## Contents

## 1 | Experiment N0 01

## 2 | Experiment Name

Study of Linear Convolution and Implementation using MATLAB code.

## 3 | Introduction

The Discrete-Time Convolution (DTC) is one of the most important operations in discrete-time signal analysis.[1] The operation relates the output sequence y(n) of a linear-time invariant (LTI) system, with the input sequence x(n) and the unit sample sequence h(n), as shown in Fig. 1. In this experiment, we found convolution manually instead using built-in function 'conv' in ***matlab***.
Formula for convolution is:

$$y(n) = \sum_{k=1}^{n} x(k) * h(n-k)$$

The above formula is an approach of direct sum to find convolution. There is another tabular method for convolution as well. In this experiment we had implemented both using ***matlab*** code.

## 4 | Objectives

The main objectives of this experiments were:

- To find the convolution of the input sequence
- To develop an algorithm to find convolution without using 'conv' function

## 5 | Equipment Required

***MATLAB***

## 6 | Algorithm

### 6.1 | Direct Sum:

- **Step 1** Constructing input sequence X(n) & input response H(n) function
- **Step 2** Running two **for loop** i.e, outer loop for n times and inner loop for k times where n is the total length and k is the length of x
- **Step 3** Multiply the two sequence elementwise & store the sum in y(n)

### 6.2 | Tabular Method:

- **Step 1** Constructing input sequence X(n) & input response H(n) function
- **Step 2** Using **while loop** adding the upper traingle diagonal's element & store the sum in y(n).
- **Step 3** Repeated step 2 for lower traingle.

## 7 | Matlab Code

Here is the input code of convolution-

### 7.1 | Direct Sum Approah

```
x=input('Enter x: ');
h=input('Enter h: ');
xl=length(x);
hl=length(h);
```

```
X=[x,zeros(1,xl)];
H=[h,zeros(1,hl)];
for n=1:xl+hl-1
    y(n)=0;
    for k=1:xl
        if n-k+1>0
          y(n)=y(n)+X(k)*H(n-k+1);
        end
    end
end
%plotting
subplot(2,2,1);
stem(x);
xlabel('n');
ylabel('x(n)');
title('Input sequence')
subplot(2,2,2);
stem(h);
xlabel('n');
ylabel('H(n)');
title('Impulse Response')
subplot(2,2,[3,4]);
stem(y);
xlabel('n');
ylabel('y(n)');
title('Output sequence)')
```

## 7.2 | Tabular Method

```
%convolution using tabular method
x=input('Enter x: ');
h=input('Enter h: ');
len=length(x);
A=[];
y=[];
for n=1:len
    g=h(n).*x;
    A=[A;g];
end
n=1;
%adding left diagonal
while(n<=len)
    y(n)=0;i=1;j=n;
    while(i<=n && j>=1)
        y(n) = y(n)+A(i,j);
        i=i+1;
        j=j-1;
    end

    n=n+1;
end
%adding right diagonal
n=len;
while(n>=2)
    y(n+len-1)=0;i=n;j=len;
    while(i<=4 && j>=n)
        y(n+len-1) = y(n+len-1)+A(i,j);
```

```
          i=i+1;
          j=j-1;
      end
      n=n-1;
end
```

## 8 | Result & Discussion

Here is the outcome of above code-

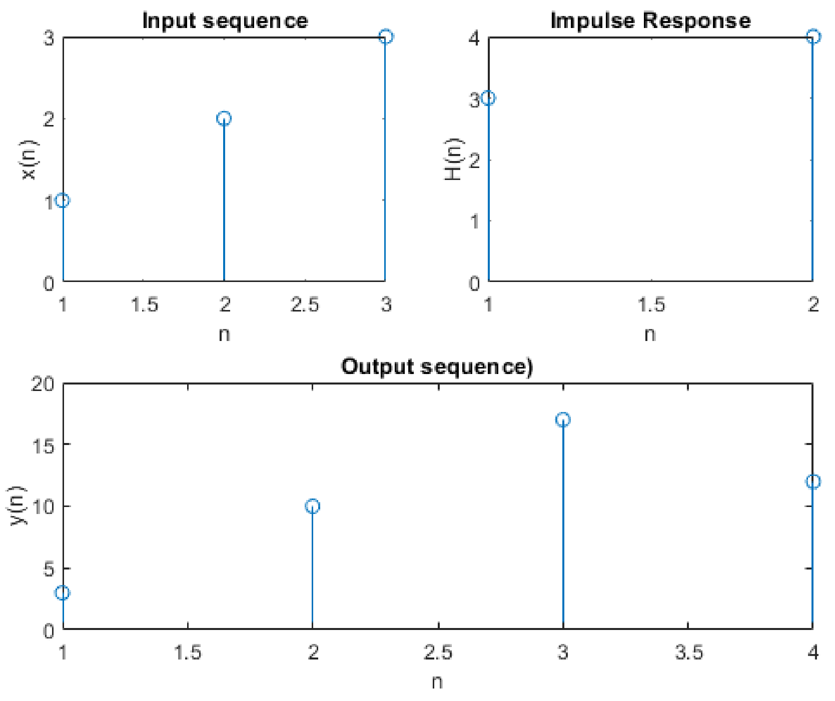

**Figure 8.1:** Graphical Plot of x(n), h(n), and y(n) signals

The outcomes of this experiment were achieved as desired i.e. the convolution of input signals without using the 'conv' function. The direct sum approach handled inputs of different lengths i.e. the size of these two signals need not be the same necessarily. On the contrary, the tabular method used here needs both inputs to be of the same size. Because the code could add the right & left diagonal elements of a square matrix

## 9 | Conclusion

The experiment was successful & we did not encounter any error while running the ***matlab*** code.

## 10 | References

[1] Discrete Time Convolution Properties — Discrete Time Signal, 11 2017.