EENG 479 : Digital Signal Processing (DSP)

Lecture #4:

2.4 Discrete Time systems
2.5 Time Domain characterization of LTI Discrete-Time Systems
2.6 Simple Interconnection Schemes

Prof. Mohab A. Mangoud
Professor of Wireless Communications (Networks, IoT and AI)
University of Bahrain, College of Engineering
Department of Electrical and Electronics Engineering
P.O.Box 32038- Kingdom of Bahrain
mmangoud@uob.edu.bh
http://mangoud.com

2.4 Discrete Time systems

- 2.4.1 Discrete time system example:
  - Accumulator
  - Moving average filter ex2.13
  - Median Filter ex2.14 , prog 2.5 and Median Filter Demo

2.4 Discrete-Time Systems

The function of a discrete-time system is to process a given *input sequence* to generate an *output sequence*. In most applications, the discrete-time system used is a single-input, single-output system, as shown schematically in Figure 2.23. The output sequence is generated sequentially, beginning with a certain value of the time index n, and thereafter progressively increasing the value of n. If the beginning time index is n_o, the output $y[n_o]$ is first computed, then $y[n_o + 1]$ is computed, and so on. We restrict our attention in this text to this class of discrete-time systems with certain specific properties as described later in this section.

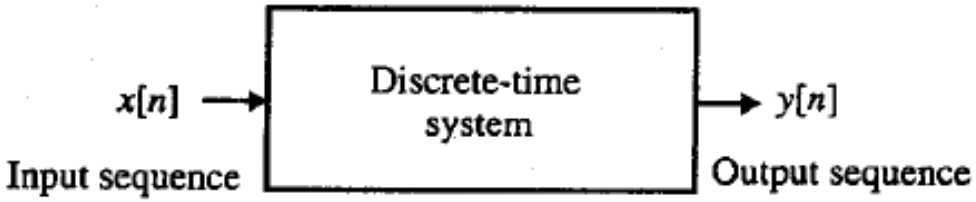


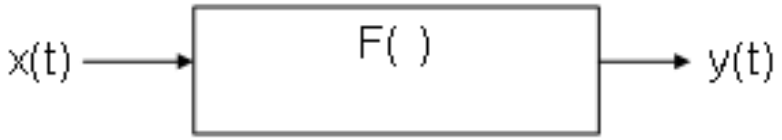
Figure 2.23: Schematic representation of a discrete-time system.

In a practical discrete-time system, all signals are digital signals, and operations on such signals also lead to digital signals. Such a discrete-time system is usually called a digital filter. However, if there is no ambiguity, we shall refer to a discrete-time system also as a digital filter, whether or not it has been implemented using finite precision arithmetic.

We consider here first several simple discrete-time systems that often find applications in practice. A study of these systems also will provide us with insights into the operation of more complex systems. We then describe several classifications of discrete-time systems. We next define two important characteristics of a discrete-time system.

Systems

- A *system* transforms a signal into a new signal or a different signal representation



- $y(t) = F(\ x(t)\ )$
- Examples: $y(t) = 2*x(t)$
  $y(t) = [x(t)]^2$
  $y(t) = x(t-2)$

Systems (cont.)

- A discrete-time system is the same concept:

$$y[n] = 2*x[n]$$
$$y[n] = \{x[n]\}^2$$
$$y[n] = x[n-2]$$

- Convert continuous-time signal to discrete-time signal:

$$y[n] = x(nT_s),$$

where T_s is the sampling period

Discrete-Time System

- Input and Output are discrete-time sequences:

$$y[n] = F(x[n])$$

- Some systems depend only on the current input:

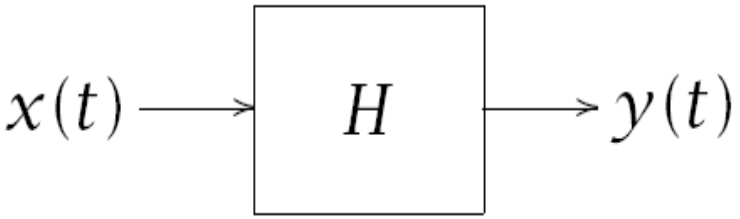
$$y[n] = 5x[n]$$

$$y[n] = 3(x[n])^2$$

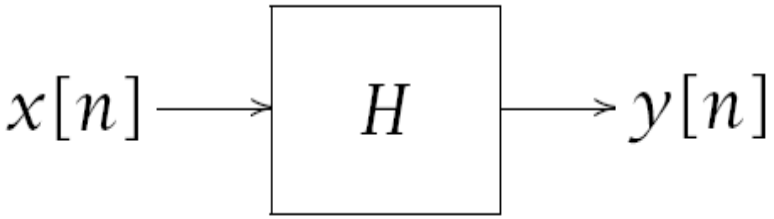
Systems

A *system* can be defined as an interconnection of operations that transforms input signals into output signals. In the simplest case — a *single-input, single-output (SISO)* system — we define an overall *operator* H that describes the transformation of the input into the output.

- In continuous time: $y(t) = H\{x(t)\}$.
- In discrete time: $y[n] = H\{x[n]\}$.



Continuous time



Discrete time

This slide is copied from the [1]

Discrete time system examples

Accumulator

A very simple example of a slightly more complex discrete-time system is the *accumulator,* defined by the input–output relation

$$y[n] = \sum_{\ell=-\infty}^{n} x[\ell]$$
$$= \sum_{\ell=-\infty}^{n-1} x[\ell] + x[n] = y[n-1] + x[n]. \tag{2.59}$$

The output $y[n]$ at time instant n is the sum of the input sample value $x[n]$ at time instant n and the previous output $y[n-1]$ at time instant $n-1$, which is the sum of all previous input sample values from $-\infty$ to the time instant $n-1$. The system therefore cumulatively adds; that is, it accumulates all input sample values from $-\infty$ to n. The accumulator can be considered as a discrete-time equivalent of a continuous-time integrator.

The above equation can also be written in the form

$$y[n] = \sum_{\ell=-\infty}^{-1} x[\ell] + \sum_{\ell=0}^{n} x[\ell] = y[-1] + \sum_{\ell=0}^{n} x[\ell], \qquad n \geq 0. \tag{2.60}$$

The second form of the accumulator is used for a causal input sequence, in which case $y[-1]$ is called the *initial condition.*

Moving-Average Filter

In Example 2.1, we pointed out that often data cannot be measured very accurately because of random variations in the measurements, and in the case of the data being corrupted by an additive noise, the n-th sample of the measured data $x[n]$ is modeled as $x[n] = s[n] + d[n]$, where $s[n]$ and $d[n]$ denote the n-th samples of the data and the noise. As demonstrated in this example, if multiple measurements of the same set of data samples are available, a reasonably good estimate of the uncorrupted data vector can be found by evaluating the ensemble average. In applications where data measurements cannot be repeated, a commonly used estimate of the data sample $s[n]$ at instant n from M measurements of the noise-corrupted data sample $x[\ell]$ available for the range $n - M + 1 \leq \ell \leq n$ is the *M-point average* or *mean* $y[n]$ given by

$$y[n] = \frac{1}{M} \sum_{\ell=0}^{M-1} x[n - \ell]. \tag{2.61}$$

An estimate of the spread of the mean value $y[n]$ from the actual value $s[n]$ is usually given by the *standard deviation* defined by [Tha98]

$$\sigma[n] = \sqrt{\frac{\sum_{\ell=0}^{M-1} (x[\ell] - y[n])^2}{M}}. \tag{2.62}$$

The discrete-time system implementing Eq. (2.61) is usually called the *M-point moving-average filter.* In most applications, the data $x[n]$ is a bounded sequence, and as a result, the M-point average $y[n]$ is also

Matlab implementation

$$y[n] = \frac{1}{M}\left(\sum_{\ell=0}^{M-1} x[n-\ell] + x[n-M] - x[n-M]\right)$$

M add

$$= \frac{1}{M}\left(\sum_{\ell=1}^{M} x[n-\ell] + x[n] - x[n-M]\right)$$

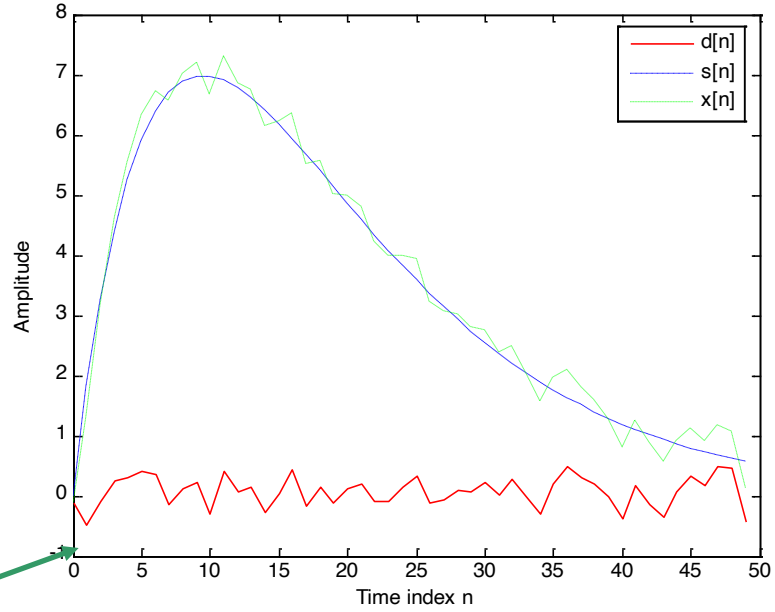
$$= \frac{1}{M}\left(\sum_{\ell=0}^{M-1} x[n-1-\ell] + x[n] - x[n-M]\right),$$

5

$$y[n] = y[n-1] + \frac{1}{M}\left(x[n] - x[n-M]\right).$$

2 add

```
% Program 2_4
% Signal Smoothing by a Moving-Average Filter
%
R = 50;
d = rand(R,1)-0.5;
m = 0:1:R-1;
s = 2*m.*(0.9.^m);
x  = s + d';
plot(m,d,'r-',m,s,'b--',m,x,'g:')
xlabel('Time index n'); ylabel('Amplitude')
legend('d[n]','s[n]','x[n]');
pause
M = input('Number of input samples = ');
b = ones(M,1)/M;
y = filter(b,1,x);
plot(m,s,'r-',m,y,'b--')
legend('s[n]','y[n]');
xlabel ('Time index n');ylabel('Amplitude')
```



FILTER One-dimensional digital filter.

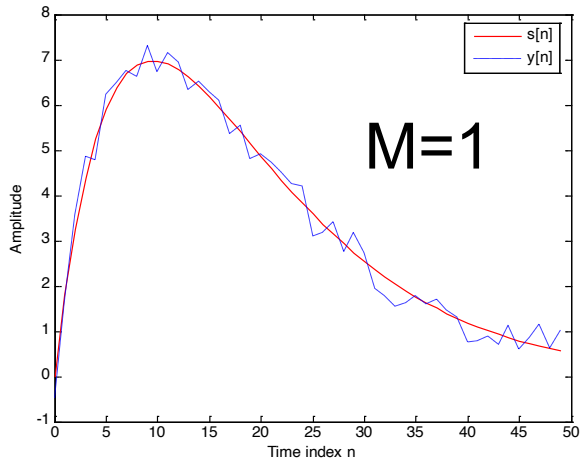
Y = FILTER(B,A,X) filters the data in vector X with the filter described by vectors A and B to create the filtered data Y. The filter is a "Direct Form II Transposed" implementation of the standard difference equation:

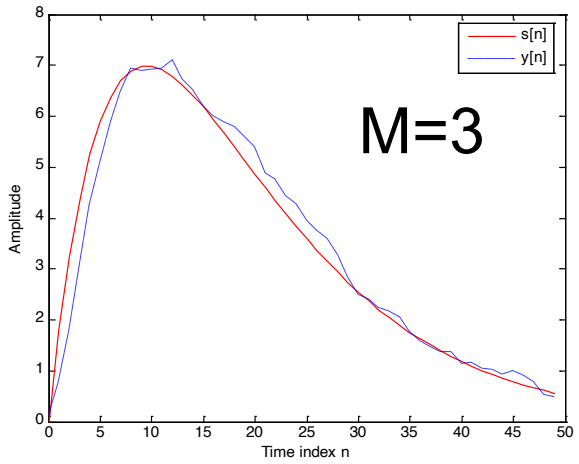
a(1)*y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb)
- a(2)*y(n-1) - ... - a(na+1)*y(n-na)

If a(1) is not equal to 1, FILTER normalizes the filter coefficients by a(1).

$$y[n] = \frac{1}{M} \sum_{\ell=0}^{M-1} x[n-\ell].$$

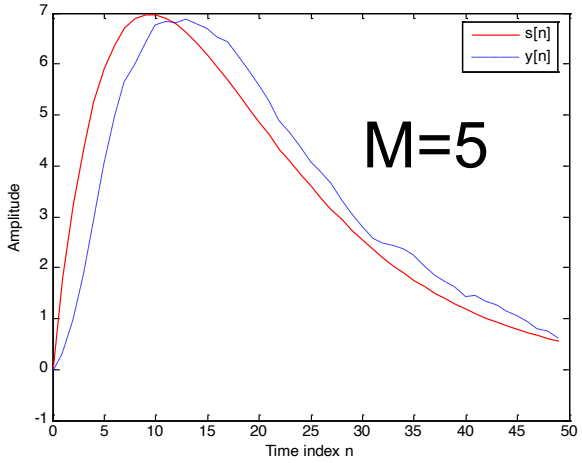
Figure 2.24: Pertinent signals of Example 2.13: $s[n]$ is the original uncorrupted sequence, $d[n]$ is the noise sequence, $x[n] = s[n] + d[n]$, and $y[n]$ is the output of the moving-average filter.

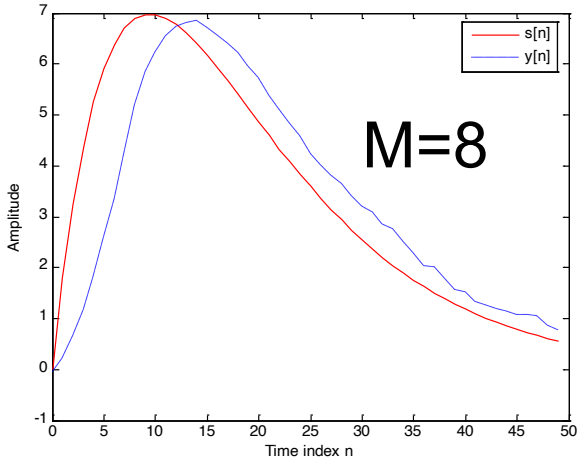




a delay of $(M-1)/2$ samples is inherent in an M-point moving-average filter.

Note: it is discrete signal





original uncorrupted signal. The proper choice of M depends on the nature of the noise corrupting the original signal. In some applications, higher quality smoothed output may be obtained by using a cascade of identical moving average filters with a smaller value of M to process the noise-corrupted signal.

Exponentially Weighted Running Average Filter

The average computed using Eq. (2.61) places equal emphasis on all M data samples. In some applications, it may be necessary to place more emphasis on data samples near the time instant n and less emphasis on data samples that are further away in determining the average. Such an average can be computed using the *exponentially weighted running average filter* given by [Tha98]:

$$y[n] = \alpha y[n-1] + x[n], \qquad 0 < \alpha < 1. \tag{2.64}$$

Computation of the running average using the above equation requires only 2 additions and 1 multiplication. Moreover, it requires only the storage of the previous running average and does not require the storage of past input data samples.

For $0 < \alpha < 1$, the exponentially weighted average filter of Eq. (2.64) places more emphasis on current data samples and less emphasis on past data samples by exponentially weighting the data samples. To illustrate this property, we observe that $y[n]$ can be rewritten as

$$\begin{aligned} y[n] &= \alpha\left(\alpha y[n-2] + x[n-1]\right) + x[n] \\ &= \alpha^2 y[n-2] + \alpha x[n-1] + x[n] \\ &= \alpha^2\left(\alpha y[n-3] + x[n-2]\right) + \alpha x[n-1] + x[n] \\ &= \alpha^3 y[n-3] + \alpha^2 x[n-2] + \alpha x[n-1] + x[n], \end{aligned}$$

by substituting the expression for $y[n-1]$ in Eq. (2.64) and then substituting the expression for $y[n-2]$ in the subsequent expression. Since $0 \leq \alpha < 1$, it can be seen from the last equation given above that the weights of past input data samples get progressively smaller at an exponential rate.

Linear Interpolator

Another example of a discrete-time system is the linear interpolator often employed to estimate sample values between pairs of adjacent sample values of a discrete-time sequence. The linear interpolation is

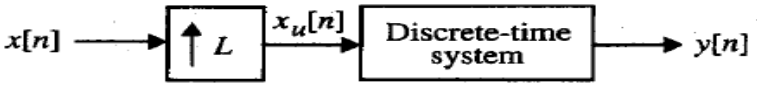


Figure 2.25: A factor-of-L interpolator.

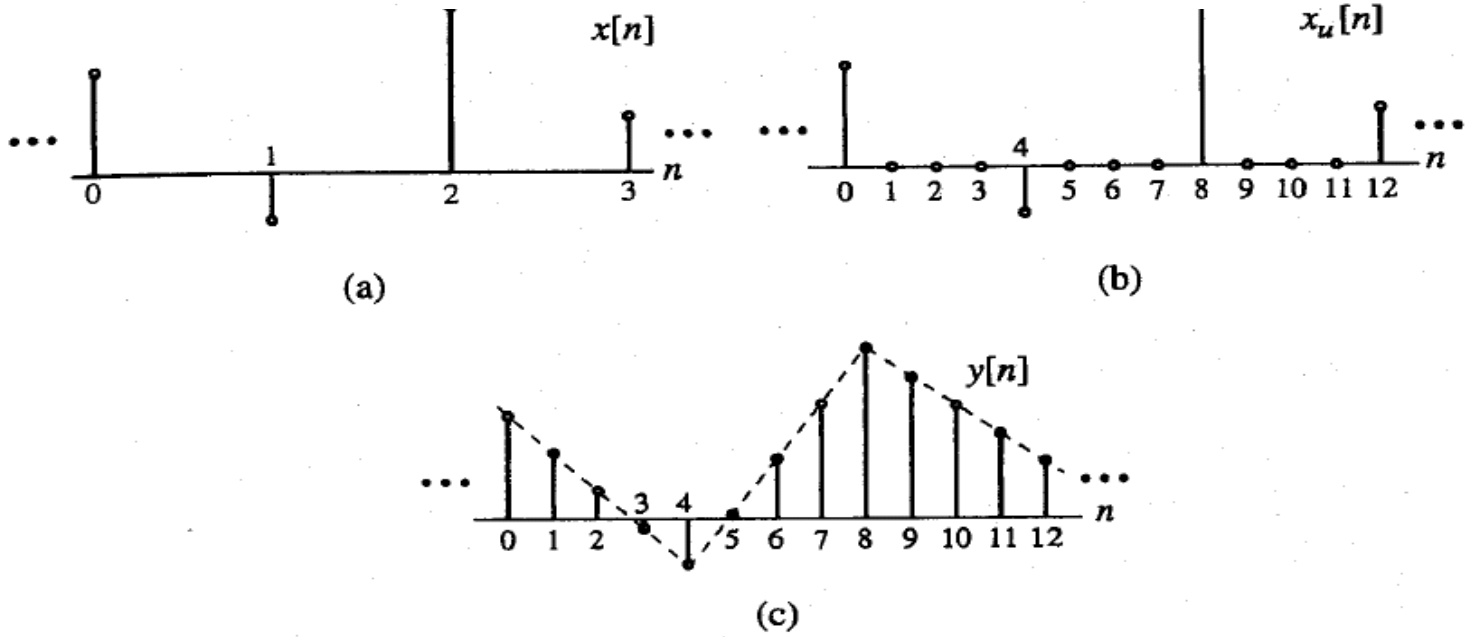


Figure 2.26: Illustration of the linear interpolation method.

implemented by first passing the input sequence $x[n]$ to be interpolated through an up-sampler whose output $x_u[n]$ is then passed through a second discrete-time system that "fills in" the zero-valued samples inserted by the up-sampler with values obtained by a linear interpolation of the pair of input samples surrounding the zero-valued samples, as indicated in Figure 2.25. The interpolated samples thus lie on a straight line joining the pair of input samples, as illustrated in Figure 2.26 for a factor-of-4 interpolation.

Bilinear interpolation

We develop the input–output relation of a linear factor-of-2 interpolator. Here, if $x_u[n]$ is a zero-valued sample inserted between a pair of input samples, it is replaced with the average of the two original input samples, $x_u[n-1]$ and $x_u[n+1]$:

$$y[n] = x_u[n] + \frac{1}{2}\left(x_u[n-1] + x_u[n+1]\right). \tag{2.65}$$

On the other hand, if $x_u[n]$ is one of the original input samples, its neighbors, $x_u[n-1]$ and $x_u[n+1]$, are both equal to 0. In this case, it follows from Eq. (2.65) that $y[n] = x_u[n]$, or, in other words, the input sample is left unchanged. Thus, the cascade of the factor-of-2 up-sampler and the discrete-time system defined by Eq. (2.65) implements a factor-of-2 interpolator. Equation (2.65) is also known as the *bilinear interpolation*.

Median Filter

The median of a set of $(2K+1)$ numbers is the number such that K numbers from the set have values greater than this number, while the other K numbers have values smaller. The median can be determined by rank-ordering the numbers in the set by their values and then choosing the number at the middle. For example,

consider the set of numbers $\{2, -3, 10, 5, -1\}$. The rank-ordered set is given by $\{-3, -1, 2, 5, 10\}$. Hence, $\text{med}\{2, -3, 10, 5, -1\} = 2$.

The *median filter* is implemented by sliding a window of odd length over the input sequence $\{x[n]\}$ one sample at a time [Reg93],[Tuk74]. At any instant, the output of the filter is the median value of the input samples inside the window. More specifically, the output sample $y[n]$ at the nth instant of the median filter with a window of length-$(2K+1)$ is given by

$$y[n] = \text{med}\{x[n-K], \ldots, x[n-1], x[n], x[n+1], \ldots, x[n+K]\}. \tag{2.67}$$

In practice, to process a finite-length sequence $\{x[n]\}$ of length N by a median filter with a window of length M, where $M < N$, $(M-1)/2$ zero-valued samples are appended to both sides of the input sequence $\{x[n]\}$ to create a new sequence $\{x_e[n]\}$ of length $N+M-1$:

$$x_e[n] = \begin{cases} 0, & -\frac{(M-1)}{2} \le n \le -1, \\ x[n], & 0 \le n \le N-1, \\ 0, & N \le n \le N-1+\frac{(M-1)}{2}. \end{cases}$$

The sequence $\{x_e[n]\}$ when processed by the median filter generates an output sequence $\{y[n]\}$ also of length N.

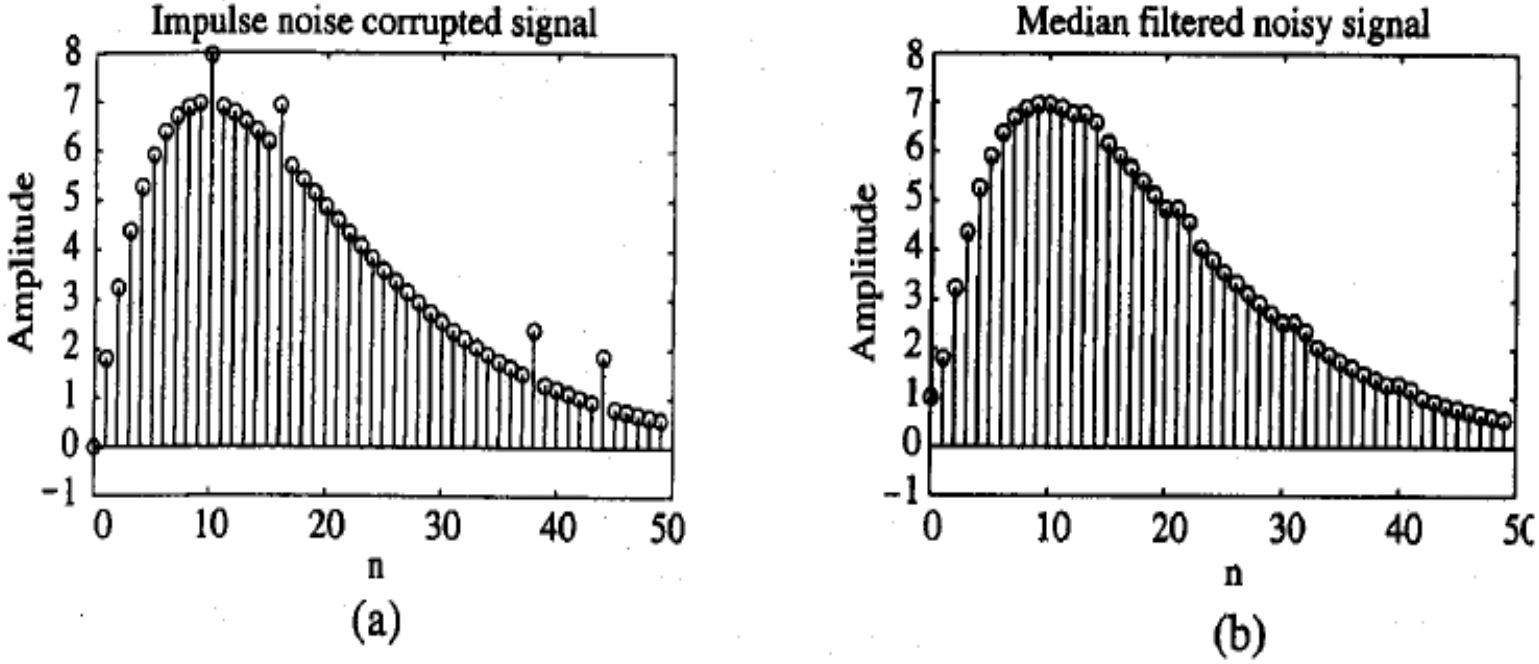


Figure 2.28: (a) Impulse-noise corrupted signal and (b) output of a length-3 median filter.

The median filter finds applications in removing additive random impulse noise, which show up as sudden large errors in the corrupted signal. In such cases, a linear lowpass filter, such as the moving-average filter or the exponentially weighted average filter, not only smooths out the sudden large-valued errors in the data but also distorts severely naturally occurring discontinuities in the original data. These types of

2.4 Discrete Time systems

2.4.2 Classification of Discrete- Time systems

- Linear system .. ex 2.15,16
- shift invariant system .. ex 2.17
- causal system
- stable system ex 2.18,19,20
- passive and step responses

- Impulse and step responses ex 2.21,22,23

Classification of Discrete time systems

- Linear system=L
- Shift invariant system=TI
- Causal System
- Stable System=BIBO
- Passive and lossless systems

Linear System

The most widely used discrete-time system, and the one that we shall be concerned with throughout most of this text, is a *linear system* for which the superposition principle always holds. More precisely, for a linear discrete-time system, if $y_1[n]$ and $y_2[n]$ are the responses to the input sequences $x_1[n]$ and $x_2[n]$, respectively, then for an input

$$x[n] = \alpha x_1[n] + \beta x_2[n],$$

the response is given by

$$y[n] = \alpha y_1[n] + \beta y_2[n].$$

The superposition property must hold for any arbitrary constants, α and β, and for all possible inputs, $x_1[n]$ and $x_2[n]$. The above property makes it very easy to compute the response to a complicated sequence that can be decomposed as a weighted combination of some simple sequences, such as the unit sample sequences or the complex exponential sequences. In this case, the desired output is given by a similarly weighted combination of the outputs to the simple sequences.

We examine the linearity property of the accumulator in Example 2.15.

Example Linearity Property of the accumulator

if

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

then the o/p $y[n]$ due to an input $(\alpha x_1[l]+\beta x_2[l])$

$$y[n]=\sum_{l=-\infty}^{n} (\alpha x_1[l]+\beta x_2[l])$$

$$y[n] = \alpha \sum_{l=-\infty}^{n} x_1[l] + \beta \sum_{l=-\infty}^{n} x_2[l]$$

$$y[n] = \alpha\, y_1[n] + \beta\, y_2[n]$$

this system is Linear

- Prove that the median filter is

  Nonlinear discrete time system

EXAMPLE 2.16 A Nonlinear Discrete-Time System

The median filter of Eq. (2.67) is a nonlinear discrete-time system. To show this, consider a median filter with a window of length 3. The output of the median filter for an input sequence $\{x_1[n]\} = \{3, 4, 5\}, 0 \le n \le 2$, is a length-3 sequence $\{y_1[n]\} = \{3, 4, 4\}$, and that for an input sequence $x_2[n] = \{2, -1, -1\}, 0 \le n \le 2$, is a length-3 sequence $\{y_2[n]\} = \{0, -1, -1\}$. On the other hand, the output for an input $\{x[n]\} = \{x_1[n] + x_2[n]\}$ is the sequence $\{y[n]\} = \{3, 4, 3\}$. It can be seen that $\{y_1[n] + y_2[n]\} = \{3, 3, 3\}$ is not equal to $\{y[n]\}$.

Median filter with a window length $M = 3$

$x_1[n] = \{3\ 4\ 5\} \longrightarrow N = 3$

$\therefore x_{e_1}[n] = [0\ 3\ 4\ 5\ 0]$ $\therefore y_1[n] = \{3\ 4\ 4\}$

3 4 4

$x_{e_2}[n] = [0\ 2\ -1\ -1\ 0]$ $y_2[n] = \{0\ -1\ -1\}$

0 -1 -1

Shift-Invariant System

The *shift-invariance* property is the second condition imposed on most digital filters in practice. For a shift-invariant discrete-time system, if $y_1[n]$ is the response to an input $x_1[n]$, then the response to an input $x[n] = x_1[n - n_o]$ is simply $y[n] = y_1[n - n_o]$, where n_o is any positive or negative integer. This relation between the input and output must hold for any arbitrary input sequence and its corresponding output. In the case of sequences and systems with indices n related to discrete instants of time, the above restriction is more commonly called the *time-invariance* property. The time-invariance property ensures that for a specified input, the output of the system is independent of the time the input is being applied.

It can be shown that the median filter of Eq. (2.67) is a time-invariant system (Problem 2.37).

Causal System

In addition to the above two properties, we impose, for practicality, additional restrictions of causality and stability on the class of discrete-time systems we deal with in this text. In a *causal* discrete-time system, the n_oth output sample $y[n_o]$ depends only on input samples $x[n]$ for $n \leq n_o$ and does not depend on input samples for $n > n_o$. Thus, if $y_1[n]$ and $y_2[n]$ are the responses of a causal discrete-time system to the inputs $u_1[n]$ and $u_2[n]$, respectively, then

$$u_1[n] = u_2[n] \quad \text{for } n < N \qquad \text{implies also that} \qquad y_1[n] = y_2[n] \quad \text{for } n < N.$$

Simply speaking, for a causal system, changes in output samples do not precede changes in the input samples. It should be pointed out here that the definition of causality given above can be applied only to discrete-time systems with the same sampling rate for the input and the output.[6]

It can be easily shown that the discrete-time systems of Eqs. (2.18), (2.59), (2.60), and (2.61) are causal systems. However, the discrete-time systems defined by Eqs. (2.65) and (2.66) are noncausal systems. It should be noted that these two noncausal systems can be implemented as causal systems by simply delaying the output by one and two samples, respectively.

Stable System

There are various definitions of stability. We define a discrete-time system to be *stable* if and only if for every bounded input, the output is also bounded. This implies that, if the response to $x[n]$ is the sequence $y[n]$ and if

$$|x[n]| < B_x \qquad \text{for all values of } n, \text{ then,} \qquad |y[n]| < B_y$$

for all values of n where B_x and B_y are finite positive constants. This type of stability is usually referred to as *bounded-input, bounded-output* (BIBO) *stability.*

Passive and Lossless Systems

A discrete-time system is said to be *passive* if, for every finite energy input sequence $x[n]$, the output sequence $y[n]$ has, at most, the same energy; that is,

$$\sum_{n=-\infty}^{\infty} |y[n]|^2 \leq \sum_{n=-\infty}^{\infty} |x[n]|^2 < \infty. \tag{2.70}$$

If the above inequality is satisfied with an equal sign for every input sequence, the discrete-time system is said to be *lossless*.

EXAMPLE 2.20 A Passive Discrete-Time System

Consider the discrete-time system defined by $y[n] = \alpha x[n - N]$, with N a positive integer. Its output energy is given by

$$\sum_{n=-\infty}^{\infty} |y[n]|^2 = |\alpha|^2 \sum_{n=-\infty}^{\infty} |x[n]|^2.$$

Hence, it is a passive system if $|\alpha| \leq 1$ and is a lossless system if $|\alpha| = 1$.

As we shall see later, in Section 12.9, the passivity and the losslessness properties are crucial to the design of discrete-time systems with very low sensitivity to changes in the filter coefficients.

2.4.3 Impulse and Step Responses

The response of a digital filter to a unit sample sequence $\{\delta[n]\}$ 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 $\{\mu[n]\}$, denoted as $\{s[n]\}$, is its *unit step response,* or simply, the *step response.*. As we show in Section 2.5, a linear time-invariant digital filter is completely characterized in the time-domain by its impulse response or its step response.

EXAMPLE 2.21 Determination of the Impulse Response

Consider an LTI discrete-time system with an input-output relation

$$y[n] = \alpha_1 x[n] + \alpha_2 x[n-1] + \alpha_3 x[n-2] + \alpha_4 x[n-3]. \tag{2.71}$$

Its impulse response $\{h[n]\}$ is obtained by setting $x[n] = \delta[n]$, resulting in

$$h[n] = \alpha_1 \delta[n] + \alpha_2 \delta[n-1] + \alpha_3 \delta[n-2] + \alpha_4 \delta[n-3].$$

The impulse response is thus a finite-length sequence of length 4 given by

$$\{h[n]\} = \{\alpha_1, \quad \alpha_2, \quad \alpha_3, \quad \alpha_4\}, \quad 0 \le n \le 3.$$

EXAMPLE 2.22 Impulse Response of the Accumulator

The impulse response $\{h[n]\}$ of the discrete-time accumulator of Eq. (2.59) is obtained by setting $x[n] = \delta[n]$, resulting in

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

which from Eq. (2.41a) is precisely the unit step sequence $\mu[n]$.

2.5 Time Domain characterization of LTI Discrete-Time Systems

2.5.1 input output relationship ex 2.24,25, 26,27, 28

2.5.2 tabular method of convolution sum computation ex 2.29, 30

2.5.3 stability condition in terms of the impulse response ex 31,32,33

2.5.4 causality condition in terms of the impulse response

2.5 Time-Domain Characterization of LTI Discrete-Time Systems

A *linear time-invariant* (LTI) discrete-time system satisfies both the linearity and the time-invariance properties. Such systems are mathematically easy to analyze and characterize and, as a consequence, easy to design. In addition, highly useful signal processing algorithms have been developed utilizing this class of systems over the last several decades. In this text, we consider almost entirely this type of discrete-time system.

In most cases, an LTI discrete-time system is designed as an interconnection of simple subsystems. Each subsystem, in turn, is implemented with the aid of the basic building blocks discussed earlier in Section 2.1.2. In order to be able to analyze such systems in the time domain, we need to develop the pertinent relationships between the input and the output of an LTI discrete-time system and the characterization of the interconnected system. We first show that the output sequence of a linear, time-invariant discrete-time system is given by the convolution sum of its impulse response sequence with the input sequence. We then outline a simple tabular method to compute the convolution sum of two finite-length sequences. From the convolution sum description of a linear, time-invariant discrete-time system, we next develop the stability condition and the causality condition in terms of its impulse response.

2.5.1 Input–Output Relationship

A consequence of the linear, time-invariance property is that an LTI discrete-time system is completely characterized by its impulse response; that is, 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, that is, the response to an input $\delta[n]$. We first compute the response of this filter to the input $x[n]$ of Eq. (2.52). Since the discrete-time system is time-invariant, its response to $\delta[n-1]$ is $h[n-1]$. Likewise, the responses to $\delta[n+2]$, $\delta[n-4]$, and $\delta[n-6]$ are, 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

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

It follows from the above result that an arbitrary input sequence $x[n]$ can be expressed as a weighted linear combination of delayed and advanced unit sample sequences in the form

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k], \tag{2.72}$$

where the weight $x[k]$ on the right-hand side denotes specifically the kth sample value of the sequence $\{x[n]\}$. The response of the LTI discrete-time system to the sequence $x[k]\delta[n-k]$ is $x[k]h[n-k]$. As a result, the response $y[n]$ of the discrete-time system to $x[n]$ is given by

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k], \tag{2.73a}$$

which can be alternately written as

$$y[n] = \sum_{k=-\infty}^{\infty} x[n-k]h[k] \tag{2.73b}$$

by a simple change of variables. The above sum in Eqs. (2.73a) and (2.73b) is called the *convolution sum* of the sequences $x[n]$ and $h[n]$ and represented compactly as

$$y[n] = x[n] \circledast h[n], \tag{2.74}$$

where the notation $\circledast$ denotes the convolution sum.[7]

EXAMPLE 2.24 Convolution of a Sequence with a Unit Sample Sequence

We evaluate

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

To this end, we rewrite the above equation as

$$y[n] = \sum_{k=-\infty}^{-1} x[n-k]\delta[k] + x[n]\delta[0] + \sum_{k=1}^{\infty} x[n-k]\delta[k].$$

Since $\delta[k] = 0$ for $k \neq 0$ and $\delta[0] = 1$, the above equation reduces to $y[n] = x[n]$, or in other words,

$$x[n] \circledast \delta[n] = x[n].$$

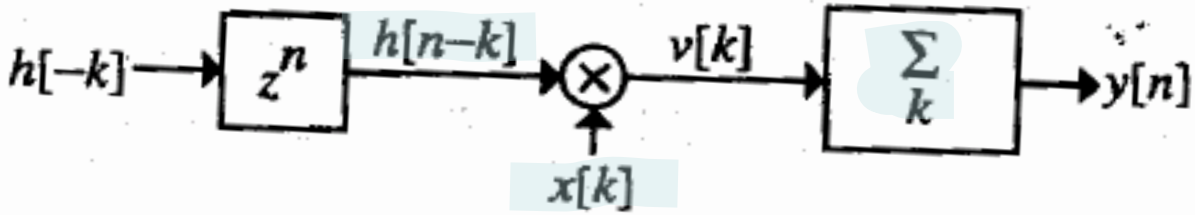


Figure 2.29: Schematic representation of the convolution sum operation.

The convolution sum operation satisfies several useful properties. First, the operation is *commutative;* that is,

$$x_1[n] \circledast x_2[n] = x_2[n] \circledast x_1[n]. \tag{2.75}$$

Second, the convolution operation, for stable and single-sided sequences, is *associative;* that is,

$$(x_1[n] \circledast x_2[n]) \circledast x_3[n] = x_1[n] \circledast (x_2[n] \circledast x_3[n]), \tag{2.76}$$

and last, the operation is *distributive;* that is,

$$x_1[n] \circledast (x_2[n] + x_3[n]) = x_1[n] \circledast x_2[n] + x_1[n] \circledast x_3[n]. \tag{2.77}$$

Proof of these properties is left as exercises (Problems 2.54 to 2.56).

The convolution sum operation of Eq. (2.73a) can be interpreted as follows. We first time-reverse the sequence $h[k]$, arriving at $h[-k]$. We then shift $h[-k]$ to the right by n sampling periods if $n > 0$, or to the left by n sampling periods if $n < 0$, to form the sequence $h[n-k]$. Next, we form the product sequence $v[k] = x[k]h[n-k]$. Summing all samples of $v[k]$ then yields the nth sample of $y[n]$ of the convolution sum. The process of generating $v[k]$ is illustrated in Figure 2.29. This process is implemented for each value of n in the range $-\infty < n < \infty$. The representation of the alternate form of the convolution sum operation given by Eq. (2.73b) is obtained by interchanging the sequences $x[k]$ and $h[k]$ in Figure 2.29.

It is clear from the above discussion that the impulse response $\{h[n]\}$ completely characterizes an LTI discrete-time system in the time domain because, knowing the impulse response, we can compute, in principle, the output sequence $y[n]$ for any given input sequence $x[n]$ using the convolution sum of Eq. (2.73a) or (2.73b). 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 systems that involve only finite sums of products.

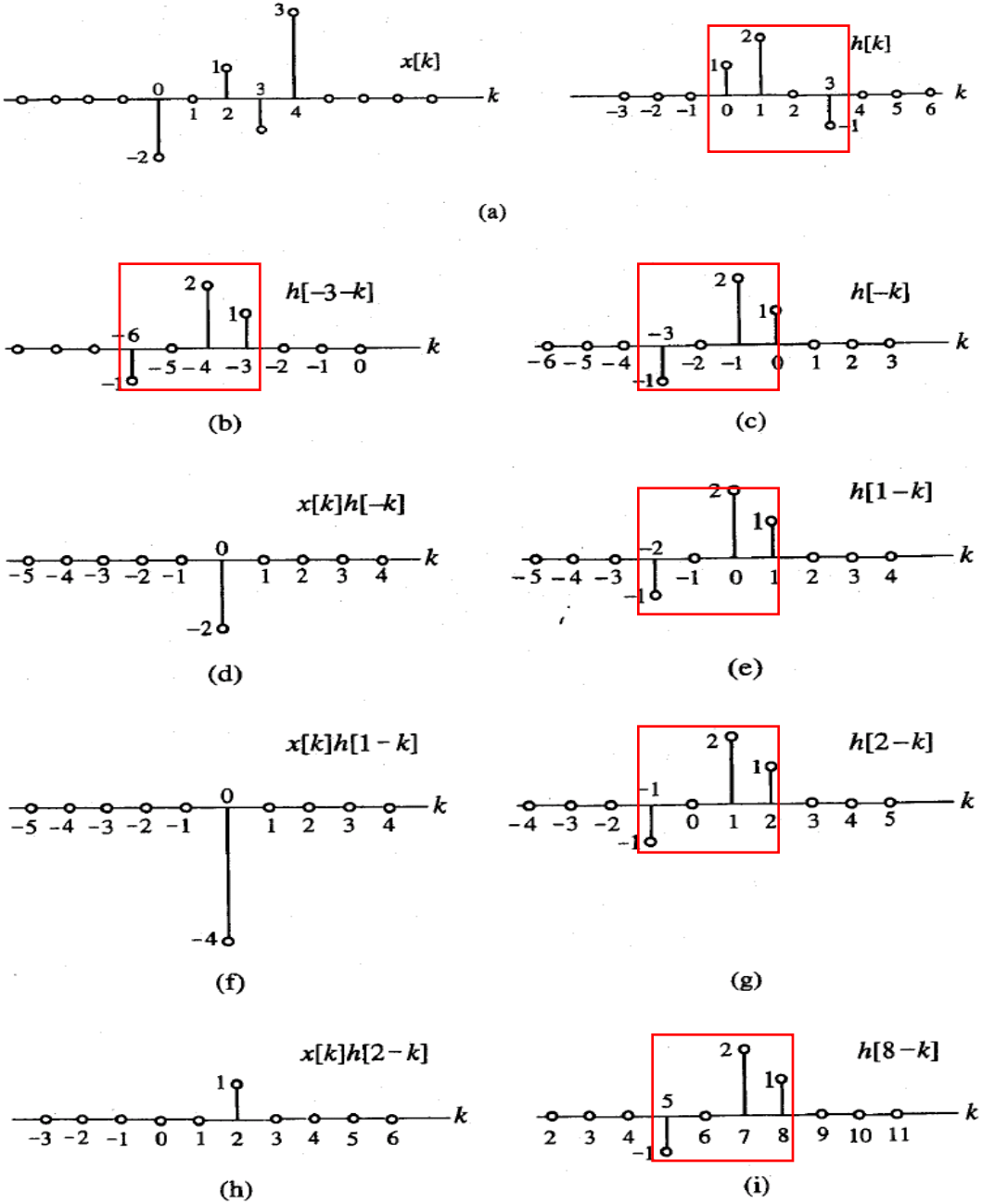


Figure 2.30: Illustration of the convolution process.

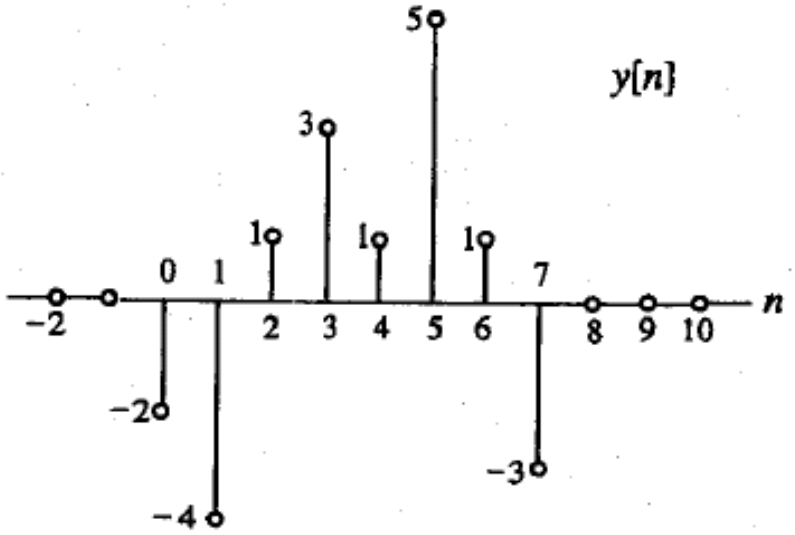


Figure 2.31: Sequence generated by the convolution.

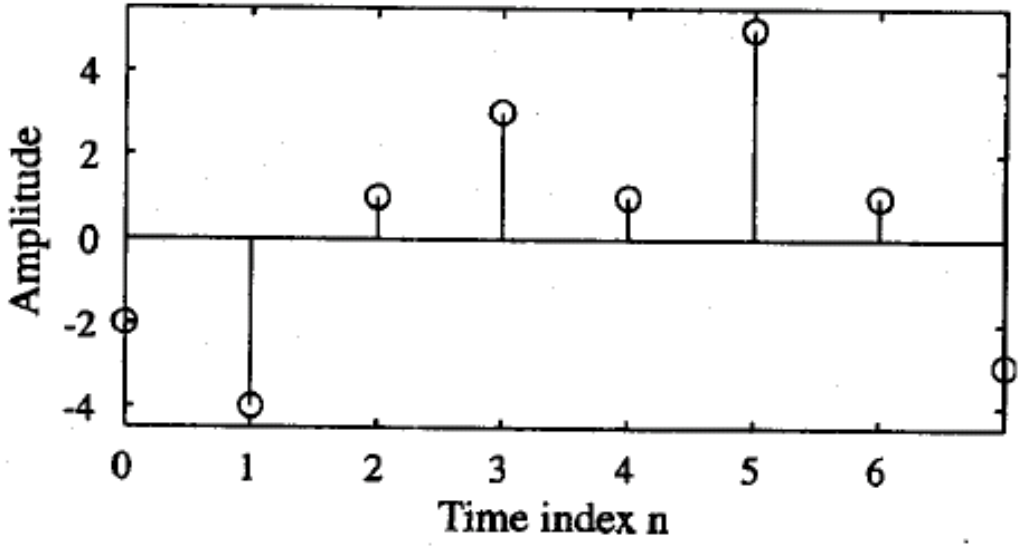


Figure 2.32: Sequence generated by convolution using MATLAB.

```
% Program 2_6
% Illustration of Convolution
%
a = input('Type in the first sequence = ');
b = input('Type in the second sequence = ');
c = conv(a, b);
M = length(c)-1;
n = 0:1:M;
disp('output sequence =');disp(c)
stem(n,c)
xlabel('Time index n'); ylabel('Amplitude');
```

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

Type in the first sequence = [-2 0 1 -1 3]

Type in the second sequence = [1 2 0 -1]

output sequence =

2.5.2 Tabular Method of Convolution Sum Computation

The calculation of the convolution sum of two finite-length sequences can be carried out simply either by using a tabular method similar to that used in conventional multiplication of two numbers [Pie96] or using a method based on the multiplication of two polynomials. These two method are simpler to carry out and easy to remember than the graphical method outlined in Section 2.5.1. We illustrate the first method in this section. The polynomial multiplication method of computing the convolution sum is described in Section 6.6.

Without any loss of generality we consider the evaluation of the convolution of the sequence $\{g[n]\}$, $0 \leq n \leq 3$, with the sequence $\{h[n]\}$, $0 \leq n \leq 2$ generating the sequence

$$y[n] = g[n] \circledast h[n], \qquad 0 \leq n \leq 5.$$

The samples of these two sequences are then multiplied using the conventional multiplication method but without any carry operations on the columns. First, each sample of $\{g[n]\}$ is multiplied with $h[0]$ and the samples of the product sequence are placed in a row beginning at time index $n = 0$. Next, each sample of $\{g[n]\}$ is multiplied with $h[1]$ and the samples of the product sequence are placed in a second row beginning at time index $n = 1$. Finally, each sample of $\{g[n]\}$ is multiplied with $h[2]$ and the samples of the product sequence are placed in a third row beginning at time index $n = 2$. The process is indicated below.

n :	0	1	2	3	4	5
$g[n]$:	$g[0]$	$g[1]$	$g[2]$	$g[3]$		
$h[n]$:	$h[0]$	$h[1]$	$h[2]$	-		
	$g[0]h[0]$	$g[1]h[0]$	$g[2]h[0]$	$g[3]h[0]$		
	-	$g[0]h[1]$	$g[1]h[1]$	$g[2]h[1]$	$g[3]h[1]$	
	-	-	$g[0]h[2]$	$g[1]h[2]$	$g[2]h[2]$	$g[3]h[2]$
$y[n]$:	$y[0]$	$y[1]$	$y[2]$	$y[3]$	$y[4]$	$y[5]$

It should be noted that each line in the above table corresponds to a delayed, weighted impulse response. The samples of the sequence $\{y[n]\}$ generated by the convolution sum are obtained by adding the three entries in the column above each sample and are given by

$$y[0] = g[0]h[0],$$
$$y[1] = g[1]h[0] + g[0]h[1],$$
$$y[2] = g[2]h[0] + g[1]h[1] + g[0]h[2],$$
$$y[3] = g[3]h[0] + g[2]h[1] + g[1]h[2],$$
$$y[4] = g[3]h[1] + g[2]h[2],$$
$$y[5] = g[3]h[2].$$

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

EXAMPLE 2.29 Convolution of Two One-Sided Sequences Using the Tabular Method

We develop the convolution sum of the two sequences $\{x[n]\}$ and $\{h[n]\}$ of Example 2.26 using the above method. The process is illustrated below.

n:	0	1	2	3	4	5	6	7
$x[n]$:	−2	0	1	−1	3			
$h[n]$:	1	2	0	−1	-			
	−2	0	1	−1	3			
	-	−4	0	2	−2	6		
	-	-	0	0	0	0	0	-
	-	-	-	2	0	−1	1	−3
$y[n]$:	−2	−4	1	3	1	5	1	−3

Thus, the convolution sum of the two sequences $x[n]$ and $h[n]$ yields

$$\{y[n]\} = \{-2 \quad -4 \quad 1 \quad 3 \quad 1 \quad 5 \quad 1 \quad -3\}, \qquad 0 \le n \le 7$$

which is seen to be identical to that derived in Example 2.26.

The tabular method can also be employed to evaluate the convolution sum of two finite-length two-sided sequences [Pie96]. In this case, a decimal point is first placed to the right of the sample with the time index $n = 0$. Next, the convolution is computed ignoring the locations of the decimal point. Finally, the decimal point is inserted according to the rules of conventional multiplication. The sample immediately to the left of the decimal point is then located at the time index $n = 0$.

EXAMPLE 2.30 Convolution of Two-Sided Sequences Using the Tabular Method

We determine the convolution sum $y[n]$ of the two sequences:

$$\{g[n]\} = \{3 \quad \underset{\uparrow}{-2} \quad 4\}, \qquad \{h[n]\} = \{4 \quad \underset{\uparrow}{2} \quad -1\}.$$

The convolution sum of the above two sequences are next evaluated using the tabular method as shown below, where the samples at time index $n = 0$ are indicated by the solid circles placed immediately to their right.

$g[n]$:			3	$-2_\bullet$	4
$h[n]$:			$4_\bullet$	2	-1
			-3	2	-4
		6	-4	8	-
	12	-8	16	-	-
$y[n]$:	12	$-2_\bullet$	9	10	-4

Hence, we have

$$\{y[n]\} = \{12 \quad \underset{\uparrow}{-2} \quad 9 \quad 10 \quad -4\}.$$

2.5.3 Stability Condition in Terms of the Impulse Response

Recall from Section 2.4.2 that a discrete-time system is defined to be stable, or precisely, bounded-input, bounded-output (BIBO) stable, if the output sequence $\{y[n]\}$ of the system remains bounded for all bounded input sequences $\{x[n]\}$. We now develop the stability condition for an LTI discrete-time system. We show that an LTI digital filter is BIBO stable if and only if its impulse response sequence $\{h[n]\}$ is absolutely summable, that is,

$$S = \sum_{n=-\infty}^{\infty} |h[n]| < \infty. \tag{2.78}$$

We prove the above statement for a real impulse response $\{h[n]\}$. The extension of the proof for a complex impulse response sequence is left as an exercise (Problem 2.74). Now, if the input sequence $\{x[n]\}$ is bounded, that is, $|x[n]| \leq B_x < \infty$, then the output amplitude at time instant n, from Eq. (2.73b), is

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

$$\leq B_x \sum_{k=-\infty}^{\infty} |h[k]| = B_x S < \infty. \tag{2.79}$$

Thus, $S < \infty$ implies $|y[n]| \leq B_y < \infty$, indicating that the sequence $\{y[n]\}$ is also bounded. To prove the converse, assume that the sequence $\{y[n]\}$ is bounded, that is, $|y[n]| \leq B_y$. Now, consider the

EXAMPLE 2.31 Stability Condition of a Causal First-Order LTI Discrete-Time System

Consider a causal LTI discrete-time system with an impulse response given by

$$h_1[n] = \alpha^n \mu[n].$$

For this system,

$$S = \sum_{n=-\infty}^{\infty} |\alpha^n \mu[n]|$$

$$= \sum_{n=0}^{\infty} |\alpha|^n = \frac{1}{1-|\alpha|}, \quad \text{for } |\alpha| < 1.$$

Therefore, $S < \infty$ if $|\alpha| < 1$ for which the above system is BIBO stable. On the other hand, if $|\alpha| \geq 1$, the infinite series $\sum_{n=0}^{\infty} |\alpha|^n$ does not converge, and as a result, the above causal system is not BIBO stable.

2.5.4 Causality Condition in Terms of the Impulse Response

$$h[k] = 0 \qquad \text{for } k < 0. \tag{2.85}$$

As a result, an LTI discrete-time system is *causal* if and only if its impulse response sequence $\{h[n]\}$ is a causal sequence satisfying the condition of Eq. (2.85).

2.6 Simple Interconnection Schemes

Two widely used schemes for developing complex LTI discrete-time systems from simple LTI discrete-time system sections are described next.

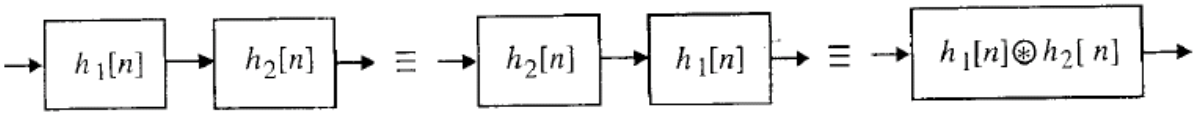


Figure 2.33: The cascade connection.

2.6.1 Cascade Connection

Figure 2.33 shows the *cascade connection* of two LTI discrete-time systems in which the output of one system is connected to the input of the second system. The overall impulse response $h[n]$ of the cascade of the two systems with impulse responses $h_1[n]$ and $h_2[n]$, respectively, is given by their linear convolution, that is,

$$h[n] = h_1[n] \circledast h_2[n]. \tag{2.86}$$

Hence, the cascaded system is equivalent to a system with an impulse response $h_1[n] \circledast h_2[n]$. If there are more than two LTI systems, the impulse response of the overall cascade is given by the linear convolution of the impulse responses of the individual systems. 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 stable systems is stable. Likewise, the cascade connection of passive (lossless) systems is passive (lossless).

An application of the cascade connection scheme is in the development of an *inverse system.* If the two LTI systems in the cascade connection of Figure 2.33 are such that

$$h_1[n] \circledast h_2[n] = \delta[n], \tag{2.87}$$

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. This is accomplished by designing an inverse system if the impulse response of the channel is known.

EXAMPLE 2.34 Inverse of a Discrete-Time Accumulator

From Example 2.22, the impulse response of the discrete-time accumulator is the unit step sequence $\mu[n]$. Therefore, from Eq. (2.87), the inverse system must satisfy the condition

$$\mu[n] \circledast h_2[n] = \delta[n]. \tag{2.88}$$

It follows from Eq. (2.88) that $h_2[n] = 0$ for $n < 0$ and

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

As a result,

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

Thus, the impulse response of the inverse system is given by

$$h_2[n] = \delta[n] - \delta[n-1],$$

which is called a *backward difference system.*

2.6.2 Parallel Connection

The connection scheme of Figure 2.34 is called the *parallel* connection, and here the outputs of the two LTI discrete-time systems are added to form the new output while the same input is fed to both systems. The impulse response of the overall system is thus given by

$$h[n] = h_1[n] + h_2[n]. \tag{2.89}$$

Likewise, if more than two LTI discrete-time systems are in parallel, the impulse response of the overall system is given by the sum of the impulse responses of the individual systems.

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

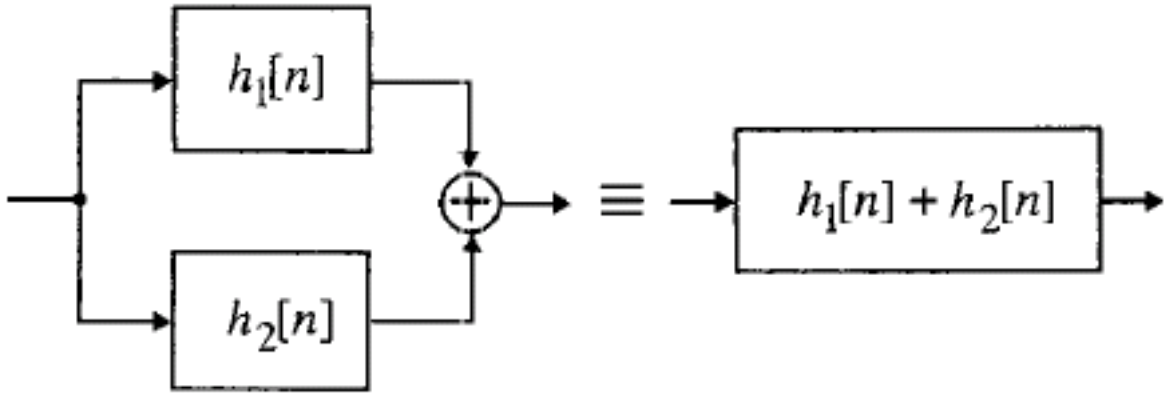
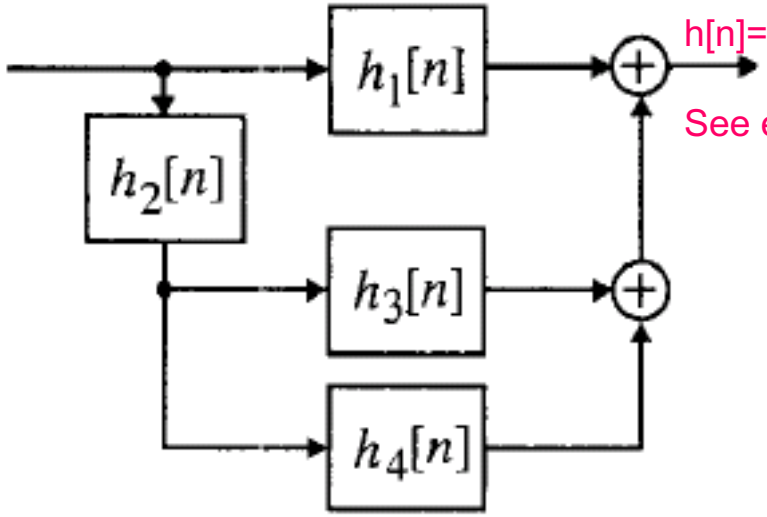


Figure 2.34: The parallel connection.



h[n]=h1[n]+h2[n]*h3[n]+h2[n]*h4[n]

See example 2.35

Figure 2.35: The discrete-time system of Example 2.35.

Hybrid: cascade and parallel connections of several systems

Ex. 2.35

$$h_1 = \delta[n] + \frac{1}{2}\delta[n-1]$$

$$h_2 = \frac{1}{2}\delta[n] - \frac{1}{4}\delta[n-1]$$

$$h_3 = 2\,\delta[n]$$

$$h_4 = -2\left(\frac{1}{2}\right)^n \mu[n]$$

the overall IR

$$h[n] = h_1 + h_2 \circledast h_3 + h_2 \circledast h_4$$

$$h_2 \circledast h_3 = \delta[n] - \frac{1}{2}\delta[n-1]$$

$$h_2 \circledast h_4 = \left(\frac{1}{2}\delta[n] - \frac{1}{4}\delta[n-1]\right) \circledast -2\left(\frac{1}{2}\right)^n \mu[n]$$

$$= -\left(\frac{1}{2}\right)^n \mu[n] + \frac{1}{2}\left(\frac{1}{2}\right)^{n-1} \mu[n-1] = -\delta[n]$$

$$\therefore\ h[n] = \delta[n] + \frac{1}{2}\delta[n-1] + \delta[n] - \frac{1}{2}\delta[n-1] - \delta[n]$$

Common Impulse Responses

EQUATION 7-1
The delta function is the identity for convolution. Any signal convolved with a delta function is left unchanged.

$$x[n] * \delta[n] = x[n]$$

EQUATION 7-2
A system that amplifies or attenuates has a scaled delta function for an impulse response. In this equation, k determines the amplification or attenuation.

$$x[n] * k\delta[n] = kx[n]$$

EQUATION 7-3
A relative shift between the input and output signals corresponds to an impulse response that is a shifted delta function. The variable, s, determines the amount of shift in this equation.

$$x[n] * \delta[n+s] = x[n+s]$$