**Digital Signal Processing**

M Usman RP-20EE-420

Um7995367@gmail.com

**Introduction:**

In digital signal processing (DSP), convolution plays a fundamental role in various applications, such as image processing, audio processing, and communication systems. It is a mathematical operation that combines two signals to produce a third signal, often used for filtering and feature extraction. At its core, convolution is a process of overlaying one signal, known as the input signal or the source signal, onto another signal called the impulse response or the kernel.

The kernel represents the system's characteristics or the filter that will be applied to the input signal. The output signal, also known as the convolution result, represents the combined effect of the input signal and the kernel. Mathematically, convolution is expressed as follows:

y[n] = ∑[k=0 to K-1] x[k] \* h[n-k]

where: y[n] is the output signal at time index n. x[k] is the input signal at time index k. h[n-k] is the kernel (impulse response) at time index n-k. K is the length of the kernel.

To perform the convolution, you slide the kernel over the input signal, multiplying each corresponding sample and summing the results. The result of this sum becomes the output signal at that specific time index. The process continues for all time indices to obtain the complete convolution result. Convolution has several important properties that make it widely used in DSP. One key property is linearity, meaning that if you scale the input signal, the output signal scales by the same factor. Convolution is also commutative, which means that the order of the input signal and the kernel can be swapped without affecting the result. In practice, convolution can be implemented using various algorithms, such as direct convolution, fast convolution (e.g., using the Fast Fourier Transform), or specialized hardware like digital signal processors (DSPs) or graphics processing units.

**MATLAB IMPLEMENTATION:**

**Task 01**

**CODE:**

clc

clear all

close all

x=[1,2,3,1];

nx=-1:2;

h=[1,2,1,-1];

nh=0:3;

nyb=nx(1)+nh(1);

nye=nx(length(x))+nh(length(h));

ny=nyb:nye;

y=conv(x,h);

title('example-1');

subplot(3,1,1);

stem(x);

xlabel('samples of x');

subplot(3,1,2);

stem(h);

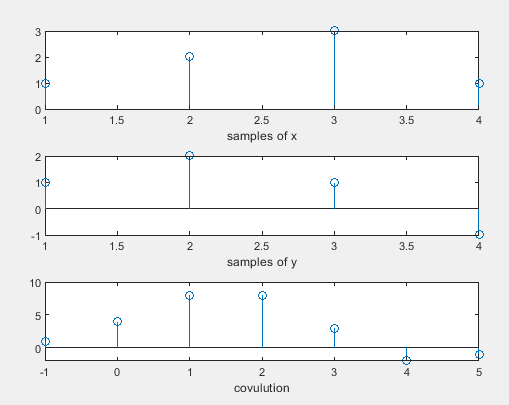
xlabel('samples of y');

subplot(3,1,3);

stem(ny,y);

xlabel('covulution');

**Result:**



**Task 2**

**code**

clc

clear all

close all

x=[1,4,5,1];

nx=-2:1;

h=[1,4,2,-1];

nh=-1:2;

nyb=nx(1)+nh(1);

nye=nx(length(x))+nh(length(h));

ny=nyb:nye;

y=conv(x,h);

title('example-1');

subplot(3,1,1);

stem(x);

xlabel('samples of x');

subplot(3,1,2);

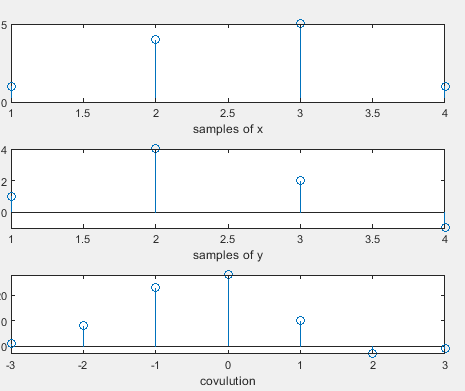
stem(h);

xlabel('samples of y');

subplot(3,1,3);

stem(ny,y);

xlabel('covulution');



**Task 3**

**Code**

clc

clear all

close all

x=[-1,4,6,2];

nx=0:4;

h=[4,3,2,-1];

nh=-2:1;

nyb=nx(1)+nh(1);

nye=nx(length(x))+nh(length(h));

ny=nyb:nye;

y=conv(x,h);

title('example-1');

subplot(3,1,1);

stem(x);

xlabel('samples of x');

subplot(3,1,2);

stem(h);

xlabel('samples of y');

subplot(3,1,3);

stem(ny,y);

xlabel('covulution');

