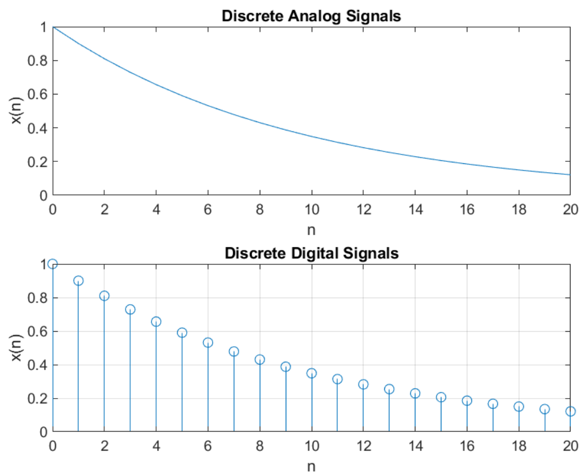
# Digital Signal Processing Project of Chapter 2

Homework 1 – 1

1. Write a MATLAB program to generate a discrete-time exponential signal. Use this function to plot the exponential $x[n]=(0.9)^n$ over the range $n$=0, 1, 2, …, 20.

```
%% Homework 1 - 1
%------------------------------------------------------------------------------
%  Write a MATLAB program to generate a discrete-time exponential signal. Use this
%  function to plot the exponential x[n]=(0.9) n  over the range n=0, 1, 2, …, 20.
%------------------------------------------------------------------------------
n = 0:20;
x= power(0.9 ,n);
%--------------------------------
figure(1)
subplot(2,1,1);
plot(n,x);
xlabel('n');
ylabel('x(n)');
title('Discrete Analog Signals');
%---------------------------------
subplot(2,1,2);
x = stem(n,x);
xlabel('n');
ylabel('x(n)');
title('Discrete Digital Signals');
%---------------------------------
grid
saveas(gcf,'Hw1-1.png')
```




The homework 1-1 have 2 figures , Discrete Analog Signals (right top) and Discrete Digtial Signals(right bottom).

This code said :

1. Use n = 0: 20 arranged the range is 0 to 20.
2. The mathematical function is $0.9^n$.
3. First use plot() function to plot the mathematical function.
4. Second use stem() founction to stem the mathematical function.
5. Finally , save the total figure.

Home work 1 – 2a

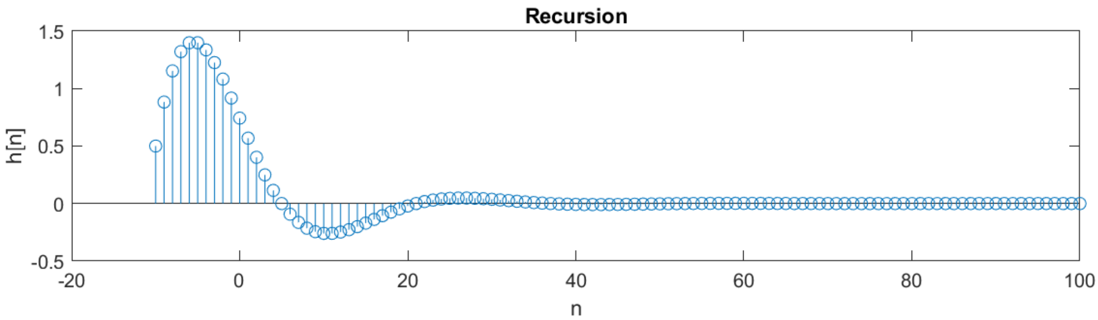
Using recursion $y[n] = 1.8\cos\left(\frac{\pi}{16}\right)y[n-1] - 0.81y[n-2] + x[n] + \frac{1}{2}x[n-1]$ generate and plot the impulse response h[n].

```
%% Homework 1 - 2a
%-------------------------------------------------------------------------
% y[n] - 1.8*cos(pi/16)*y[n-1] + 0.81*y[n-2] = x[n] +1/2 x[n-1]
%-------------------------------------------------------------------------
n = -10 :100 ;
y_axis = [];
for i = n
    y_axis(i+11) = func(y_axis,i);
end

figure(2)
subplot(2,1,1);
stem(n,y_axis);
xlabel('n');
ylabel('h[n]');
title('Recursion');
```

```
%% 1-2a function
function x = delta(n)  // impulse
    if n == 0
        x=1 ;
    else
        x=0;
    end
end

function y = func(y_s, i )
    if ( i < -10)
        y = 0;
    elseif(i == -10)
        y =  delta( i + 11 ) + 1/2*delta( i+ 11 -1 );
    elseif(i == -9)
        y = 1.8 *cos(pi/16) * y_s(i-1 + 11) + delta(i + 11) + 1/2* delta(i+11 - 1) ;
    else
        y = 1.8*cos(pi/16)* y_s(i - 1 + 11) - 0.81*y_s(i - 2 + 11) +delta(i + 11)+ 1/2 * delta(i-1 +11);
    end
end
```




Homework 1-2a use a recursion to generate signals , but the recursion is too slow to run. So when it run the recursion , it must record it variables itself.

Function 1 $x[n] = \delta[n]$

$$\begin{cases} x[n] = 1, n = 0, \\ x[n] = 0, n - 1. \end{cases}$$

Function 2 Recursion

$$\begin{cases} y[n] = 0 & , n < 0, \\ y[n] = \delta[n] + \frac{1}{2}\delta[n-1] & , n = 0, \\ y[n] = 1.8\cos\left(\frac{\pi}{16}\right)y[n-1] + \delta[n] + \frac{1}{2}[n-1] & , n = 1, \\ y[n] = 1.8\cos\left(\frac{\pi}{16}\right)y[n-1] - 0.81y[n-2] + \delta[n] + \frac{1}{2}[n-1], otherwise. \end{cases}$$

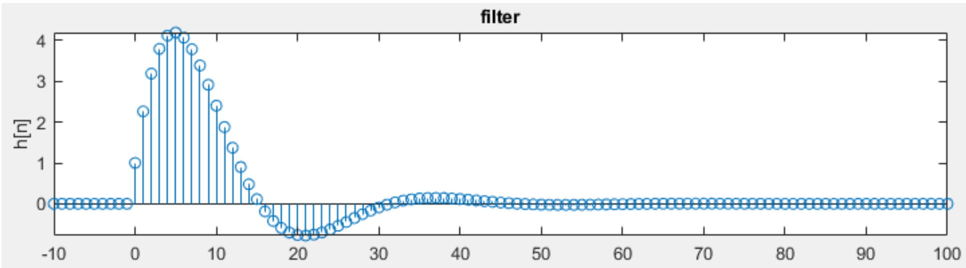
Homework 1-2b

Using the filter function generate and plot the impulse response h[n].

```
%% Homework 1 - 2b
%-------------------------------------------------------------------------------
x = zeros(1,111);

x(1,11) = 1 ;

A = [1,0.5];
B = [1, -1.8 *cos(pi/16) , 0.81];
y =filter(A,B,x);
subplot(2,1,2);
stem(-10:100 , y);
xlabel('n');
ylabel('h[n]');
title('filter');
axis tight;
saveas(gcf,'Hw1-2.png')
```




Using filter function is easy and no Time-consuming.

Just dismantle the recursion :

$$\left[1, -1.8\cos\left(\frac{\pi}{16}\right), 0.81\right]$$

And use filter() to finish the work.