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

Batch: C1 Roll No.: 16010122323

Experiment No. 4

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:

$$y_c[n] = g[n] \, Ⓝ \, h[n]$$

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

$$g[n] \, Ⓝ \, h[n] \equiv h[n] \, Ⓝ \, g[n]$$

Example Of Linear Convolution:

Example Of Circular Convolution:

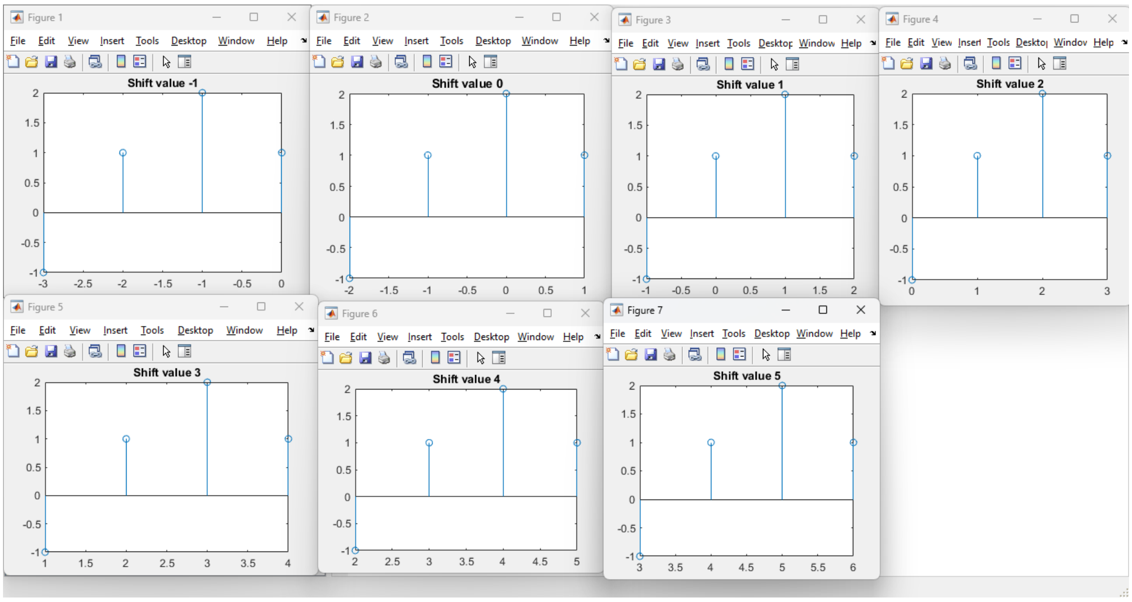
Implementation details along with screenshots:

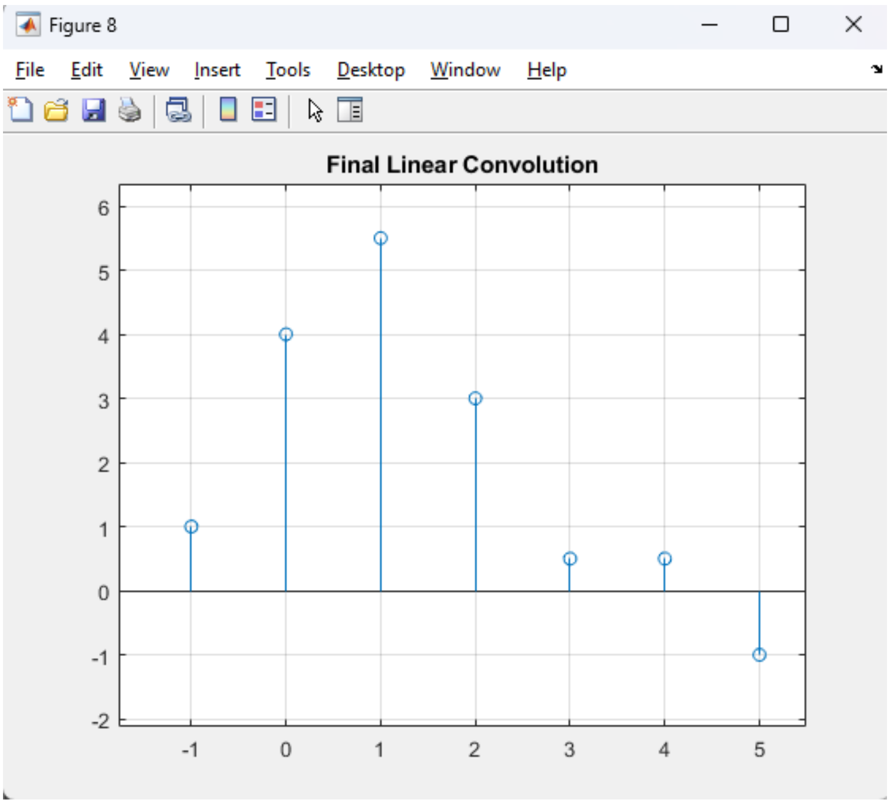
Linear Convolution:

```
x1 = input('Enter first signal: ');

x2 = input('Enter second signal: ');

nx1 = input('Enter the starting point of first signal: ');

nx2 = input('Enter the starting point of second signal: ');

Nx1 = length(x1);

Nx2 = length(x2);

Ny = Nx1 + Nx2 - 1;

ny = nx1 + nx2;

indices_x1 = zeros(1,Nx1);

indices_x2 = zeros(1,Nx2);

for n = 1:Nx1

indices_x1(n) = nx1 + n -1;

end

for n = 1:Nx2

indices_x2(n) = nx2 + n -1;

end

folded_indices = -flip(indices_x2);

folded_x2 = flip(x2);

y = zeros(1,Ny);

finalX = zeros(1,Ny);

for n = 1:Ny

finalX(n) = ny + n -1;

end

shiftedX = zeros(1,Nx2);

for i = 1:Ny
```

```
value = 0;
for j = 1:Nx2
for k = 1:Nx1
if (folded_indices(j) + finalX(i)) == indices_x1(k)
value = value + (x1(k) * folded_x2(j));
end
end
shiftedX(j) = folded_indices(j) + finalX(i);
end
y(i) = value;
figure;
stem(shiftedX, folded_x2);
title(sprintf('Shift value %d', finalX(i)));
end
figure;
stem(finalX, y);
title('Final Linear Convolution');
grid on;
```

```
Command Window
>> exp4_334_Linear
Enter first signal: [1,2,0.5,1]
Enter second signal: [1,2,1,-1]
Enter the starting point of first signal: 0
Enter the starting point of second signal: -1
>>
```



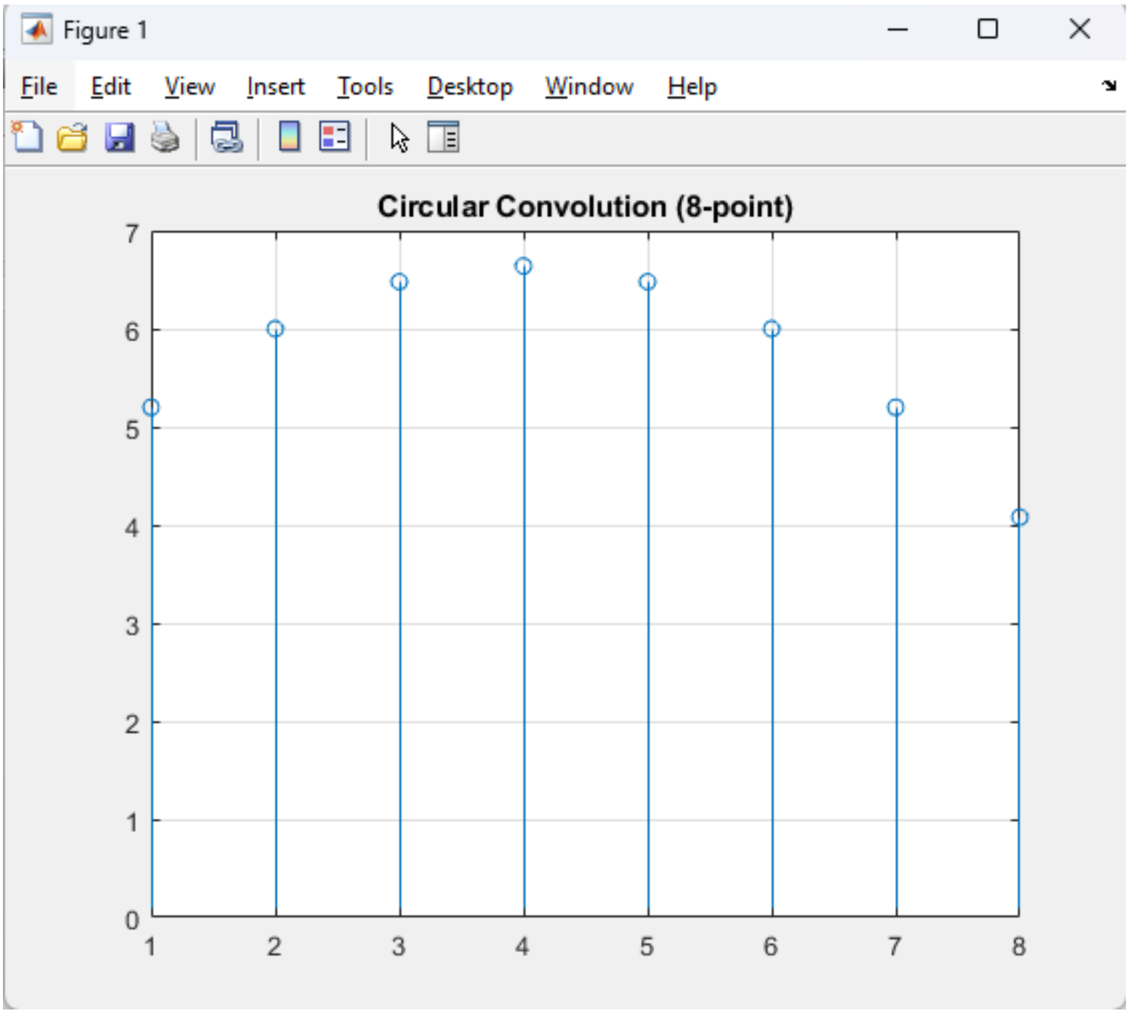


Circular Convolution:

```
x1 = input('Enter first signal: ');

x2 = input('Enter second signal: ');

% Set the desired length for circular convolution (4 or 8 points)

N = input('Enter the length for circular convolution (4 or 8): ');

% Ensure both signals are padded to the desired length N

x1 = [x1, zeros(1, N - length(x1))];

x2 = [x2, zeros(1, N - length(x2))];

% 1. Change of Index: Represent the signals with indices from 1 to N

% MATLAB uses 1-based indexing

indices_x1 = 1:N;

indices_x2 = 1:N;

% 2. Selective Folding: Flip pairs, except the first and middle
elements

folded_x2 = x2; % Start with the original x2

half_N = N / 2; % Calculate the half-length for determining opposite
pairs

% Loop through and flip the pairs (skip first and middle element)

for i = 2:half_N % Skip the first and the middle element

% Flip opposite pairs (x(i) with x(N-i+1))

folded_x2(i) = x2(N-i+2);

folded_x2(N-i+2) = x2(i);

end

% 3. Rotation: For each step in the convolution, shift folded_x2
circularly

y = zeros(1, N); % Initialize the output

for n = 1:N

value = 0;

for m = 1:N

% Rotate the folded signal by index n (wrap around with mod)
```

```
rotated_idx = mod(m - n, N) + 1;

% 4. Multiplication: Multiply the two signals element-wise

value = value + x1(m) * folded_x2(rotated_idx);

end

% 5. Summation: Store the result for this index

y(n) = value;

end

% Plot the result of the circular convolution

figure;

stem(1:N, y);

title(['Circular Convolution (' num2str(N) '-point)']);

grid on;
```

```
Command Window
>> exp4_334_Circular
Enter first signal: [0.2,0.4,0.6,0.8,1.0,1.2,1.4,1.6]
Enter second signal: [0.1,0.3,0.5,0.7,0.9,1.1,1.3,1.5]
Enter the length for circular convolution (4 or 8): 8
>>
```



Conclusion:- In this experiment, we wrote a program to compute linear and circular convolution of two discrete time signal sequences 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 describes how an input signal interacts with a system to produce an output signal. It is used to analyze linear time-invariant (LTI) systems and is represented as:

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

where:

- x(n) is the input signal,
- h(n) is the system's impulse response,
- y(n) is the output signal.

Role of Convolution in Signal Processing:

1. **Filtering:** Used in designing digital filters (low-pass, high-pass, band-pass) to modify signals.
2. **System Response Analysis:** Helps determine the output of an LTI system given its impulse response.
3. **Image Processing:** Applied in blurring, edge detection, and feature extraction.
4. **Feature Extraction in Machine Learning:** Convolutional Neural Networks (CNNs) use convolution for object detection.
5. **Echo and Reverberation Effects:** Used in audio signal processing for effects like echo and reverb.

2. Explain the difference between linear and circular convolution?

Feature	Linear Convolution	Circular Convolution
Definition	Convolution of two signals where the output length is (M+N−1), where M and N are the lengths of the input signals.	Convolution that assumes periodic extension of signals, resulting in an output length of max(M,N).
Formula	$y(n) = \sum_{k=0}^{M-1} x(k)h(n-k)$	$y(n) = \sum_{k=0}^{M-1} x(k)h(\mathrm{mod}(n-k, M))$
Padding Requirement	No padding required.	Zero-padding required to avoid time-aliasing effects.

Feature	Linear Convolution	Circular Convolution
Application	Used in most real-world signal processing systems like filtering and system response	Used in FFT-based computation for efficient convolution in digital signal processing (DSP).

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**, follow these steps:

Example:
Let x(n)=[1,2,3] and h(n)=[4,5,6].

1.**Compute Linear Convolution:**
Using the formula for linear convolution,

y(n)=[4,13,28,27,18]

The output length is (3+3−1)=5.

2.**Apply Zero Padding to Match the Length:**
Circular convolution requires equal-length sequences. Pad both x(n) and h(n) with zeros to make their lengths equal to the next power of two (e.g., 6).

x(n)=[1,2,3,0,0,0] , h(n)=[4,5,6,0,0,0]

3.**Compute Circular Convolution using DFT/IDFT or Direct Method:**
Using the circular convolution formula, we get a periodic result where values beyond length 3 wrap around.

To transform **circular convolution** back into **linear convolution**:

- Extend the input sequences with zeros.
- Perform circular convolution.
- Truncate the result to the expected linear convolution length.

This transformation is used in FFT-based fast convolution techniques.