolution
Definition	Convolution of two discrete signals as per the standard formula.	Convolution where the signal wraps around, assuming periodicity.
Mathematical Formula	$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n -$	$y[n] = \sum_{k=0}^{N-1} x[k]h[(n-k) \mod N]$

Length of Output	If x[n] has length M and h[n] has length N, the output length is M+N-1.	The output has the same length as the periodic sequence (usually max of M,N).
Applications	Used for LTI systems, filtering, and time-domain analysis.	Used in FFT-based convolution, periodic signal processing, and spectral analysis.
Aliasing Effect	No aliasing occurs.	Aliasing (overlap) occurs due to periodic summation.
Implementation	Direct computation or FFT-based optimization.	Performed efficiently using Fast Fourier Transform (FFT).

Relation Between Linear and Circular Convolution:

Zero-Padding: If the input sequences are zero-padded properly to size M+N-1, then circular convolution is equivalent to linear convolution.

DFT Property: Circular convolution can be computed using the Discrete Fourier Transform (DFT) as:

y[n] = IDFT{X[k]*H[k]),

where X[k] and H[k] are the DFTs of x[n] and h[n], respectively.

3. **Explain with the help of an example the steps required to transform linear convolution with circular convolution and vice-versa.**

To transform linear convolution into circular convolution or vice versa, we need to followcertain steps. Let's first consider how to transform linear convolution into circularconvolution:

Transforming Linear Convolution into Circular Convolution:

Suppose we have two sequences x[n] and h[n] and we want to transform their linear convolution into circular convolution. The steps involved in this transformation are as follows:

Step 1: Pad the sequences x[n] and h[n] with zeros to make them of length N + M - 1,

where N is the length of x[n] and M is the length of h[n]. This is to ensure that the linearconvolution is equivalent to circular convolution.

Step 2: Take the DFT (Discrete Fourier Transform) of the two sequences x[n] and h[n]

using an FFT (Fast Fourier Transform) algorithm.

Step 3: Multiply the DFT of the two sequences element-wise to obtain the DFT of the

circular convolution.

Step 4: Take the inverse DFT of the result from step 3 using an IFFT (Inverse Fast FourierTransform) algorithm to obtain the circular convolution sequence.

For example, let's consider the following sequences:

x[n] = {1, 2, 3}

h[n] = {2, 1}

To transform their linear convolution into circular convolution, we first pad the

sequences with zeros:

x[n] = {1, 2, 3, 0}

h[n] = {2, 1, 0, 0}

Then we take their DFT using an FFT algorithm:

X[k] = {6, -1+1.73j, -1-1.73j, 0}

H[k] = {3, 1-1j, 1+1j, 0}

Next, we multiply the DFT of the two sequences element-wise:

Y[k] = X[k] * H[k] = {18, -4+4.59j, -4-4.59j, 0}

Finally, we take the inverse DFT of the result from step 3 using an IFFT algorithm:

y[n] = IDFT(Y[k]) = {2.5, 0.5, 5, 0}

Therefore, the circular convolution of x[n] and h[n] is given by the sequence y[n] = {2.5,0.5, 5, 0}.

Transforming Circular Convolution into Linear Convolution:

Similarly, we can transform circular convolution into linear convolution by following thebelow steps:

Step 1: Take the DFT of the circular convolution sequence using an FFT algorithm.

Step 2: Divide the DFT of the circular convolution sequence by the DFT of one of the input sequences, either x[n] or h[n].

Step 3: Take the inverse DFT of the result from step 2 using an IFFT algorithm to obtainthe linear convolution sequence.

For example, let's consider the following circular convolution sequence:

y[n] = {2.5, 0.5, 5, 0}

To transform this circular convolution sequence into linear convolution, we first take its DFT:

Y[k] = {8, 2-2j, 2+2j, 0}

Next, we divide the DFT of the circular convolution sequence by the DFT of one of the input sequences, let's say x[n]:

X[k] = {6, -1+1.73j, -1-1.73j, 0}

H[k] = {2, 1-1j, 1+1j, 0}

Therefore, we divide Y[k] by X[k]