Review of Convolution

Lab Exercises on January 2, 2022

Department of Electronics and Computer Engineering

Pulchowk Campus, Lalitpur

Ashlesh Pandey

PUL074BEX007

Contents

List of Figures

Listings

1 Objectives

- To be able to perform convolution of two given signal using basic formula.
- To be able to perform convolution of two given signals using Python (`NumPy`) functions.

2 Functions Used

- `conv`: `conv` is an in-built Matlab function to find convolution of input arrays.
- `sinc`: `sinc` is an in-built Matlab function to return the $sinc$ response of the input.

3 Packages Used

- `NumPy`: `NumPy` is a fundamental package for scientific computing with Python.
- `Matplotlib`: `Matplotlib` is a comprehensive library for creating static, animated, and interactive visualizations in Python.

4 Background Theory

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 $\circledast$, and defined as,

$$y(t) = x(t) \circledast h(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \tag{1}$$

$$y[n] = x[n] \circledast h[n] = \int_{k=-\infty}^{\infty} x[k]h[n-k] \tag{2}$$

Equation 1 is known as convolution integral and is defined for continuous time signals where as Equation 2 is known as convolution sum and is defined for discrete time signals.

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

$$y[n] = \int_{k=0}^{n} x[k]h[n-k] \tag{3}$$

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
```

Listing 1: Matlab function for folding

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_m(x,m,n0)
    n=m+n0;
    y=x;
end
```

Listing 2: Matlab function for shifting

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
```

Listing 3: Matlab function for multiplication

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

Example: *Convolution of signals $x[n]$ and $h[n]$ by graphical, tabular and* `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_graphical=zeros(1,length(n));
[x_folded,n_folded]=sigfold(x,n1);
temp=1;
for i=n
    [x_shift,n_shift]=sigshift_m(
    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
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
y_conv=conv(x,h);
l=tiledlayout(3,1);
title(l,{'Example convolution from
    labsheet','(PUL074BEX007)'})
xlabel(l,'n')
ylabel(l,'Value')
nexttile
stem(n,y_graphical,'Linewidth'
    ,1.5)
title('Graphical method')
nexttile
stem(n,y_tabular,'Linewidth',1.5)
title('Tabular method')
nexttile
stem(n,y_conv,'Linewidth',1.5)
title('Conv function')
print('../Figures/
    lab_3_example_all','-depsc')
```

Listing 4: Matlab script to find convolution of example using graphical, tabular and conv

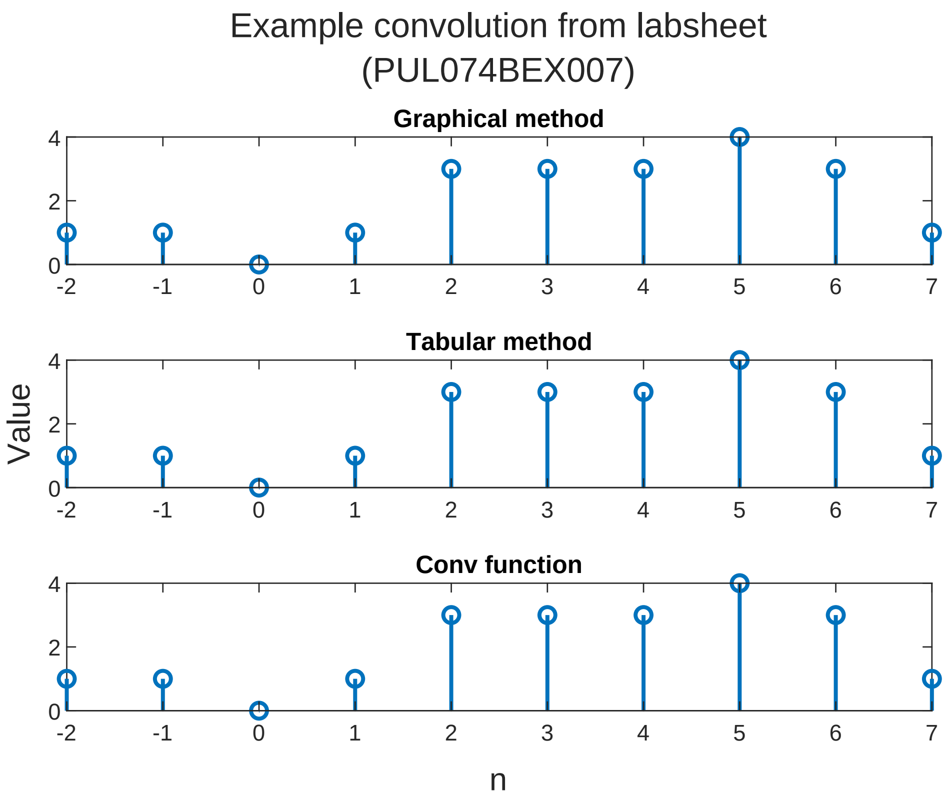

Figure 1: Example convolution using graphical, tabular and conv

5 Lab Exercises

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
```

```
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','(PUL074BEX007)'})
nexttile
stem(n,y,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
print('../Figures/lab_3_1_ml','-depsc')
```

Listing 5: Matlab script for convolution using basic convolution formula

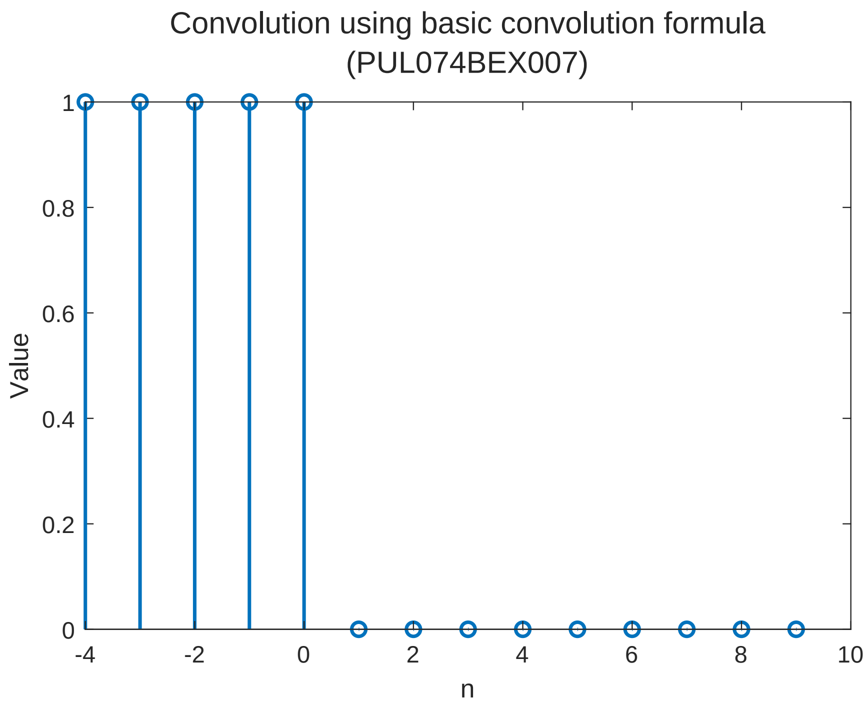

Figure 2: Observation of convolution using basic convolution formula (Matlab)

```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\n(PUL074BEX007)')
plt.xlabel('n')
plt.ylabel('Value')
plt.savefig('../Figures/lab_3_1_py.eps', format='eps')
plt.show()
```

Listing 6: Python script for convolution using basic convolution formula

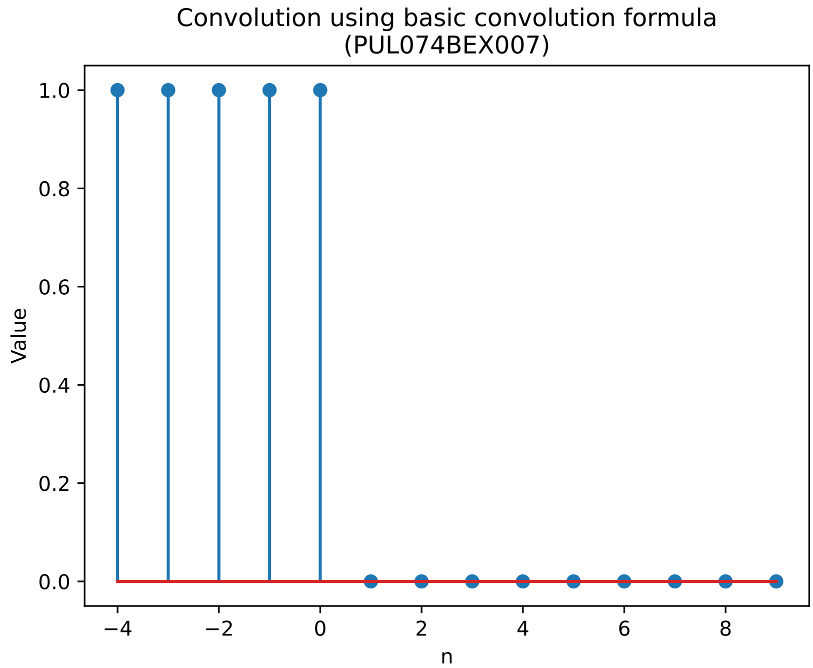

Figure 3: Observation of convolution using basic convolution formula (Python)

Problem 2

Find the convolution using `conv` and `convolve` function.

a.

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

```
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','(PUL074BEX007)'
    })
nexttile
stem(n,y,'Linewidth',1.5)
xlabel('n')
ylabel('Value')
print('../Figures/lab_3_2_a_ml','-
    depsc')
```

Listing 7: Matlab script for DT convolution using conv function

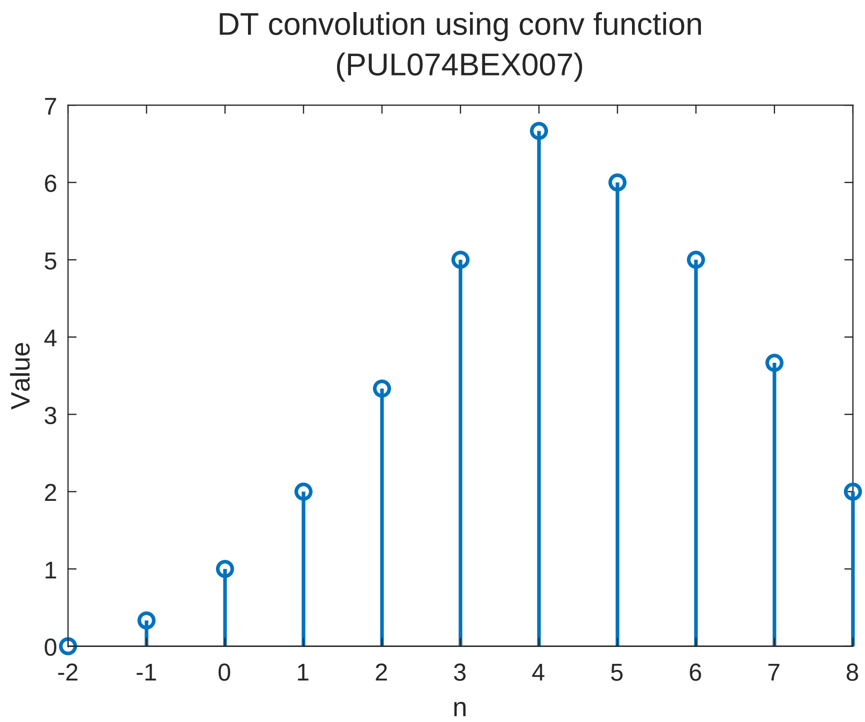

Figure 4: Observation of DT convolution using conv function (Matlab)

```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\n(PUL074BEX007)')
plt.xlabel('n')
plt.ylabel('Value')
plt.savefig('../Figures/lab_3_2_a_py.eps', format='eps')
plt.show()
```

Listing 8: Python script for DT convolution using convolve function

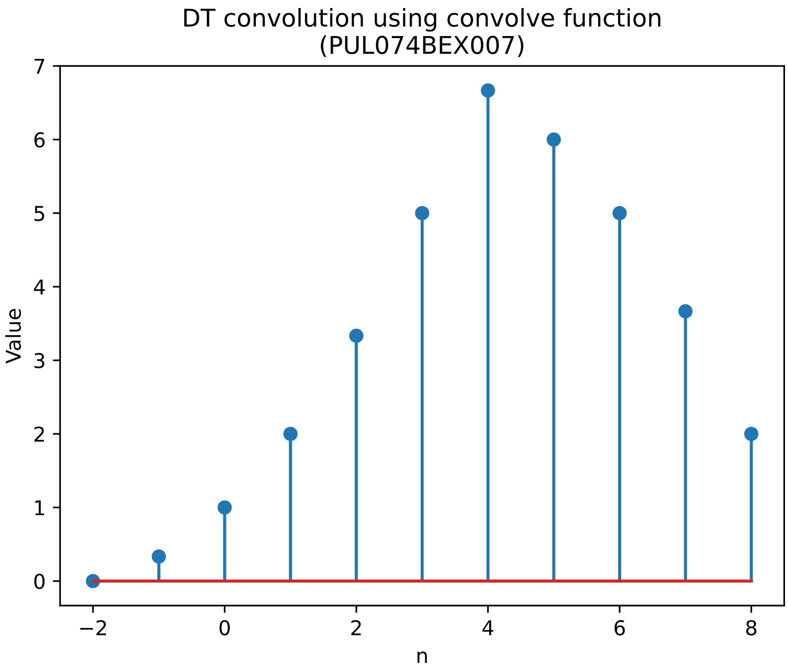

Figure 5: Observation of DT convolution using convolve function (Python)

b.

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

```
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','(PUL074BEX007)'
    })
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)')
```

```
print('../Figures/lab_3_2_b_ml','-depsc')
```

Listing 9: Matlab script for CT convolution using conv function

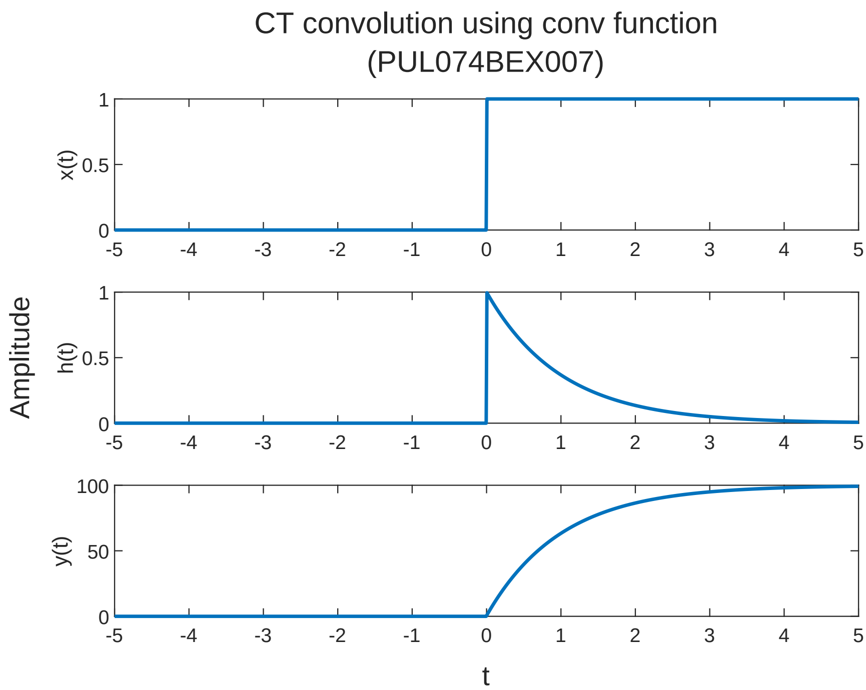

Figure 6: Observation of CT convolution using conv function (Matlab)

```
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\n(PUL074BEX007)')
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('../Figures/lab_3_2_b_py.eps', format='eps')
plt.show()
```

Listing 10: Python script for CT convolution using convolve function

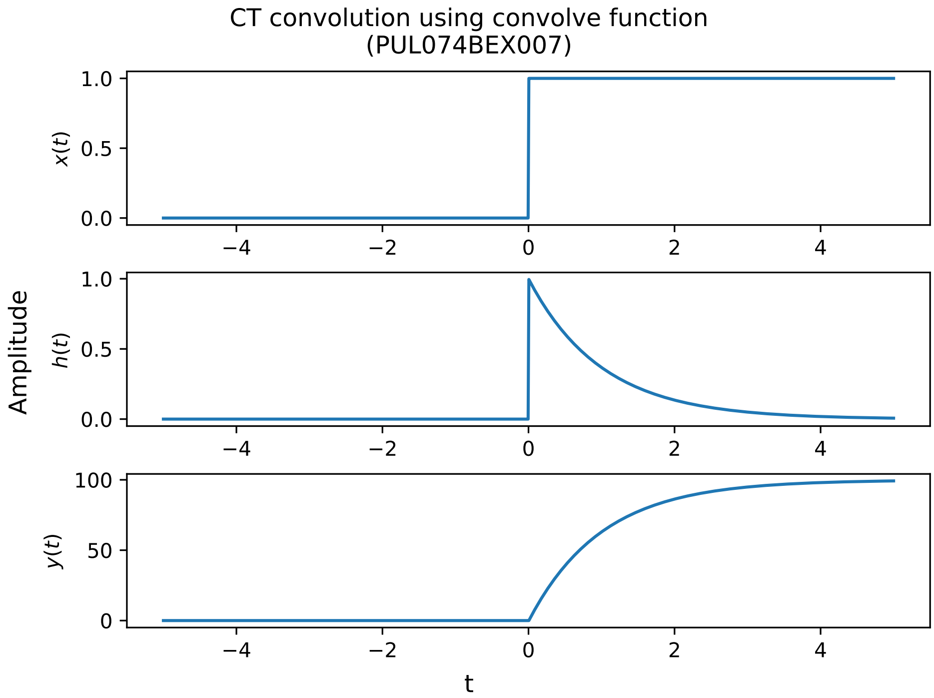

Figure 7: Observation of CT convolution using convolve function (Python)

Problem 3

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

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

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.

```
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);
```

```matlab
nh=length(h);
n=nmin:nx+nh+nmin-2;
y=conv(x,h);
l=tiledlayout(1,1);
title(l,{'Response of LTI system',
    '(PUL074BEX007)'})
nexttile
stem(n,y,'Linewidth',1.5)
xlabel('n')
ylabel('y[n]')
print('../Figures/lab_3_3_ml','-
    depsc')
```

Listing 11: Matlab script for finding response of LTI system

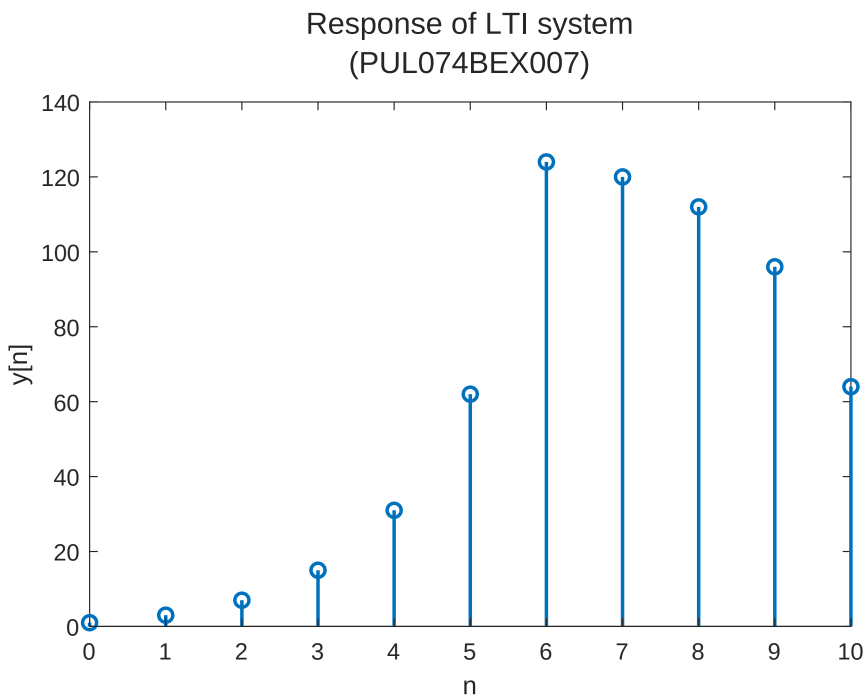

Figure 8: Observation of response of LTI system (Matlab)

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

```python
plt.stem(n, y)
plt.title('Response of LTI system\n(PUL074BEX007)')
plt.xlabel('n')
plt.ylabel('Value')
plt.savefig('../Figures/lab_3_3_py.eps', format='eps')
plt.show()
```

Listing 12: Python script for finding response of LTI system

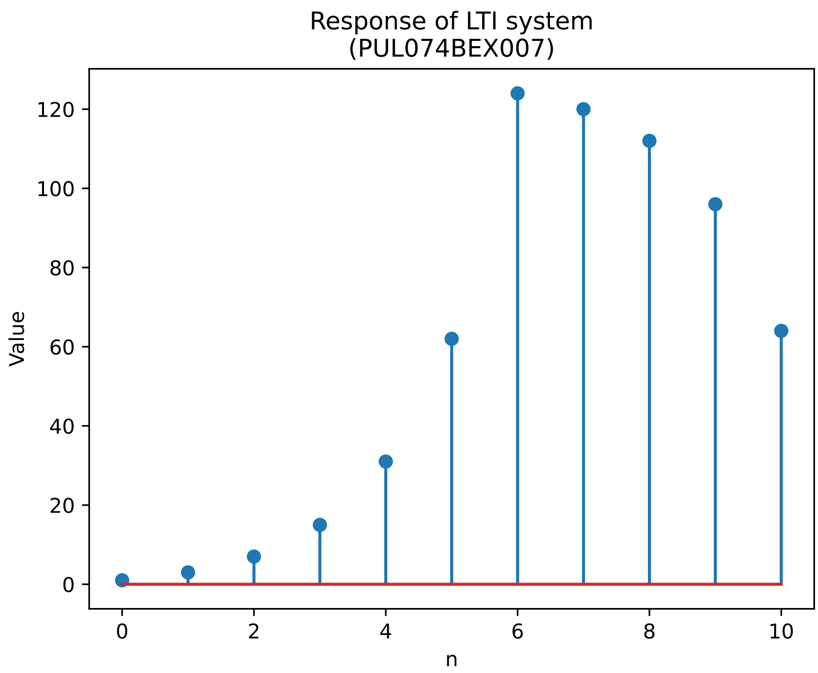

Figure 9: Observation of response of LTI system (Python)

The output response for the given LTI system goes on increasing up to $n = 6$ and then decays for the rest of the observable time.

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,

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

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

```
t=linspace(-100,100,1000);
x=t>=1;
tau=double(input('Enter value of  : '));
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 with sinc function as impulse response',sprintf('for  =%d',tau),'(PUL074BEX007)'})
nexttile
plot(t,y,'Linewidth', 1.5)
xlabel('x')
ylabel('y(t)')
print('../Figures/lab_3_4_ml','-depsc')
```

Listing 13: Matlab script for finding response of LTI system with sinc function as impulse response

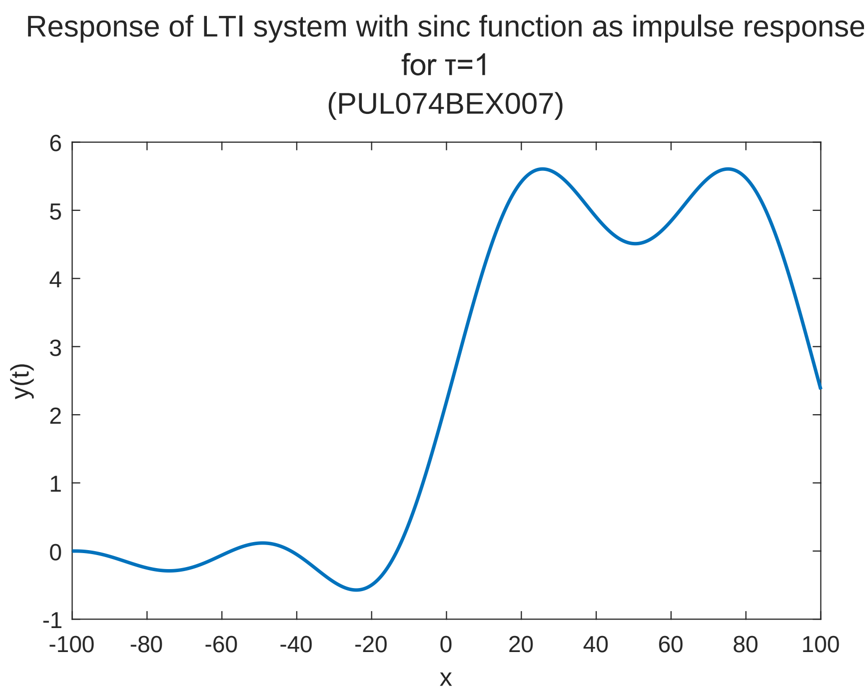

Figure 10: Observation of response of LTI system with sinc function as impulse response (Matlab)

```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 : '))
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\nwith ={:.2f} \n(PUL074BEX007)'.format(tau))
plt.xlabel('t')
plt.ylabel('y(t)')
plt.savefig('../Figures/lab_3_4_py.eps', format='eps')
plt.show()
```

Listing 14: Python script for finding response of LTI system with sinc function as impulse response

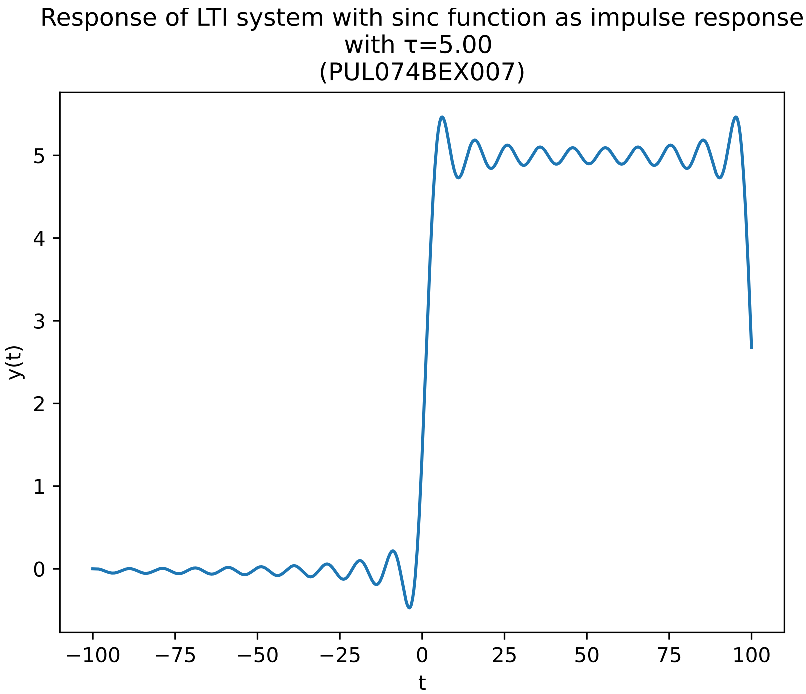

Figure 11: Observation of response of LTI system with sinc function as impulse response (Python)

The output response for the given LTI system where the impulse response is a sinc function changes with change in the value of τ.

6 Discussion and Conclusion

In this lab experiment, we performed convolution using different techniques. Initially, with the example problem in the lab sheet, convolution using graphical, tabular and in-built `conv` function was visualized in Matlab. Then, we performed convolution with the basic formula in both Matlab and Python. Following that, we performed convolution of DT and CT signals with `conv` function in Matlab and `convolve` function in Python using `NumPy`. The output response of an LTI system in DT was also visualized, which was the convolution sum of the input $x[n]$ and impulse response $h[n]$. Finally, output response of an LTI system in CT with the `sinc` function as the impulse response was visualized.
Hence, the objectives of the lab experiment were fulfilled.