**Batch: A3 Roll No.: 1811053**

**Experiment No. 3**

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

**Signature of the Staff In-charge with date**

|  |
| --- |
| **Title:** To compute convolution and correlation of two discrete time 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] = Σ h[k]x [n-k] =h[n]\*x[n] k=-∞

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:



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



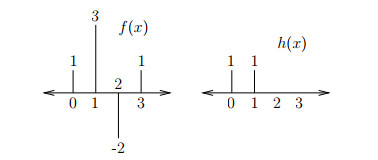
As in linear convolution circular convolution is commutative.

i.e.



**Example of Linear Convolution:**

**Consider the convolution of the following two functions:**

****

This convolution can be performed graphically by reflecting and shifting h(x)

The samples of f(s) and h(s − x) that line up vertically are multiplied and summed:

g(0) = f(−1)h(1) + f(0)h(0) = 0 + 1 = 1

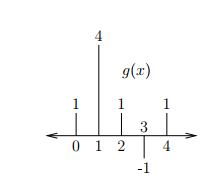
g(1) = f(0)h(1) + f(1)h(0) = 1 + 3 = 4

g(2) = f(1)h(1) + f(2)h(0) = 3 + −2 = 1

g(3) = f(2)h(1) + f(3)h(0) = −2 + 1 = −1

g(4) = f(3)h(1) + f(4)h(0) = 1 + 0 = 1

The result of the convolution is as shown below:



**Example of Circular Convolution:**

For M = 4, the convolution can be performed using circular reflection and shifts of h(x).

The samples of f(s) and h((s − x) mod M) that line up vertically are multiplied and summed:

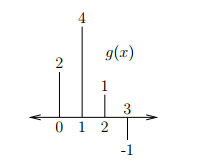
g(0) = f(3)h(1) + f(0)h(0) = 1 + 1 = 2

g(1) = f(0)h(1) + f(1)h(0) = 1 + 3 = 4

g(2) = f(1)h(1) + f(2)h(0) = 3 + −2 = 1

g(3) = f(2)h(1) + f(3)h(0) = −2 + 1 = −1

The result of the convolution is as shown below:



Notice that f(x) and h(x) are both treated as if they are of length 4, and the circular convolution is also of length 4.

**Example of Correlation:**

x(n) = (1,1,0,1) y(n) = (4,-3,-2,1)

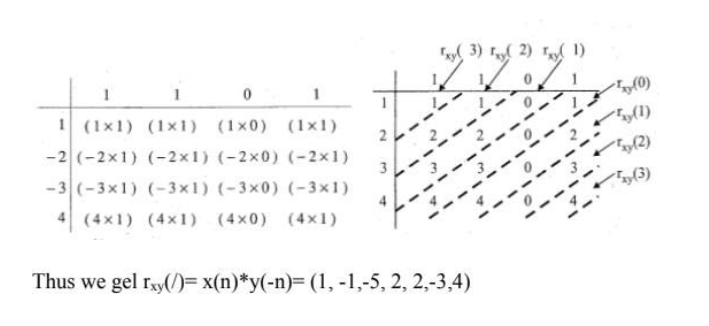
To obtain y (-n), write down sequence y (n) in the reverse order. Here arrow is marked at sample (-2). Do not change the arrow. Thus y (-n ) = (1, -2,-3,4)

We will decide the range of n for convolution of x(n ) and y (-n)

Lowest index = -2+ (- 1 ) = —3, Highest index = 1 + 2 = 3

Thus output sequence will be from -3 to +3.

Perform Convolution of x(n) and y(-n):



**Implementation details along with screenshots:**

**Linear convolution:**

**Code:**

startr = input('Enter the start range of X : ');

starth = input('Enter the start range of H : ');

n1 = input('Enter number of samples X: ');

samples = [];

start=0;

for i = 1:n1

x = input('Enter the values ');

samples = [samples x];

end

n2 = input('Enter number of samples H : ');

hsamples = [];

for i = 1:n2

x = input('Enter the values ');

hsamples = [hsamples x];

end

len\_y=n1+n2-1;

sumt=0;

fin=[];

for i=1:len\_y

sumt=0;

for k=1:n1

st = i-k;

if st<start || st>=n2

st=0;

else

st=hsamples(((i-k)-start)+1);

end

y = samples(k).\* st;

sumt = sumt + y;

end

fin=[ fin sumt ];

end

fin

x1 = startr:1:startr+n1-1;

subplot(1,3,1)

stem(x1,samples);

title('x(n)');

xlabel('N(u)');

ylabel('Amplitude');

x1 = starth:1:starth+n2-1;

subplot(1,3,2);

stem(x1,hsamples);

title('h(n)');

xlabel('N(u)');

ylabel('Amplitude');

x1 = startr:1:startr+len\_y-1;

subplot(1,3,3);

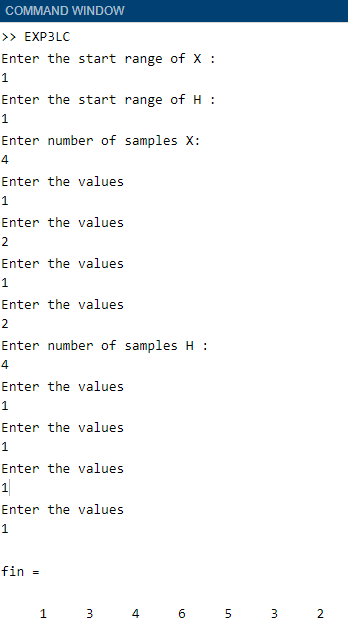
stem(x1,fin);

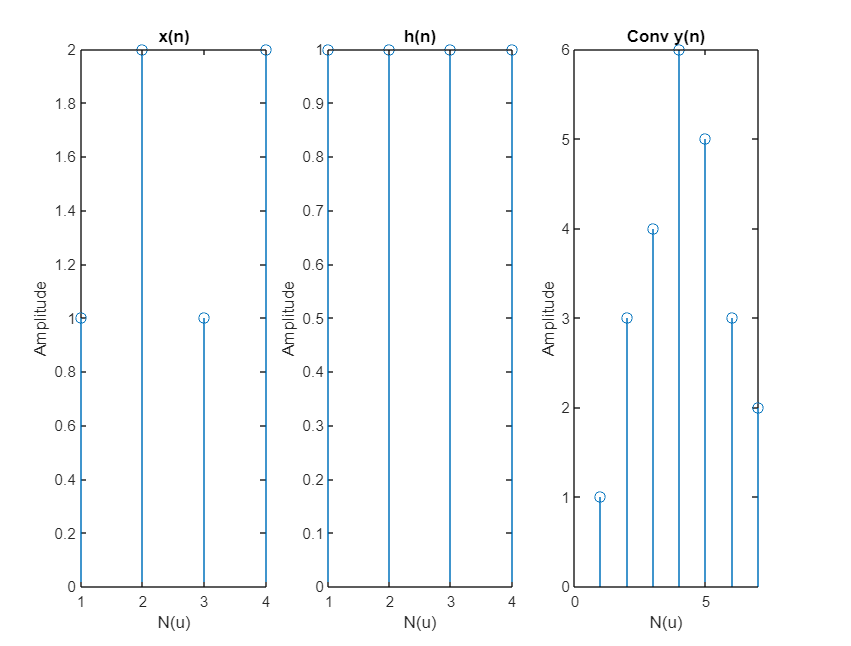
title('Conv y(n)');

xlabel('N(u)');

ylabel('Amplitude');

**Output:**





**Circular convolution:**

function y=circular\_convolution(x,h)

x= input ('enter X sequence');

h= input ('enter H sequence');

N1= length(x);

N2= length(h);

N=max(N1,N2);%length of sequence

x=[x zeros(1,N-N1)]; %modified first sequence

h=[h zeros(1,N-N2)]; %modified second sequence

for n=0:N-1;

y(n+1)=0;

for i=0:N-1

j=mod(n-i,N);

y(n+1)=y(n+1)+x(i+1)\*h(j+1); %shifting and adding

end

end

n=N:1:N+N1-1;

subplot(1,3,1);

stem(n,x);

title('x(n)');

xlabel('N(u)');

ylabel('Amplitude');

title('Input signal x(n)');

n=N:1:N+N2-1;

subplot(1,3,2);

stem(n,h);

title('h(n)');

xlabel('N(u)');

ylabel('Amplitude');

title('Input signal h(n)');

disp('Output sequence of circular convolution');

disp(y)

subplot(1,3,3);

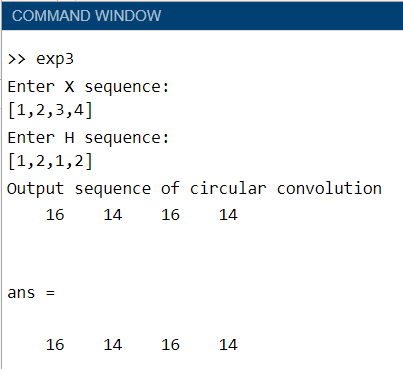
stem(n1,y);

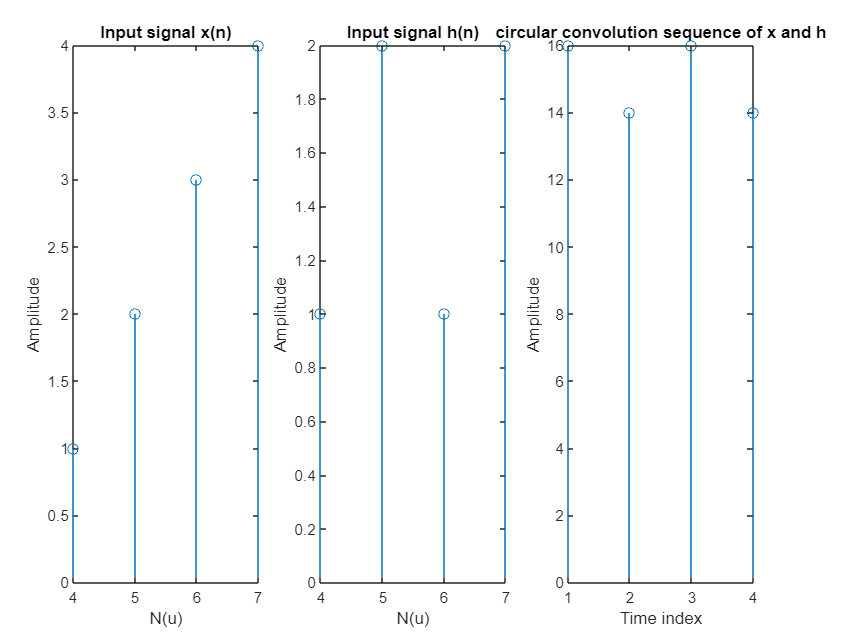
xlabel('Time index');

ylabel('Amplitude');

title('circular convolution sequence of x and h');

**Output:**





**Cross Correlation:**

disp("Cross Correlation");

org\_x = input('Enter origin index of signal x: ');

xmin = 1-org\_x;

x = input('Enter input signal x sequence: ');

xmax = xmin + length(x) -1;

org\_h = input('Enter origin index of signal h: ');

hfmin = 1-org\_h;

h = input('Enter input signal h sequence: ');

hf = fliplr(h);

hfmax = hfmin + length(h) -1;

min\_n = min(xmin, hfmin);

N = length(x) + length(h) - 1; y = zeros(1,N);

if(max(xmax,hfmax)==hfmax)

x = [x zeros(1,abs(hfmax-xmax))];

else

hf = [hf zeros(1,abs(xmax-hfmax))];

end

for i = 1:N

for j = 1:length(x)

try

y(i) = y(i) + (x(j)\*hf(i-j+1));

catch

end

end

end

disp("Result after Cross Correlation: ");

y

subplot(2,2,1);

stem((xmin:1:xmin+length(x)-1),x);

title('x(n) input signal');

xlabel('No. of samples (n)');

ylabel('Amplitude R(n)');

subplot(2,2,2);

stem((hfmin:1:hfmin+length(h)-1),h);

title('h(n) input signal');

xlabel('No. of samples (n)');

ylabel('Amplitude R(n)');

subplot(2,2,3);

stem((org\_h-length(h):1:org\_h-1),fliplr(h));

title('h(-n) Folded signal');

xlabel('No. of samples (n)'); ylabel('Amplitude R(n)');

subplot(2,2,4);

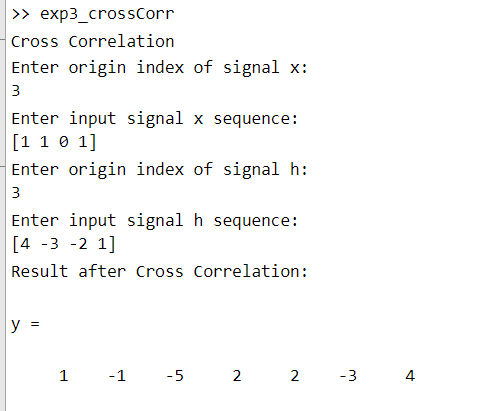
stem((min\_n:1:min\_n+length(y)-1),y);

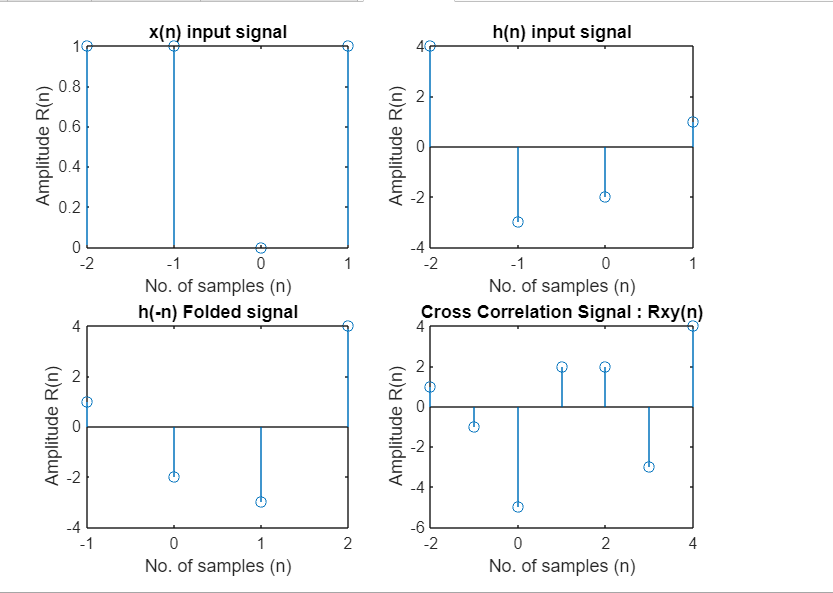
title('Cross Correlation Signal : Rxy(n)');

xlabel('No. of samples (n)');

ylabel('Amplitude R(n)');

**Output:**





**Auto correlation:**

disp("Auto Correlation");

org\_x = input('Enter origin index of signal x:');

xmin = 1-org\_x;

x = input('Enter input signal x sequence: ');

xf = fliplr(x);

min\_n = min(xmin, hfmin);

N = 2\*length(x)-1;

y = zeros(1,N);

for i = 1:N

for j = 1:length(x)

try

y(i) = y(i) + (x(j)\*xf(i-j+1));

catch

end

end

end

disp("Result after Auto Correlation is: ");

y;

subplot(2,2,1);

stem((xmin:1:xmin+length(x)-1),x);

title('x(n) input signal');

xlabel('No. of samples (n)');

ylabel('Amplitude R(n)');

subplot(2,2,2);

stem((org\_x-length(x):1:org\_x-1),xf);

title('x(-n) Folded signal');

xlabel('No. of samples (n)');

ylabel('Amplitude R(n)');

subplot(2,2,3);

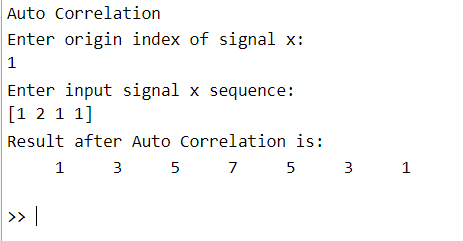
stem((min\_n:1:min\_n+length(y)-1),y);

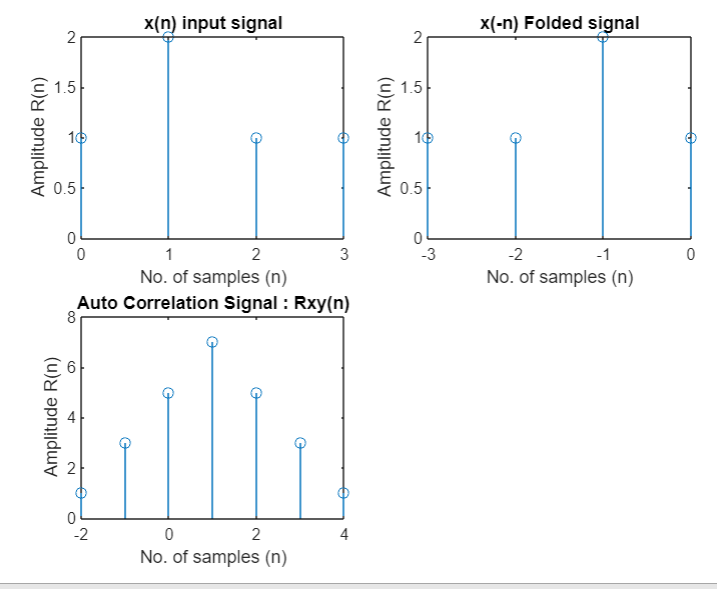
title('Auto Correlation Signal : Rxy(n)');

xlabel('No. of samples (n)');

ylabel('Amplitude R(n)');

**Output:**



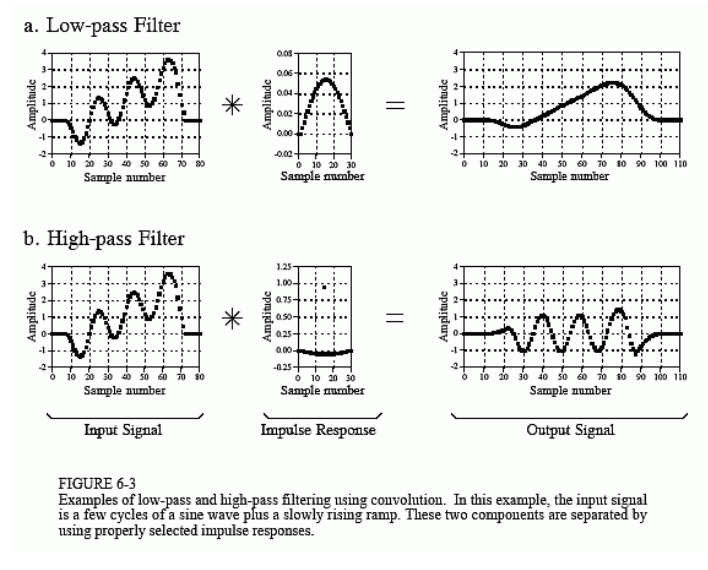


**Conclusion:-** We have successfully computed linear and circular convolution of two discrete time sequences using Matlab.

**Date: 17/02/2021 Signature of faculty in-charge**

**Post Lab Descriptive Questions**

* 1. Explain the role of convolution in signal processing.
* Convolution is a mathematical operation on two functions (f and g) that produces a third function expressing how the shape of one is modified by the other. The term convolution refers to both the result function and to the process of computing it. It is defined as the integral of the product of the two functions after one is reversed and shifted.
* Convolution has applications that include probability, statistics, computer vision, natural language processing, image and signal processing, engineering, and differential equations.
* Generalizations of convolution have applications in the field of numerical analysis and numerical linear algebra, and in the design and implementation of finite impulse response filters in signal processing.
* Figure below show convolution being used for low-pass and high-pass filtering. The example input signal is the sum of two components: three cycles of a sine wave (representing a high frequency), plus a slowly rising ramp (composed of low frequencies). In (a), the impulse response for the low-pass filter is a smooth arch, resulting in only the slowly changing ramp waveform being passed to the output. Similarly, the high-pass filter, (b), allows only the more rapidly changing sinusoid to pass.



* 1. Explain the difference between linear and circular convolution?

|  |  |  |
| --- | --- | --- |
| **Features** | **Linear Convolution** | **Circular Convolution** |
| **Definition** | To calculate the output for any linear time invariant system given its input and its impulse response. | Multiplication of two DFTs when the support of the signal is periodic |
| **Shifting** | Linear | Circular |
| **Number of samples in result** | N1+N2-1 | max(N1,N2) |
| **Domain** | Two sequences are multiplied in time domain | Two sequences are multiplied in frequency domain |

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

Ans:

Linear and circular convolution are fundamentally different operations. However, there are conditions under which linear and circular convolution are equivalent. Establishing this equivalence has important implications. For two vectors, x and y, the circular convolution is equal to the inverse discrete Fourier transform (DFT) of the product of the vectors' DFTs. Knowing the conditions under which linear and circular convolution are equivalent allows you to use the DFT to efficiently compute linear convolutions.

The linear convolution of an N-point vector, x, and an L-point vector, y, has length N + L - 1.

For the circular convolution of x and y to be equivalent, you must pad the vectors with zeros to length at least N + L - 1 before you take the DFT. After you invert the product of the DFTs, retain only the first N + L - 1 elements.

Create two vectors, x and y, and compute the linear convolution of the two vectors.

x = [2 1 2 1];

y = [1 2 3];

clin = conv(x,y);

The output has length 4+3-1.

Pad both vectors with zeros to length 4+3-1. Obtain the DFT of both vectors, multiply the DFTs, and obtain the inverse DFT of the product.

xpad = [x zeros(1,6-length(x))];

ypad = [y zeros(1,6-length(y))];

ccirc = ifft(fft(xpad).\*fft(ypad));

The circular convolution of the zero-padded vectors, xpad and ypad, is equivalent to the linear convolution of x and y. You retain all the elements of ccirc because the output has length 4+3-1.

Plot the output of linear convolution and the inverse of the DFT product to show the equivalence.

subplot(2,1,1)

stem(clin,'filled')

ylim([0 11])

title('Linear Convolution of x and y')

subplot(2,1,2)

stem(ccirc,'filled')

ylim([0 11])

title('Circular Convolution of xpad and ypad')

