1. Write a MATLAB program to find the impulse response and step response of the system given by

y(n) = x(n) + x(n-1) – 0.7y(n-1) at $-20 \le n \le 100$

THEORY

The impulse response of a given system is its response to impulse function input. We knowthat y[n]=impulse response h[n],when input x[n] is unit impulse function Step response of a system isits output for step function.

ALGORITHM

1. Start
2. Input the coefficients of x(n).
3. Generate impulse signal.
4. Input the coefficients of y(n).
5. Obtain the impulse response using filter function
6. Plot the impulse response
7. Generate step signal.
8. Obtain the step response using filter function.
9. Plot the step response
10. 10. Stop.

MATLAB FUNCTIONS USED

FILTER: One dimensional digital filter. Y=FILTER (B, A, X) filter the data in vector X with the filter described by vectors A and B to create the filtered data Y. The filter is a "Direct Form II Transposed"

implementation of the standard difference equation:

a(1) * y(n) = b(1) * x(n) +b(2) * x(n - 1) + ... +b(nb + 1) * x(n - nb) - a(2) * y(n - 1) - . . . a(na + 1) * y(n - na) If a(1) is not equal to 1,

filternormalizes the filter coefficients by a(1).

Code:

```
clear all
close all
x = [ 1 1];
y = [1 0.7];
n = -20:1:10;
imp = [zeros(1,20) 1 zeros(1,10)];
h = filter (x, y, imp);
subplot (2, 1, 1);
stem (n,h);
title(' Impulse Response ')
xlabel (' Samples ');
ylabel (' Amplitude ');
stp = [zeros(1,10 1 ones(1,20)];
```

```
h = filter(x,y,stp);
subplot ( 2, 1, 2);
stem(n,h);
title (' Step Response ');
xlabel (' Samples ');
ylabel (' Amplitude ');
```

1. An equation: y(n) = a^x(n).
   where x1(n) ={0,1,2,3}, x2(n)={1,2,3,4}, a1=a2=1, a=2.

   So, How can you write MATLAB code to test the system is linear or not?

Two important special cases of linearity property.

- **scaling property**: $\boxed{\mathcal{T}\{a\,x[n]\} = a\mathcal{T}[x[n]]}$
  Note that from $a = 0$ we see that zero input signal implies zero output signal for a linear system.
- **additivity property**: $\boxed{\mathcal{T}\{x_1[n] + x_2[n]\} = \mathcal{T}\{x_1[n]\} + \mathcal{T}\{x_2[n]\}}$
  Using proof-by-induction, one can easily extend this property to the general **superposition property**:

$$\boxed{\mathcal{T}\left[\sum_{k=1}^{K} x_k[n]\right] = \sum_{k=1}^{K} \mathcal{T}\{x_k[n]\}\,.}$$

Example: Proof that the accumulator is a linear system, where $y[n] = \sum_{k=-\infty}^{n} x[k]$.

Method:

- Find output signal $y_1[n]$ for a general input signal $x_1[n]$.
- "Repeat" for input $x_2[n]$ and $y_2[n]$.
- Find output signal $y[n]$ when input signal is $x[n] = a_1\,x_1[n] + a_2\,x_2[n]$.
- If $y[n] = a_1\,y_1[n] + a_2\,y_2[n]$, $\forall n$, then the system is linear.

For the accumulator, $y_1[n] = \sum_{k=-\infty}^{n} x_1[n]$ and $y_2[n] = \sum_{k=-\infty}^{n} x_2[n]$. If the input is $x[n] = a_1\,x_1[n] + a_2\,x_2[n]$, then th output is

$$y[n] = \sum_{k=-\infty}^{n} x[n] = \sum_{k=-\infty}^{n} (a_1\,x_1[n] + a_2\,x_2[n]) = a_1 \sum_{k=-\infty}^{n} x_1[n] + a_2 \sum_{k=-\infty}^{n} x_2[n] = a_1\,y_1[n] + a_2\,y_2[n]\,.$$

Since this holds for all n, for all input signals $x_1[n]$ and $x_2[n]$, and for any constants a_1 and a_2, the accumulator is linear.

Example: To show that $y[n] = \sqrt{x[n]}$ is nonlinear, all that is needed is a counter-example to the above properties. The scalin property will usually suffice. Let $x_1[n] = 2$, a constant signal. Then $y_1[n] = \sqrt{2}$. Now suppose the input is $x[n] = 3\,x_1[n] =$ (then the output is $y[n] = \sqrt{6} \neq 3\,y_1[n]$, so the system is nonlinear.

2. Test for causal non causal

y(k+2)+2y(k+1)+1/2*y(k)=x(k+1)+x(k)
Where x: input to the system and y: output of the system

```
clc;

close all;
k=2;%delay
n=0:2+k;
x=[10 2 5 zeros(1,k)]; %x(n)
subplot(411)
stem(n,x)
xdelay=[zeros(1,k) x(1:3)]; %x(n-2)
subplot(412)
stem(n,xdelay)
y=x+n.*xdelay;  %y(n)=x(n)+n*x(n-2)
% delayed output y'(n)=x(n-k)+(n-k)*x(n-k-2)
nk=(0:length(n)-1+k)-k;
ydelayed=[xdelay zeros(1,k)]+nk.*[zeros(1,k) xdelay]
subplot(413)
stem(0:length(ydelayed)-1,ydelayed)
n1=(0:length(n)-1+k);
ydin=[xdelay zeros(1,k)]+n1.*[zeros(1,k) xdelay]  % output due to delayed input
subplot(414)
stem(0:length(ydin)-1,ydin)
%%ydelayed is not equal system is non causal
```

Write a MATLAB program to find whether the given systems are linear, stable and causal.

1. y(n) = x(n) – 0.9y(n-1)

2. y(n) = exp x(n)

THEORY

A system is linear if and only if T [a1x1[n] + a2x2n]] = a1T [x1[n]] + a2T [x2[n]], for any arbitrary constants a1 and a2 .Let x[n] be bounded input sequence and h[n] be impulse response of the system and y[n], the output sequence. Necessary and sufficient boundary condition for stability is8n = -8|h[n]| < 8.Causal system is a system where the output depends on past or current inputs but not on the future inputs. The necessary condition is h[n] = 0; < 0, where h[n] is the impulse response.

ALGORITHM

1. Start
2. Input the coefficients of x(n) and y(n)
3. Generate random signals x1 and x2.
4. Check for linearity using filter function of the given system and display
5. Generate impulse signal.
6. Check for causality using filter function of the given system and display
7. Obtain the absolute value of impulse response and check for the stability of the system and display.
8. Stop

MATLAB FUNCTIONS USED

• RAND: Uniformly distributed pseudo random numbers. R= RAND (N) returns an N-by-N matrixcontaining pseudo-random values drawn from a uniform distribution on the unit interval. RAND (M ,N)or RAND([M, N]) returns an M-by-N matrix.

DISP: Display array. DISP(X) displays the array, without printing the array name. In all other ways it‟s the same as leaving the semicolon of an expression except that empty arrays don‟t display.

• ABS: Absolute value. ABS(X) is the absolute value of the elements of X.

• SUM : Sum of elements. S=SUM(X) is the sum of the elements of the vector X.

Program:

3. To find a linearity of a system

```
Clc
clear all
close all
% System 1% Linearity
b = [1 0];
a = [1 0.9];
x1 = rand (1,10);

x2 = rand (1,10);
```

```
y2 = filter (b, a, 2.*x1);
y5 = filter (b, a,2.* x2);
y = filter (b, a, x1);
y0 = filter (b, a, x2);
y6 = y2+y5;
y7 = y+y0;
if (y6-y7 ~= 0)
disp (' Linear ')
else
disp (' Non Linear ')
end;

% Causality--------%%%

n= -10:1:10;
x = [zeros(1,10) 1 zeros(1,10)];
y1 = filter (a, b, x);
subplot (2, 1, 1);
stem (n1,y1);
xlabel (' Samples ');
ylabel (' Amplitude ');
% Stability
T = abs (y1);
t = sum (T);
if (t < 1000)
disp (' Stable ')
else
disp (' Unstable ')
end;

%%%%%%%%%%% System 2%%%%%%%%%%%%%%%%%%%%%%
% 1 % Linearity

y8 = exp (x2);
y9 = exp ( x3);
y10 = 5*y8+5*y9;
y11 = exp(5*x2+5*x3);
if (y11-y10~=0)
disp (' 2Linear ')
else
disp (' 2Non Linear ')
end;


% 2 % Causality
Y8= exp (x);
subplot (2, 1, 2);
stem (n1,y8);
xlabel (' Samples ');
ylabel (' Amplitude ');
```

```
% 3 % Stability
T1= abs (y8);
t1 = sum (T1);
if (t1 < 1000)
disp (' 2Stable ')
else
disp (' 2Unstable ')
end;
```