3. DISCRETE-TIME SYSTEM

A system is a combination of elements that act together to perform function not possible with any of individual part. The system can be viewed as any process that results in the transformation of signals. Thus, a system has an input signal and output signal, which is related to the input through the system transformation.

In the *discrete-time system* (DTS) discrete–time inputs is transformed into discrete–time outputs (see Figure 1). These systems will be represented symbolically as

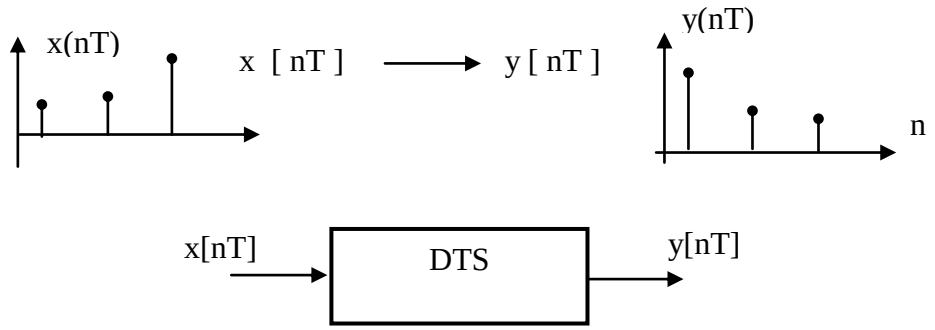


Figure 1

3.1 Examples of DTS:

3.1.1 Accumulator

$$y(n) = \sum_{k=-\infty}^{k=\infty} x(k) = \sum_{k=-\infty}^{-1} x(k) + \sum_{k=0}^{n} x(k) = y(-1) + \sum_{k=0}^{n} x(k) \qquad (3.1)$$

Where y(-1) is called the initial condition.

3.1.2 Upsampler

$$y(n) = \begin{cases} x\left(\frac{n}{L}\right), & 0, \pm L, \pm 2L, 3L, \ldots \\ 0 & otherwise \end{cases} \qquad (3.2)$$

L-up-sampling factor is a positive integer greater than 1.

Upsampling is operates by inserting L-1 zero-valued equispaced samples between each consecutive pair of input sequenses.

Matlab files torealise upsampling for L=3 is given below.

```
n=0:9;
x = [1 2 3 4 5 7 3 2 3 1];
L=3;
subplot(2,1,1)
stem(n,x,'fill')
y=zeros(1,L*length(x));
y([1:L:length(y)])=x;
subplot(2,1,2)
stem(n,y(1:length(x)),'fill')
```

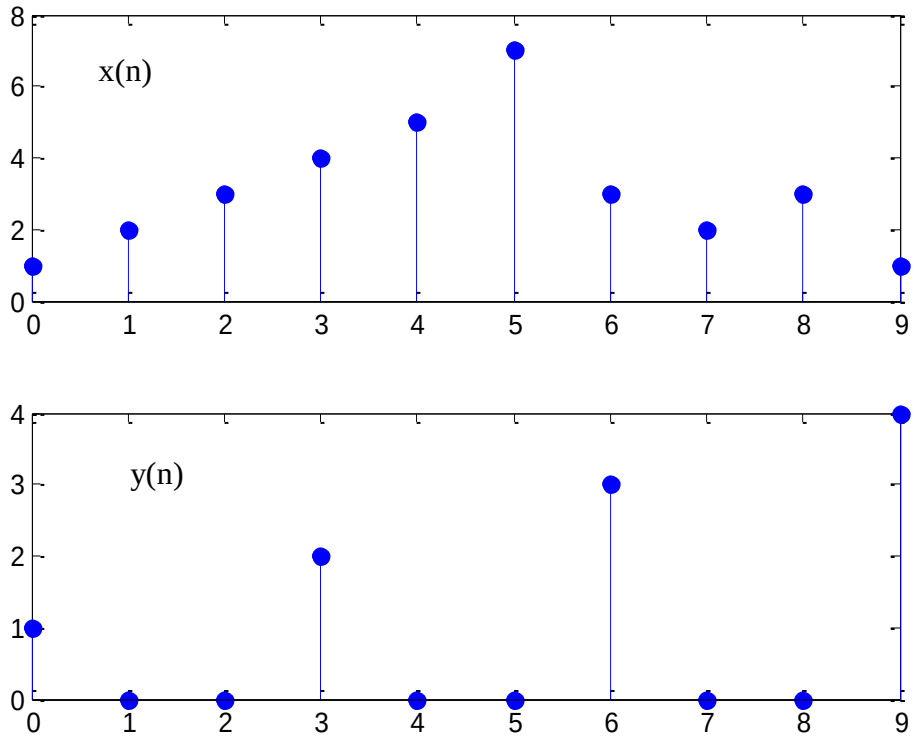


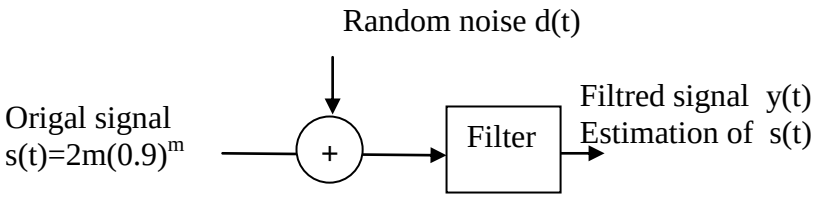
Figure 2

3.1.3 M-point Moving average

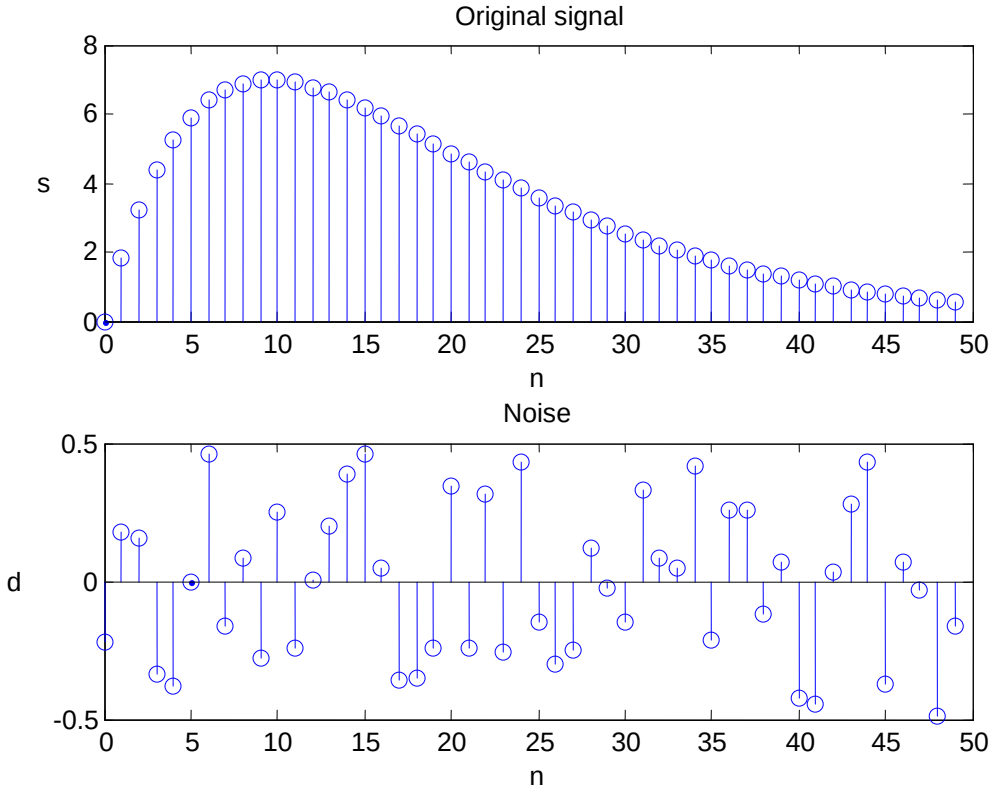
$$y(n) = \frac{1}{M}\sum_{k=0}^{M-1} x(n-k)\,; \qquad M = 3 \tag{3.3}$$

The average filtering is used in stock market analysis for determining slowly varying trend from data that contents high frequency fluctuations. In this case, a possible approach is to average data over an interval in order to smooth out the fluctuations and keep only trend.

Application of Moving Average to reduse noise from original signal




```
%Generation the
original signal
R=50;
m=0:1:R-1;
s=2*m.*(0.9.^m);
subplot (2,1,1)
stem(m,s);
xlabel('n');ylabel('s');
title('Original signal')
%Generation random noise
d=rand(R,1)-0.5;
subplot (2,1,2)
stem(m,d);
xlabel('n');ylabel('d');
title('Noise')
```




```
%Generation noisly signal
x=s+d';
plot (m,x)
xlabel('n');ylabel('x');
hold on
M=3;
Filtering of noisly signal
b=ones(M,1)/M;
y=filter(b,1,x);
plot (m,y,'r','linewidth',2)
xlabel('n');ylabel('y')
```

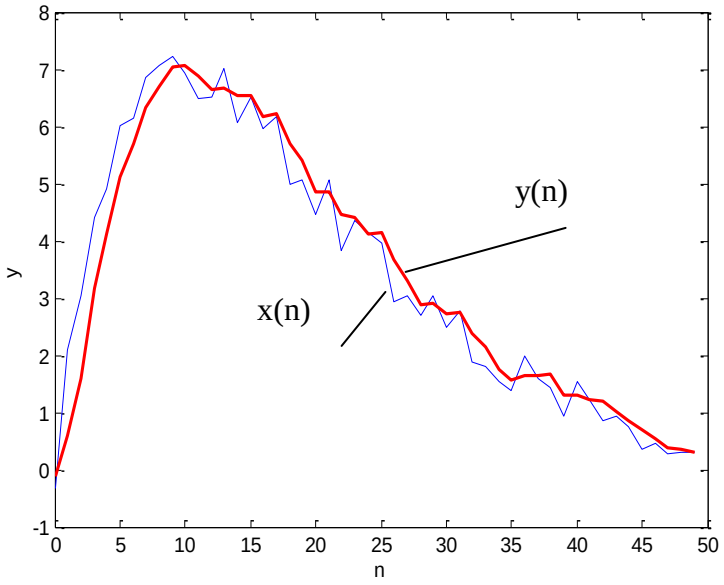


Figure 3

3.1.4 Linear Interpolation

Factor 2-interpolation:

$$y(n)=x(n)+\frac{1}{2}x(n-1)+x(n+1) \tag{3.4}$$

Factor 3-interpolation:

In the following figure is shown factor-3 interpolation.

$$y(n)=x(n)+\frac{1}{3}[x(n-1)+x(n+2)]+\frac{2}{3}[x(n-2)+x(n+1)] \tag{3.5}$$

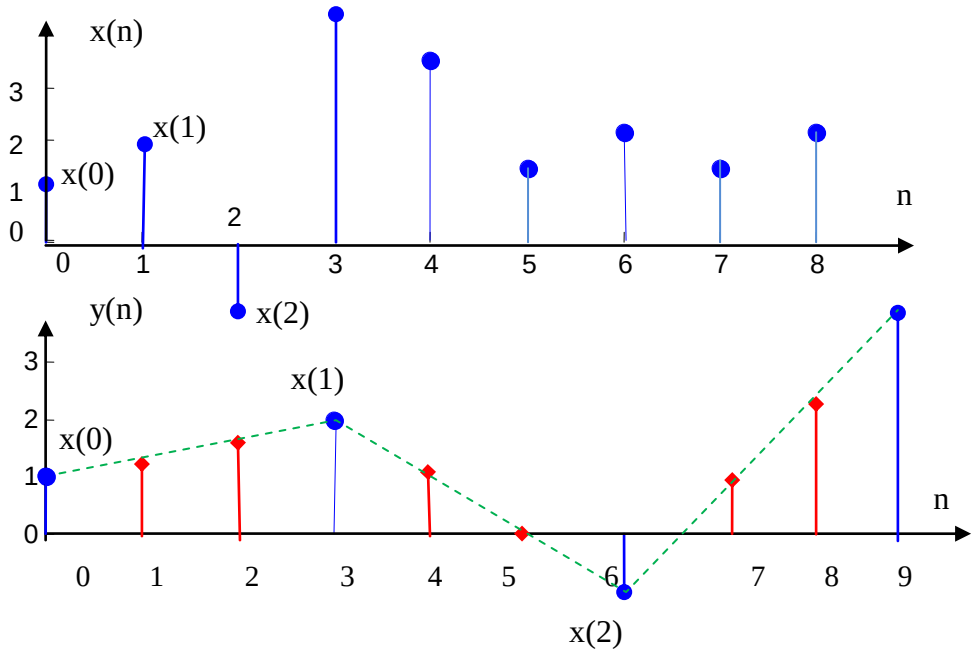


Figure 4

3.2 Properties of DTS

In this section, we introduce and discuss a number of basic properties of discrete-time systems.

3.2.1 Systems with and without memory

A system is said to be *memory less* if its output for each value n_0 of the independent variable n is dependents only on the input at time n_0. The output of the memoryless system is independent of the input applied before or after n_0.

One particularly simple memory less system is the *identity system*, whose output is identical to its input. That is,

$$y[n] = x[n]$$

An example of a *system with memory is unit time delay system*

$$y[n] = x[n-1]$$

or accumulator described by

$$y[n] = \sum_{k=0}^{n} x[k] = x(0) + x(1) + x(2) + \ldots + x(n)$$

3.2.2 Invertiblity and Inverse System

A system is invertible if by observing its output; we can determine its input. For the discrete–time case, we can construct an inverse system which when cascaded with the original system yields an output z [n] equal to the input x [n] to the first system. Thus, the series interconnection in Figure 5 has an overall input – output relationship that is the same as that for the identity system.

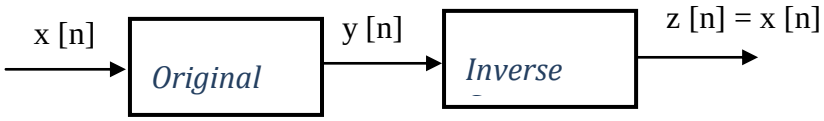


Figure 5

Find the inverse system for invertible system described by

$$y(n) = 2\,x(n)$$

for which the inverse system is

$$z(n) = (1/2)\ y(n)$$

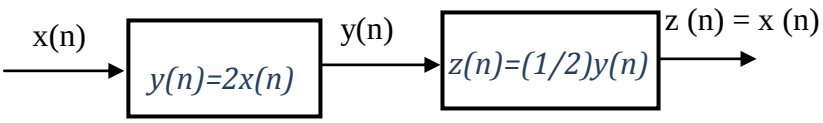


Figure 6

The inverse system is widely used in communication system to avoid distortion introduced by channel. For this purpose inverse of channel model filter is introduced to the receiver.

3.2.3 Causality

A system is causal if the output any time depends only on values of the input at the present time and in the past.

A system described by y(n)=ax(n+1) is non-causal or anticipative. Output depends of the future input.

The *forward difference system* defined by the relationship

$y[n] = x[n + 1] - x[n]$

is not causal, since the current value of the output depends on a future value of the input. The *backward difference system*, defined as

$y[n] = x[n] - x[n - 1]$

has an output that depends only on the present and past values of the input. This system is causal by definition.

Example

$$y(n)=a_0x(n)+ a_1x(n-1)+ a_2x(n-2)+ a_3x(n-3)+ a_4x(n-4)$$

Such a system is often referred to as being non anticipative, as the system output does not anticipate future values of the input. Example: The motion of an automobile is casual since it does not anticipate future actions of the driver.

Causality is a necessary condition for a system to be built in the real word. Causality is importance for communications and control systems that operate in real time. In speech, geophysical or meteorological data processing systems primary data can be stored in tapes and then processing later. Causality is not importance for image processing systems, in which independent variable is not time. Therefore, non-causal system can be used non-real time systems.

Example of noncausal system is average filtering described as

$$y[n]=\frac{1}{2M+1}\sum_{k=-M}^{M}x[n-k] \qquad (3.6)$$

Example for M=1, we have $y[n]=\frac{1}{3}\{x[n-1]+x[n]-x[n+1]\}$

Not that all memoryless systems (upsampler, accumulator)are causal.

3.2.4 Stability

Every system designed to process signals must be stable. If an electrical system is not stable, it will likely burn out or saturate after the application of an input, no matter how small. If a mechanical system or a structure is not stable, it will generally blow up or disintegrate. If a computer program is unstable, it will overflow. Thus, every system must be designed to be stable in order to function properly.

A system is said to be BIBO (bounded-input bounded-output) stable if every bounded input excites a bounded output.

This implies that if,

$$|x(n)| \leq B_u$$

Then for stable system $|y(n)| \leq B_y$

A system is said to be *marginally stable* or stable in the sense of Lyapunov if the response excited every finite initial state is bounded. It is said to be *asymptotically stable* if the response excited by every finite initial state is bounded and approaches zero at $n \to \infty$. These definitions are applicable only to linear systems.

Example 2.4. Output-input relationship of system is described by following equation

$$y[n]=\sum_{k=0}^{n} x[k] \qquad (3.7)$$

The system is stable?

Suppose x[n] = μ[n] is unit step sequence. Then y[0]=1, y[1]=2, y[2]=3, . . . or y[n]=[n+1] [n], system output increases without bound, that is the system is unstable.

Example 2.5. Considerer the averaging filter described by

$$y[n] = \frac{1}{M+1} \sum_{k=-M}^{M} x[n-k] \qquad (3.8)$$

Suppose x[n] =μ [n], then y[0]=1, y[1]=1, y[2]=1,…. The input is bounded by 1, output also is bounded by 1. System is stable.

3. 2. 5 Time Invariance

A system is time invariant if a time shift in the input signal causes a time in the output signal. Specifically, if y [n] is the output of a discrete – time– invariant system when x [n] is the input, then y [n - no] is the output when x [n – no] is applied.

$$x(n)=x_1(n-n_0)$$
$$y(n)=y(n-n_0)$$

Exersice. The system defined by the relation

$$y(n) = x(Mn)$$

with *M* a positive integer, is called a compressor. Specifically, it discards *(M* − 1*)* samples out of *M*; i.e., it creates the output sequence by selecting every *M* th sample. This system is not time invariant.We can show that it is not by considering the response $y\,1[n]$ to the input $x1[n] = x[n-n0]$. For the system to be time invariant, the output of the system when the input is $x1[n]$ must be equal to $y[n - n0]$. The output $y\,1[n]$ that results from the input $x1[n]$ can be directly computed

$y\,1[n] = x1[Mn] = x[Mn - n0]$

Delaying the output $y[n]$ by $n0$ samples yields $y[n - n0] = x[M(n - n0)]$. Comparing these two outputs, we see that $y[n-n0]$ is not equal to $y\,1[n]$ for all *M* and $n0$, and therefore, the system is not time invariant.

Example. Let us consider the continuous – time system defined by

$$y\,(n) = \sin\,[x\,(n)]$$

Suppose x(n)=μ(n); (0)=1, then y(0) = sin1; if (2)=1, then y (2) = sin 1, in general, if x(t)=(t)=1, the output is always *sin 1* no matter what *n* is. Thus, the system is time invariant.

Example 2.7 Consider y (n) = (sin*n*)x (n).

Suppose x(n)=μ(n); if x (n)=1 at n=0, then y(0) = (sin 0)x1=0; if x (n)=1 at t=1, then y(1) = (sin 1)x1=0.84x1=0.84. We find that the output is different if the same input is applied at different time. Thus, the system described by y (n) = (sin n)x (n) is time varying.

3. 2. 6 Linearity

Let us $y_1(n)$ is a response of a system to $x_1(n)$ and $y_2(n)$ be a response to $x_2(n)$.

$$x_1(n) \rightarrow y_1(n)$$

$$x_2(n) \rightarrow y_2(n)$$

A system is linear if:

$$ax(n) \rightarrow ay(n)$$

$$x_1(n) + x_2(n) \rightarrow y_1(n) + y_2(n)$$

The condition (2.2) is referred to as homogeneity (or scaling) property; the condition (2.3) is referred to as additivity property.

The two properties defining a linear system can be combined into a single statement, which is written below for both the continuous-time and discrete-time cases:

$$x(n) = ax_1(n) + bx_2(n)$$

$$y(n) = ay_1[n] + by_2[n]$$

where *a* and *b* are any complex constants. This very important fact is known as the superposition property.

Example. The system defined by

$$y[n] = \sum_{k=0}^{n} x[k]$$

is called the accumulator system, since the output at time *n* is the accumulation or sum of the present and all previous input samples. The accumulator system is a linear system. Since this may not be intuitively obvious, it is a useful exercise to go through
the steps of more formally showing this. We begin by defining two arbitrary inputs
*x*1[*n*] and *x*2[*n*] and their corresponding outputs

$$y_1[n] = \sum_{k=-\infty}^{n} x_1[k] \qquad\qquad y_2[n] = \sum_{k=-\infty}^{n} x_2[k]$$

When the input is $x_3[n] = ax_1[n] + bx_2[n]$, the superposition principle requires the output $y3[n] = ay_1[n] + by_2[n]$ for all possible choices of *a* and *b*.

$$y_3[n] = \sum_{k=-\infty}^{n} x_3[k] = \sum_{k=-\infty}^{n} (ax_1[k] + \sum_{k=-\infty}^{n} bx_2[k]) = a\sum_{k=-\infty}^{n} x_1[k] + b\sum_{k=-\infty}^{n} x_2[k] =$$

$$= y_1[n] + by_2[n].$$

Thus, the accumulator system of s atisfies the superposition principle for all inputs and is therefore linear.

Linear systems possess another important property, which is that zero input yields zero output. For example, if $x[n] \rightarrow y[n]$, then the scaling property tells us that

$$0 = 0.x\,[n] \rightarrow 0.y\,[n] = 0$$

A linear time-invariant system satisfies both the linearity and time –invariance properties. Consider then the system

$$y\,[n] = 2x\,[n] + 3$$

We see that this system not linear, since y [n] = 3, if x [n] = 0. On the other hand, this system falls into the class of *incrementally linear systems.*

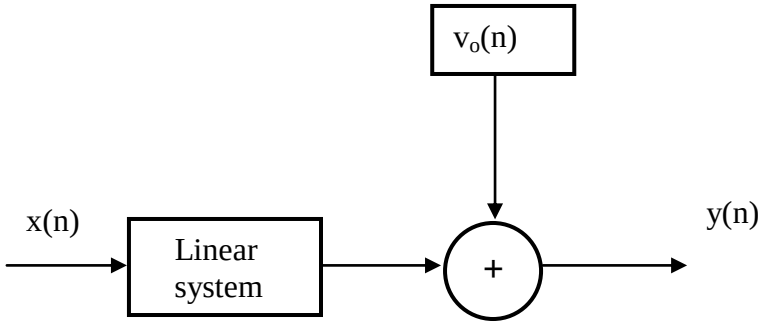


Figure 7

An incrementally linear system in continuous or discrete time is one that responds linearly to changes in the input. That is, the difference in the responses to any two inputs to an incrementally linear system is a linear function of the difference between the two inputs.

Any incrementally linear system can be shown as shown in Figure 2.13.

That is , theresponse of such a system equals the sum of a linear system and of another signal that is unaffected by the input.

Example. Determine if each of the following systems is linear?

1. y (n) = sin [x (n)]
2. y (n) = x(n)sin(n)
3. y (n) = x^2 (n)
4. y (n) = (sin n^2)x (n)

Solution:

1. y (n) = sin [x_1 (n) + x_2(n)] ≠ sin [x_1 (n)] + sin [x_2 (n)] → sysnem is non linear
2. y (n) = sin n(ax_1(n)+bx_2(n)) = ax_1(n)sin n + bx_2(n)sin n → system is linear
3. y (n) = [ax_1(n) + bx_2(n)]^2 ≠ [ax_1(n) + bx_2(n)] → system is non linear
4. y (n) = (sin n^2) [ax_1(n) + bx_2(n)] = ax_1(n)sin n^2 + bx_2(n)sin n^2→ system is linear

3. 3 Impulse and Step Response

The response of digital fitler to a unit sample sequence {δ[n]} is called the *unit sample* response, or simply, the *impulse response,* and is denoted as {h[n]}. Correspondingly, the response of a discrete-time system to a unit step sequence {μ[n]}, denoted as {s[n]}, is its *unit step response* or simply, the *step response*. As we show next, a linear time-invariant digital filter is completely characterized in the time-domain by its impulse response or its step response.

Example. The impulse response {h[n]} of the discrete-time accumulator is obtained by setting x[n]= δ [n] resulting in

$$h[n] = \sum_{l=-\infty}^{n} \delta[l],$$

which is precisely the unit step sequence μ [n].

Example. The impulse response {h[n]} of the factor-of-2 interpolator is obtained by setting x[n]= δ [n] and is given by

$$h[n] = \delta[n] + \frac{1}{2}(\delta[n-1] + \delta[n+1]).$$

The impulse response is seen to be a finite length sequence of length 3 and can be written alternately as example

h[n]={0.5, 1, 0.5}

3.4 Input-Output Relationship

A consequence of the linear, time-invariance property is that an LTI discrete-time system a completely specified by its impulse response, i.e., knowing the impulse response, we can compute the output of the system to any arbitrary input. We develop this relationship now.

Let h[n] denote the impulse response of the LTI discrete-time system of interest, i.e., the response to an input $\delta[n]$. we first compute the response of this filter to the input x[n]of eq. since the discrete-time system is time invariant, its response to $\delta[n-1]$ will be h[n-1]. Likewise, the responses to $\delta[n+2]$, $\delta[n-4]$ *and* $\delta[n-6]$ will be, respectively, h[n+2], h[n-4], and h[n-6]. Because of linearity, the response of the LTI discrete-time system to the input

$$x[n] = 0.5\delta[n+2] + 1.5\delta[n-1] - \delta[n-2] + \delta[n-4] + 0.75\delta[n-6]$$

Will be simply

$$y[n] = 0.5h[n+2] + 1.5h[n-1] - h[n-2] + h[n-4] + 0.75h[n-6].$$

We now generalize the above result for an arbitrary input sequence x[n] that can be expressed as

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k]. \qquad (3.9)$$

In the above expression, x[k] denotes specifically the k-th sample value. The response of the LTI discrete-time system to the sequence $x[k]\delta[n-k]$ will be x[k]h[n-k]. As a result, the response y[n] of the discrete-time system to x[n] will be given by

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k],$$

which can be alternately written as

$$y[n] = \sum_{k=\infty}^{\infty} x[n-k]h[k] \tag{3.10}$$

by a simple change of variables. The above sum in Eqs. Is called the convolution sum of the sequences x[n] and h[n], and represented compactly as:

$$y[n] = x[n] \otimes h[n]$$

where the notation $\otimes$ denotes the convolution sum.

The convolution sum operation satisfies several useful properties. First, the operation is commutative, i.e.,

$$x_1[n] \otimes x_2[n] = x_2[n] \otimes x_1[n].$$

Second, the convolution sum operation, for stable and single-sided sequences, is associative, i.e.,

$$(x_1[n] \otimes x_2[n]) \otimes x_3[n] = x_1[n] \otimes (x_2[n] \cdot \otimes x_3[n]).$$

And last, the operation is distributive, i.e.,

$$x_1[n] \otimes (x_2[n] + x_3[n]) = x_1[n] \otimes x_2[n] + x_1[n] \otimes x_3[n].$$

It clear from the above discussion that the impulse response {h[n]} completely characterizes an LTI discrete-time system in the time-domain because, knows the impulse response, we can compute, in principle, the output sequence y[n] for any given input sequence x[n] using the convolution. The computation of an output sample is simply a sum of products involving fairly simple arithmetic operations such as additions, multiplications, and delays. However, in practice, the convolution sum can be employed to compute the output sample at any instant only if either the impulse response sequence and/or the input sequence is of finite length, resulting in a finite sum of products. Note that if both the input and the impulse response sequences are of finite length, the output sequence is also of finite length. In the case of a discrete-time system with an infinite-length impulse response, it is obviously not possible to compute the output using the convolution sum if the input is also of infinite length. We

shall therefore consider alternative time-domain descriptions of such system that involve only finite sums of products.

To understand the process involved in generating the new sequence y[*n*], let us first compute the sample y[0] that is given by

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]\, h[n-k].$$

For n=0

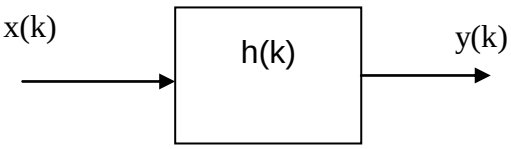
$$y[0] = \sum_{k=-\infty}^{\infty} x[k]\, h[-k].$$

Thus, to compute the output sample y[0] we first time-reverse the sequence h[*k*] to arrive at h[-*k*]. For n=1

$$y[1] = \sum_{k=-\infty}^{\infty} x[k]\, h[1-k],$$

which indicates that first the time-reserved sequence h[-*k*] is shifted to the right by one sampling period to arrive at the sequence h[1-*k*]. Then the *k*-th sample of this sequence is multiplied with the *k*-th input sample x[*k*] and summed over all values of k to arrive at y[1]. This process is continued for increasing values of *n*.

Example**.**



x(k) = (-2 0 1 -1 3)

h(k) = (1 2 0 -1)

Find y(k). Sketch sequences.

We systematically develop the sequence y[n] generated by the convolution of the two finite-length sequence x[n] and h[n] show in figure. As can be seen from the plot of the shifted time-reversed version {h[n-k]} for n<0 sketched in figure for n=-3, for any value of the sample index k, the kth sample of either {x[k]} or {h[n-k]} is zero. As a result the product of the k-th samples is always zero for any k, and consequently, the convolution sum of eq. leads to

y[n]=0 for n<0.

Consider now the calculation of y[0]. We form {h[-k]} as shown in figure. The product sequence {x[k]h[-k]} is potted in figure, which has a single nonzero sample for k=0, x[0]h[0]. Thus, y[0]=x[0]h[0]=-2.

For the computation of y[1], we shift {h[-k]} to the right by one sample period to form {h[1-k]} as sketched in figure. The product sequence {x[k]h[1-k]} shown in figure has one nonzero sample or k=0, x[0]h[1]. As a result,

Y[1]=x[0]h[1]+x[1]h[1] =-4

Y[2]=x[0]h[2]+x[1]h[1]+x[2]h[0]=0+0+1=1

y[3]=x[0]h[3]+x[1]h[2]+x[2]h[1]+x[3]h[0]=2+0+0+1=3,

y[4]=x[1]h[3]+x[2]h[2]+x[3]h[1]+x[4]h[0]=0+0-2+3=1,

y[5]=x[2]h[3]+x[3]h[2]+x[4]h[1]=-1+0+6=5,

y[6]=x[3]h[3]+x[4]h[2]=1+0=1,

y[7]=x[4]h[3]=-3.

From the plot of {h[n-k]} for n >7 as shown in figure and the plot of {x[k]} in figure, it can be seen that there is no overlap between these two sequence. As a result y[n]=0 for n>7. Output sequences generated by the convolution is given below.

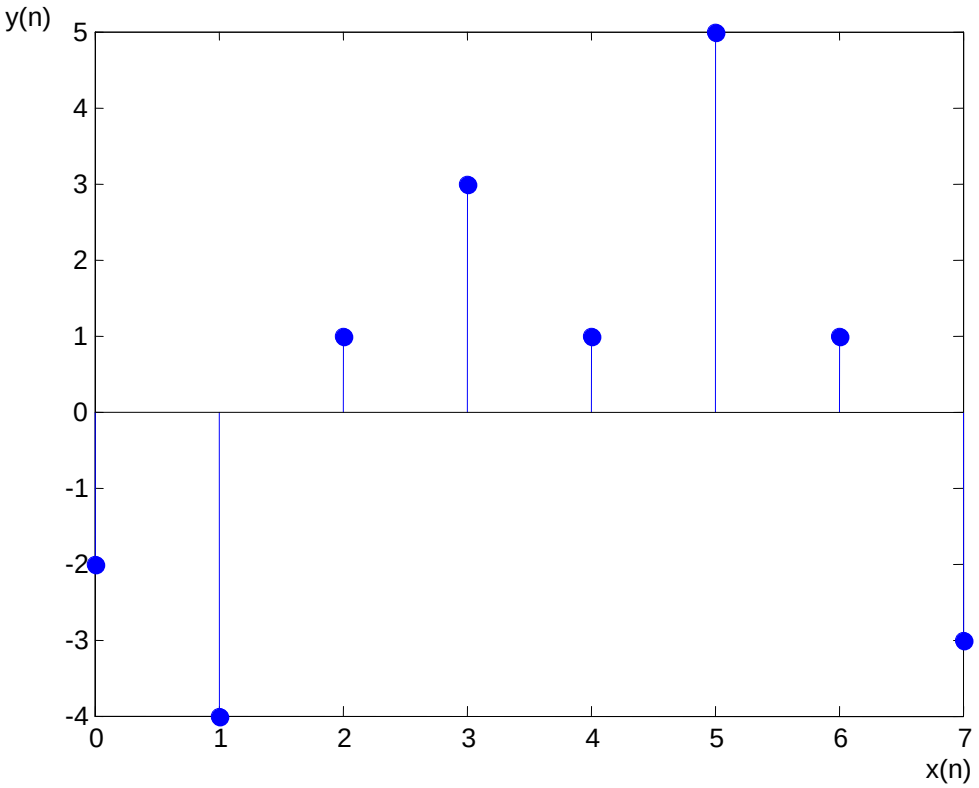


Figure 8

3.4. 1 Convolution Using Table

In addition to direct substitution, discrete convolution can also be carried out using the table. The table lists *h*[*k*] and *x*[*k*] on the top row and left-most column. The products of these

elements yield the entries of the table . The sums of those along the dotted lines yield the convolution . Note that the positions of *h[k]* & *x[k]* can be interchanged and the result will still be the same .

This is a convenient method of computing discrete convolution by hand.

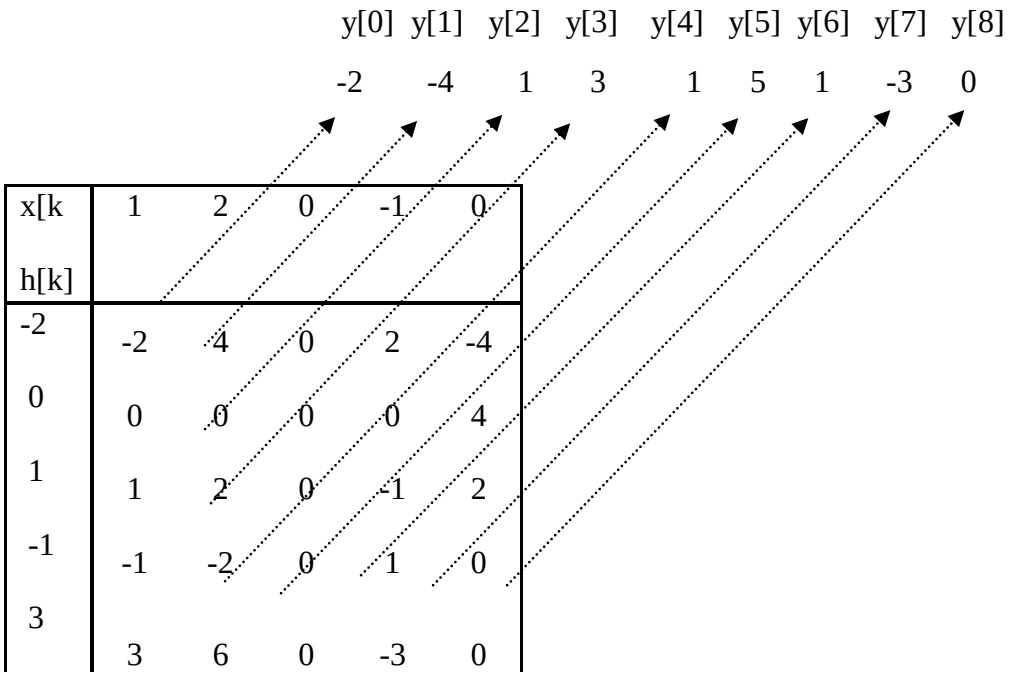


Figure 9

Matlab-files for calculation of convolution is given below.

```
% Convolution
x=[-2  0  1  -1  3];
h=[1  2  0  -1];
y=conv(x,h);
M=length(y)-1;
n=0:1:M;
disp(y);
stem(n,y,'fill','linewidth',3)
xlabel('n');ylabel('y')
```

3. 5 Interconnection of Systems

a) Series or cascade interconnection

A *series or cascade interconnection* of two systems is illustrated in Figure 2.1 (a). We will refer to diagrams such as this as *block diagrams.* Here the output of DTS_1 is the input of DTS_2, and the overall impulse response.

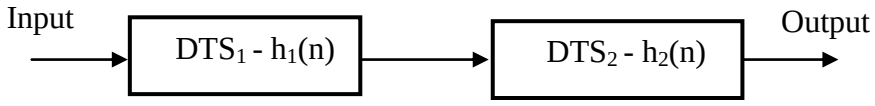


Figure 10

$$h[n] = h_1[n] \otimes h_2[n].$$

Note that, in general, the ordering of the filters in the cascade has no effect on the overall impulse response because of the commutative property of convolution.

It can be shown that the cascade connection of two stable systems is stable. Likewise, the cascade connection of the passive (lossless) systems is passive (lossless).

The cascade connection schema is employed in the development of an inverse system. If the two LTI system in the cascade connection of figure are such that

$$h_1[n] \otimes h_2[n] = \delta[n],$$

Then the LTI system $h_2[n]$ is said to be the inverse of the LTI system $h_1[n]$, and vice-versa. As a result of the above relation, if the input to the cascaded system is x[n], its output is also x[n]. An application of this concept is in the recovery of a signal from its distorted version appearing at the output of a transmission channel is known.

The following example illustrates the development of an inverse system.

Example.

From example the impulse of the discrete time accumulator is the unit step response $\mu[n]$. Therefore, from eq. the inverse system must satisfy the condition

$$\mu[n] \otimes h_2[n] = \delta[n].$$

It follows from eq. that $h_2[n] = 0$ for n<0 and

$$h_2[0]=1, \qquad \sum_{l=0}^{n} h_2[l]=0, \qquad n\geq 1.$$

As a results,

$$h_2[1]=-1 \text{ and } \qquad h_2[n]=0, \qquad \text{for } n\geq 2.$$

Thus the impulse response of the overall filter is given here by

$$h[n]=h_1[n]+h_2[n].$$

It is a simple exercise to show that the parallel connection of two stable systems is stable. However, the parallel connection of two passive (lossless) systems may or may not be passive (lossless).

b) Parallel interconnection

A *parallel interconnection* of two systems is illustrated in figure 11. Here the same input signal is applied to DTS_1 and DTS_2. The symbol "$\oplus$" in the figure denotes addition, so that the output of the parallel interconnection is the sum of the outputs of DTS_1 and DTS_2. We can also define parallel interconnections of more than two system , and we can combine both cascade and parallel interconnections to obtain more complicated interconnections . Interconnections such as systems can be used to construct new system out of existing ones.

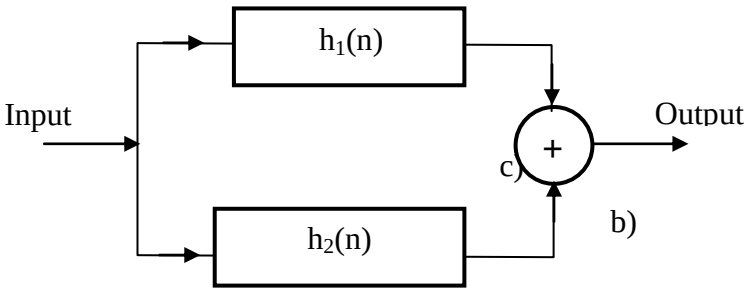


Figure 11

$$h[n]=h_1[n]+h_2[n].$$

Example. Design system to compute arithmetic expression

$$y[n] = (2x[n] - x[n]^2)^2$$

In the Figure 12 is shown block-diagram of desired system.

In addition to providing a mechanism that allows us to build new systems, interconnections also allow us to view an existing system as an interconnection of its component parts. For example, electrical circuits involve interconnections of basic

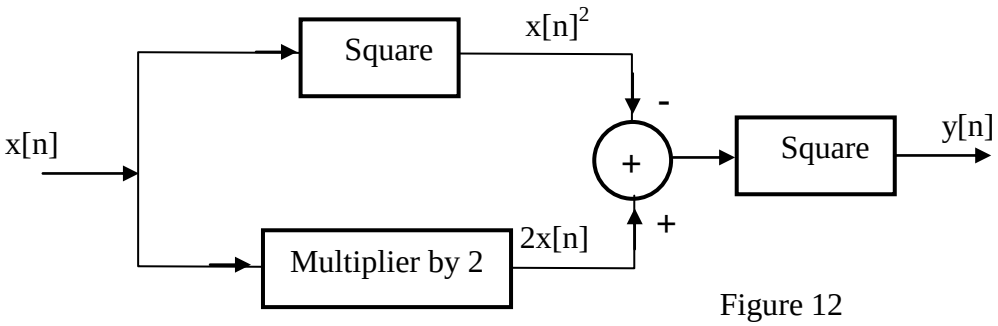
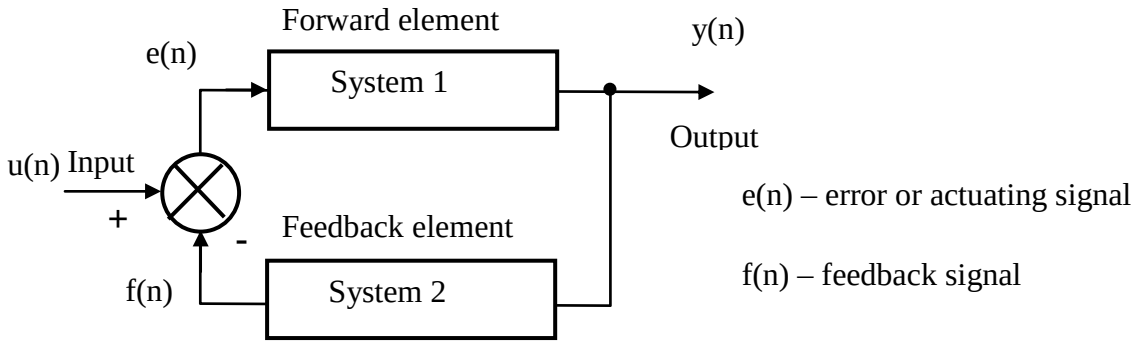


Figure 12

circuit elements (resistor, capacitor, inductors). Similarly, the operation of an automobile can be broken down into the interconnected operation of the carburetor, pistons, crankshaft, and so on. Viewing a complex system in this manner is often useful in facilitating the analysis of the system.

c) Feedback Interconnection

An example of which is illustrated in Figure 13. Here the output of the System 1 is the input to System 2, while the output of System 2 is fed back and added to the external input to produce the actual input to System 1. Feedback system arises in a wide variety of applications. For example, a speed governor on an automobile senses vehicle velocity and adjusts the input from the driver in order to keep the speed at a safe level. Also, electrical circuits are often useful viewed as containing feedback interconnections.



Fiigure 13

3.6 Stability Condition of LTI Systems

We know that DTS is defined to be BIBO stable if , if the output y(n) remain bounded for all bounded input x(n). LTI system is BIBO stable if and only if its impulse response h(n) is absolutely sumable

$$S = \sum_{i=0}^{\infty} |h(n)| < \infty \tag{3.11}$$

Example. h(n) $=(\alpha)^n \mu(n)$

$$S = \sum_{i=0}^{\infty} |h(n)| = \sum_{i=0}^{\infty} |(\alpha)^n \mu(n)| = \sum_{i=0}^{\infty} |(\alpha)^n| = \frac{1}{1-|\alpha|}$$

if $|\alpha| < 1$, S $< \infty$ system is BIBO stable. If, on the other hand, $|a| \geq 1$, then the sum is infinite and the system is unstable

3.7 Finite Dimensional LTI Discrete-Time Systems

An important subclass of LTI discrete-time systems is characterized by a linear constant coefficient difference equation of the form

$$\sum_{k=0}^{N} b_k y[n-k] = \sum_{k=0}^{M} a_k x[n-k] \ , \tag{3.12}$$

If we assume the system to be causal, then we can rewrite equation to express y[n] explicitly as a function of x[n]:

$$y[n] = -\sum_{k=1}^{N} \frac{b_k}{b_0} y[n-k] + \sum_{k=0}^{M} \frac{b_k}{b_0} x[n-k] \ ,$$

Where x[n] and y[n] are, respectively, the input and output of the system, and $\{b_k\}$ and $\{a_k\}$ are constants. The order of the discrete-time system is said to be *N* (assuming $N \geq M$), which is the order of the difference equation characterizing the system. It is possible to implement an LTI system characterized by eq. since the computation here involves two infinite sum of products. In general, such a system may not be causal, as illustrated by the following example.

Example.

The filter described by

$$y[n] - ay[n-1] = x[n]$$

has an impulse response h[n] (obtained by setting $x[n] = \delta[n]$);

$$h[n] = a^n \mu[n] + ca^n ,$$

where c is any arbitrary constant. Consider two solutions:

$$h_1[n] = a^n \mu[n], \qquad h_2[n] = -a^n \mu[-n-1].$$

The first solution $h_1[n]$, obtained for c=0, represents a causal system and is stable for $|a|<1$, where as the second solution $h_2[n]$, obtained for c=-1, represents a non causal system that is stable for $|a| > 1$.

Example.

There exist infinite-impulse response LTI discrete-time systems that cannot be characterized by the difference equation form of equation. The system defined by the impulse response

$$h[n] = \frac{1}{n^2} \mu[n-1]$$

is such an example. Note that the above system is causal and also BIBO stable.

If we assume the system to be causal, then we can rewrite equation to express y[n] explicitly as a function of x[n]:

The following example illustrates the computation of the impulse responses of an LTI system

Example.

Program given below can be employed to compute the impulse response of a causal finite-dimensional LTI discrete-system of the form of equation. The program calls for the following input data: desired length of the impulse response and the filter coefficient vectors p and d that must be entered inside square brackets. The program then plots the impulse response sequence. To illustrate its application we determine the first samples of the impulse response of the causal LTI system defined

$$y[n]+0.7y[n-1]-0.45y[n-2]-0.6y[n-3] = 0.8x[n]-0.44x[n-1]+0.36x[n-2]+0.02x[n-3].$$

```
%İmpulse Response calculation
N=41;
p=[0.8 -0.44 0.36 0.02];
d=[1 0.7 -0.45 -0.6];
x=[1 zeros(1,N-1)];
y=filter(p,d,x);
k=0:1:N-1;
stem(k,y,'fill')
```

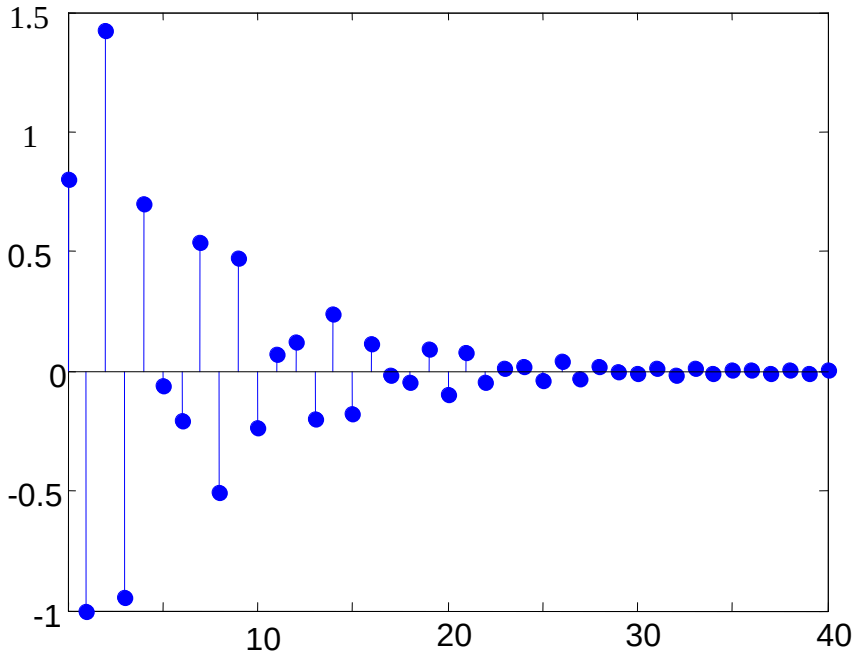


Figure 14

The impulse response sequence generated by the program is then plotted as indicated in figure. Note that the impulse response a discrete-time finite-dimensional system can also be computed in MATLAB using the function *impz.*

To determine the step response we replace in the above program the statement x = [1 zeros (1, N-1)] with the statement x= [ones (1, N)]. The computed first 41 samples of the step response are indicated in figure.

From the plots of figures we can conclude that most likely the LTI system of equation is BIBO stable. However, it is impossible to check the stability of a system just by examining only a finite segment of its impulse response as in these figures.

3. 8 Classification of LTI Discrete-Time Systems

Linear time-invariant (LTI) discrete-time systems are usually classified either according to the length of their impulse response sequences or according to the method of calculation employed to determine the output samples.

3.8.1 Classification Based on Impulse Response Length.

If h[n] is of finite length,

$$h[n]=0 \quad \text{for } n < N_1 \quad \text{and } n > N_2 \quad \text{with } N_1 < N_2, \tag{3.13}$$

then it is known as a finite impulse response (FIR) discrete-time system, for which the convolution sum reduces to

$$y[n] = \sum_{k=N_1}^{N_2} h[k]x[n-k]. \tag{3.14}$$

Note that the above convolution sum, being a finite sum, can be used to calculate y[n] directly. The basic operations involved are simply multiplication and addition. Note that when $N_1 = 0,$ the calculate of the present value of the output sequence involves the present value and $N_2 - N_1$ previous values of the input sequence and involves the $N_2 - N_1 + 1$ impulse response samples describing the FIR discrete-time system.

Examples of FIR discrete-time systems are the moving average system and the linear interpolators.

If h[n] is of infinite length, then it is known as an infinite impulse response (IIR) discrete-time system. In this case the convolution sum cannot be used to numerically calculate the output sequence samples due to the infinite nature of the sum unless the input sequence is of finite length. The class of IIR filter we are concerned with in this text is the causal system

characterized by the linear constant coefficient difference equation of equation. Note that here also the basic operations needed in the output calculations are multiplication and addition. An example of an IIR system is the accumulator of equations.

3.8.2 Classification Based on the Output Calculation Process.

If the output sample can be calculated sequentially, knowing only the present and past input samples, the filter is said to be a *nonrecursive* discrete-time system. If, on the order hand, the computation of the output involves past output in addition to the present and past input samples, it is known as a recursive discrete-time system. An example of a nonrecursive system is the FIR discrete-time system implemented using equation. The IIR discrete-time system implemented using the difference equation of eq. is an example of the recursive system. This equation permits the recursive computation of the output response beginning at some instant $n = n_0$ and progressively for higher values of n provided the initial conditions $y[n_0 - 1]$ through $y[n_0 - N]$ are known. However, it is possible to implement an FIR system using a recursive computational scheme and an IIR system using a nonrecursive computational scheme. The former case is illustrated in the example below.

Example.

Consider the FIR discrete-time system given by the impulse response

$\{h[n]\}=\{1,1,1,1,1,1\}.$

$\uparrow$

Substituting the above in equation

$$y[n] = \sum_{k=N_1}^{N_2} h[k]x[n-k].$$

we get the input-output relation

$$y[n]=x[n]+x[n-1]+x[n-2]+x[n-3]+x[n-4]+x[n-5].$$

To develop the recursive version of this filter, we note from the above

$$y[n-1]=x[n-1]+x[n-2]+x[n-3]+x[n-4]+x[n-5]+x[n-6],$$

which can be rewritten as

$$y[n-1]-x[n-6]=x[n-1]+x[n-2]+x[n-3]+x[n-4]+x[n-5].$$

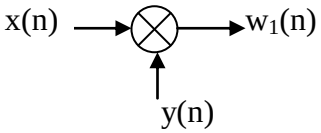
Substituting the above in equation we arrive at

$$y[n]=x[n]-x[n-6]+y[n-1],$$

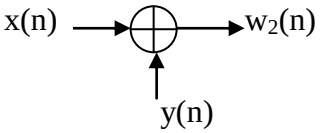
which is a difference equation of the form of equation.

3.9 Realization o f DTS

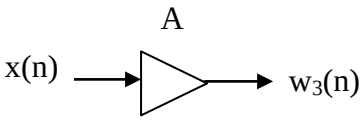
Basic elements:

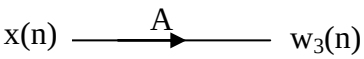


Modulation: $w_1(n)= x(n).y(n)$

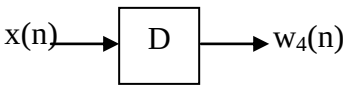


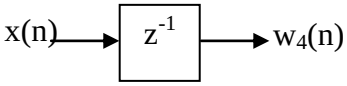
Addition: $w_2(n)= x(n) + y(n)$



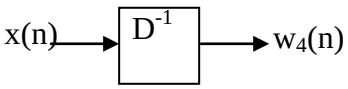


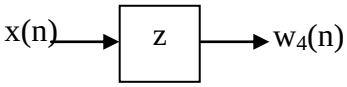
Multiplication: $w_1(n)= Ax(n)$





Unit delay: $w_1(n)= x(n-1)$





Unit advance: $w_1(n)= x(n-+1)$

3.9.1 Finite Impulse Response Systems

$$y(n) = \sum_{k=0}^{N} h(k)x(n-k)$$

$$H(z) = \frac{y(z)}{x(z)} = \sum_{k=0}^{N} h(k)z^{-k}$$ Transfer function

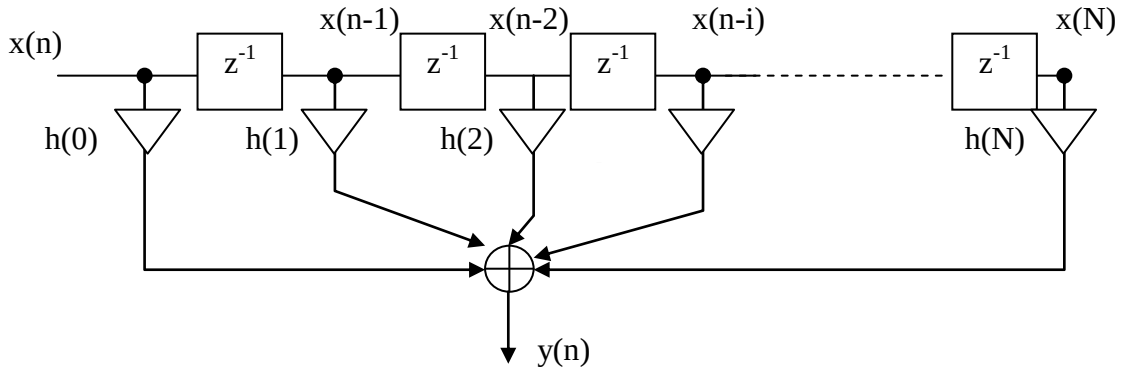


Figure 15

3.9.2 Infinite Impulse Response Systems

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k) = \sum_{k=0}^{N} b_k x(n-k) - \sum_{k=0}^{M} a_k y(n-k)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{N} b_k z^{-k}}{1 + \sum_{k=0}^{M} a_k z^{-k}}$$

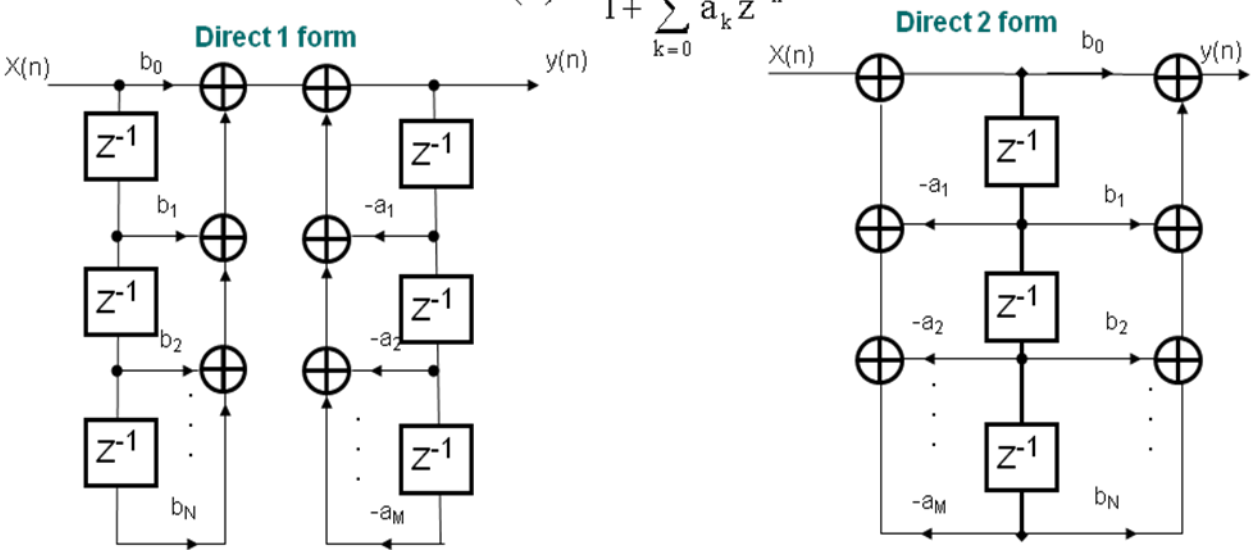


Figure 16

Example: Develope the relation between x(n) and y(n)

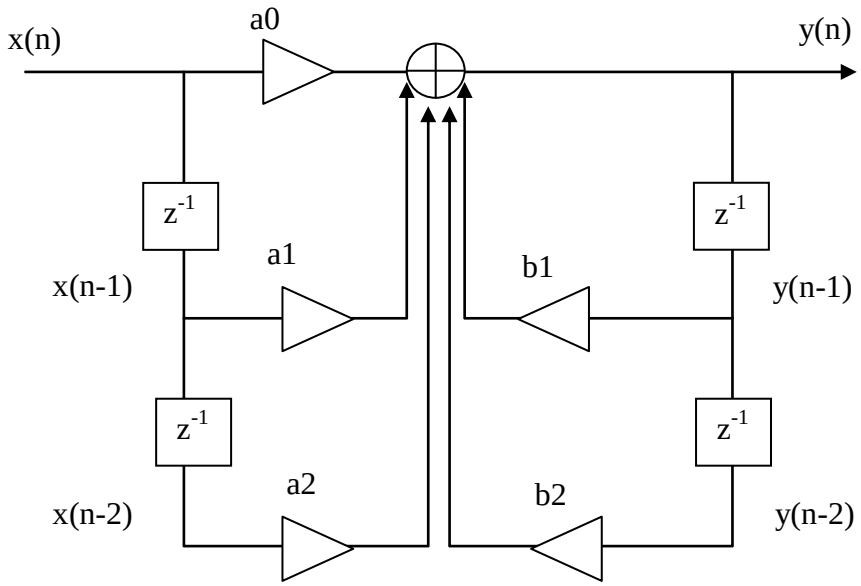


Figure 17

y(n) =a0x(n) + a1x(n-1) + a2x(n-2) + b1y(n-1) + b2y(n-2)

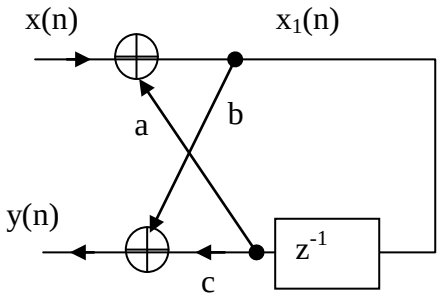


Figure 18

$x_1(z) = x(z) + x_1(z)z^{-1}a$

$$y(z) = bx_1(z) + x_1(z)z^{-1}c$$
$$x_1(z) - x_1(z)z^{-1}a = x(z)$$

$$x(z) = x_1(z) - x_1(z)z^{-1}a$$
$$x_1(z)(1 - a\,z^{-1}) = x(z)$$
$$x_1(z) = \frac{x(z)}{1-a^{-1}}$$

$$y(z) = b\frac{x(z)}{1 - az^{-1}} + \frac{x(z)cz^{-1}}{1 - az^{-1}}$$

Transfer Function of System

$$\boldsymbol{H(z) = \frac{y(z)}{x(z)} = \frac{b}{1-az^{-1}} + \frac{cz^{-1}}{1-az^{-1}}}$$

Output: **y(n)= ay(n-1)+bx(n)+cx(n-1)**