National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department of Electrical Engineering and Computer Science

Faculty Member: Dr. Salman Ghafoor Dated: 17/10/2022

Semester: 5th Section: BEE 12C

EE-232: Signals and Systems

Lab 6: Convolution

Group Members

Name	Reg. No	PLO4 – CLO3 Viva / Quiz / Lab Performance	PLO5 - CLO3 Analysis of data in Lab Report	PLO8 - CLO4 Modern Tool Usage	PLO9 - CLO4 Ethics and Safety
		5 Marks	5 Marks	5 Marks	5 Marks
Danial Ahmad	331388				
Muhammad Umer	345834				
Syeda Fatima Zahra	334379				

1 Table of Contents

2 Introduction to Properties of Systems

2.1 Objectives

The goal of this exercise is to gain familiarity with the convolution function provided in MATLAB and to write a function that would perform the convolution operation. The convolution examples considered in this lab will relate only to the case of discrete time signals.

- How to use the convolution operator in MATLAB
- How to write a function that will perform convolution operation
- Applications of Convolution Operator
- How to use spfirst convolution GUIs

2.2 Equipment

Software

- *MATLAB*

2.3 Lab Instructions

All questions should be answered precisely to get maximum credit. Lab report must ensure following items:

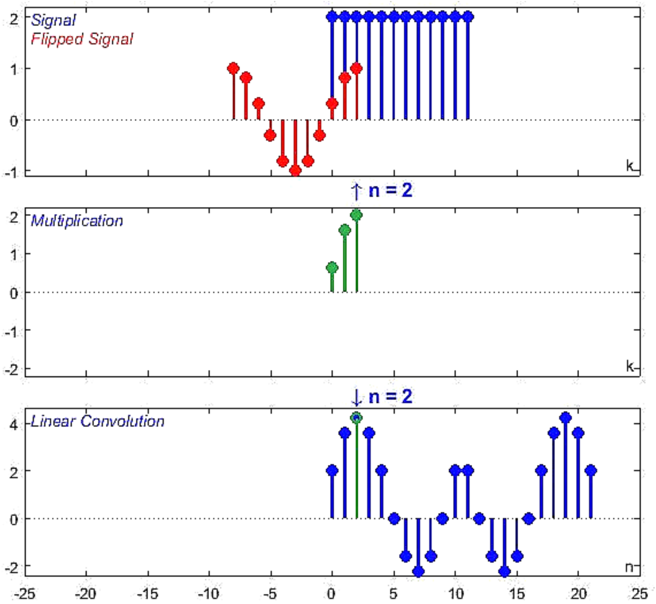
- Lab objectives
- MATLAB codes
- Results (Graphs/Tables) duly commented and discussed
- Conclusion

3 Lab Tasks

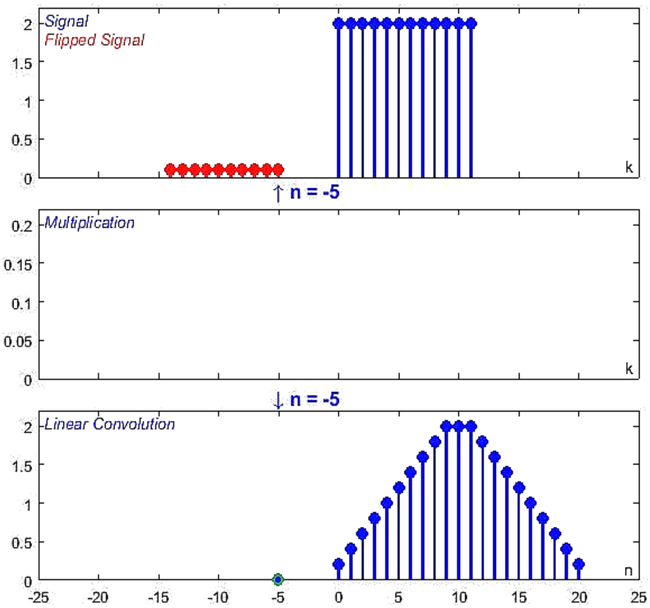
3.1 Convolution GUIs

3.1.1 Discrete-Time Convolution Demo

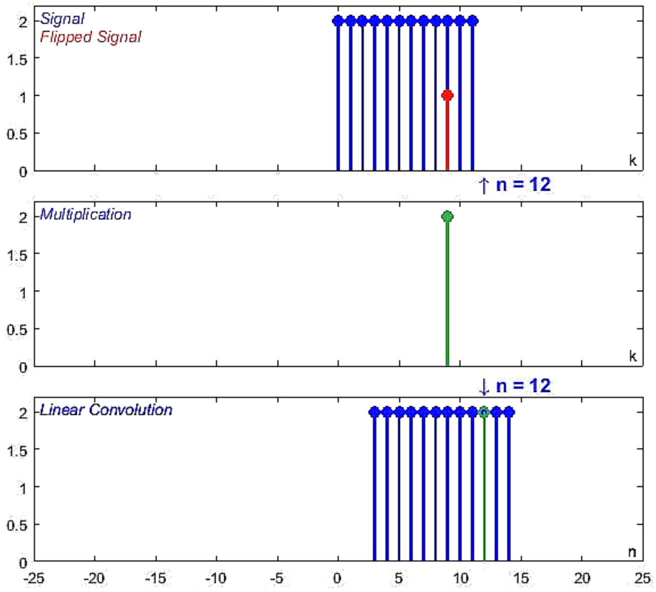
1. Set the input to a finite-length pulse: x[n] = 2 {u[n] - u[n - 12]}.
2. Set the filter's impulse response as a cosine function with equation h[n]= cos (0.2*pi*n * w[n]). Where w[n]= u[n]-u[n-11].



3. Set the filter's impulse response to a length-10 average, i.e., h[n] = 0.1{u[n] - u[n - 10]}. Use the GUI to produce the output signal.



4. Set the filter's impulse response to a shifted impulse, i.e., h[n] = delta[n - 3]. Use the GUI to produce the output signal.

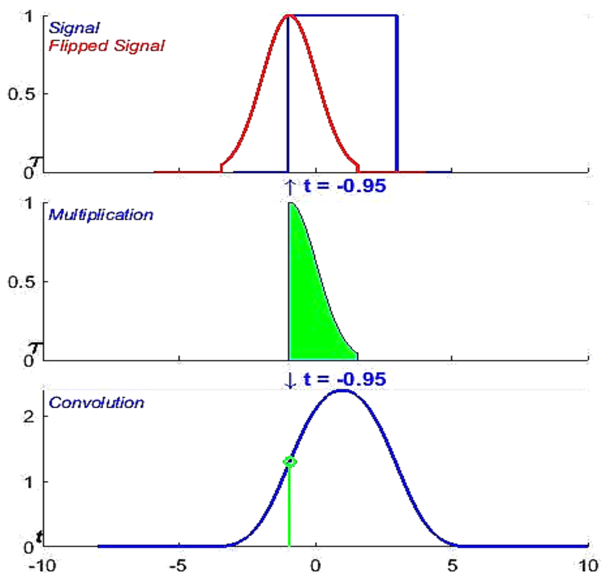


5. Compare the outputs from parts (c), (e) and (f). Notice the different shapes (triangle, rectangle or trapezoid), the maximum values, and the different lengths of the outputs.

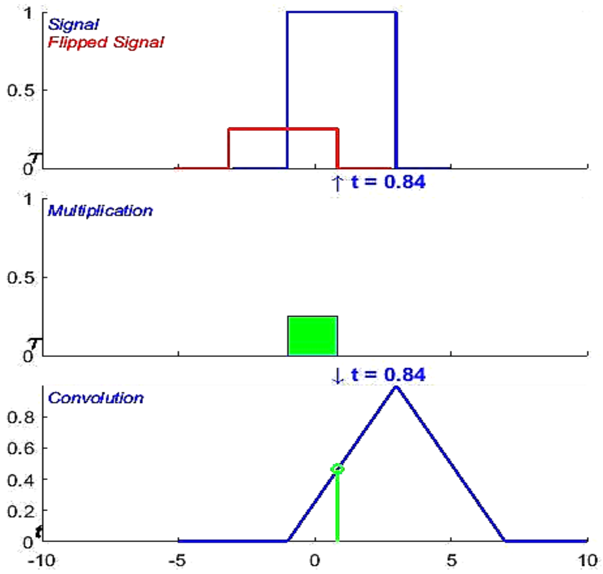
Part c: Unit step signal convolved with a 'cosine' response results in a sinusoidal shaped output Part e: Unit step signal convolved with a 'averager' response results in a triangular shaped output Part f: Unit step signal convolved with a shifted impulse results in a rectangular shaped output

3.1.2 Continuous-Time Convolution Demo

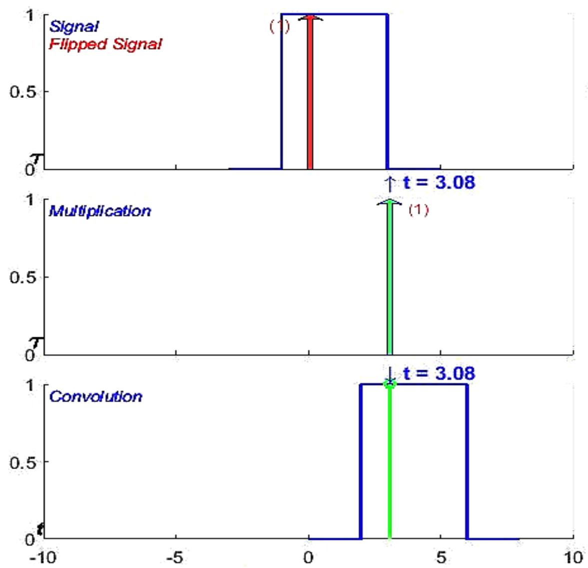
1. Set the input to a pulse x(t) = u(t+1) - u(t - 3).
2. Set the filter's impulse response as h(t)=$e^{-0.5t\wedge 2}$ [u(t)- u(t-5)]



3. Set the filter's impulse response to a 4-second pulse with amplitude 0.25, i.e., h(t) = 0.25{u(t) u(t-4)}. Use the GUI to produce the output signal.



4. Set the filter's impulse response to a shifted impulse, i.e., h(t) = delta(t - 3). Use the GUI to produce the output signal.



5. Compare the outputs from parts (c), (d) and (e). Notice the different shapes (triangle, rectangle or trapezoid), the maximum values, and the different lengths of the outputs.

Part c: Unit step signal convolved with an 'exponential signal' response results in a gaussian pulse Part e: Unit step signal convolved with a 'pulse' response results in a triangular shaped output Part f: Unit step signal convolved with a shifted impulse results in a rectangular shaped output

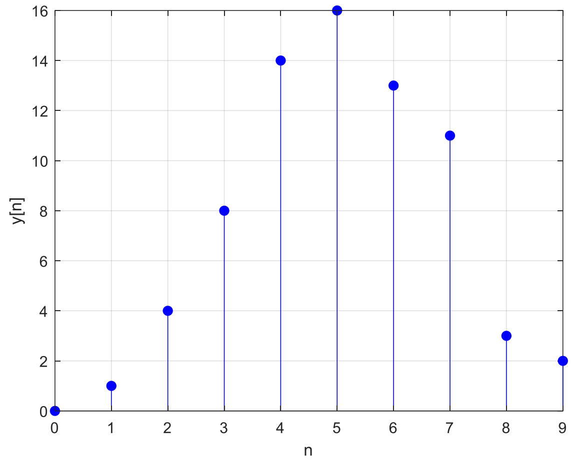
3.2 Introduction to Systems

3.2.1 Pre-Lab

The output y[n] is given by the convolution of the input x[n] with the system impulse response h[n]. This operation can be written as the convolution sum: $y[n] = \sum_{m=-\infty}^{\infty} x[m]h[n-m]$

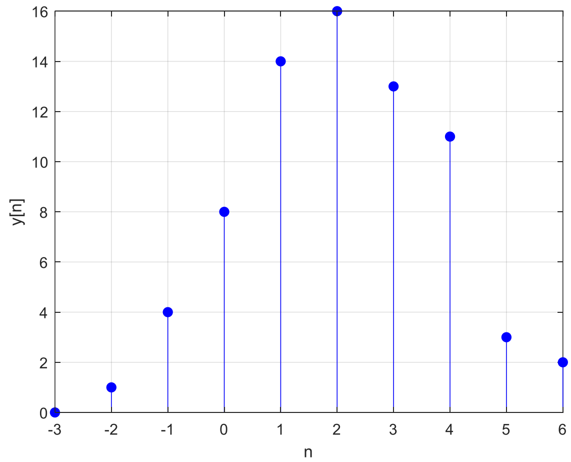
1. Calculate the result of convolving x[n]= {$\underline{1}$ 2 1 2} with h[n]= {$\underline{0}$ 1 2 3 4 1 1}

```
x = [1 2 1 2];
h = [0 1 2 3 4 1 1];
y = conv(x, h);
n = 0:1:(length(x) + length(h) - 2);
```



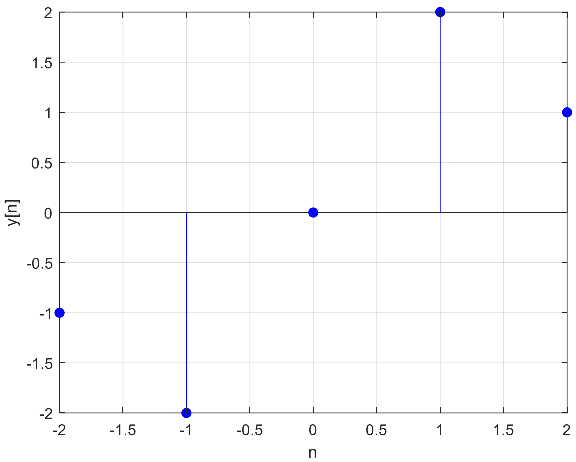
2. Calculate the result of convolving x[n]= {1 $\underline{2}$ 1 2} with h[n]= {0 1 $\underline{2}$ 3 4 1 1}

```
x = [1 2 1 2];
h = [0 1 2 3 4 1 1];
y = conv(x, h);
n = -3:1:(length(x) + length(h) - 5);
```



3. Calculate the result of convolving x[n]= {-1 0 1} with h[n]= {1 2 1}

```
x = [-1 0 1];
h = [1 2 1];
y = conv(x, h);
n = -2:1:(length(x) + length(h) - 4);
```



3.2.2 Lab Exercise: Convolution of Sequences

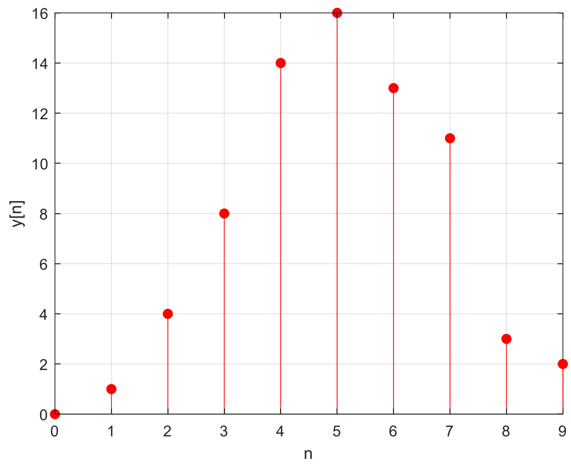
Given the following two sequences, x[n]= {1,2,1,2} h[n]= {0,1,2,3,4,1,1} Where _ indicates the zero position.

1. Write a MATLAB function '***my_conv***' that will convolve the signal x[n] with the system impulse response h[n] and produce the output y[n]. Plot the output y[n] on a graph with correct axis.

```
function y_n = my_conv(x_n, x_limit, h_n, h_limit)
    len_conv = length(x_n) + length(h_n) - 1;
    n = ((x_limit(1)) + (h_limit(1))):len_conv - length(x_n);
    y_n = zeros(1, len_conv);

    for i = 1:length(x_n)
        for j = 1:length(h_n)
            y_n(i + j - 1) = y_n(i + j - 1) + x_n(i) * h_n(j);
        end
    end
    stem(n, y_n, 'filled', 'red');
    grid on
    ylabel('y[n]')
    xlabel('n')

end
```



2. Compare your result with built in function of MATLAB ‘***conv***’.

```
>> y_n = my_conv(x_n, h_n)
y_n =
     0     1     4     8    14    16    13    11     3     2
>> y_n = conv(x_n, h_n)
y_n =
     0     1     4     8    14    16    13    11     3     2
```

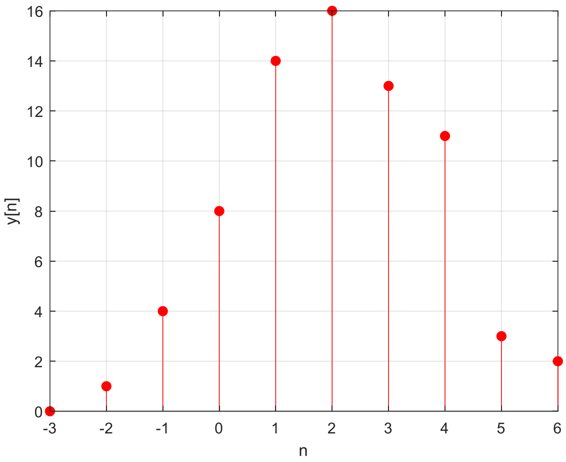
3. If x[n] starts from -1 and h[n] starts from -2 then what will be the result of convolution using ‘***my_conv***’ and ‘***conv***’? Is the result of ‘***my_conv***’ similar to the result you get on paper? If not, how will you get the correct result with respect to position of signal values.

The convolution result using my_conv and conv has the first index 1 since MATLAB doesn’t support negative indices. To fix this, we create a vector “time” with the value at which the convolution starts as its first index and plot output against time using stem or plot depending on nature of signal.

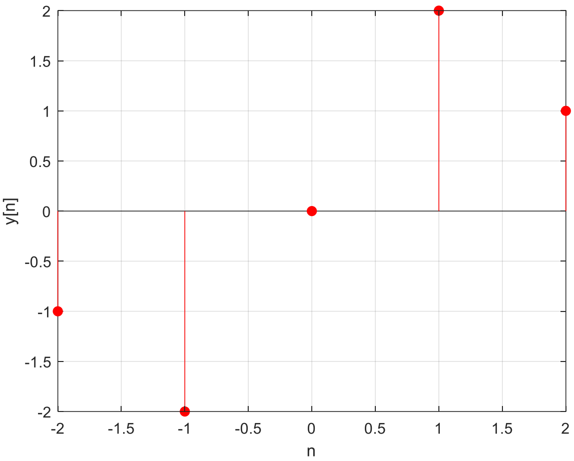
```
function y_n = my_conv(x_n, x_limit, h_n, h_limit)
    len_conv = length(x_n) + length(h_n) - 1;
    n = ((x_limit(1)) + (h_limit(1))):len_conv - length(x_n);
    y_n = zeros(1, len_conv);

    for i = 1:length(x_n)
        for j = 1:length(h_n)
            y_n(i + j - 1) = y_n(i + j - 1) + x_n(i) * h_n(j);
        end
    end
    stem(n, y_n, 'filled', 'red');
    grid on
    ylabel('y[n]')
    xlabel('n')

end
```



4. Convolution of x[n]= {-1 0 1} with h[n]= {1 2 1} results in y[n] = {-1 -2 0 2 1}. Verify this by using the function ***'my_conv'*.**



4 Conclusion

After performing this lab, we conclude that spfirst Convolution GUI serves as an easy and efficient way of finding and visualizing convolution whether the signal is CT or DT. MATLAB built-in functions such as conv and convn also help in computing convolution. We also implemented our own version of 1-D convolution from scratch and verified the results with the built-in conv function.