INSTITUTE OF ENGINEERING
CENTRAL CAMPUS,PULCHOWK

DIGITAL SIGNAL PROCESSING

LAB #3

Convolution

Submitted BY:

Name: Amrit Prasad Phuyal

Campus Roll: 074BEX403

IOE Roll: PUL074BEX004

Submitted To:

Anila Kansakar

Department of Electronics and Computer Engineering

January 16, 2022

Table of Contents

List of Matlab and Python codes

List of Figures

1 Title

Convolution

2 Objective

- To be able to perform convolution of two given signal using basic formula.
- To be able to perform convolution of two given signals using Matlab function.
- To be able to perform convolution of two given signals using Python package.

3 Theory

3.1 Matlab Function Used

`Conv,Sinc etc.`

3.2 Python Package Used

`NumPy and Matlplotlib`

3.3 Background

3.3.1 Convolution Sum

The output of any Linear Time Invariant (LTI) system is some sort of operation between input and system response; the operation is nothing but convolution, denoted by symbol {*}, and defined as

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

$$y[n] = x[n] * h[n] = \int_{k=-\infty}^{\infty} x[k]h[n-k] \text{ - - - For Discrete Time}$$

For a causal LTI system, the convolution sum is given by,

$$y[n] = \int_{k=0}^{n} x[k]h[n-k]$$

The process of computing the convolution between x[k] and h[k] involves the following four steps:

1. **Folding**: Fold $h[k]$ about $k = 0$ to obtain $h[-k]$.

```
function [y,n]=sigfold(x,n)
    y=fliplr(x);
    n=-fliplr(n);
end
```

Code 1: Matlab Function for Folding Signal

2. **Shifting**: Shift $h[-k]$ by n_0 to the right (left if n_o is positive (negative), to obtain $h[n_o - k]$.

```
function[y,n]=sigshift(x,m,n0)
    n=m+n0;
    y=x;
end
```

Code 2: Matlab Function for Shifting Signal

3. **Multiplication**: Multiply $x[k]$ by $h[n_o - k]$ to obtain the product sequence $V_{n_o}[k]$ where, $V_{n_o}[k] = x[k].h[n_o - k]$.

```
function[y,n]=sigmulti(x1,n1,x2,n2)
    n=min(min(n1),min(n2)):max(max(n1),max(n2));
    y1=zeros(1,length(n));
    y2=y1;
    y1(find((n>=min(n1))&(n<=max(n1))==1))=x1;
    y2(find((n>=min(n2))&(n<=max(n2))==1))=x2;
    y=y1.*y2;
end
```

Code 3: Matlab Function for Multiplication of Signal

4. **Summation**: Sum all the values of the product sequence V_{n_o} to obtain the value of the output at times $n = n_o$.

3.3.2 Example

Find convolution of x[n] and h[n], where

```
x=[1,0,-1,1,2,1]
n1=[-2,-1,0,1,2,3]
h=[1,1,1,1,1]
n2=[0,1,2,3,4]
```

1. **Graphical Method**

```
x=[1,0,-1,1,2,1];
n1=[-2,-1,0,1,2,3];
h=[1,1,1,1,1];
n2=[0,1,2,3,4];
nmin=min(min(n1),min(n2));
nx=length(x);
nh=length(h);
n=nmin:nx+nh+nmin-2;
y_graphical=zeros(1,length(n));
[x_folded,n_folded]=sigfold(x,n1);
temp=1;
for i=n
    [x_shift,n_shift]=sigshift(x_folded,n_folded,i);
    [x_multi,n_multi]=sigmulti(x_shift,n_shift,h,n2);
    for j=1 :length(n_multi)
        y_graphical(temp)=y_graphical(temp)+x_multi(j);
    end
    temp=temp+1;
end

stem(n,y_graphical,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
title('Graphical method')
```

Code 4: Matlab code for Convolution using Graphical method

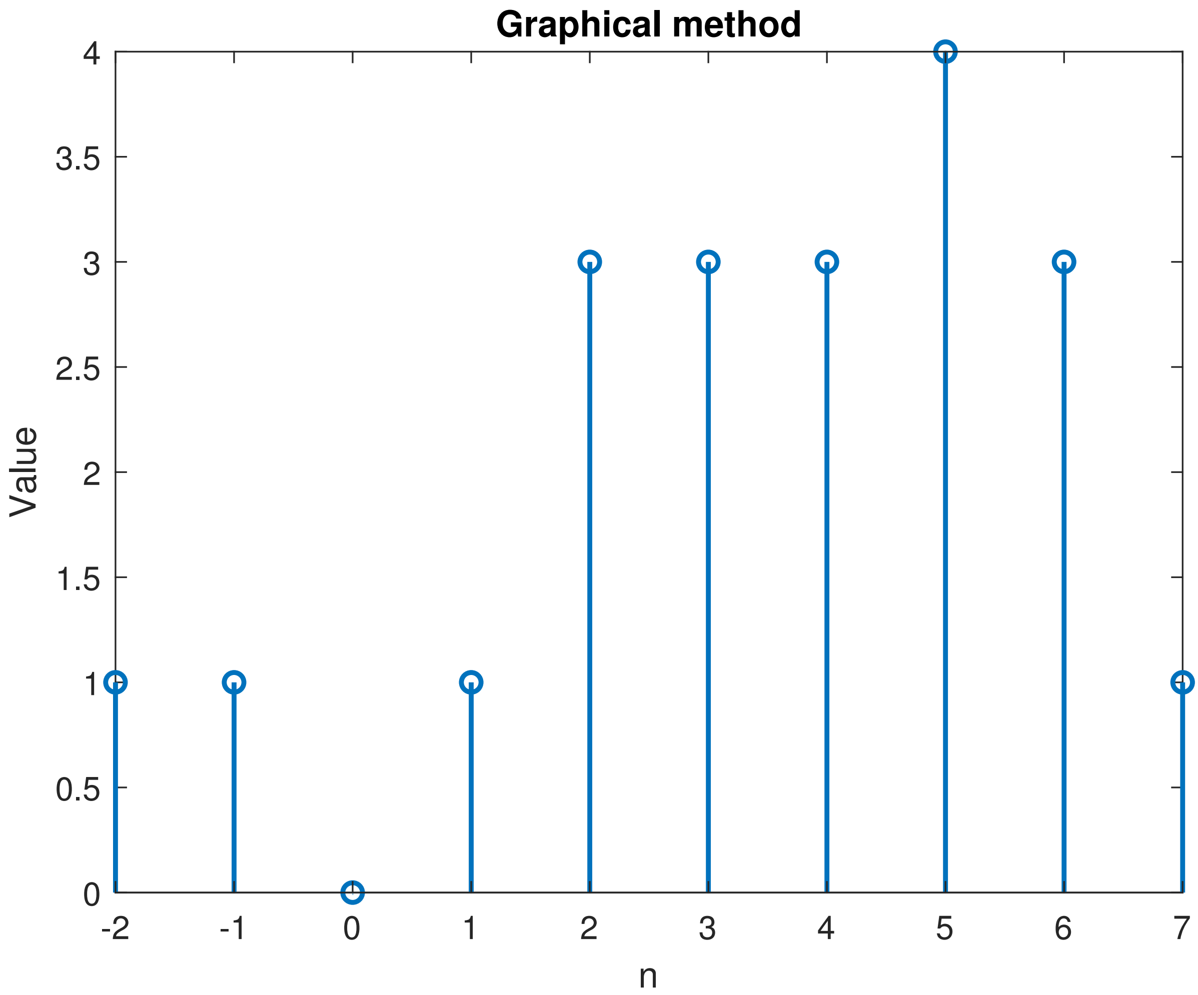

Figure 1: Convolution using Graphical method

2. **Tabular Method**

```
x=[1,0,-1,1,2,1];
n1=[-2,-1,0,1,2,3];
h=[1,1,1,1,1];
n2=[0,1,2,3,4];
nmin=min(min(n1),min(n2));
nx=length(x);
nh=length(h);
n=nmin:nx+nh+nmin-2;

y_tabular=zeros(1,length(n));
prod=x'.*h;
for i = 1:nx
    y_tabular(i:i+nh-1)=y_tabular(i:i+nh-1)+prod(i,:);
end

stem(n,y_tabular,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
title('Tabular method')
```

Code 5: Matlab code for Convolution using Tabular method

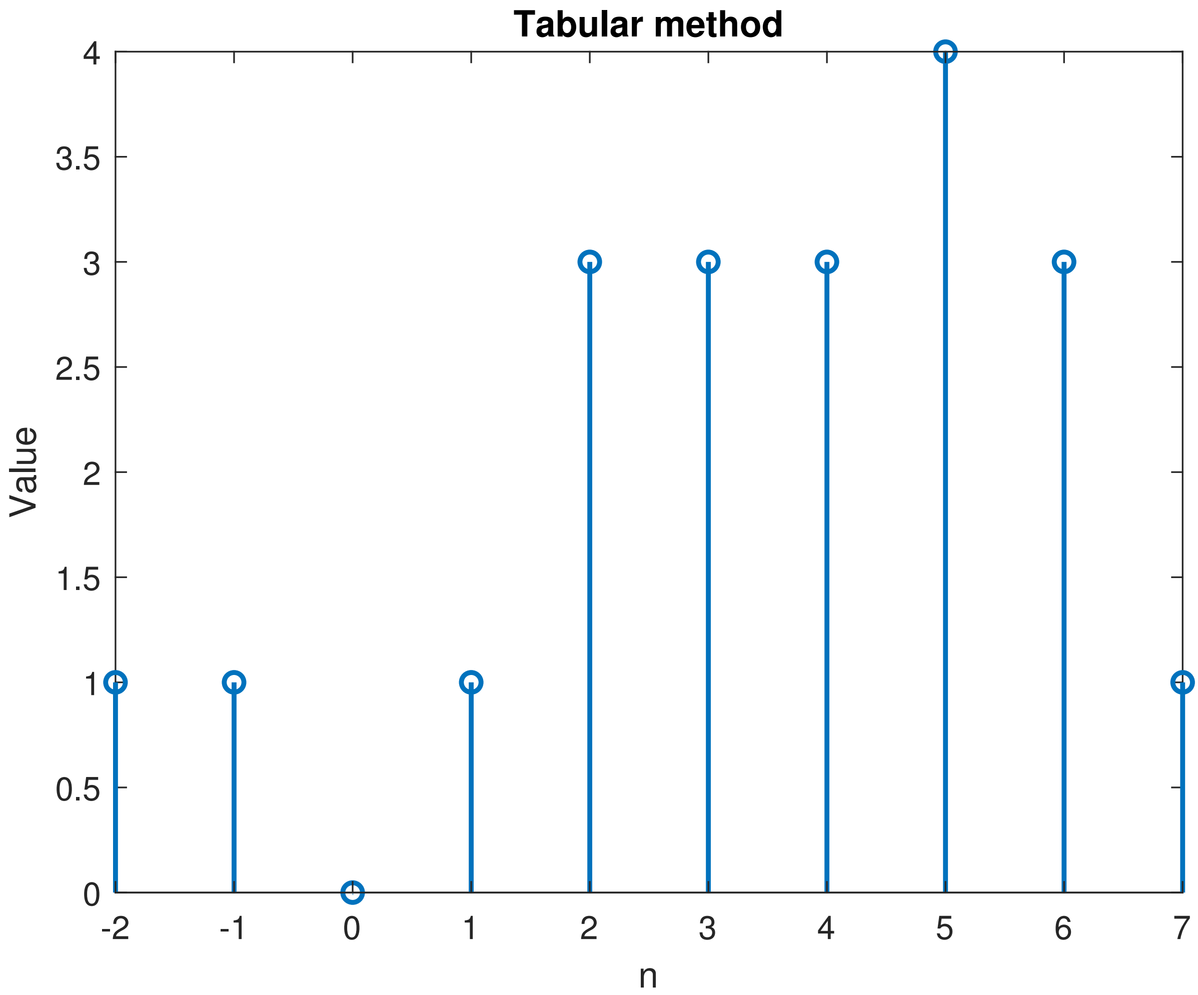

Figure 2: Convolution using Tabular method

3. ***conv* Function**

```
x=[1,0,-1,1,2,1];
n1=[-2,-1,0,1,2,3];
h=[1,1,1,1,1];
n2=[0,1,2,3,4];
nmin=min(min(n1),min(n2));
nx=length(x);
nh=length(h);
n=nmin:nx+nh+nmin-2;

y_conv=conv(x,h);

stem(n,y_conv,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
title('Conv function')
```

Code 6: Matlab code for Convolution using *conv* function

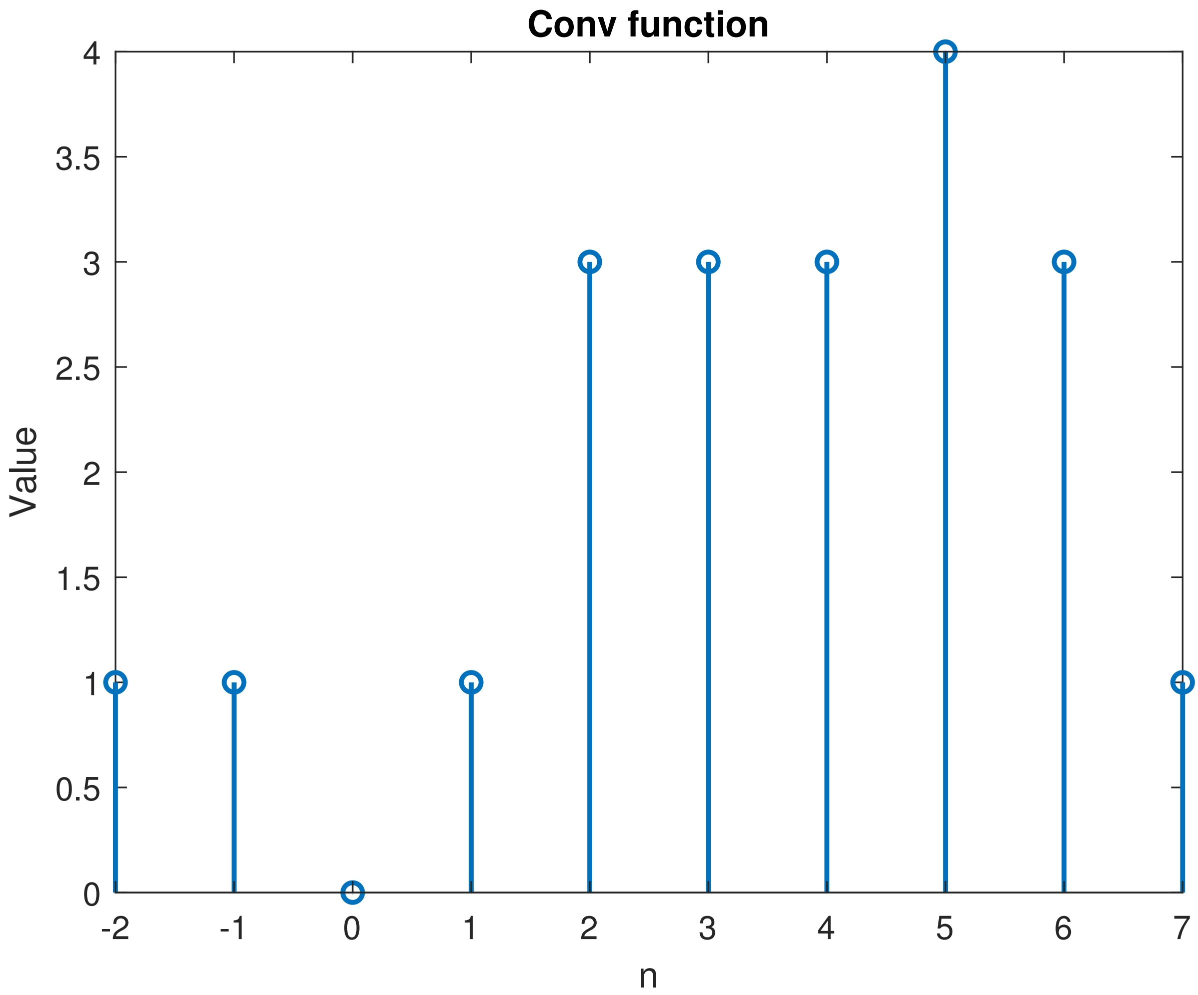

Figure 3: Convolution using *conv* function

4 Lab Problems

4.1 Problem 1

Find the convolution result of the following signal using basic convolution formula :

```
X1(n1)=[1,1,1,1,1]
n1=[-2,-1,0,1,2]
X2(n2)=[1,0,0,0,0,0,0,0,0,0]
n2=[-4,-3,-2,-1,0,1,2,3,4,5]
X2 is a periodic signal.
Y2=X1*X2
```

Matlab

```
x=[1,1,1,1,1];
n1=[-2,-1,0,1,2];
h=[1,0,0,0,0,0,0,0,0,0];
n2=[-4,-3,-2,-1,0,1,2,3,4,5];
nmin=min(min(n1),min(n2));
nx=length(x);
nh=length(h);
```

```
n=nmin:nx+nh+nmin-2;
y=zeros(1,length(n));
prod=x'.*h;
for i = 1:nx
    y(i:i+nh-1)=y(i:i+nh-1)+prod(i,:);
end
l=tiledlayout(1,1);
title(l,{'Convolution using basic convolution formula'})
nexttile
stem(n,y,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
```

Code 7: Matlab code for Convolution using basic convolution formula

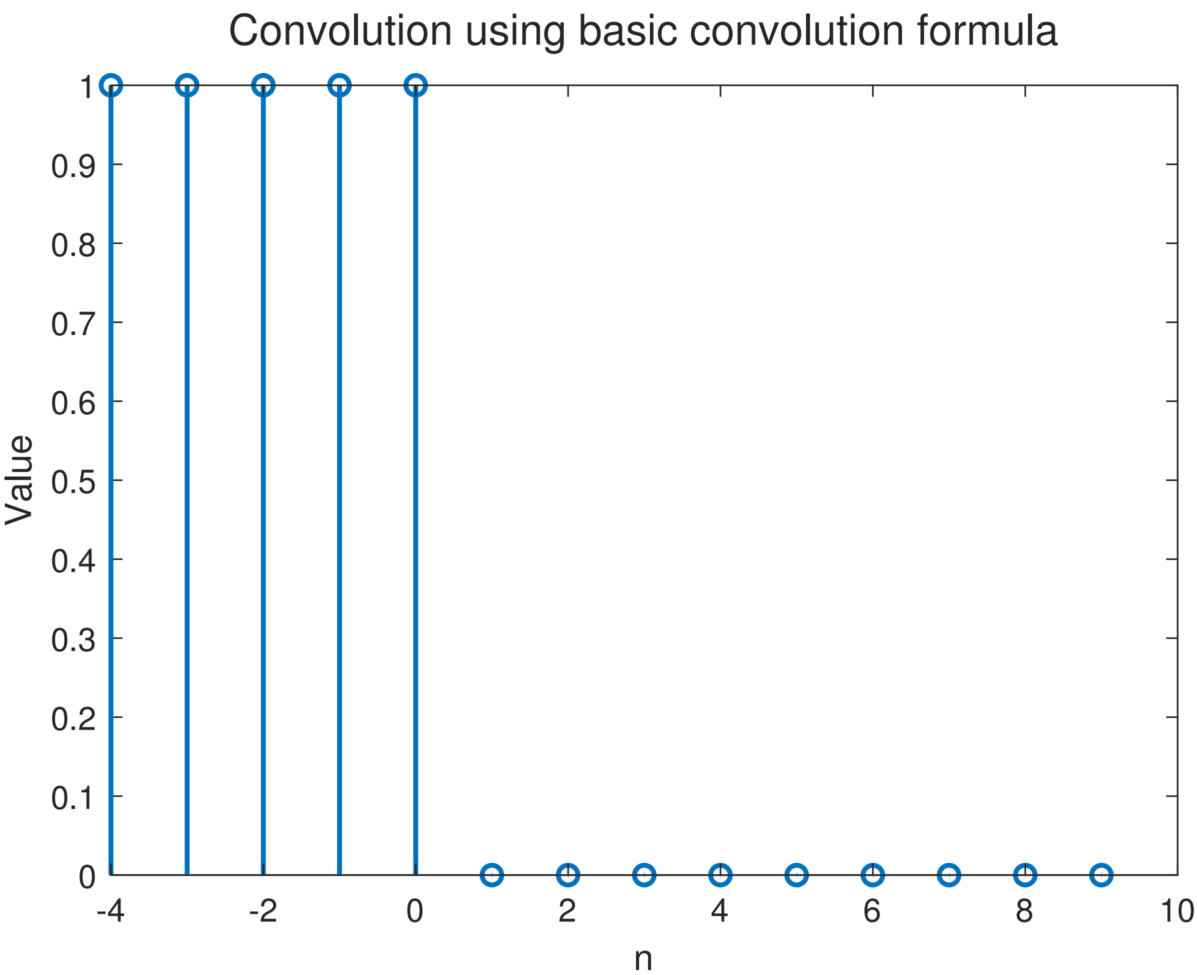

Figure 4: Convolution using basic formula {Matlab}

Python

```
import numpy as np
from matplotlib import pyplot as plt
x1 = np.array([1, 1, 1, 1, 1])
n1 = np.arange(start=-2, stop=3)
x2 = np.array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0])
n2 = np.arange(start=-4, stop=6)
```

```
n_min = min(np.amin(n1), np.amin(n2))
n = np.arange(start=n_min, stop=n_min+n1.size+n2.size-2+1)
prod = np.transpose(np.column_stack(x1))*x2
y = np.zeros(n.size)
for i in range(0, x1.size):
    y[i:i+x2.size] = y[i:i+x2.size]+prod[i, :]
plt.stem(n, y)
plt.title('Convolution using basic convolution formula')
plt.xlabel('n')
plt.ylabel('Value')
plt.savefig('FIG/p1p.eps', format='eps')
plt.show()
```

Code 8: Python code for convolution using basic convolution formula

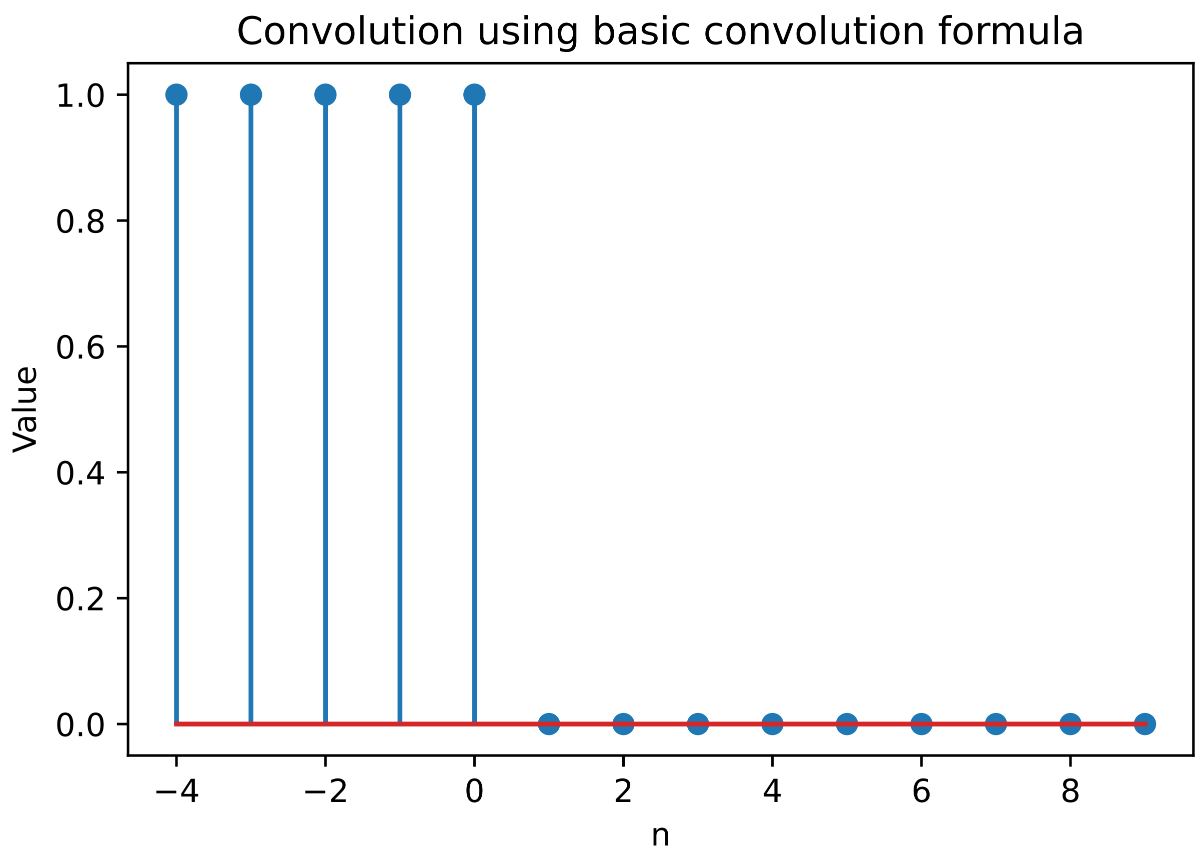

Figure 5: Convolution using basic convolution formula {Python}

4.2 Problem 2

Find the convolution of using {`conv`} function for matlab and {`convolve`} functon from numpy package for python

a.

$$x[n] = \begin{cases} \frac{1}{3}n & \text{for} \quad 0 \le n \le 6 \\ 0 & \text{else} \end{cases} \qquad \text{and,} \quad h[n] = \begin{cases} 1 & \text{for} \quad -2 \le n \le 2 \\ 0 & \text{else} \end{cases}$$

Matlab

```
n1=0:6;
x=n1./3;
n2=-2:2;
h=n2 >= -2 & n2 <= 2;
nmin=min(min(n1),min(n2));
nx=length(x);
nh=length(h);
n=nmin:nx+nh+nmin-2;
y=conv(x,h);
l=tiledlayout(1,1);
title(l,{'DT convolution using conv function'})
nexttile
stem(n,y,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
```

Code 9: Matlab code for DT Convolution using {conv} function

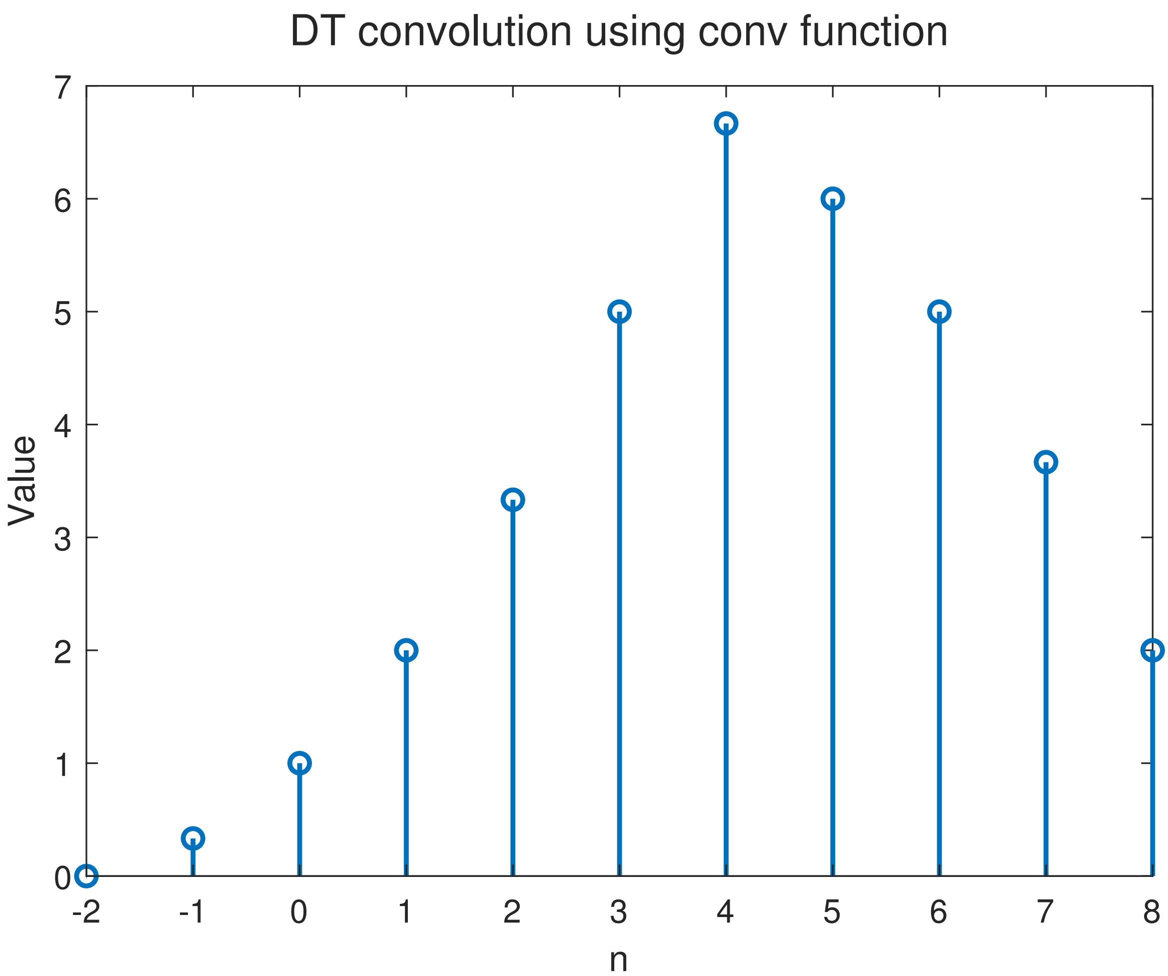

Figure 6: DT Convolution using {conv} function {Matlab}

Python

```
import numpy as np
from matplotlib import pyplot as plt
n1 = np.arange(start=0, stop=7)
```

```
x = np.where(np.logical_and(n1 >= 0, n1 <= 6), n1/3, 0)
n2 = np.arange(start=-2, stop=3)
h = np.where(np.logical_and(n2 >= -2, n2 <= 2), 1, 0)
n_min = min(np.amin(n1), np.amin(n2))
n = np.arange(start=n_min, stop=n_min+n1.size+n2.size-2+1)
y = np.convolve(x, h)
plt.stem(n, y)
plt.title('DT convolution using convolve function')
plt.xlabel('n')
plt.ylabel('Value')
plt.savefig('FIG/p2ap.eps', format='eps')
plt.show()
```

Code 10: Python code for DT Convolution using {convolve} function

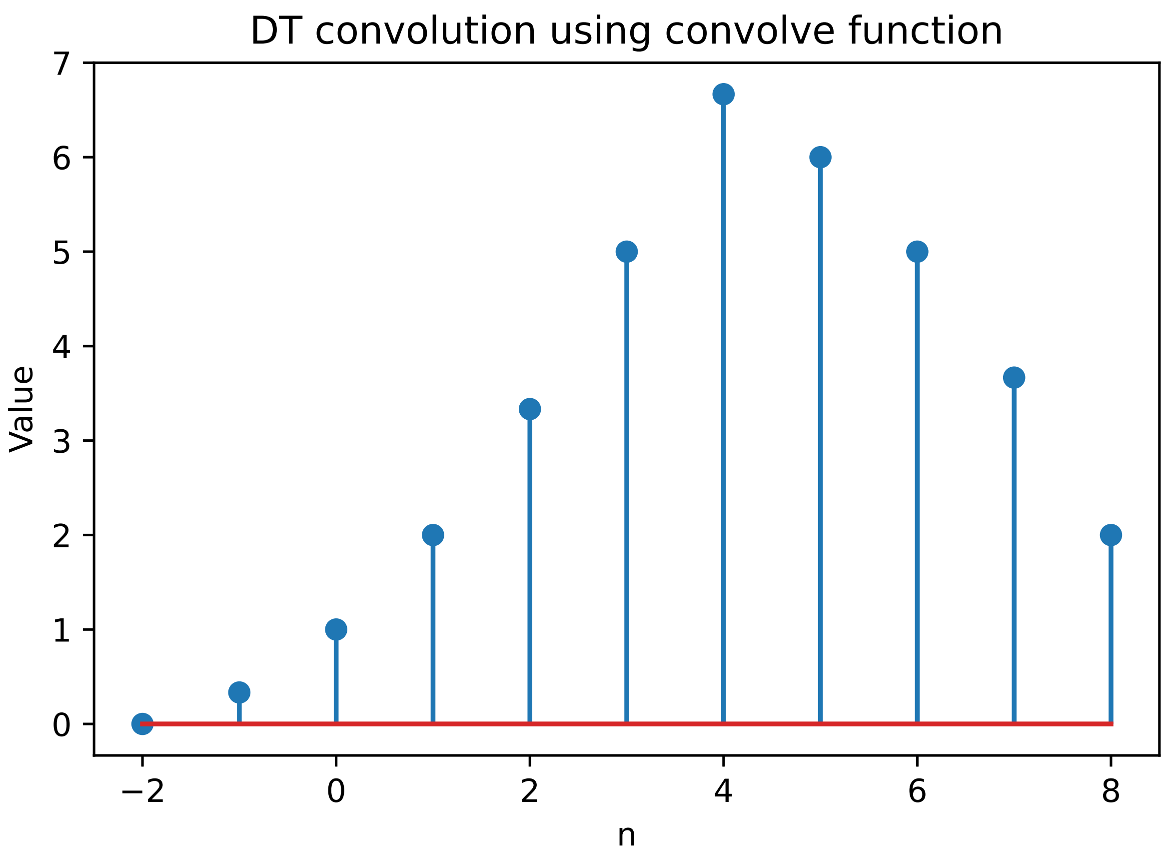

Figure 7: DT Convolution using {convolve} function {Python}

b.

$$x(t) = u(t) \quad \text{and,} \quad h(t) = e^{-at}u(t), \text{ where } a > 0$$

Matlab

```
a=1;
t=linspace(-5,5,1000);
x=t>=0;
h=x.*exp(-a.*t);
y=conv(x,h,'same');
l=tiledlayout(3,1);
title(l,{'CT convolution using conv function'})
```

```
xlabel(l,'t')
ylabel(l,'Amplitude')
nexttile
plot(t,x,'Linewidth',1.5)
ylabel('x(t)')
nexttile
plot(t,h,'Linewidth',1.5)
ylabel('h(t)')
nexttile
plot(t,y,'Linewidth',1.5)
ylabel('y(t)')
```

Code 11: Matlab code for CT Convolution using {conv} function

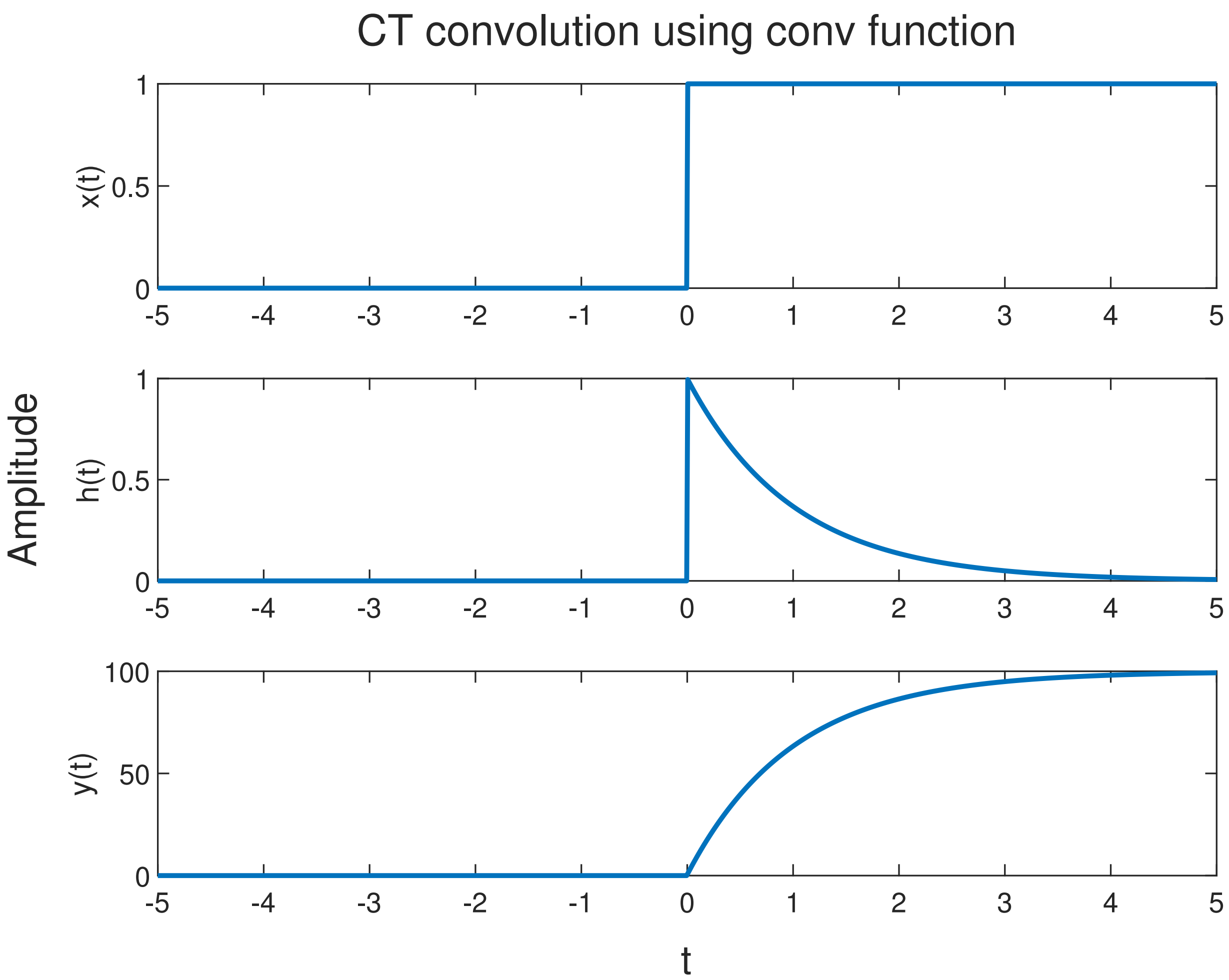

Figure 8: CT Convolution using {conv} function {Matlab}

Python

```
import numpy as np
from matplotlib import pyplot as plt
a = 1
t = np.linspace(-5, 5, 1000)
x = np.where(t >= 0, 1, 0)
h = x*np.exp(-a*t)
y = np.convolve(x, h, mode='same')
```

```
fig, (ax1, ax2, ax3) = plt.subplots(3,1,constrained_layout=True)
fig.suptitle('CT convolution using convolve function')
fig.supxlabel('t')
fig.supylabel('Amplitude')
ax1.plot(t, x)
ax1.set_ylabel('$x(t)$')
ax2.plot(t, h)
ax2.set_ylabel('$h(t)$')
ax3.plot(t, y)
ax3.set_ylabel('$y(t)$')
plt.savefig('FIG/p2bp.eps', format='eps')
plt.show()
```

Code 12: Python code for CT Convolution using {convolve} function

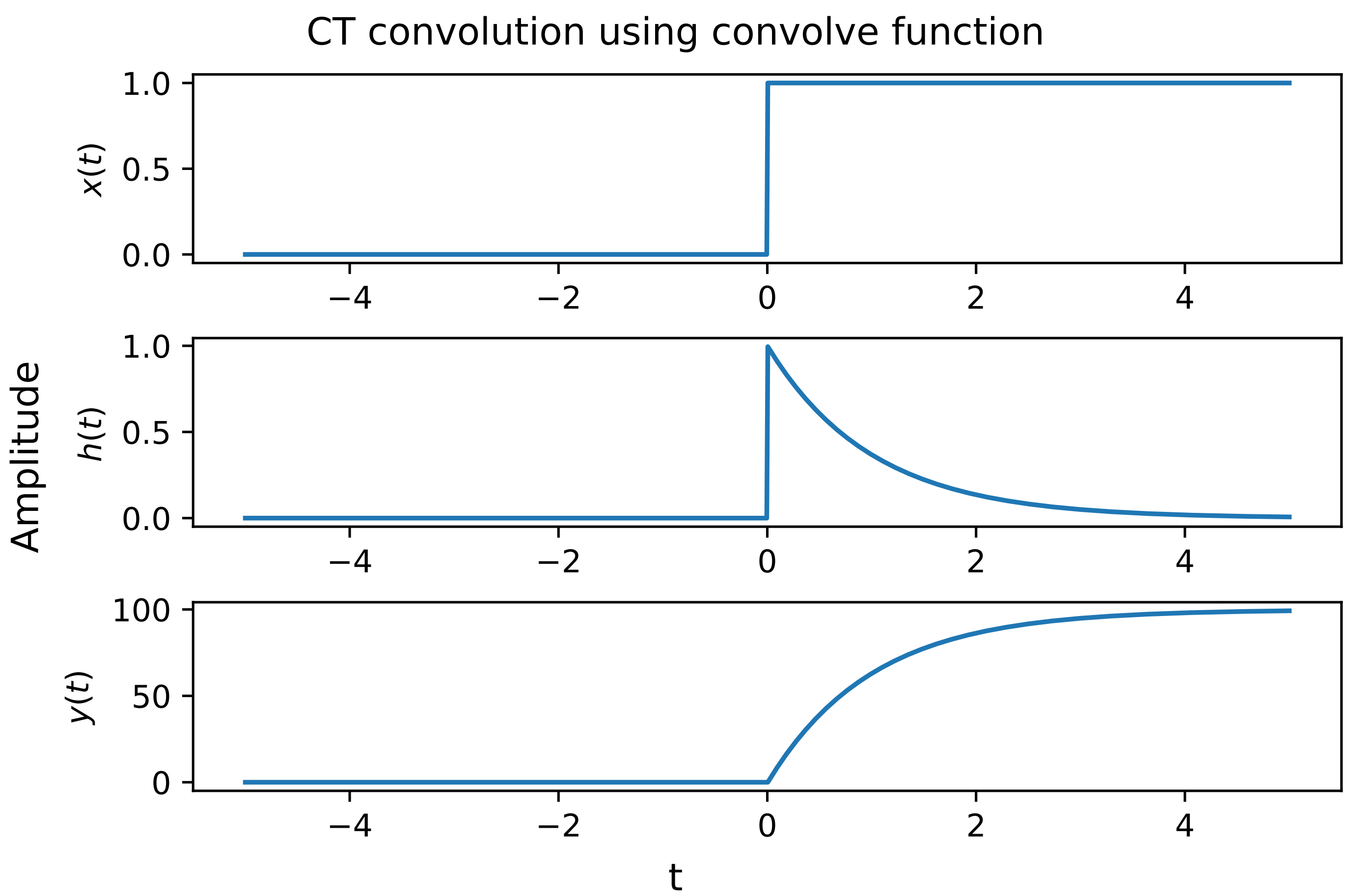

Figure 9: CT Convolution using {convolve} function {Python}

4.3 Problem 3

Consider two discrete time sequences $x[n]$ and $h[n]$ given by

$$x[n] = \begin{cases} 1 & for \quad 0 \leq n \leq 4 \\ 0 & elsewhere \end{cases} \qquad \textbf{and,} \quad h[n] = \begin{cases} 2^n & for \quad 0 \leq n \leq 6 \\ 0 & elsewhere \end{cases}$$

a. Find the response of the LTI system with impulse response $h[n]$ to input $x[n]$.

b. Plot the signals and comment on the result.

Matlab

```
n1=0:4;
x=n1>=0&n1<=4;
n2=0:6;
h=(n2>=0&n2<=6).*(2.^n2);
nmin=min(min(n1),min(n2));
nx=length(x);
nh=length(h);
n=nmin:nx+nh+nmin-2;
y=conv(x,h);
l=tiledlayout(1,1);
title(l,{'Response of LTI system'})
nexttile
stem(n,y,'Linewidth',1.5)
xlabel('n')
ylabel('y[n]')
```

Code 13: Matlab code to find the response of the LTI system

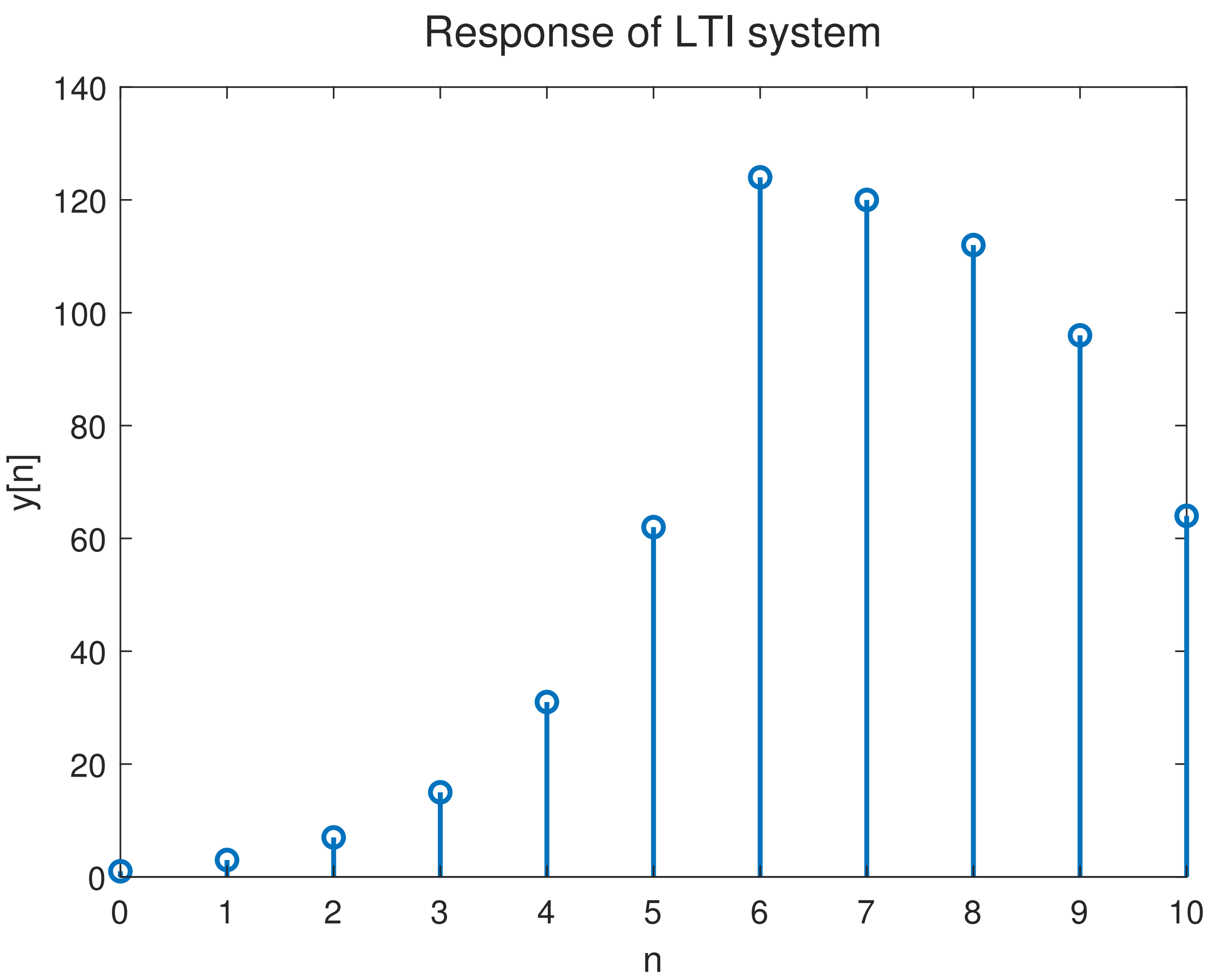

Figure 10: Response of LTI system {Matlab}

Python

```
import numpy as np
from matplotlib import pyplot as plt
n1 = np.arange(start=0, stop=5)
```

```python
x = np.where(np.logical_and(n1 >= 0, n1 <= 4), 1, 0)
n2 = np.arange(start=0, stop=7)
h = np.where(np.logical_and(n2 >= 0, n2 <= 6), 2**n2, 0)
n_min = min(np.amin(n1), np.amin(n2))
n = np.arange(start=n_min, stop=n_min+n1.size+n2.size-2+1)
y = np.convolve(x, h)
plt.stem(n, y)
plt.title('Response of LTI system')
plt.xlabel('n')
plt.ylabel('Value')
plt.savefig('FIG/p3p.eps', format='eps')
plt.show()
```

Code 14: Python code to find the response of the LTI system

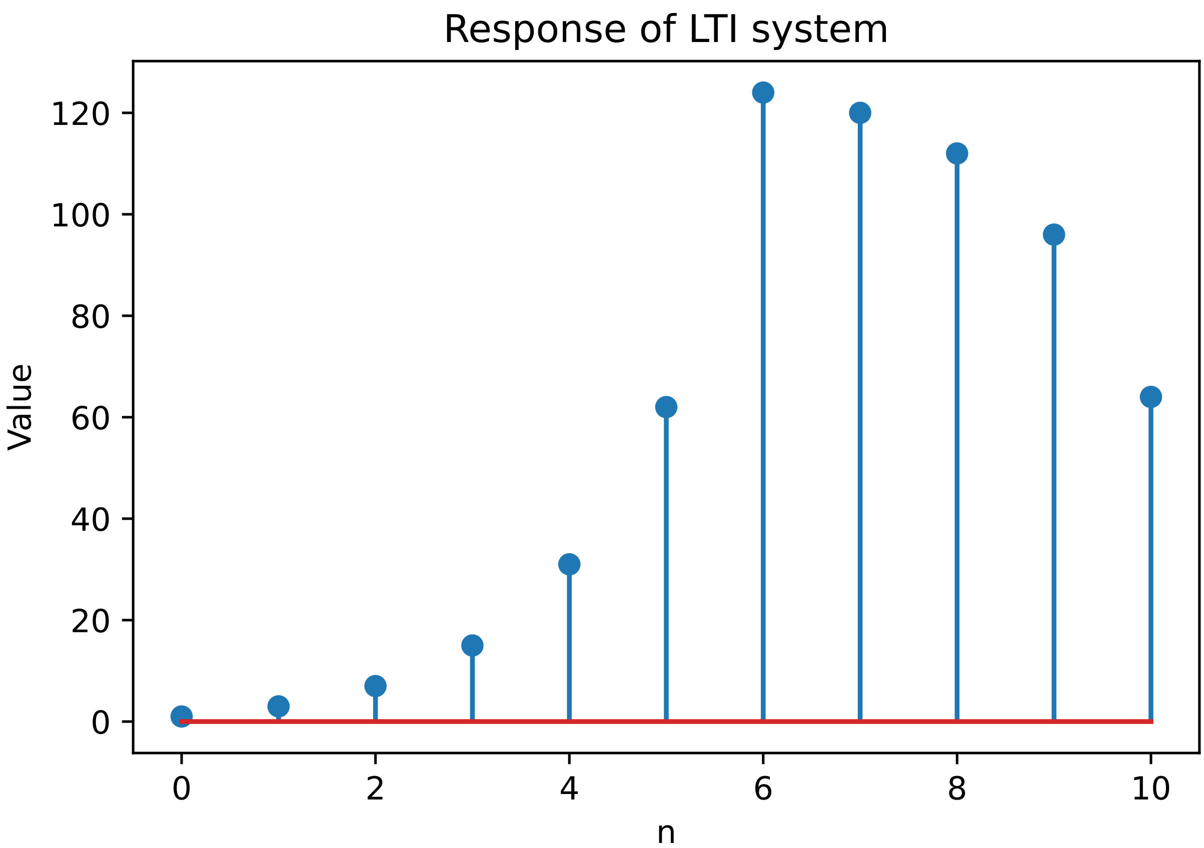

Figure 11: Response of LTI system {Python}

4.4 Problem 4

If the impulse response $h(t)$ of a LTI system is given by `sinc` function and input signal $x(t)$ is a rectangular wave given as,

$$h(t) = \frac{2\tau}{T_p} sinc\left(\frac{2\tau t}{T_p}\right) \quad \textbf{and,} \quad x(t) = \begin{cases} 1 & \textbf{for} \quad 1 \le t \le 100 \\ 0 & \textbf{elsewhere} \end{cases}$$

Find output of the system for different values of τ. Comment on the result.

Matlab

```
t=linspace(-100,100,1000);
x=t>=1;
tau=double(input('Enter value of TAU: '));
Tp=50;
h=2*tau/Tp.*sinc(t.*2*tau/Tp);
y=conv(x,h,'same');
l=tiledlayout(1,1);
title(l,{'Response of LTI system',sprintf('for t=%d',tau)})
nexttile
plot(t,y,'Linewidth', 1.5)
xlabel('x')
ylabel('y(t)')
```

Code 15: Matlab code to find output of the LTI system where sinc function as impulse response

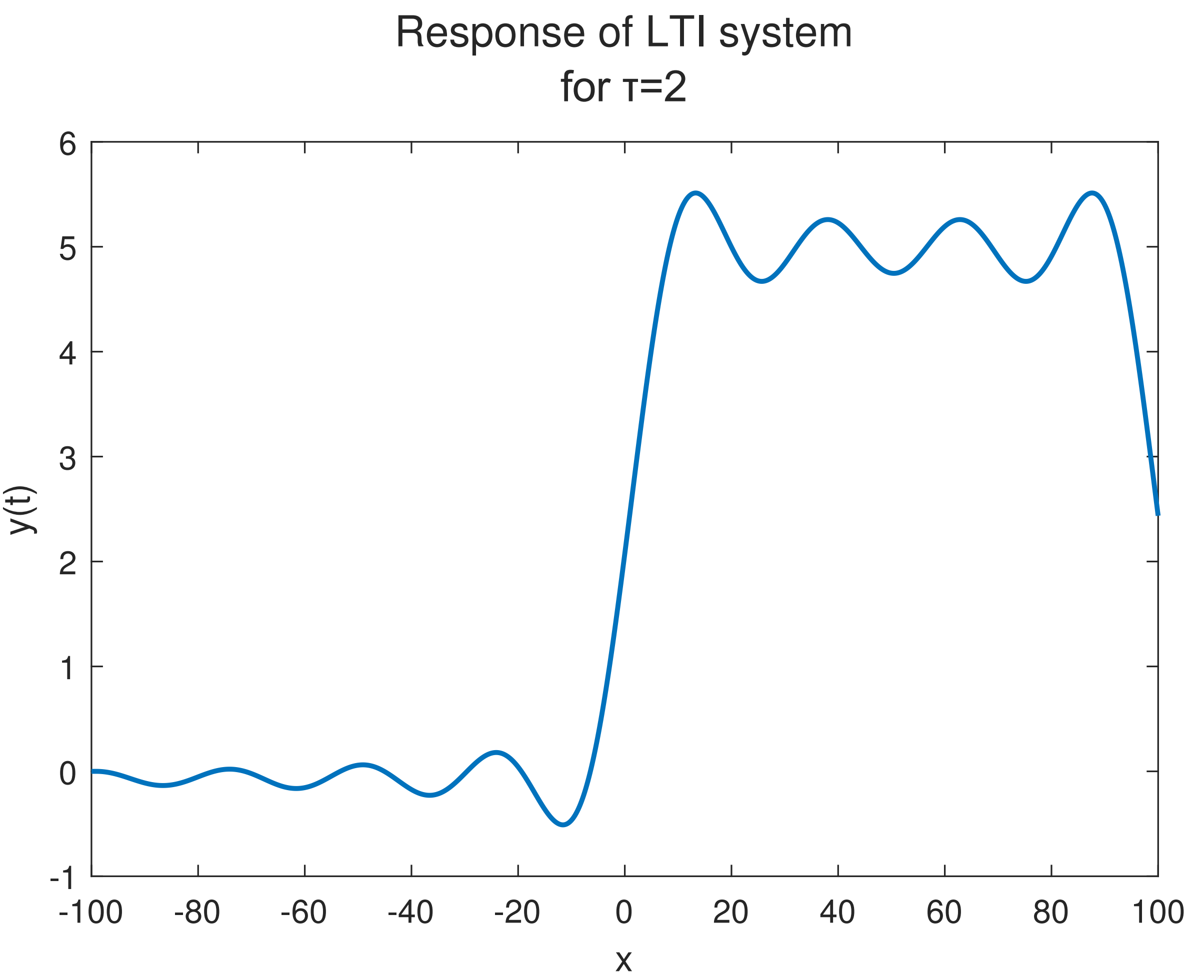

Figure 12: Output of the LTI system where sinc function as impulse response {Matlab}

Python

```
import numpy as np
from matplotlib import pyplot as plt
t = np.linspace(-100, 100, 1000)
x = np.where(np.logical_and(t>=1,t<=100), 1, 0)
tau=float(input('Enter value of T: '))
Tp=50
```

```
h = 2*tau/Tp*np.sinc(t*2*tau/Tp)
y = np.convolve(x, h, mode='same')
plt.plot(t, y)
plt.title('Response of LTI system with sinc function as impulse response\n with T
    ={:.2f} '.format(tau))
plt.xlabel('t')
plt.ylabel('y(t)')
plt.savefig('FIG/p4p.eps', format='eps')
plt.show()
```

Code 16: Python code to find output of the LTI system where sinc function as impulse response

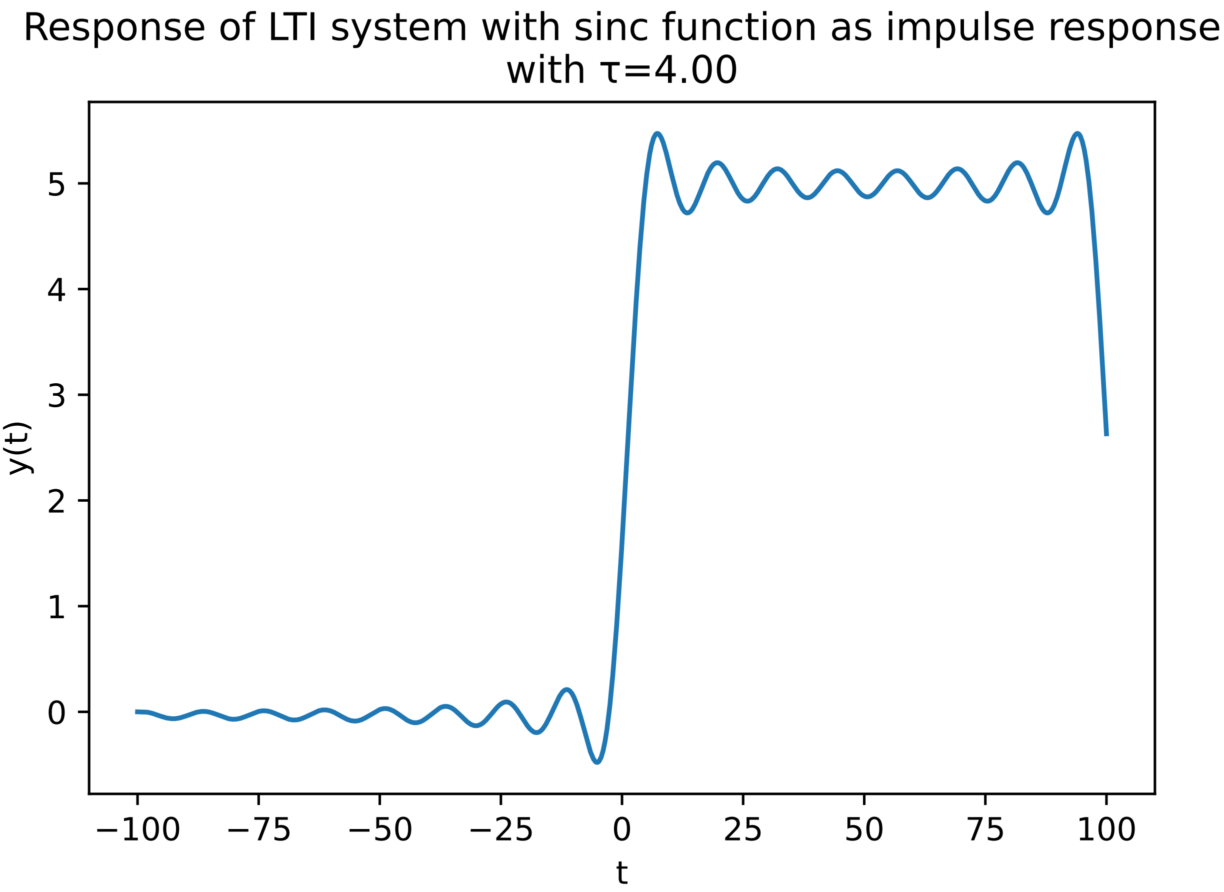

Figure 13: Output of the LTI system where sinc function as impulse response {Python}

5 Discussion and Conclusion

In this Lab we familiarize ourself with Matlab and Python Programming (Numpy and Matplotlib package) with basic of Convolution with Continuous time and Discrete time signals. We explored Graphical, Tabular and builtin functions of Matlab and Python (numpy package). We also used the concept of convolution and convolution theorem to find the response of the LTI system. We experiment with {`conv`} function of matlab and {`convolve`} function of numpy package. We used the concept of DT and CT signals to find the response of the LTI system. We used the concept of sinc function to find the output of the LTI system. To visualize result in python we used the concept of {`plot`} function of matplotlib package.