EE387 – BASIC SIGNAL REPRESENTATION AND CONVOLUTION IN MATLAB

MOHAMMED M.H.H.
E/15/131
SEMESTER 06
28/03/2020

PART 1: Basic Signal Representation in MATLAB

1. **Write a MATLAB program and necessary functions to generate the following signal**
   y(t) = 3r(t+3) – 6r(t+1) +3r(t) – u(t-3)

ramp function

```
function y = ramp(t,m,ad)
    % t: length of time
    % m: slope of the ramp function
    % ad: advance (positive), delay (negative) factor
    y = zeros(1,length(t));
    for i = 1:length(y)
        if (t(i) >= -ad)
        y(i) = m*(t(i) + ad);
        end
    end
end
```

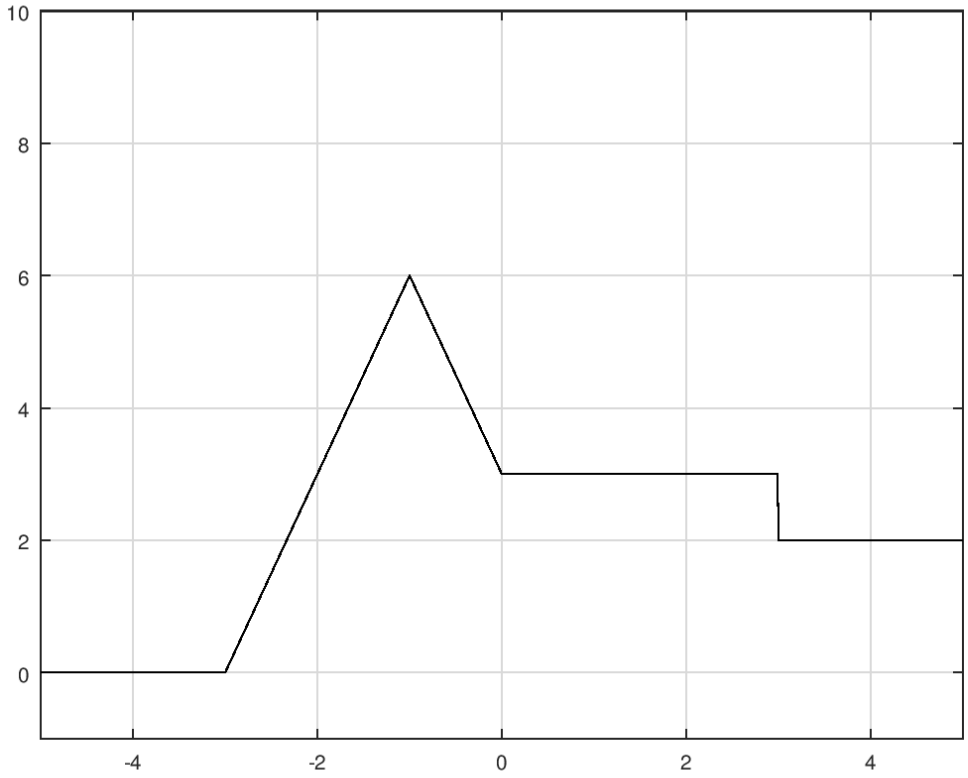
ustep function

```
function y = ustep(t,ad)
    % t: length of time
    % ad: advance (positive), delay (negative) factor
    y = zeros(1,length(t));
    for i = 1:length(y)
        if t(i) >= -ad
            y(i) = 1;
        end
    end
end
```

program to generate the signal y(t) = 3r(t+3) – 6r(t+1) +3r(t) – u(t-3)

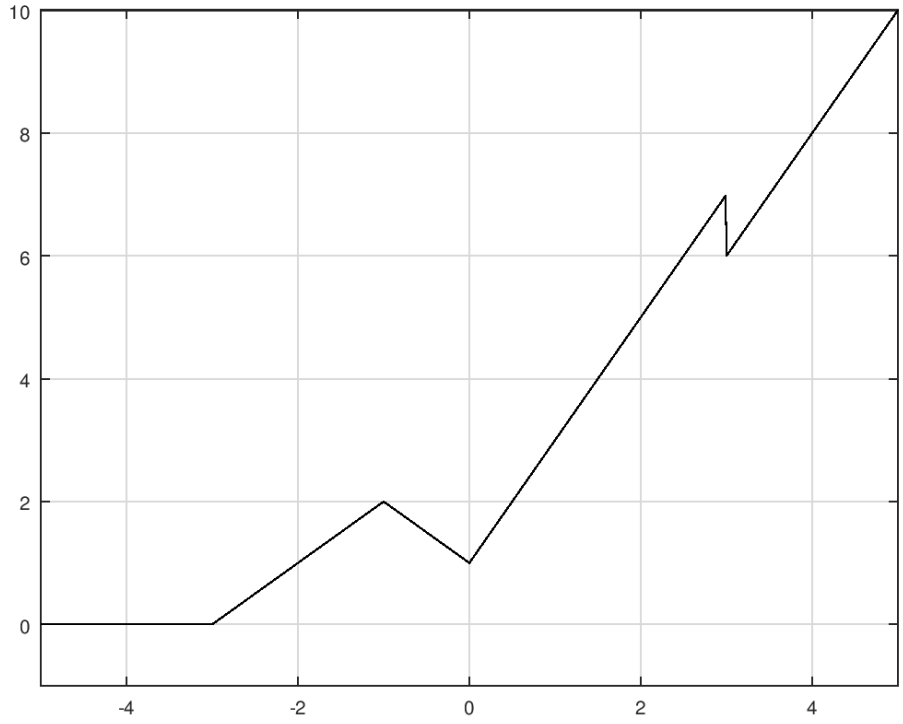
```
clear all;
Ts = 0.01;
t = -5:Ts:5;
y1 = ramp(t,3,3);
y2 = ramp(t,-6,1);
y3 = ramp(t,3,0);
y4 = ustep(t,-3);
y = y1+y2+y3-y4;
plot(t,y,'k'); grid on;
axis([-5 5 -1 7]);
```

Output Result for y(t) = 3r(t+3) – 6r(t+1) +3r(t) – u(t-3)



Program for plot y(t) = r(t+3) – 2r(t+1) +3r(t) – u(t-3)

```
clear all;
Ts = 0.01;
t = -5:Ts:5;
y1 = ramp(t,1,3);
y2 = ramp(t,-2,1);
y3 = ramp(t,3,0);
y4 = ustep(t,-3);
y = y1+y2+y3-y4;
plot(t,y,'k'); grid on;
axis([-5 5 -1 10]);
```

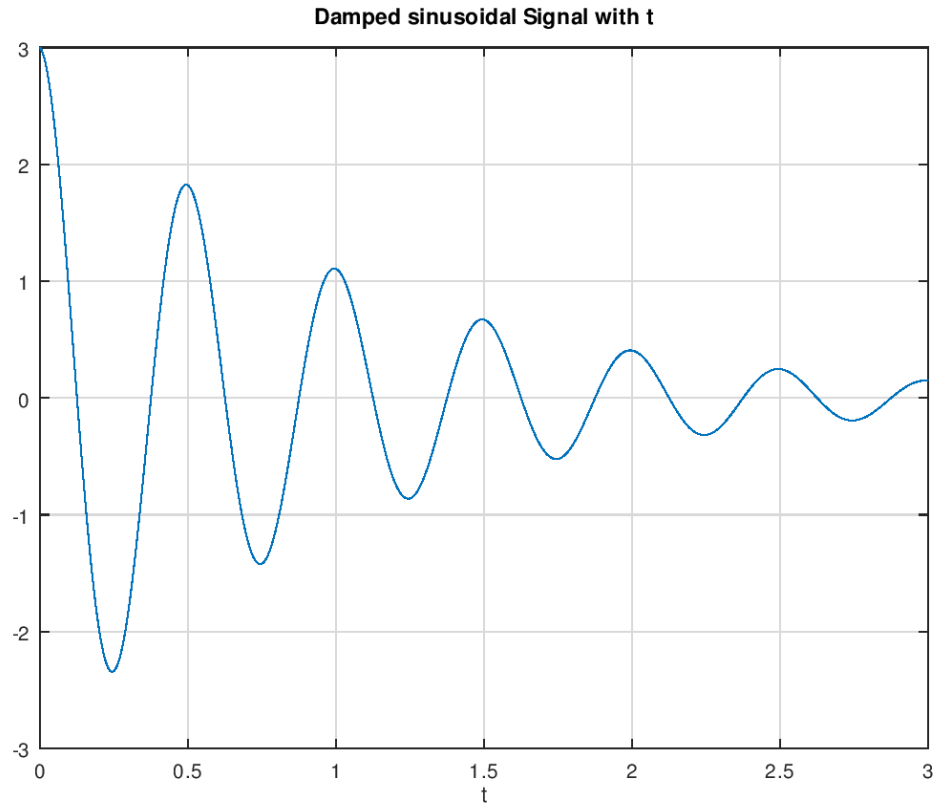


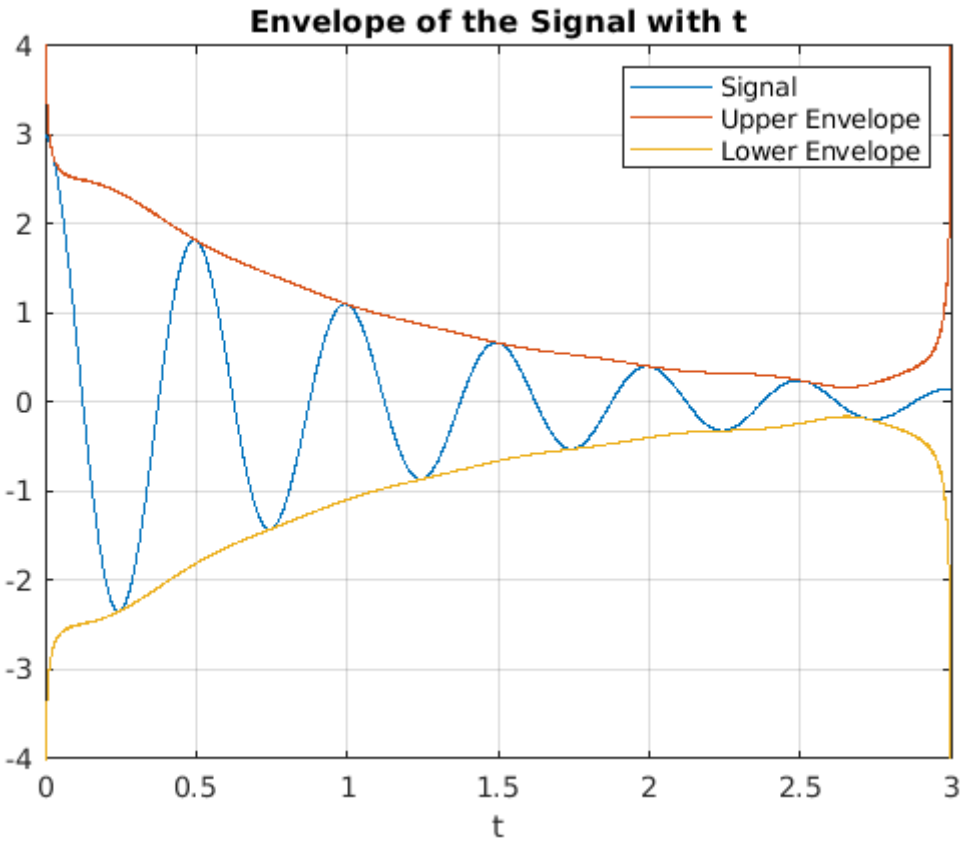
2. **For the damped sinusoidal signal $x(t) = 3e^{-t}\cos(4\pi t)$ write a MATLAB program to generate x(t) and its envelope, then plot.**

Program to plot sinusoidal and its envelop

```
Ts = 0.001; t1 = 0:Ts:3;
y1 = 3.*exp(-t1).*cos(4*pi*t1);
[up,lo] = envelope(y1);

figure(1);
plot(t1,y1); grid on; xlabel('t');
title('Damped sinusoidal Signal with t');
figure(2);

plot(t1,y1,t1,up,t1,lo); grid on; xlabel('t');
legend("Signal","Upper Envelope","Lower Envelope")
title('Envelope of the Signal with t');
axis([0 3 -4 4]);
```

Output Plot


Damped sinusoidal Signal with t
3
2
1
0
-1
-2
-3
0
0.5
1
1.5
2
2.5
3
t


Envelope of the Signal with t
Signal
Upper Envelope
Lower Envelope
4
3
2
1
0
-1
-2
-3
-4
0
0.5
1
1.5
2
2.5
3
t

PART 2: Time-Domain Convolution

Creating a rectangular pulse in MATLAB & Elementary signal operations

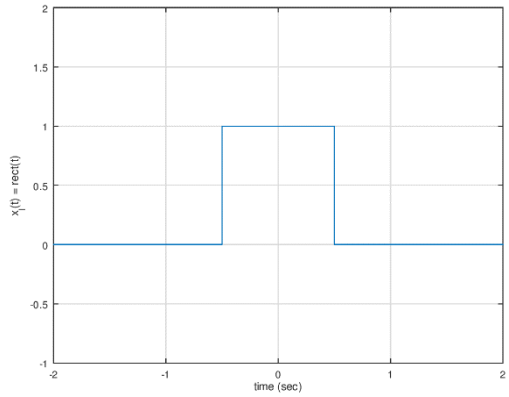
rect function

```
function x = rect(t)
    % This function takes in a vector t of sample instants and outputs the
    % corresponding rectangular pulse
    x = zeros(1,length(t));
    for i =1:length(x)
        if t(i) >= -0.5 && t(i) < 0.5
            x(i) = 1;
        end
    end
end
```

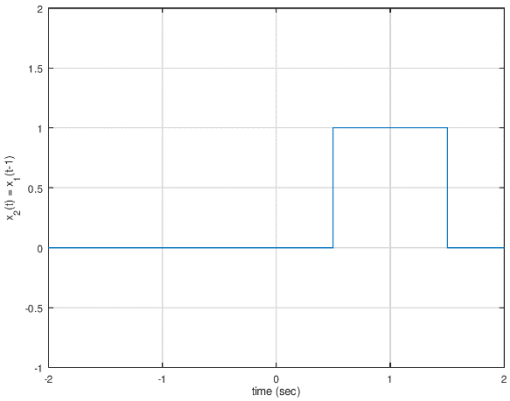
program to plot rectangular pulse and elementary operations on rectangular signal

```
clear all;
f_s = 1000; T_s = 1/f_s;
t = -5:T_s:5;
x1 = rect(t);
x2 = rect(t-1);
x3 = rect(t/2);
x4 = rect(t)+(1/2)*rect(t-1);
x5 = rect(-t)+(1/2)*rect(-t-1);
x6 = rect(1-t)+(1/2)*rect(-t);
figure(1);
plot(t,x1); axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('x_1(t) = rect(t)');
figure(2);
plot(t,x2); axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('x_2(t) = x_1(t-1)');
figure(3);
plot(t,x3); axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('x_3(t) = x_1(t/2)');
figure(4);
plot(t,x4); axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('x_4(t) = x_1(t)+(1/2)x_1(t-1)');
figure(5);
plot(t,x5); axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('x_5(t) = x_4(-t)');
figure(6);
plot(t,x6); axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('x_6(t) = x_4(1-t)');
```

Output Plots

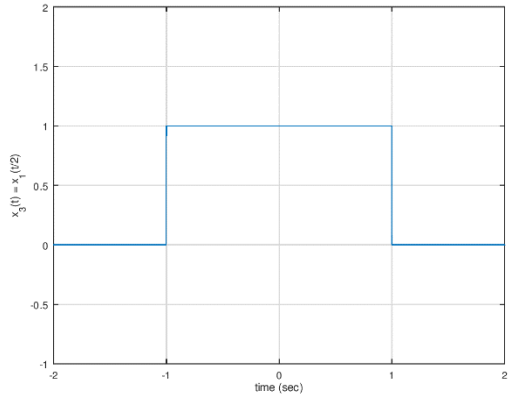


Rectangular Pulse signal



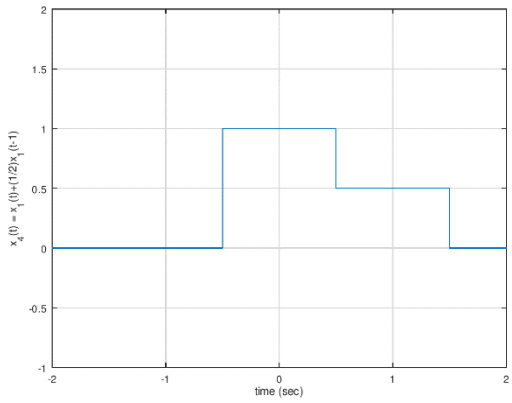
Time-Delayed Signal,

x2(t) = rect(t-1)

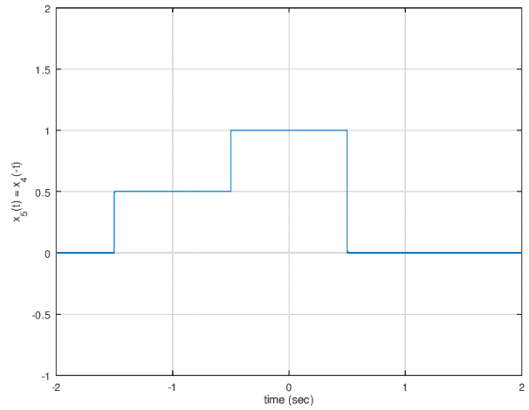


Time-Scaled Signal,

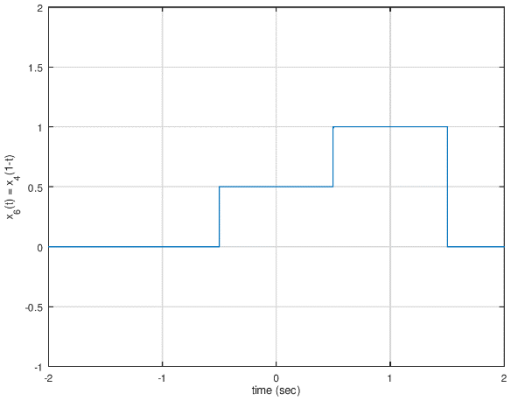
x3(t) = rect(t/2)



Signal x4(t) = rect(t)+(1/2)rect(t-1)



Time Reversal Signal x5(t) = x4(-t)

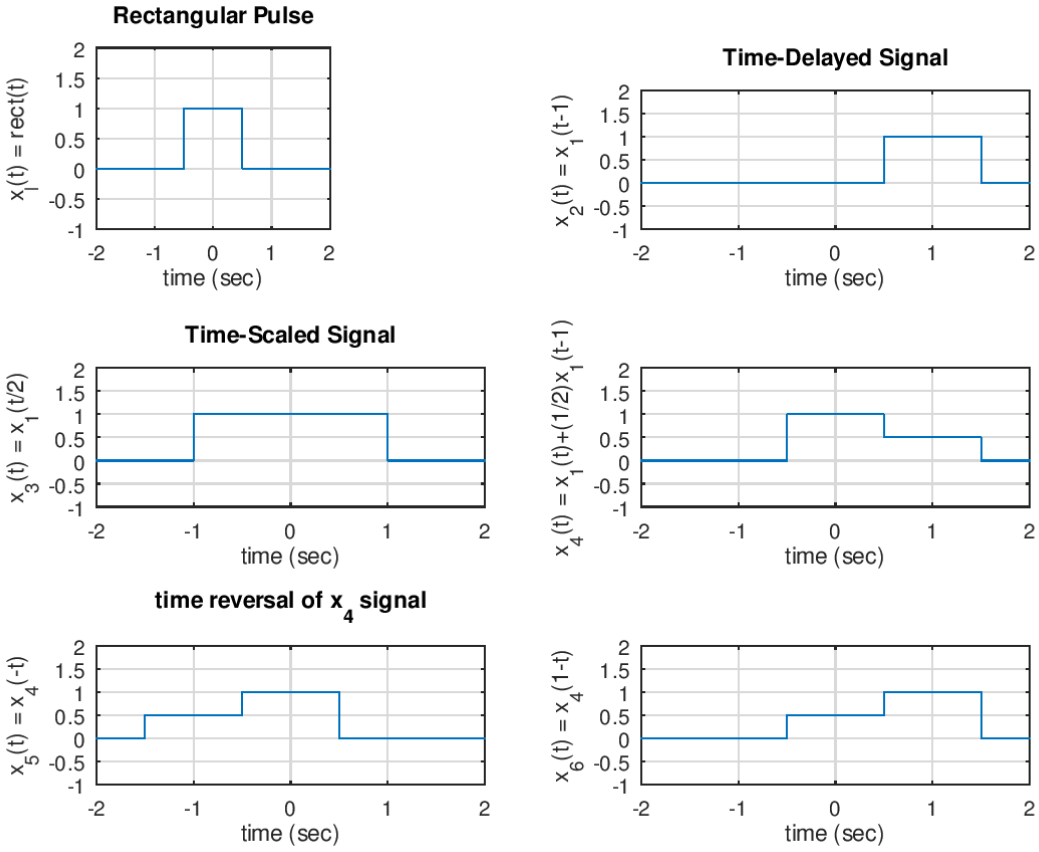


x6(t) = x4(1-t) = rect(1-t)+(1/2)rect(-t)

Program to plot all the above signals plots in single plot

```
clear all;
f_s = 1000; T_s = 1/f_s;
t = -5:T_s:5;
x1 = rect(t);
x2 = rect(t-1);
x3 = rect(t/2);
x4 = rect(t)+(1/2)*rect(t-1);
x5 = rect(-t)+(1/2)*rect(-t-1);
x6 = rect(1-t)+(1/2)*rect(-t);

subplot(3,3,1)
plot(t,x1); grid on; axis([-2 2 -1 2]);
title('Rectangular Pulse');
xlabel('time (sec)'); ylabel('x_1(t) = rect(t)');

subplot(3,2,2)
plot(t,x2); grid on; axis([-2 2 -1 2]);
title('Time-Delayed Signal');
xlabel('time (sec)'); ylabel('x_2(t) = x_1(t-1)');

subplot(3,2,3)
plot(t,x3); grid on; axis([-2 2 -1 2]);
title('Time-Scaled Signal');
xlabel('time (sec)'); ylabel('x_3(t) = x_1(t/2)');

subplot(3,2,4)
plot(t,x4); grid on; axis([-2 2 -1 2]);
xlabel('time (sec)'); ylabel('x_4(t) = x_1(t)+(1/2)x_1(t-1)');

subplot(3,2,5)
plot(t,x5); grid on; axis([-2 2 -1 2]);
title('time reversal of x_4 signal');
xlabel('time (sec)'); ylabel('x_5(t) = x_4(-t)');

subplot(3,2,6)
plot(t,x6); grid on; axis([-2 2 -1 2]);
xlabel('time (sec)'); ylabel('x_6(t) = x_4(1-t)');
```

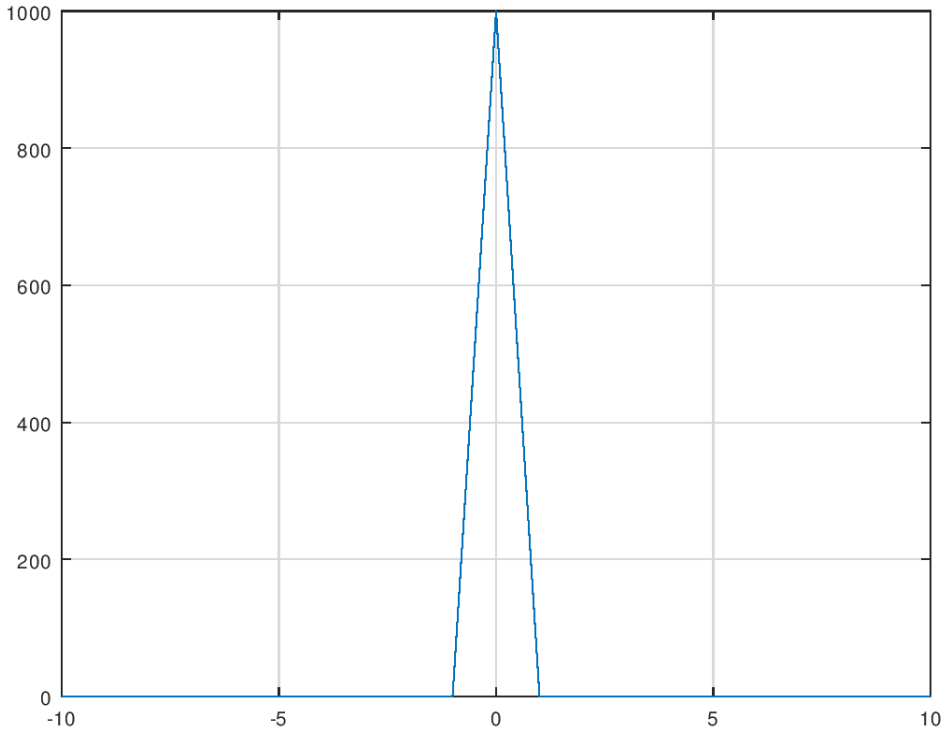
Output Plot



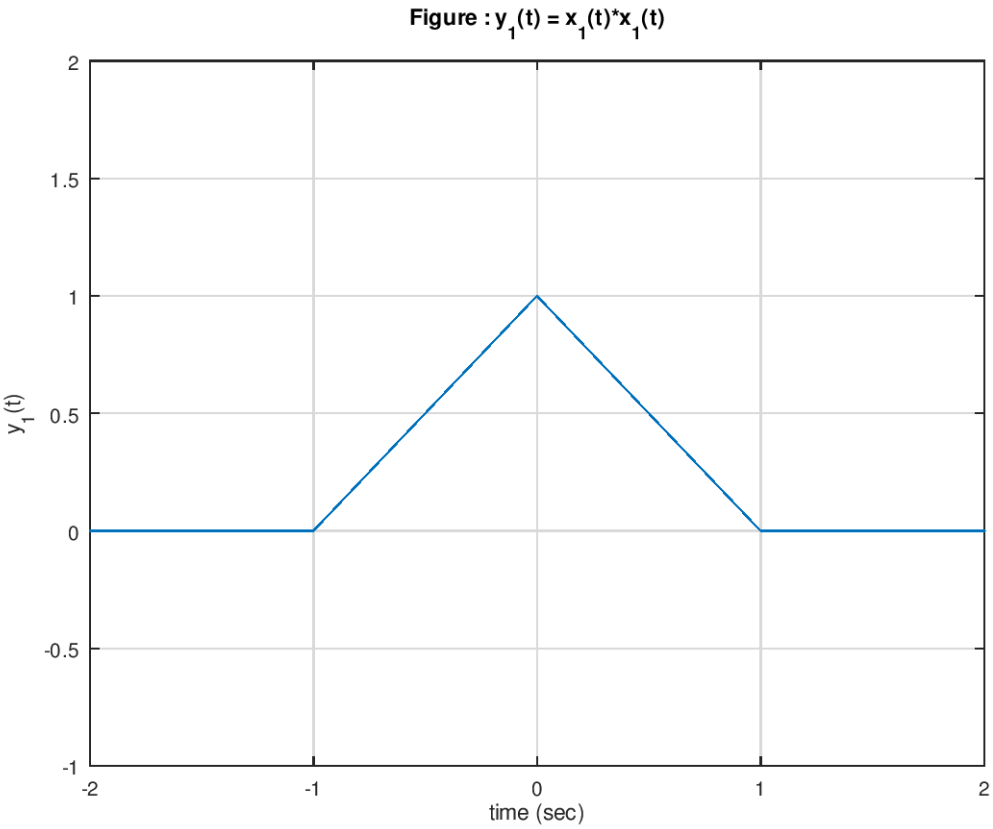
Convolution

```
clear all;
f_s = 1000;
T_s = 1/f_s;
t = -5:T_s:5;
x1 = rect(t);
y = conv(x1,x1);
close all;
%plot(t,y); %Error Message "vector lengths must match"

length(y);
length(t);
t_y = -10:T_s:10;
%plot(t_y,y); grid on; %Error, Wrong Plot, Plot with the peak of 1000

y1 = T_s*conv(x1,x1); %Correction
plot(t_y,y1);
axis([-2 2 -1 2]); grid on;
xlabel('time (sec)'); ylabel('y_1(t)');
title('Figure : y_1(t) = x_1(t)*x_1(t)');
```

Output Plots



Wrong Plot with the Peak of f_s = 1000



Plot of the convolution of two rectangular pulses

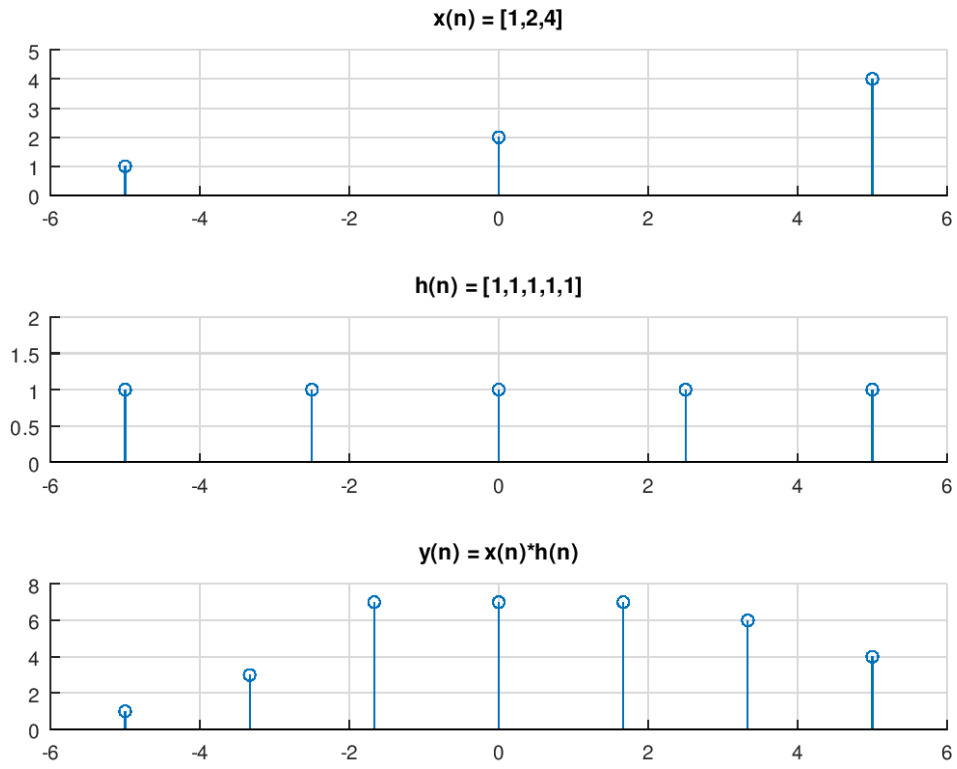
Exercise

1. **Perform convolution on discrete time signals x(n) and h(n), i.e., y(n) = x(n)*h(n) using MATLAB. For each set of signals, plot x(n), h(n) and y(n) as subplots in the same figure.**

   x(n) = { 1,2,4 }, h(n) = {1,1,1,1,1}

```
clear all;
x = [1,2,4]; lenx = length(x);
n1 = linspace(-5,5,lenx);
subplot(3,1,1);
stem(n1,x); grid on; axis([-6 6 0 5]); title('x(n) = [1,2,4]');

h = ones(1,5); lenh = length(h);
n2 = linspace(-5,5,lenh);
subplot(3,1,2);
stem(n2,h); grid on; axis([-6 6 0 2]); title('h(n) = [1,1,1,1,1]');

y = conv(x,h); leny = length(y);
n3 = linspace(-5,5,leny);
subplot(3,1,3);
stem(n3,y); grid on; axis([-6 6 0 8]); title('y(n) = x(n)*h(n)');
```

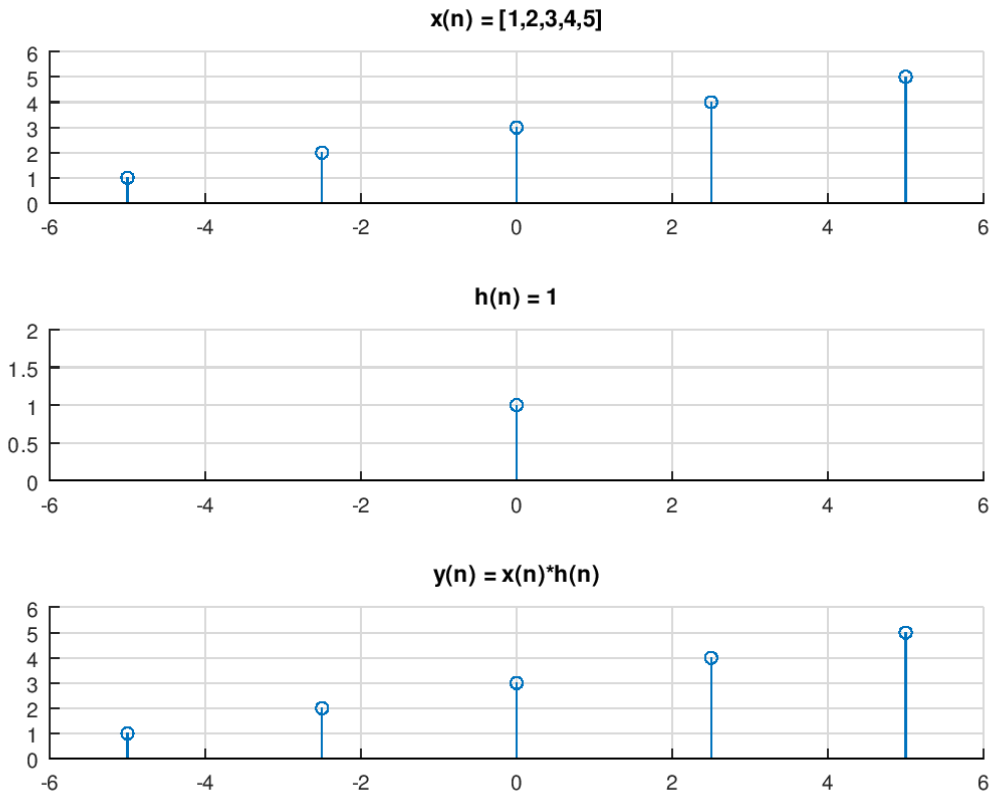
   Output Plots



x(n) = { 1,2,3,4,5 }, h(n) = {1}

```
clear all;
x = 1:5; lenx = length(x);
n1 = linspace(-5,5,lenx);
subplot(3,1,1);
stem(n1,x); grid on; axis([-6 6 0 6]); title('x(n) = [1,2,3,4,5]');

h = 1; lenh = length(h);
n2 = 0;
subplot(3,1,2);
stem(n2,h); grid on; axis([-6 6 0 2]); title('h(n) = 1');

y = conv(x,h); leny = length(y);
n3 = linspace(-5,5,leny);
subplot(3,1,3);
stem(n3,y); grid on; axis([-6 6 0 6]); title('y(n) = x(n)*h(n)');
```

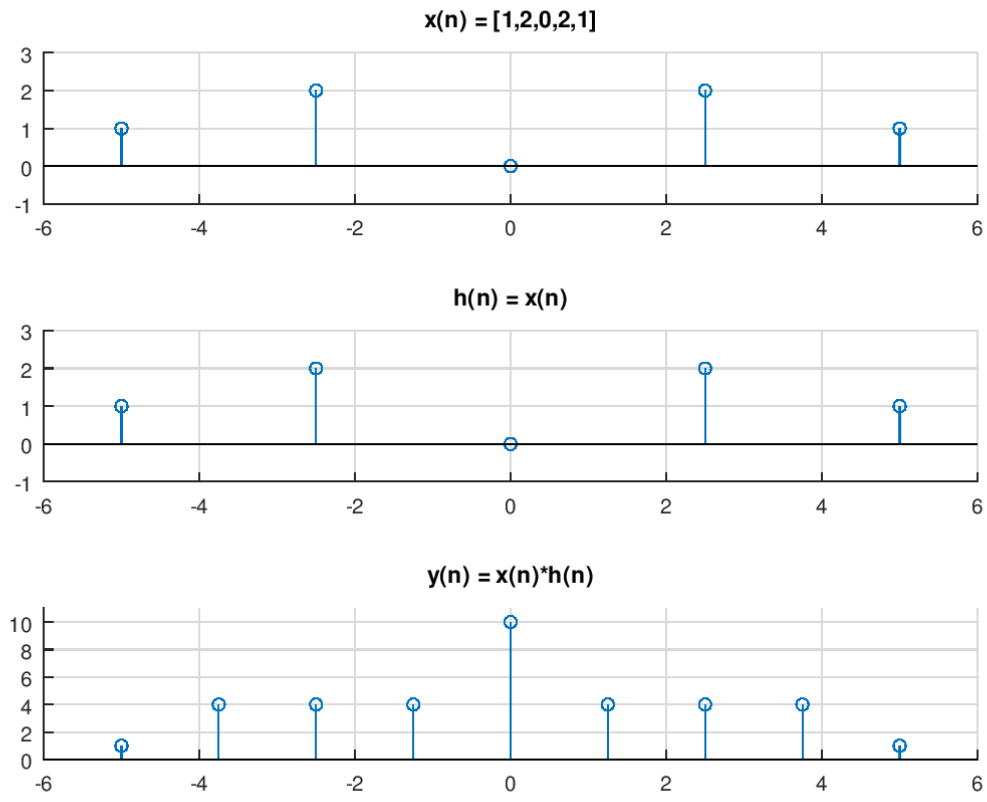
Output Plots



x(n) = h(n) ={ 1,2,0,2,1}

```
clear all;
x = [1,2,0,2,1]; lenx = length(x);
n1 = linspace(-5,5,lenx);
subplot(3,1,1);
stem(n1,x); grid on; axis([-6 6 -1 3]); title('x(n) = [1,2,0,2,1]');

h = x;
subplot(3,1,2);
stem(n1,h); grid on; axis([-6 6 -1 3]); title('h(n) = x(n)');

y = conv(x,h);leny = length(y);
n2 = linspace(-5,5,leny);
subplot(3,1,3);
stem(n2,y); grid on; axis([-6 6 0 11]); title('y(n) = x(n)*h(n)');
```

Output Plots



2. **Assume a system with the following impulse response:**

   h(n) = $(0.5)^n$ for 0 <= n < 4

   = 0 elsewhere

   Determine the input x (n) that will generate the output sequence y(n) = {1, 2, 2.5, 3, 3, 3, 2, 1,0...}. Plot h(n), y (n) and x (n) in one figure.

```
clear all;
n = linspace(0,6,7);
len = length(n);
h = zeros(1,len);
for i = 1:len
    if n(i)>=0 && n(i)<4
        h(i) = (1/2)^n(i);
    end
end
subplot(3,1,1);
stem(n,h);grid on;
axis([-1 7 -1 1.5]);
title('h(n) vs n');

y = [1,2,2.5,3,3,3,2,1,0];
leny = length(y);
n1 = linspace(0,6,leny);
subplot(3,1,2);
stem(n1,y); grid on;
axis([-1 7 -1 4]);
title('y(n) vs n');

x = deconv(y,h);
lenx = length(x);
n2 = linspace(0,6,lenx);
subplot(3,1,3);
stem(n2,x); grid on;
axis([-1 7 0 2]);
title('Deconvolution of y(n) & h(n)');
```

Output Plots