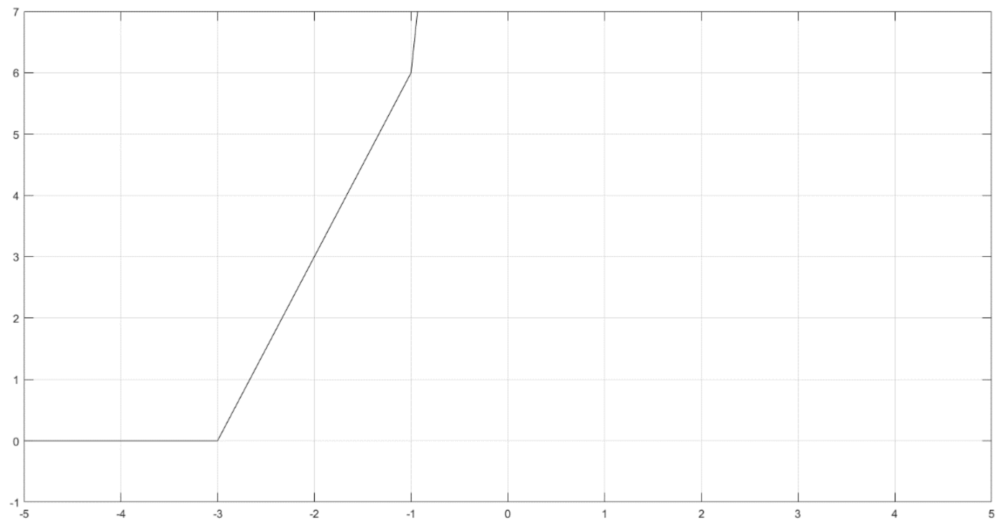
EE387: Signal Processing
Lab 1: Basic Signal Representation and Convolution in MATLAB

Ranage R.D.P.R. - E/19/310

PART 1: Basic Signal Representation in MATLAB

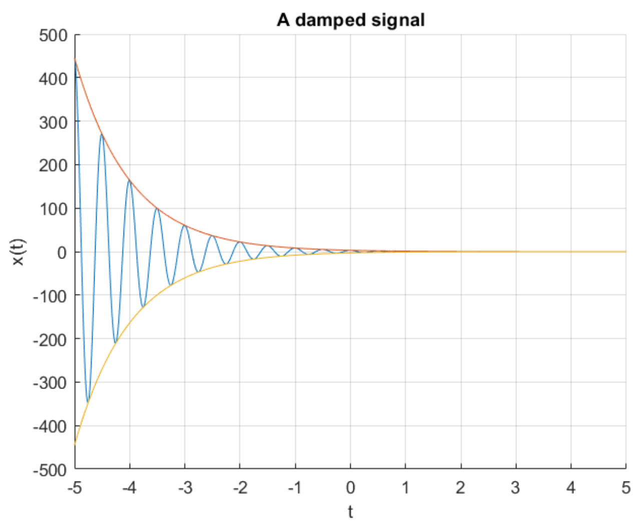
1. Write a Matlab program and necessary functions to generate the following signal:
y(t) = r(t+3) – 2r(t+1) +3r(t) – u(t-3)
Then plot it and verify analytically that the obtained figure is correct.

```
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 = y1-2*y2+3*y3-y4;

plot(t, y, 'k');
axis([-5 5 -1 7]);
grid

function y = ramp(t, m, ad)
    % t: length of time
    % m: slope of the ramp function
    % ad: advance (positive), delay (negative) factor
    t = t(:)';
    % Initialize y
    y = zeros(size(t));
    % Find the indices where t >= -ad
    idx = find(t >= -ad);
    % Calculate ramp function
    y(idx) = m * (t(idx) + ad);
end

function y = ustep(t, ad)
    % t: length of time
    % ad: advance (positive), delay (negative) factor
    t = t(:)';
    y = zeros(size(t));
    % Find the indices where t >= -ad
    idx = t >= -ad;
    % Set the values to 1 where t >= -ad
    y(idx) = 1;
end
```



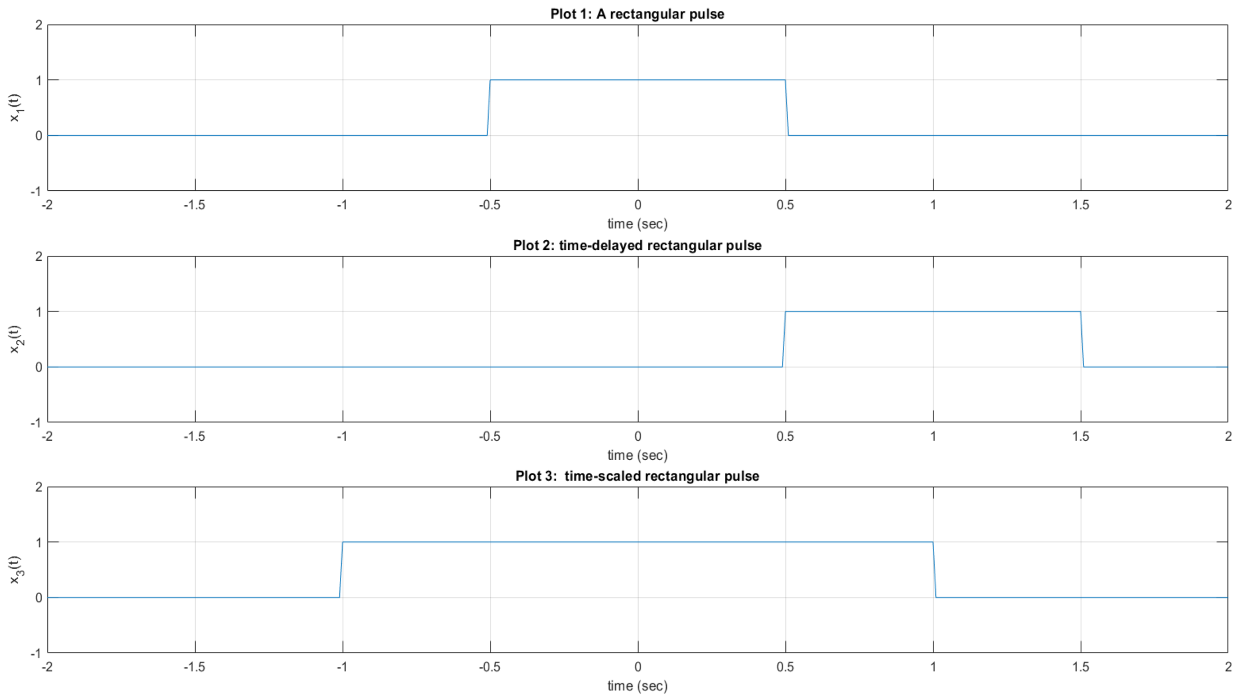
2. For the damped sinusoidal signal x(t) = 3e-tcos(4πt) write a MATLAB program to generate x(t) and its envelope, then plot.

```
clear all;
close all;
clc;

Ts=0.01; %Sampling time
t=-5:Ts:5; %Time vector

x= 3 * exp(-1*t) .* cos(4*pi*t);
envelope= 3 * exp(-1*t);

figure;
hold on;
plot(t,x);
plot(t, envelope);
plot(t, -1*envelope);
title('A damped signal');
xlabel('t');
ylabel('x(t)');
grid
```



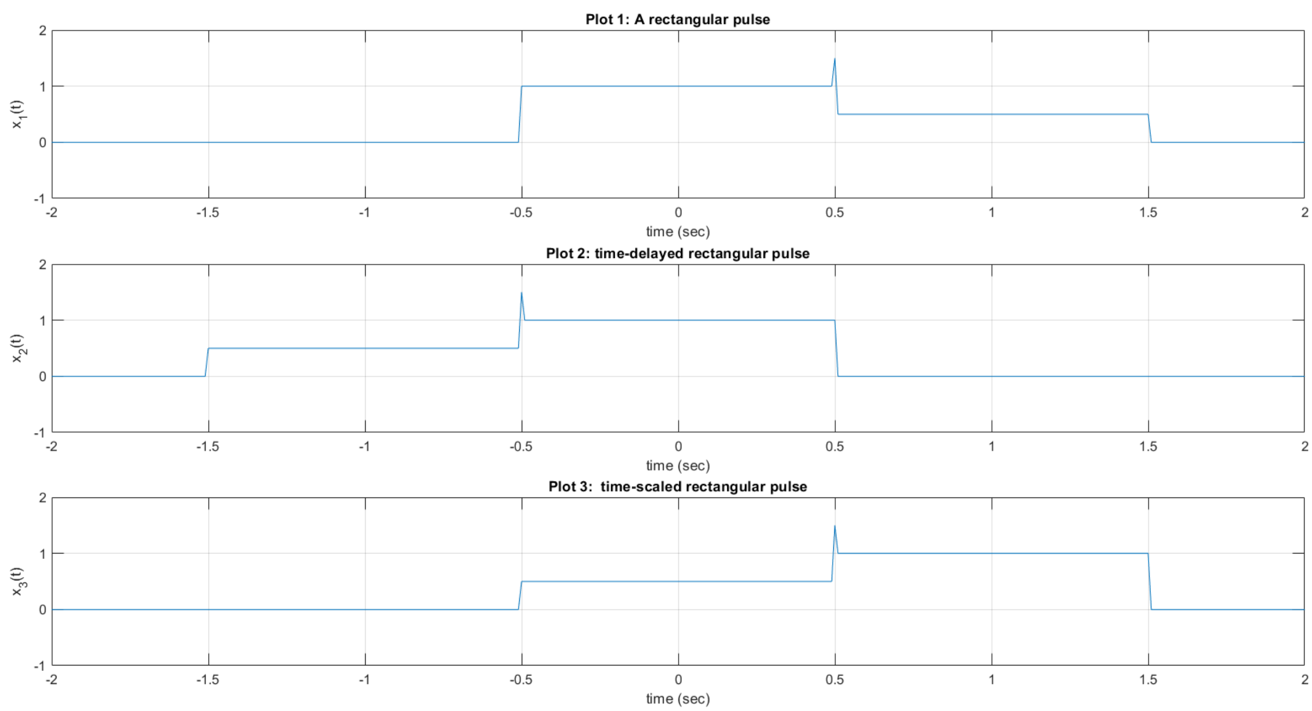
PART 2: Time-Domain Convolution
Creating a rectangular pulse in MATLAB

```
f_s = 100;
T_s = 1/f_s;
t = [-5:T_s:5];

x1 = rect(t);
subplot(3,1,1);
plot(t,x1);
axis( [-2 2 -1 2]);
title ('Plot 1: A rectangular pulse');
xlabel( 'time (sec)' ) ;
ylabel( 'x_1(t)' ) ;
grid on;

x2 = rect(t-1);
subplot(3, 1, 2);
plot(t,x2);
axis( [-2 2 -1 2]);
title ('Plot 2: time-delayed rectangular pulse');
xlabel( 'time (sec)' ) ;
ylabel( 'x_2(t)' ) ;
grid on;

x3 = rect(t/2);
subplot(3, 1, 3);
plot(t,x3);
axis( [-2 2 -1 2]);
title ('Plot 3:  time-scaled rectangular pulse');
xlabel( 'time (sec)' ) ;
ylabel( 'x_3(t)' ) ;
grid on;

% Define rectangular pulse function
function x = rect(t)
    %
    % RECT rectangular pulse
    %
    % Usage: x = rect(t)
    %
    % This function takes in a vector t of sample instants and
outputs the
    % corresponding rectangular pulse contained in the function x
    x = (abs(t) <= 0.5);
end
```

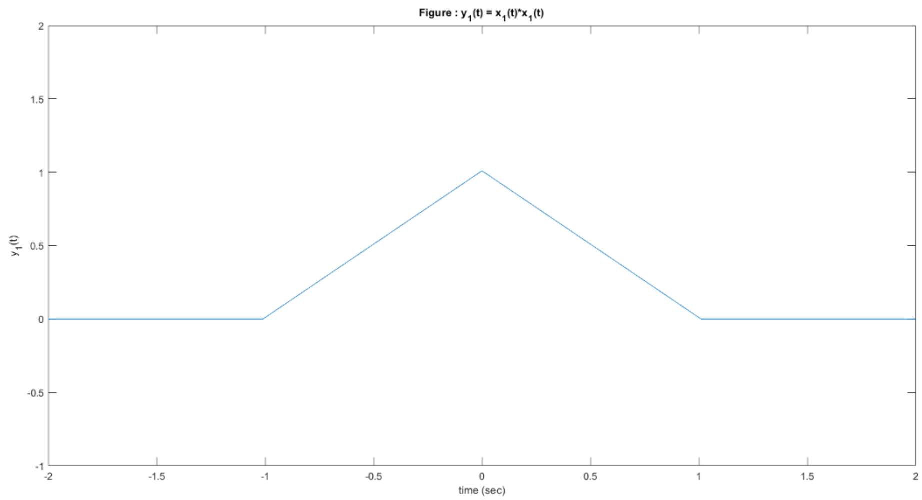



```
f_s = 100;
T_s = 1/f_s;
t = [-5:T_s:5];

x4 = rect(t) + (1/2) * rect(t-1);
subplot(3,1,1);
plot(t,x4);
axis( [-2 2 -1 2]);
title ('Plot 1: A rectangular pulse');
xlabel( 'time (sec)' ) ;
ylabel( 'x_1(t)' ) ;
grid on;

x5 = rect(-t) + (1/2) * rect(-t-1);
subplot(3, 1, 2);
plot(t,x5);
axis( [-2 2 -1 2]);
title ('Plot 2: time-delayed rectangular pulse');
xlabel( 'time (sec)' ) ;
ylabel( 'x_2(t)' ) ;
grid on;

x6 = rect(1-t) + (1/2) * rect(-t);
subplot(3, 1, 3);
plot(t,x6);
```

```
axis( [-2 2 -1 2]);
title ('Plot 3:  time-scaled rectangular pulse');
xlabel( 'time (sec)' ) ;
ylabel( 'x_3(t)' ) ;
grid on;

% Define rectangular pulse function
function x = rect(t)
    %
    % RECT rectangular pulse
    %
    % Usage: x = rect(t)
    %
    % This function takes in a vector t of sample instants and
outputs the
    % corresponding rectangular pulse contained in the function x
    x = (abs(t) <= 0.5);
end
```



```
y = conv(x1,x1);
close all;
%plot ( t, y);

length(y)
length(t)

t_y = -10:T_s:10;

plot( t_y, y);

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



Exercise

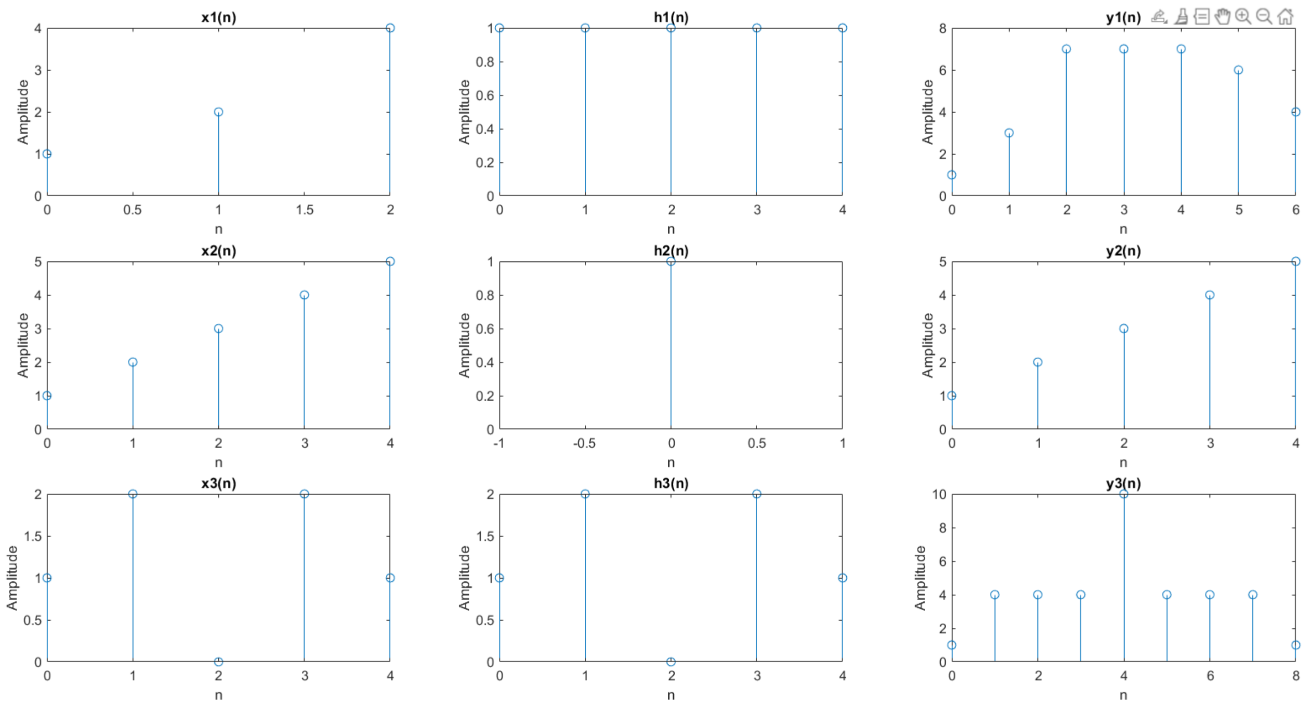
1. Perform convolution on discrete time signals x(n) and h(n), i.e., y(n) = x(n)*h(n) using MATLAB. For each set of signals, plot x(n), h(n) and y(n) as subplots in the same figure.

- **x(n) = { 1,2,4 }, h(n) = {1,1,1,1,1}**
- **x(n) = { 1,2,3,4,5 }, h(n) = {1}**
- **x(n) = h(n) ={ 1,2,0,2,1}**

```
% Define the signals
x1 = [1, 2, 4];
h1 = [1, 1, 1, 1, 1];

x2 = [1, 2, 3, 4, 5];
h2 = [1];

x3 = [1, 2, 0, 2, 1];
h3 = [1, 2, 0, 2, 1];

% Perform convolution
y1 = conv(x1, h1);
y2 = conv(x2, h2);
y3 = conv(x3, h3);

% Plot the signals and convolution results
figure;

% Subplot for x1(n)
subplot(3, 3, 1);
stem(0:length(x1)-1, x1);
```

```
title('x1(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for h1(n)
subplot(3, 3, 2);
stem(0:length(h1)-1, h1);
title('h1(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for y1(n)
subplot(3, 3, 3);
stem(0:length(y1)-1, y1);
title('y1(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for x2(n)
subplot(3, 3, 4);
stem(0:length(x2)-1, x2);
title('x2(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for h2(n)
subplot(3, 3, 5);
stem(0:length(h2)-1, h2);
title('h2(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for y2(n)
subplot(3, 3, 6);
stem(0:length(y2)-1, y2);
title('y2(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for x3(n)
subplot(3, 3, 7);
stem(0:length(x3)-1, x3);
title('x3(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for h3(n)
subplot(3, 3, 8);
stem(0:length(h3)-1, h3);
```

```
title('h3(n)');
xlabel('n');
ylabel('Amplitude');

% Subplot for y3(n)
subplot(3, 3, 9);
stem(0:length(y3)-1, y3);
title('y3(n)');
xlabel('n');
ylabel('Amplitude');
```



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=0:3;
h=0.5.^n;
y=[1, 2, 2.5, 3, 3, 3, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0 ];
[x,R]=deconv(y,h);
figure
subplot(3,1,1);
stem(h);
title('Impulse response');
xlabel('n');
ylabel('h(n)');
subplot(3,1,2);
stem(y);
title('Output');
xlabel('n');
ylabel('y(n)');
subplot(3,1,3);
stem(x);
title('Input');
xlabel('n');
ylabel('x(n)');
```