CS-215: Experiment 4B

Soumen Pradhan – 1912176

17 . 03 . 2021

1. Convolution of rectangular and triangular function

Aim

Compute and plot convolution of these 2 functions using the CONV function.

Theoretical Background

During convolution, the output to an arbitary input signal is found. The input signal can be imagined to be a sum of scaled and shifted consecutive impulses. Thus, the output will be sum of the corresponding impulse responses.

The triangular pulse has the shape of a triangle between $[-1, 1]$.

Methodology

- The HEAVISIDE function is used to simulate unit step function and TRIANGULARPULSE to simulate the input pulse.
- A vector of the convolution using CONV function is generated and plotted.
- Note that, the CONV function is discrete only and HEAVISDE$(0) = \frac{1}{2}$.

Code

```
clear all
clc

syms n x(n) h(n);
x(n) = heaviside(n-1) - heaviside(n-5);
h(n) = triangularPulse((n - 6) / 4);

range = [-100: 100];
xVec = double(x(range));
hVec = double(h(range));

stem(range, xVec, 'fill');
pbaspect([2.5, 1, 1]);
```

```
set(gca, ...
    'Box'        ,'off'            , ...
    'TickDir'    ,'out'            , ...
    'YGrid'      ,'on'             , ...
    'XTick'      ,[-20: 2: 20]     , ...
    'YTick'      ,[-20: .2: 20]    , ...
    'FontSize'   ,10                 );

axis([-.5, 16.5, 0, 1.6]);
print(gcf, '1_inputSignal.eps', '-depsc');

stem(range, hVec, 'fill');
pbaspect([2.5, 1, 1]);
set(gca, ...
    'Box'        ,'off'            , ...
    'TickDir'    ,'out'            , ...
    'YGrid'      ,'on'             , ...
    'XTick'      ,[-20: 2: 20]     , ...
    'YTick'      ,[-20: .2: 20]    , ...
    'FontSize'   ,10               );

axis([-.5, 16.5, 0, 1.6]);
print(gcf, '1_impulseResponse.eps', '-depsc');

yVec = conv(xVec, hVec, 'same');
stem(range, yVec, 'fill');
pbaspect([1.5, 1, 1]);
set(gca, ...
    'Box'        ,'off'            , ...
    'TickDir'    ,'out'            , ...
    'YGrid'      ,'on'             , ...
    'XTick'      ,[-20: 2: 20]     , ...
    'YTick'      ,[-20: .5: 20]    , ...
    'FontSize'   ,10               );

axis([-.5, 16.5, 0, 3.5]);
print(gcf, '1_outputSignal.eps', '-depsc');
```

Input Description

The 2 functions are:

$$x[n] = u[n-1] - u[n-5]$$
$$h[n] = tri\left(\frac{n-6}{4}\right)$$

The range is taken to be $[-100, 100]$ to ensure no information is lost during convolution.

Result

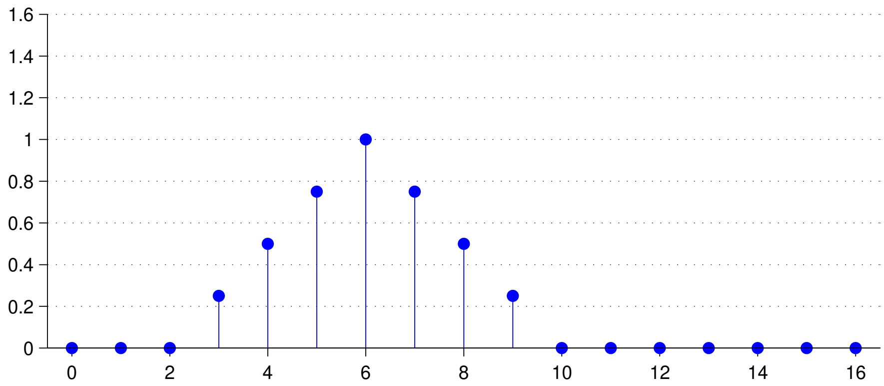

Figure 1.1: *Impulse Response, tri* $\left(\frac{n-6}{4}\right)$

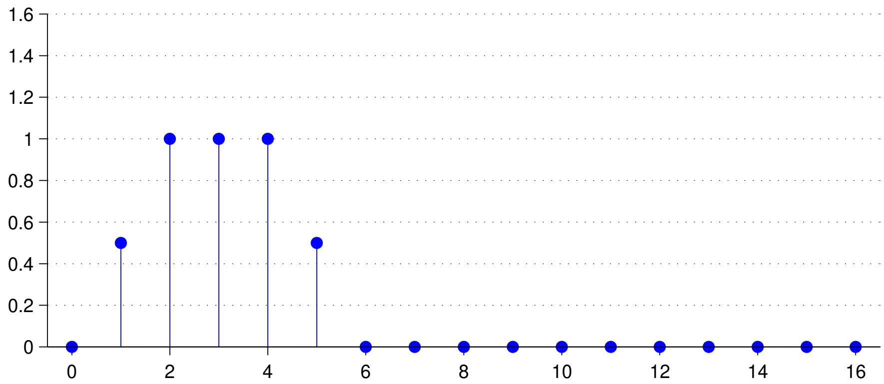

Figure 1.2: *Input Signal,* $u[n-1] - u[n-5]$

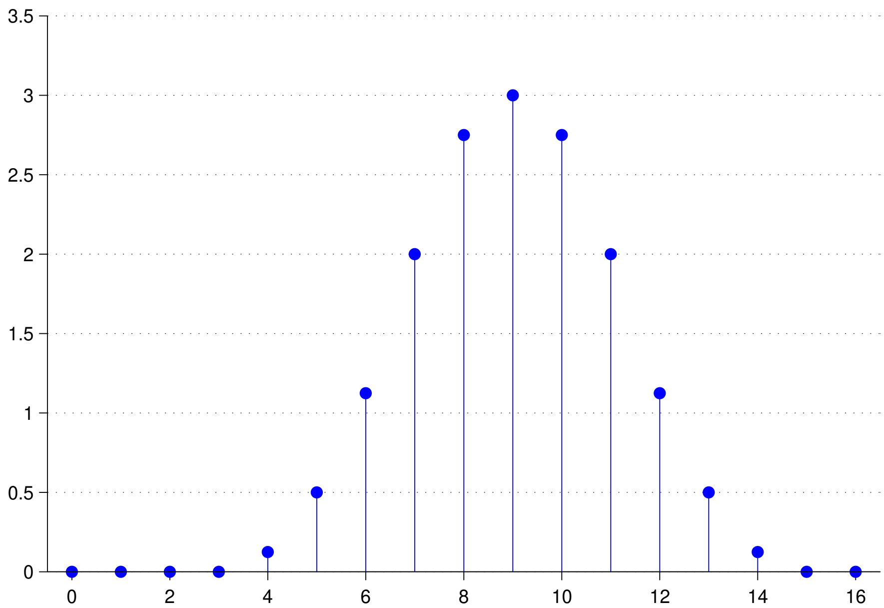

Figure 1.3: *Output Signal*

Conclusion

The resulting output is also a pulse shifted left from both the signals.

2. Convolution of Alternating function

Aim

a. Find impulse response when unit step response is

$$s[n] = 2\left[\left(-\frac{1}{2}\right)^n - 1\right]u[n]$$

b. Compute response of system to $x[n] = n \cdot u[n]$ and filter it.

Theoretical Background

The unit step function is the sum of infinte shifted unit impulse functions. Hence, the impulse response can be computed by subtracting a shifted unit response.

Filtering a signal refers to rectifying its output into a continuos signal that reasonably approximates the original signal. One such method is Running Average, where the values are substituted by the average value in a window of certain width.

Methodology

- The impulse response is computed from the given unit response.
- The convolution sum is thus, found using the impulse respnse.
- A Fitered output is generated using FILTER.

Code

```
clear all
clc

syms n s(n) h(n) x(n);
s(n) = 2 * ((-0.5) ^ n - 1) * heaviside(n);
h(n) = s(n) - s(n-1);
x(n) = n * heaviside(n);

```

```
range = [-100: 100];

sVec = double(s(range));
stem(range, sVec, 'fill');
pbaspect([1.5, 1, 1]);
set(gca, ...
    'Box'       ,'off'              , ...
    'TickDir'   ,'out'              , ...
    'YGrid'     ,'on'               , ...
    'XTick'     ,[-20: 2: 20]       , ...
    'YTick'     ,[-20: .5: 20]      , ...
    'FontSize'  ,10                 );

axis([-2.5, 10.5, -3.5, 0]);
print(gcf, '2_unitResponse.eps', '-depsc');

hVec = double(h(range));
stem(range, hVec, 'fill');
pbaspect([1.5, 1, 1]);
set(gca, ...
    'Box'       ,'off'              , ...
    'TickDir'   ,'out'              , ...
    'YGrid'     ,'on'               , ...
    'XTick'     ,[-20: 2: 20]       , ...
    'YTick'     ,[-20: .5: 20]      , ...
    'FontSize'  ,10                 );

axis([-2.5, 10.5, -3.5, 2]);
print(gcf, '2_impulseResponse.eps', '-depsc');

xVec = double(x(range));
yVec = conv(xVec, hVec, 'same');

stem(range, yVec, '');
hold on;

windowSize = 2;
b = (1/windowSize)*ones(1,windowSize);
yVecFiltered = filter(b, 1, yVec);

plot(range, yVecFiltered, 'r');
hold off;

pbaspect([1.5, 1, 1]);
set(gca, ...
    'Box'       ,'off'              , ...
    'TickDir'   ,'out'              , ...
```

```
    'YGrid'      ,'on'              , ...
    'XTick'      ,[-20: 2: 20]    , ...
    'YTick'      ,[-30: 2: 30]  , ...
    'FontSize'   ,10               );

axis([-2.5, 10.5, -20, 0]);
print(gcf, '2_outputSignal.eps', '-depsc');
```

Input Description

The unit step response is

$$s[n] = 2\left[\left(-\frac{1}{2}\right)^n - 1\right]u[n]$$

The input to the system is the ramp function

$$x[n] = n \cdot u[n]$$

The window size for the Filter function is taken to be 2.

Result

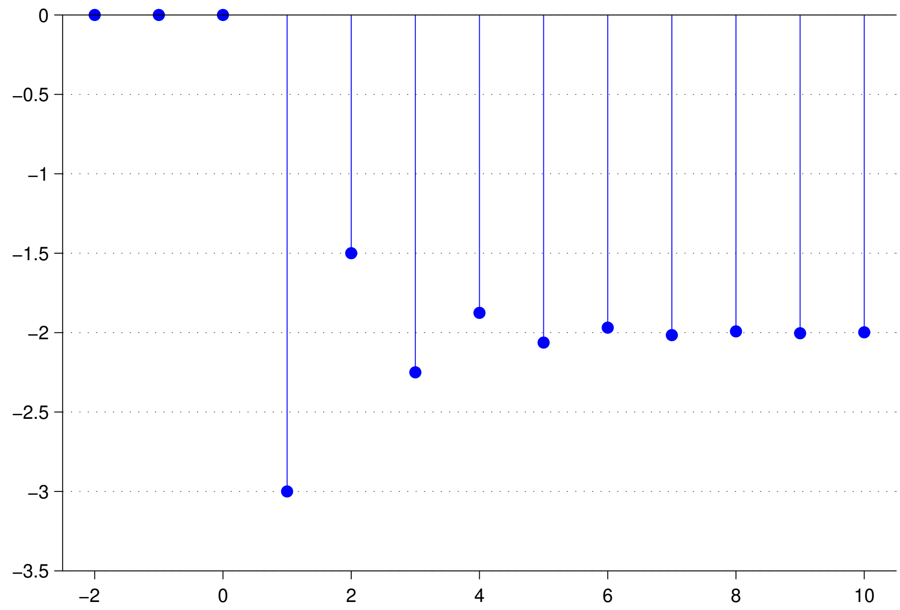

Figure 2.1: *Unit response, $s[n]$*

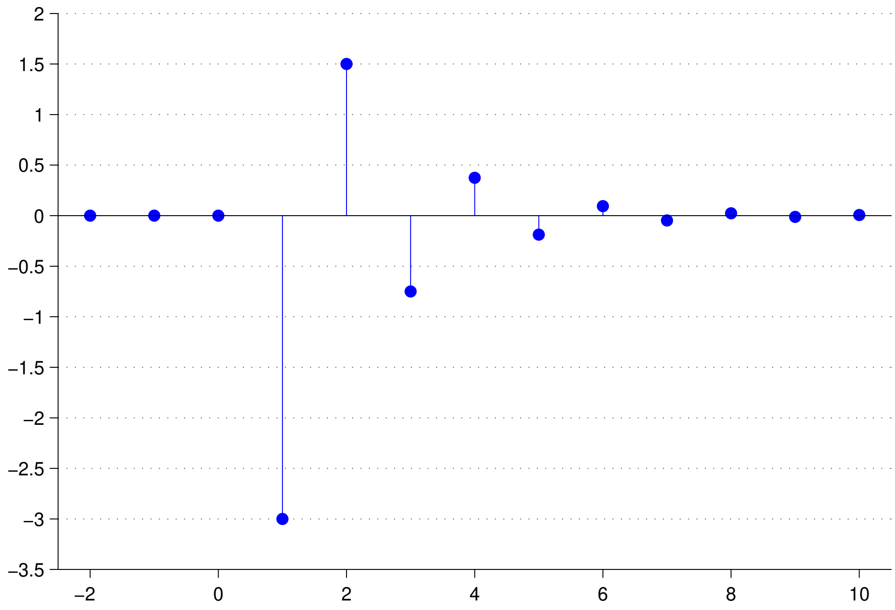

Figure 2.2: *Impulse response, $s[n] - s[n-1]$*

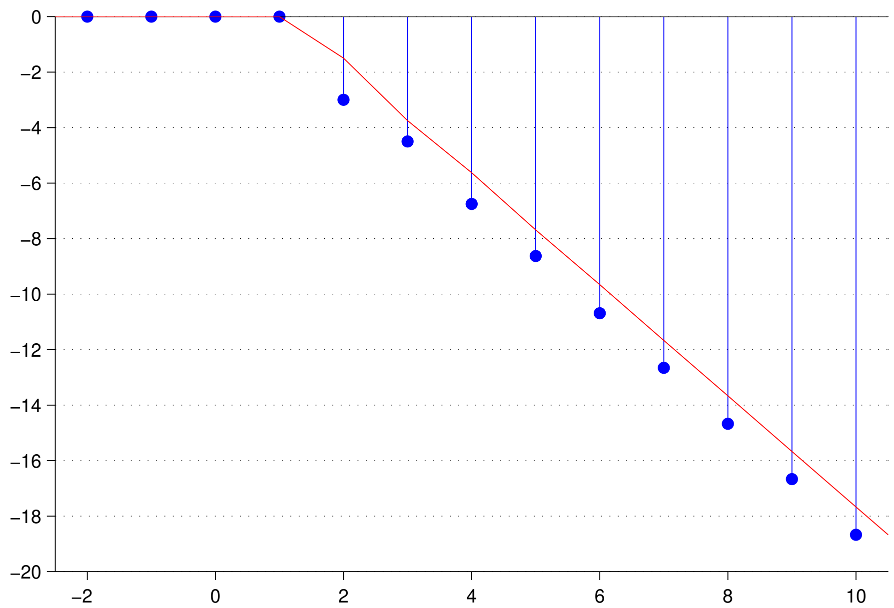

Figure 2.3: *Convolved Signal and Filtered Counterpart(red)*

Conclusion

The filtered output is a scaled version of the input.

The largest value of the impulse function (negative) dominates the convolution sums.