SRI CHANDRASEKHARENDRA SARASWATHI VISWA MAHAVIDYALAYA

(University established under section 3 of UGC Act 1956) (Accredited
With 'A' Grade by NAAC)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

COURSE MATERIAL

DIGITAL SIGNAL PROCESSING

V SEMESTER – B.E (ECE)

during the academic year 2021-2022

Prepared By:
Dr.P.VENKATESAN,
Associate Professor, ECE
SCSVMV, Kanchipuram

DIGITAL SIGNAL PROCESSING

PRE-REQUISITE:

Signal & Systems, Digital System Design, Engineering Mathematics

AIM & OBJECTIVES:

- To learn discrete Fourier transform, properties of DFT and its application to linear filtering.
- To understand the characteristics of digital filters, design digital FIR filters and apply these filters to filter undesirable signals in various frequency bands.
- To design digital IIR filters and apply these filters to filter undesirable signals in various frequency bands.
- To understand the effects of finite precision representation on digital filters.
- To understand the fundamental concepts of multi-rate signal processing and its applications.

UNIT I DISCRETE FOURIER TRANSFORM

Review of discrete-time signals and systems – Discrete Fourier Transform (DFT) and its properties, Circular convolution, Linear filtering using DFT, Filtering long data sequences - overlap-save methods - Overlap-add, Fast Fourier Transform (FFT) algorithms – Fast computation of DFT –Radix- 2 decimation in time FFT - Decimation in frequency FFT – Linear filtering using FFT.

UNIT II DESIGN OF FINITE IMPULSE RESPONSE FILTERS

Structures for FIR systems – Transversal and Linear phase structures, Design of FIR filters – Symmetric and Anti-symmetric FIR filters, Design of linear phase FIR filters using Windows (Rectangular, Hamming and Hanning windows) and Frequency sampling methods

UNIT III DESIGN OF INFINITE IMPULSE RESPONSE FILTERS

Structures for IIR systems – direct, cascade, parallel forms, Comparison of FIR and IIR, Analog filters – Butterworth filters - Chebyshev type – I filters (upto 3rd order), Analog transformation of prototype LPF to BPF/BSF/HPF, Transformation of analog filters into equivalent digital filters using Impulse invariant method and Bilinear Z-transform method.

UNIT IV FINITE WORD LENGTH EFFECTS

Representation of fixed and floating point numbers, ADC quantization -truncation and rounding - quantization noise, Coefficient quantization Error – Product quantization error – Overflow error - Round-off noise power, Limit cycle oscillation due to product round-off error- Limit cycle oscillation due to overflow in digital filters – Principle of scaling.

UNIT V MULTI-RATE SIGNAL PROCESSING

Introduction to multi-rate signal processing – Decimation – Interpolation- Sampling rate conversion by a rational factor - Polyphase decomposition of FIR filter – Multistage implementation of sampling rate conversion – Design of narrow band filters–Applications of multi-rate signal processing

UNIT -1
INTRODUCTION
REVIEW OF DISCRETE TIME SIGNALS AND SYSTEMS

Anything that carries some information can be called as signals. Some examples are ECG, EEG, ac power, seismic, speech, interest rates of a bank, unemployment rate of a country, temperature, pressure etc.

A **signal** is also defined as any physical quantity that varies with one or more independent variables.

A **discrete time signal** is the one which is not defined at intervals between two successive samples of a signal. It is represented as graphical, functional, tabular representation and sequence.

Some of the elementary discrete time signals are unit step, unit impulse, unit ramp, exponential and sinusoidal signals (as you read in signals and systems).

Classification of discrete time signals
Energy and Power signals

$$E \equiv \sum_{n=-\infty}^{\infty} |x(n)|^2$$

If the value of E is finite, then the signal x(n) is called energy signal.

$$P = \lim_{N \to \infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x(n)|^2$$

If the value of the P is finite, then the signal x(n) is called Power signal.

Periodic and Non periodic signals
A discrete time signal is said to be periodic if and only if it satisfies the condition X (N+n) =x (n), otherwise non periodic

Symmetric (even) and Anti-symmetric (odd) signals
The signal is said to be even if x(-n)=x(n)

The signal is said to be odd if x(-n)= - x(n)

Causal and non causal signal
The signal is said to be causal if its value is zero for negative values of ‘n’.

Some of the **operations** on discrete time signals are shifting, time reversal, time scaling, signal multiplier, scalar multiplication and signal addition or multiplication.

Discrete time systems
A discrete time signal is a device or algorithm that operates on discrete time signals and produces another discrete time output.

Classification of discrete time systems
Static and dynamic systems
A system is said to be static if its output at present time depend on the input at present time only.

Causal and non causal systems
A system is said to be causal if the response of the system depends on present and past values of the input but not on the future inputs.

Linear and non linear systems

A system is said to be linear if the response of the system to the weighted sum of inputs should be equal to the corresponding weighted sum of outputs of the systems. This principle is called superposition principle.

Time invariant and time variant systems

A system is said to be time invariant if the characteristics of the systems do not change with time.

Stable and unstable systems

A system is said to be stable if bounded input produces bounded output only.

TIME DOMAIN ANALYSIS OF DISCRETE TIME SIGNALS AND SYSTEMS

Representation of an arbitrary sequence

Any signal x(n) can be represented as weighted sum of impulses as given below

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k)$$

The response of the system for unit sample input is called impulse response of the system h(n)

$$\begin{aligned} y(n) &= \mathcal{T}[x(n)] = \mathcal{T}\left[\sum_{k=-\infty}^{\infty} x(k)\delta(n-k)\right] \\ &= \sum_{k=-\infty}^{\infty} x(k)\mathcal{T}[\delta(n-k)] \\ &= \sum_{k=-\infty}^{\infty} x(k)h(n.k) \end{aligned}$$

By time invariant property, we have

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

The above equation is called **convolution sum**.

Some of the properties of convolution are commutative law, associative law and distributive law.

Correlation of two sequences

It is basically used to compare two signals. It is the measure of similarity between two signals. Some of the applications are communication systems, radar, sonar etc.

The cross correlation of two sequences x(n) and y(n) is given by

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x(n)y(n-l) \qquad l = 0, \pm 1, \pm 2, \ldots$$

One of the important properties of cross correlation is given by

$$r_{xy}(l) = r_{yx}(-l)$$

The auto correlation of the signal x(n) is given by

$$r_{xx}(l) = \sum_{n=-\infty}^{\infty} x(n)x(n-l)$$

Linear time invariant systems characterized by constant coefficient difference equation
The response of the first order difference equation is given by

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

The first part contain initial condition y(-1) of the system, the second part contains input x(n) of the system.
The response of the system when it is in relaxed state at n=0 or
y(-1)=0 is called **zero state response** of the system or **forced response**.

$$y_{zs}(n) = \sum_{k=0}^{n} a^k x(n-k) \qquad n \geq 0$$

The output of the system at zero input condition x(n)=0 is called **zero input response** of the system or **natural response**.

The impulse response of the system is given by zero state response of the system

$$\begin{aligned} y_{zs}(n) &= \sum_{k=0}^{n} a^k \delta(n-k) \\ &= a^n \qquad n \geq 0 \end{aligned}$$

The total response of the system is equal to sum of natural response and forced responses.

FREQUENCY DOMAIN ANALYSIS OF DISCRETE TIME SIGNALS AND SYSTEMS

A s we have observed from the discussion o f Section 4.1, the Fourier series representation o f a continuous-time periodic signal can consist of an infinite number of frequency components, where the frequency spacing between two successive harmonically related frequencies is 1 */ T p,* and where *Tp* is the fundamental period.

Since the frequency range for continuous-time signals extends infinity on both sides it is possible to have signals that contain an infinite number of frequency components.

In contrast, the frequency range for discrete-time signals is unique over the interval. A discrete-time signal of fundamental period *N* can consist of frequency components separated by 2*n / N* radians.

Consequently, the Fourier series representation o f the discrete-time periodic signal will contain at most *N* frequency components. This is the basic difference between the Fourier series representations for continuous-time and discrete-time periodic signals.

4.2.1 The Fourier Series for Discrete-Time Periodic Signals

Suppose that we are given a periodic sequence $x(n)$ with period N, that is, $x(n) = x(n+N)$ for all n. The Fourier series representation for $x(n)$ consists of N harmonically related exponential functions

$$e^{j2\pi kn/N} \qquad k = 0, 1, \ldots, N-1$$

and is expressed as

$$x(n) = \sum_{k=0}^{N-1} c_k e^{j2\pi kn/N} \tag{4.2.1}$$

where the $\{c_k\}$ are the coefficients in the series representation.

To derive the expression for the Fourier coefficients, we use the following formula:

$$\sum_{n=0}^{N-1} e^{j2\pi kn/N} = \begin{cases} N, & k = 0, \pm N, \pm 2N, \ldots \\ 0, & \text{otherwise} \end{cases} \tag{4.2.2}$$

Note the similarity of (4.2.2) with the continuous-time counterpart in (4.1.3). The proof of (4.2.2) follows immediately from the application of the geometric summation formula

$$\sum_{n=0}^{N-1} a^n = \begin{cases} N, & a = 1 \\ \dfrac{1-a^N}{1-a}, & a \neq 1 \end{cases} \tag{4.2.3}$$

The expression for the Fourier coefficients c_k can be obtained by multiplying both sides of (4.2.1) by the exponential $e^{-j2\pi ln/N}$ and summing the product from $n = 0$ to $n = N-1$. Thus

$$\sum_{n=0}^{N-1} x(n) e^{-j2\pi ln/N} = \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} c_k e^{j2\pi(k-l)n/N} \tag{4.2.4}$$

If we perform the summation over n first, in the right-hand side of (4.2.4), we obtain

$$\sum_{n=0}^{N-1} e^{j2\pi(k-l)n/N} = \begin{cases} N, & k-l = 0, \pm N, \pm 2N, \ldots \\ 0, & \text{otherwise} \end{cases} \tag{4.2.5}$$

where we have made use of (4.2.2). Therefore, the right-hand side of (4.2.4) reduces to Nc_l and hence

$$c_l = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j2\pi ln/N} \qquad l = 0, 1, \ldots, N-1 \tag{4.2.6}$$

Thus we have the desired expression for the Fourier coefficients in terms of the signal $x(n)$.

4.2.3 The Fourier Transform of Discrete-Time Aperiodic Signals

Just as in the case of continuous-time aperiodic energy signals, the frequency analysis of discrete-time aperiodic finite-energy signals involves a Fourier transform of the time-domain signal. Consequently, the development in this section parallels to a large extent, that given in Section 4.1.3.

The Fourier transform of a finite-energy discrete-time signal $x(n)$ is defined as

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} \tag{4.2.23}$$

Physically, $X(\omega)$ represents the frequency content of the signal $x(n)$. In other words, $X(\omega)$ is a decomposition of $x(n)$ into its frequency components.

We observe two basic differences between the Fourier transform of a discrete-time finite-energy signal and the Fourier transform of a finite-energy analog signal. First, for continuous-time signals, the Fourier transform, and hence the spectrum of the signal, have a frequency range of $(-\infty, \infty)$. In contrast, the frequency range for a discrete-time signal is unique over the frequency interval of $(-\pi, \pi)$ or, equivalently, $(0. 2\pi)$. This property is reflected in the Fourier transform of the signal. Indeed. $X(\omega)$ is periodic with period 2π. that is,

$$\begin{aligned} X(\omega + 2\pi k) &= \sum_{n=-\infty}^{\infty} x(n)e^{-j(\omega+2\pi k)n} \\ &= \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}e^{-j2\pi kn} \\ &= \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} = X(\omega) \end{aligned} \tag{4.2.24}$$

Hence $X(\omega)$ is periodic with period 2π. But this property is just a consequence of the fact that the frequency range for any discrete-time signal is limited to $(-\pi, \pi)$ or $(0, 2\pi)$, and any frequency outside this interval is equivalent to a frequency within the interval.

The second basic difference is also a consequence of the discrete-time nature of the signal. Since the signal is discrete in time, the Fourier transform of the signal involves a summation of terms instead of an integral, as in the case of continuous-time signals.

Since $X(\omega)$ is a periodic function of the frequency variable ω, it has a Fourier series expansion, provided that the conditions for the existence of the Fourier series, described previously, are satisfied. In fact, from the definition of the Fourier transform $X(\omega)$ of the sequence $x(n)$, given by (4.2.23), we observe that $X(\omega)$ has the form of a Fourier series. The Fourier coefficients in this series expansion are the values of the sequence $x(n)$.

To demonstrate this point, let us evaluate the sequence $x(n)$ from $X(\omega)$. First, we multiply both sides (4.2.23) by $e^{j\omega m}$ and integrate over the interval $(-\pi, \pi)$. Thus we have

$$\int_{-\pi}^{\pi} X(\omega)e^{j\omega m}d\omega = \int_{-\pi}^{\pi}\left[\sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}\right]e^{j\omega m}d\omega \qquad (4.2.25)$$

The integral on the right-hand side of (4.2.25) can be evaluated if we can interchange the order of summation and integration. This interchange can be made if the series

$$X_N(\omega) = \sum_{n=-N}^{N} x(n)e^{-j\omega n}$$

converges uniformly to $X(\omega)$ as $N \to \infty$. Uniform convergence means that, for every ω, $X_N(\omega) \to X(\omega)$, as $N \to \infty$. The convergence of the Fourier transform is discussed in more detail in the following section. For the moment, let us assume that the series converges uniformly, so that we can interchange the order of summation and integration in (4.2.25). Then

$$\int_{-\pi}^{\pi} e^{j\omega(m-n)}d\omega = \begin{cases} 2\pi, & m = n \\ 0, & m \neq n \end{cases}$$

Consequently,

$$\sum_{n=-\infty}^{\infty} x(n)\int_{-\pi}^{\pi} e^{j\omega(m-n)}d\omega = \begin{cases} 2\pi x(m), & m = n \\ 0, & m \neq n \end{cases} \qquad (4.2.26)$$

By combining (4.2.25) and (4.2.26), we obtain the desired result that

$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)e^{j\omega n}d\omega \qquad (4.2.27)$$

If we compare the integral in (4.2.27) with (4.1.9), we note that this is just the expression for the Fourier series coefficient for a function that is periodic with period 2π. The only difference between (4.1.9) and (4.2.27) is the sign on the exponent in the integrand, which is a consequence of our definition of the Fourier transform as given by (4.2.23). Therefore, the Fourier transform of the sequence $x(n)$, defined by (4.2.23), has the form of a Fourier series expansion.

FREQUENCY DOMAIN SAMPLING: THE DISCRETE FOURIER TRANSFORM

Before we introduce the DFT, we consider the sampling of the Fourier transform of an aperiodic discrete-time sequence. Thus, we establish the relationship between the sampled Fourier transform and the DFT.

5.1.1 Frequency-Domain Sampling and Reconstruction of Discrete-Time Signals

We recall that aperiodic finite-energy signals have continuous spectra. Let us consider such an aperiodic discrete-time signal $x(n)$ with Fourier transform

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n} \tag{5.1.1}$$

Suppose that we sample $X(\omega)$ periodically in frequency at a spacing of $\delta\omega$ radians between successive samples. Since $X(\omega)$ is periodic with period 2π, only samples in the fundamental frequency range are necessary. For convenience, we take N equidistant samples in the interval $0 \leq \omega < 2\pi$ with spacing $\delta\omega = 2\pi/N$, as shown in Fig. 5.1. First, we consider the selection of N, the number of samples in the frequency domain.

If we evaluate (5.1.1) at $\omega = 2\pi k/N$, we obtain

$$X\left(\frac{2\pi}{N}k\right) = \sum_{n=-\infty}^{\infty} x(n)e^{-j2\pi kn/N} \qquad k = 0, 1, \ldots, N-1 \tag{5.1.2}$$

The summation in (5.1.2) can be subdivided into an infinite number of summations, where each sum contains N terms. Thus

$$\begin{aligned} X\left(\frac{2\pi}{N}k\right) = & \quad \cdots + \sum_{n=-N}^{-1} x(n)e^{-j2\pi kn/N} + \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N} \\ & + \sum_{n=N}^{2N-1} x(n)e^{-j2\pi kn/N} + \cdots \\ = & \sum_{l=-\infty}^{\infty} \sum_{n=lN}^{lN+N-1} x(n)e^{-j2\pi kn/N} \end{aligned}$$

If we change the index in the inner summation from n to $n - lN$ and interchange the order of the summation, we obtain the result

$$X\left(\frac{2\pi}{N}k\right) = \sum_{n=0}^{N-1}\left[\sum_{l=-\infty}^{\infty} x(n-lN)\right]e^{-j2\pi kn/N} \tag{5.1.3}$$

for $k = 0, 1, 2, \ldots, N-1$.

The signal

$$x_p(n) = \sum_{l=-\infty}^{\infty} x(n-lN) \tag{5.1.4}$$

obtained by the periodic repetition of $x(n)$ every N samples, is clearly periodic with fundamental period N. Consequently, it can be expanded in a Fourier series as

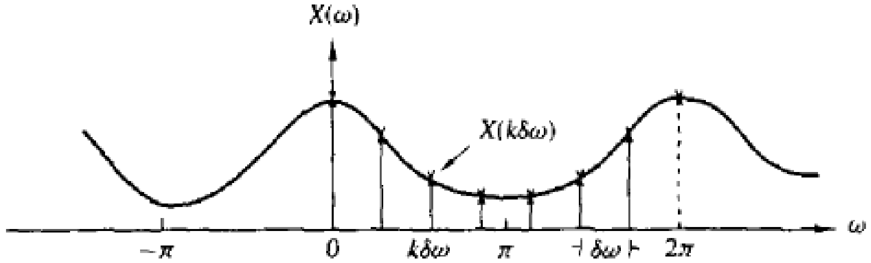


Figure 5.1 Frequency-domain sampling of the Fourier transform.

$$x_p(n) = \sum_{k=0}^{N-1} c_k e^{j2\pi kn/N} \qquad n = 0, 1, \ldots, N-1 \tag{5.1.5}$$

with Fourier coefficients

$$c_k = \frac{1}{N}\sum_{n=0}^{N-1} x_p(n)e^{-j2\pi kn/N} \qquad k = 0, 1, \ldots, N-1 \tag{5.1.6}$$

Upon comparing (5.1.3) with (5.1.6), we conclude that

$$c_k = \frac{1}{N}X\left(\frac{2\pi}{N}k\right) \qquad k = 0, 1, \ldots, N-1 \tag{5.1.7}$$

Therefore,

$$x_p(n) = \frac{1}{N}\sum_{k=0}^{N-1} X\left(\frac{2\pi}{N}k\right)e^{j2\pi kn/N} \qquad n = 0, 1, \ldots, N-1 \tag{5.1.8}$$

5.1.2 The Discrete Fourier Transform (DFT)

The development in the preceding section is concerned with the frequency-domain sampling of an aperiodic finite-energy sequence $x(n)$. In general, the equally spaced frequency samples $X(2\pi k/N)$, $k = 0, 1, \ldots, N-1$, do not uniquely represent the original sequence $x(n)$ when $x(n)$ has infinite duration. Instead, the frequency samples $X(2\pi k/N)$, $k = 0, 1, \ldots, N-1$, correspond to a periodic sequence $x_p(n)$ of period N, where $x_p(n)$ is an aliased version of $x(n)$, as indicated by the relation in (5.1.4), that is,

$$x_p(n) = \sum_{l=-\infty}^{\infty} x(n - lN) \tag{5.1.15}$$

When the sequence $x(n)$ has a finite duration of length $L \leq N$, then $x_p(n)$ is simply a periodic repetition of $x(n)$, where $x_p(n)$ over a single period is given as

$$x_p(n) = \begin{cases} x(n), & 0 \leq n \leq L-1 \\ 0, & L \leq n \leq N-1 \end{cases} \tag{5.1.16}$$

Consequently, the frequency samples $X(2\pi k/N)$, $k = 0, 1, \ldots, N-1$, uniquely represent the finite-duration sequence $x(n)$. Since $x(n) \equiv x_p(n)$ over a single period (padded by $N - L$ zeros), the original finite-duration sequence $x(n)$ can be obtained from the frequency samples $\{X(2\pi k/N\}$ by means of the formula (5.1.8).

It is important to note that *zero padding* does not provide any additional information about the spectrum $X(\omega)$ of the sequence $\{x(n)\}$. The L equidistant samples of $X(\omega)$ are sufficient to reconstruct $X(\omega)$ using the reconstruction formula (5.1.13). However, padding the sequence $\{x(n)\}$ with $N - L$ zeros and computing an N-point DFT results in a "better display" of the Fourier transform $X(\omega)$.

In summary, a finite-duration sequence $x(n)$ of length L [i.e., $x(n) = 0$ for $n < 0$ and $n \geq L$] has a Fourier transform

$$X(\omega) = \sum_{n=0}^{L-1} x(n)e^{-j\omega n} \qquad 0 \leq \omega \leq 2\pi \tag{5.1.17}$$

where the upper and lower indices in the summation reflect the fact that $x(n) = 0$ outside the range $0 \leq n \leq L-1$. When we sample $X(\omega)$ at equally spaced frequencies $\omega_k = 2\pi k/N$, $k = 0, 1, 2, \ldots, N-1$, where $N \geq L$, the resultant samples are

$$\begin{aligned} X(k) &\equiv X\left(\frac{2\pi k}{N}\right) = \sum_{n=0}^{L-1} x(n)e^{-j2\pi kn/N} \\ X(k) &= \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N} \qquad k = 0, 1, 2, \ldots, N-1 \end{aligned} \tag{5.1.18}$$

where for convenience, the upper index in the sum has been increased from $L-1$ to $N-1$ since $x(n) = 0$ for $n \geq L$.

The relation in (5.1.18) is a formula for transforming a sequence $\{x(n)\}$ of length $L \leq N$ into a sequence of frequency samples $\{X(k)\}$ of length N. Since the frequency samples are obtained by evaluating the Fourier transform $X(\omega)$ at a set of N (equally spaced) discrete frequencies, the relation in (5.1.18) is called the *discrete Fourier transform* (DFT) of $x(n)$. In turn, the relation given by (5.1.10), which allows us to recover the sequence $x(n)$ from the frequency samples

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N} \qquad n = 0, 1, \ldots, N-1 \tag{5.1.19}$$

is called the *inverse DFT* (IDFT). Clearly, when $x(n)$ has length $L < N$, the N-point IDFT yields $x(n) = 0$ for $L \leq n \leq N-1$. To summarize, the formulas for the DFT and IDFT are

DFT

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N} \qquad k = 0, 1, 2, \ldots, N-1 \tag{5.1.18}$$

IDFT

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N} \qquad n = 0, 1, 2, \ldots, N-1 \tag{5.1.19}$$

5.1.3 The DFT as a Linear Transformation

The formulas for the DFT and IDFT given by (5.1.18) and (5.1.19) may be expressed as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad k = 0, 1, \ldots, N-1 \tag{5.1.20}$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} \qquad n = 0, 1, \ldots, N-1 \tag{5.1.21}$$

where, by definition,

$$W_N = e^{-j2\pi/N} \tag{5.1.22}$$

which is an Nth root of unity.

With these definitions, the N-point DFT may be expressed in matrix form as

$$\mathbf{X}_N = \mathbf{W}_N \mathbf{x}_N \tag{5.1.24}$$

where $\mathbf{W}_N$ is the matrix of the linear transformation. We observe that $\mathbf{W}_N$ is a symmetric matrix. If we assume that the inverse of $\mathbf{W}_N$ exists, then (5.1.24) can be inverted by premultiplying both sides by $\mathbf{W}_N^{-1}$. Thus we obtain

$$\mathbf{x}_N = \mathbf{W}_N^{-1} \mathbf{X}_N \tag{5.1.25}$$

Relationship to the Fourier series coefficients of a periodic sequence. A periodic sequence $\{x_p(n)\}$ with fundamental period N can be represented in a Fourier series of the form

$$x_p(n) = \sum_{k=0}^{N-1} c_k e^{j2\pi nk/N} \qquad -\infty < n < \infty \tag{5.1.29}$$

where the Fourier series coefficients are given by the expression

$$c_k = \frac{1}{N} \sum_{n=0}^{N-1} x_p(n) e^{-j2\pi nk/N} \qquad k = 0, 1, \ldots, N-1 \tag{5.1.30}$$

If we compare (5.1.29) and (5.1.30) with (5.1.18) and (5.1.19), we observe that the formula for the Fourier series coefficients has the form of a DFT. In fact, if we define a sequence $x(n) = x_p(n)$, $0 \leq n \leq N-1$, the DFT of this sequence is simply

$$X(k) = Nc_k \tag{5.1.31}$$

Furthermore, (5.1.29) has the form of an IDFT. Thus the N-point DFT provides the exact line spectrum of a periodic sequence with fundamental period N.

Relationship to the Fourier transform of an aperiodic sequence. We have already shown that if $x(n)$ is an aperiodic finite energy sequence with Fourier transform $X(\omega)$, which is sampled at N equally spaced frequencies $\omega_k = 2\pi k/N$, $k = 0, 1, \ldots, N-1$, the spectral components

$$X(k) = X(\omega)|_{\omega=2\pi k/N} = \sum_{n=-\infty}^{\infty} x(n) e^{-j2\pi nk/N} \qquad k = 0, 1, \ldots, N-1 \tag{5.1.32}$$

are the DFT coefficients of the periodic sequence of period N, given by

$$x_p(n) = \sum_{l=-\infty}^{\infty} x(n - lN) \tag{5.1.33}$$

Thus $x_p(n)$ is determined by aliasing $\{x(n)\}$ over the interval $0 \leq n \leq N-1$. The finite-duration sequence

$$\hat{x}(n) = \begin{cases} x_p(n), & 0 \leq n \leq N-1 \\ 0, & \text{otherwise} \end{cases} \tag{5.1.34}$$

bears no resemblance to the original sequence $\{x(n)\}$, unless $x(n)$ is of finite duration and length $L \leq N$, in which case

$$x(n) = \hat{x}(n) \qquad 0 \leq n \leq N-1 \tag{5.1.35}$$

Only in this case will the IDFT of $\{X(k)\}$ yield the original sequence $\{x(n)\}$.

Relationship to the *z*-transform. Let us consider a sequence $x(n)$ having the z-transform

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)z^{-n} \tag{5.1.36}$$

with a ROC that includes the unit circle. If $X(z)$ is sampled at the N equally spaced points on the unit circle $z_k = e^{j2\pi k/N}$, 0, 1, 2, ..., $N-1$, we obtain

$$\begin{aligned} X(k) &\equiv X(z)|_{z=e^{j2\pi nk/N}} \qquad k = 0, 1, \ldots, N-1 \\ &= \sum_{n=-\infty}^{\infty} x(n)e^{-j2\pi nk/N} \end{aligned} \tag{5.1.37}$$

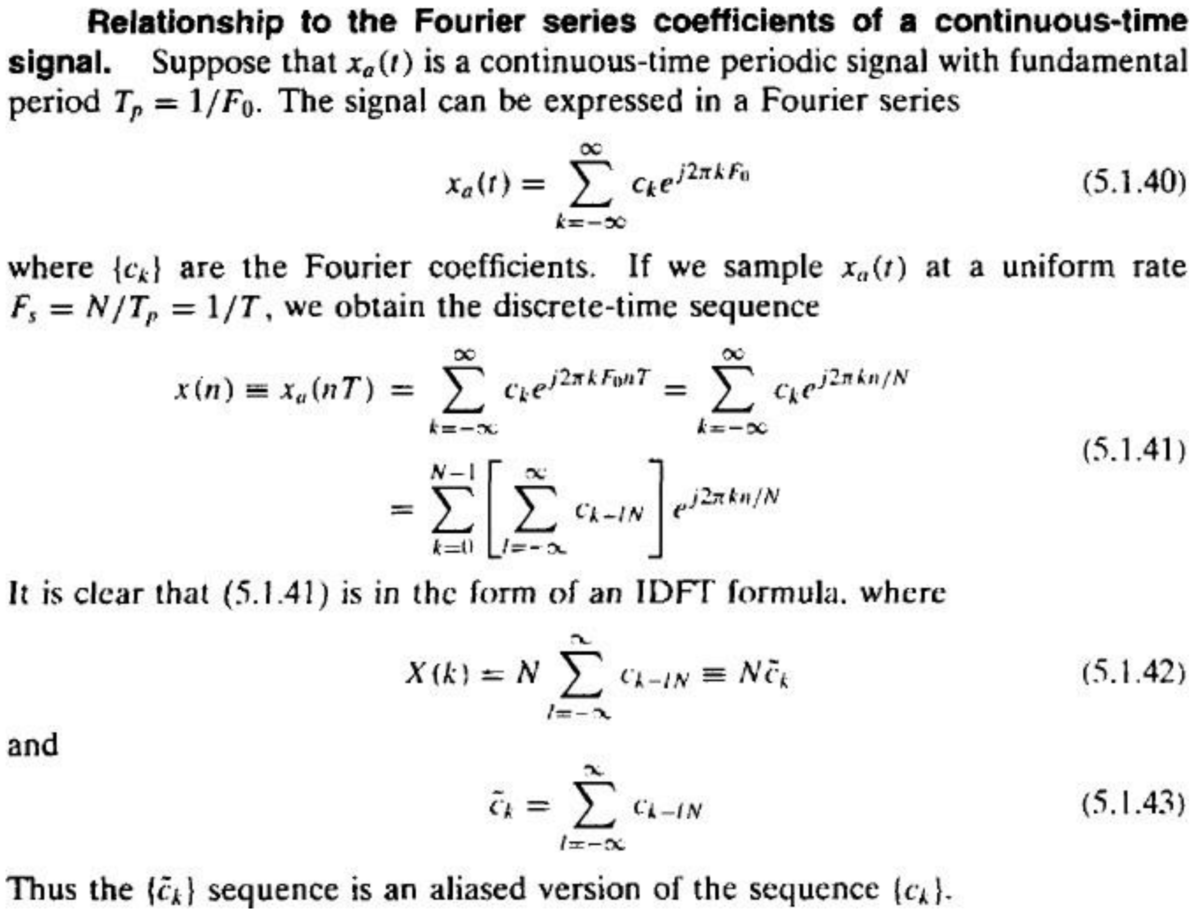
The expression in (5.1.37) is identical to the Fourier transform $X(\omega)$ evaluated at the N equally spaced frequencies $\omega_k = 2\pi k/N$, $k = 0, 1, \ldots, N-1$, which is the topic treated in Section 5.1.1.

If the sequence $x(n)$ has a finite duration of length N or less, the sequence can be recovered from its N-point DFT. Hence its z-transform is uniquely determined by its N-point DFT. Consequently, $X(z)$ can be expressed as a function of the DFT {X(k)} as follows

$$\begin{aligned} X(z) &= \sum_{n=0}^{N-1} x(n)z^{-n} \\ X(z) &= \sum_{n=0}^{N-1}\left[\frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi kn/N}\right]z^{-n} \\ X(z) &= \frac{1}{N}\sum_{k=0}^{N-1} X(k)\sum_{n=0}^{N-1}\left(e^{j2\pi k/N}z^{-1}\right)^n \\ X(z) &= \frac{1-z^{-N}}{N}\sum_{k=0}^{N-1}\frac{X(k)}{1-e^{j2\pi k/N}z^{-1}} \end{aligned} \tag{5.1.38}$$

When evaluated on the unit circle, (5.1.38) yields the Fourier transform of the finite-duration sequence in terms of its DFT, in the form

$$X(\omega) = \frac{1-e^{-j\omega N}}{N}\sum_{k=0}^{N-1}\frac{X(k)}{1-e^{-j(\omega-2\pi k/N)}} \tag{5.1.39}$$



Relationship to the Fourier series coefficients of a continuous-time signal. Suppose that $x_a(t)$ is a continuous-time periodic signal with fundamental period $T_p = 1/F_0$. The signal can be expressed in a Fourier series

$$x_a(t) = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi kF_0} \qquad (5.1.40)$$

where $\{c_k\}$ are the Fourier coefficients. If we sample $x_a(t)$ at a uniform rate $F_s = N/T_p = 1/T$, we obtain the discrete-time sequence

$$\begin{aligned} x(n) \equiv x_a(nT) &= \sum_{k=-\infty}^{\infty} c_k e^{j2\pi kF_0 nT} = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi kn/N} \\ &= \sum_{k=0}^{N-1}\left[\sum_{l=-\infty}^{\infty} c_{k-lN}\right] e^{j2\pi kn/N} \end{aligned} \qquad (5.1.41)$$

It is clear that (5.1.41) is in the form of an IDFT formula, where

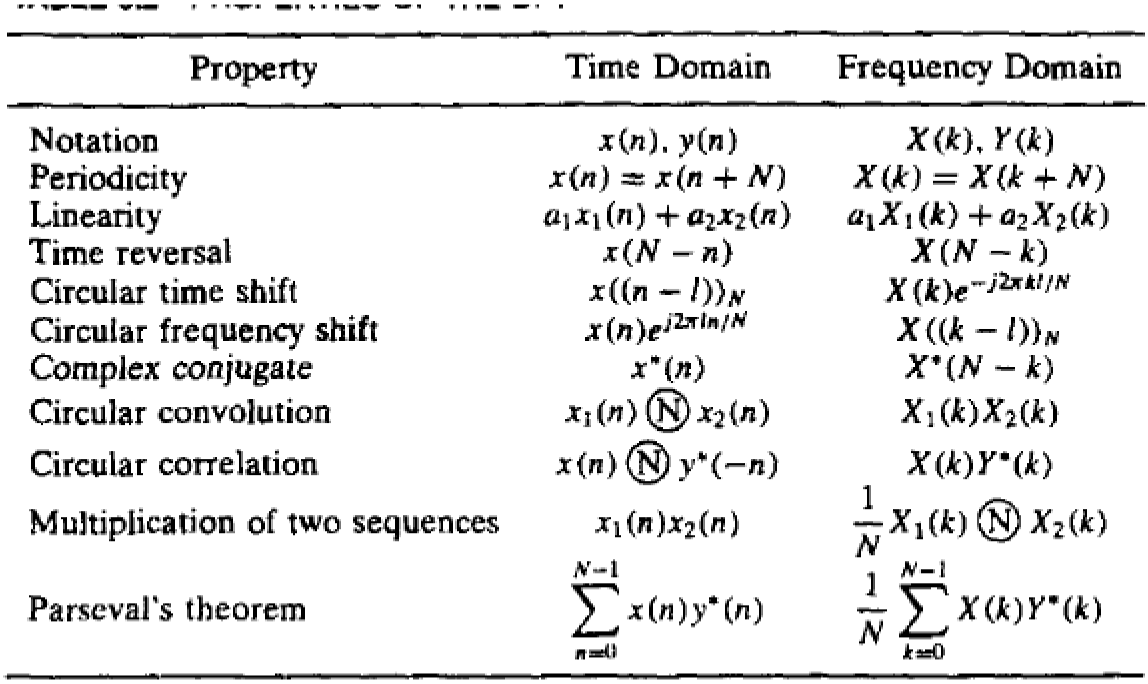
$$X(k) = N \sum_{l=-\infty}^{\infty} c_{k-lN} \equiv N\tilde{c}_k \qquad (5.1.42)$$

and

$$\tilde{c}_k = \sum_{l=-\infty}^{\infty} c_{k-lN} \qquad (5.1.43)$$

Thus the $\{\tilde{c}_k\}$ sequence is an aliased version of the sequence $\{c_k\}$.

PROPERTIES OF DFT:



Property	Time Domain	Frequency Domain
Notation	$x(n), y(n)$	$X(k), Y(k)$
Periodicity	$x(n) = x(n+N)$	$X(k) = X(k+N)$
Linearity	$a_1x_1(n) + a_2x_2(n)$	$a_1X_1(k) + a_2X_2(k)$
Time reversal	$x(N-n)$	$X(N-k)$
Circular time shift	$x((n-l))_N$	$X(k)e^{-j2\pi kl/N}$
Circular frequency shift	$x(n)e^{j2\pi ln/N}$	$X((k-l))_N$
Complex conjugate	$x^*(n)$	$X^*(N-k)$
Circular convolution	$x_1(n) \textcircled{N} x_2(n)$	$X_1(k)X_2(k)$
Circular correlation	$x(n) \textcircled{N} y^*(-n)$	$X(k)Y^*(k)$
Multiplication of two sequences	$x_1(n)x_2(n)$	$\frac{1}{N}X_1(k) \textcircled{N} X_2(k)$
Parseval's theorem	$\sum_{n=0}^{N-1} x(n)y^*(n)$	$\frac{1}{N}\sum_{k=0}^{N-1} X(k)Y^*(k)$

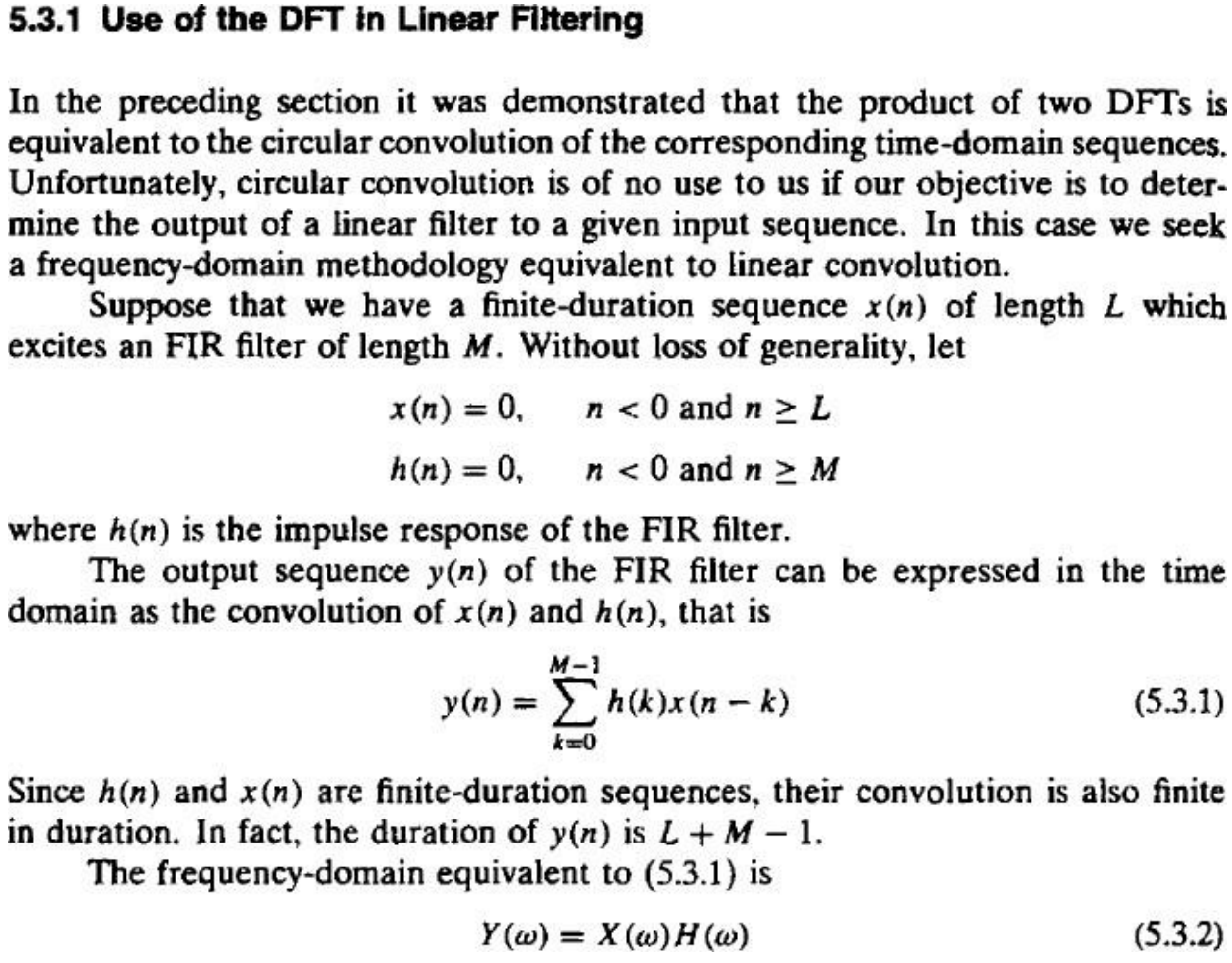
LINEAR FILTERING METHODS BASED ON THE DFT

Since the D F T provides a discrete frequency representation o f a finite-duration Sequence in the frequency domain, it is interesting to exp lore its use as a computational tool for linear system analysis and, especially, for linear filtering. We have already established that a system with frequency response *H { w) y* w hen excited with an input signal that has a spectrum possesses an output spectrum.

The output sequence *y(n)* is determined from its spectrum via the inverse Fourier transform. Computationally, the problem with this frequency domain approach is that are functions o f the

continuous variable. As a consequence, the computations cannot be done on a digital computer, since the computer can only store and perform computations on quantities at discrete frequencies.
On the other hand, the DFT does lend itself to computation on a digital computer. In the discussion that follows, we describe how the DFT can be used to perform linear filtering in the frequency domain. In particular, we present a computational procedure that serves as an alternative to time-domain convolution.

In fact, the frequency-domain approach based on the DFT, is computationally m ore efficient than time-domain convolution due to the existence of efficient algorithms for computing the DFT . These algorithms, which are described in Chapter 6, are collectively called fast Fourier transform (FFT) algorithms.



5.3.1 Use of the DFT in Linear Filtering

In the preceding section it was demonstrated that the product of two DFTs is equivalent to the circular convolution of the corresponding time-domain sequences. Unfortunately, circular convolution is of no use to us if our objective is to determine the output of a linear filter to a given input sequence. In this case we seek a frequency-domain methodology equivalent to linear convolution.

Suppose that we have a finite-duration sequence $x(n)$ of length L which excites an FIR filter of length M. Without loss of generality, let

$$x(n) = 0, \qquad n < 0 \text{ and } n \geq L$$

$$h(n) = 0, \qquad n < 0 \text{ and } n \geq M$$

where $h(n)$ is the impulse response of the FIR filter.

The output sequence $y(n)$ of the FIR filter can be expressed in the time domain as the convolution of $x(n)$ and $h(n)$, that is

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \tag{5.3.1}$$

Since $h(n)$ and $x(n)$ are finite-duration sequences, their convolution is also finite in duration. In fact, the duration of $y(n)$ is $L + M - 1$.

The frequency-domain equivalent to (5.3.1) is

$$Y(\omega) = X(\omega)H(\omega) \tag{5.3.2}$$

If the sequence $y(n)$ is to be represented uniquely in the frequency domain by samples of its spectrum $Y(\omega)$ at a set of discrete frequencies, the number of distinct samples must equal or exceed $L+M-1$. Therefore, a DFT of size $N \geq L+M-1$, is required to represent $\{y(n)\}$ in the frequency domain.

Now if

$$\begin{aligned} Y(k) &\equiv Y(\omega)|_{\omega=2\pi k/N} \qquad && k = 0, 1, \ldots, N-1 \\ &= X(\omega)H(\omega)|_{\omega=2\pi k/N} \qquad && k = 0, 1, \ldots, N-1 \end{aligned}$$

then

$$Y(k) = X(k)H(k) \qquad k = 0, 1, \ldots, N-1 \tag{5.3.3}$$

where $\{X(k)\}$ and $\{H(k)\}$ are the N-point DFTs of the corresponding sequences $x(n)$ and $h(n)$, respectively. Since the sequences $x(n)$ and $h(n)$ have a duration less than N, we simply pad these sequences with zeros to increase their length to N. This increase in the size of the sequences does not alter their spectra $X(\omega)$ and $H(\omega)$, which are continuous spectra, since the sequences are aperiodic. However, by sampling their spectra at N equally spaced points in frequency (computing the N-point DFTs), we have increased the number of samples that represent these sequences in the frequency domain beyond the minimum number (L or M, respectively).

Since the $N = L+M-1$-point DFT of the output sequence $y(n)$ is sufficient to represent $y(n)$ in the frequency domain. it follows that the multiplication of the N-point DFTs $X(k)$ and $H(k)$, according to (5.3.3), followed by the computation of the N-point IDFT. must yield the sequence $\{y(n)\}$. In turn, this implies that the N-point circular convolution of $x(n)$ with $h(n)$ must be equivalent to the linear convolution of $x(n)$ with $h(n)$. In other words, by increasing the length of the sequences $x(n)$ and $h(n)$ to N points (by appending zeros), and then circularly convolving the resulting sequences, we obtain the same result as would have been obtained with linear convolution. Thus with zero padding, the DFT can be used to perform linear filtering.

FAST FOURIER TRANSFORM

EFFICIENT COMPUTATION OF DFT:

In this section we represent several methods for computing dft efficiently. In view of the importance of the DFT in various digital signal processing applications such as linear filtering, correlation analysis and spectrum analysis, its efficient computation is a topic that has received considerably attention by many mathematicians, engineers and scientists. Basically the computation is done using the formula method.

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad 0 \le k \le N-1$$

where

$$W_N = e^{-j2\pi/N}$$

In general, the data sequence $x(n)$ is also assumed to be complex valued.

Similarly, the IDFT becomes

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk} \qquad 0 \le n \le N-1$$

We observe that for each value of k, direct computation of $X(k)$ involves N complex multiplications ($4N$ real multiplications) and $N-1$ complex additions ($4N-2$ real additions). Consequently, to compute all N values of the DFT requires N^2 complex multiplications and $N^2 - N$ complex additions.

6.1.1 Direct Computation of the DFT

For a complex-valued sequence $x(n)$ of N points, the DFT may be expressed as

$$X_R(k) = \sum_{n=0}^{N-1} \left[x_R(n) \cos \frac{2\pi kn}{N} + x_I(n) \sin \frac{2\pi kn}{N} \right] \qquad (6.1.6)$$

$$X_I(k) = -\sum_{n=0}^{N-1} \left[x_R(n) \sin \frac{2\pi kn}{N} - x_I(n) \cos \frac{2\pi kn}{N} \right] \qquad (6.1.7)$$

The direct computation of (6.1.6) and (6.1.7) requires:

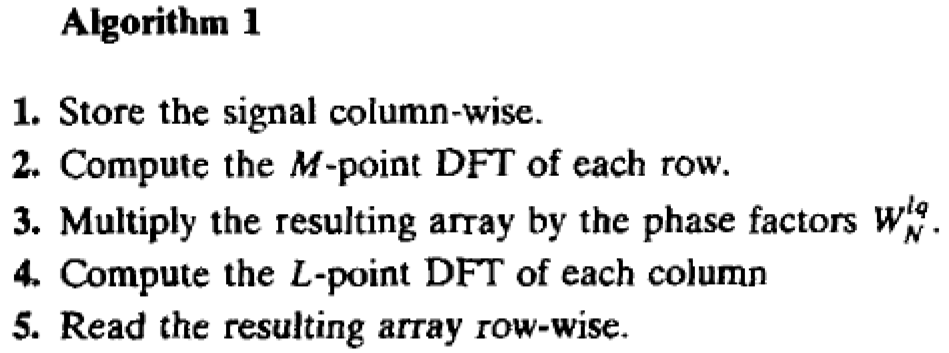
1. $2N^2$ evaluations of trigonometric functions.
2. $4N^2$ real multiplications.
3. $4N(N-1)$ real additions.
4. A number of indexing and addressing operations.

These operations are typical of DFT computational algorithms. The operations in items 2 and 3 result in the DFT values $X_R(k)$ and $X_I(k)$. The indexing and addressing operations are necessary to fetch the data $x(n)$, $0 \le n \le N-1$, and the phase factors and to store the results. The variety of DFT algorithms optimize each of these computational processes in a different way.

Divide-and-Conquer Approach to Computation of the DFT

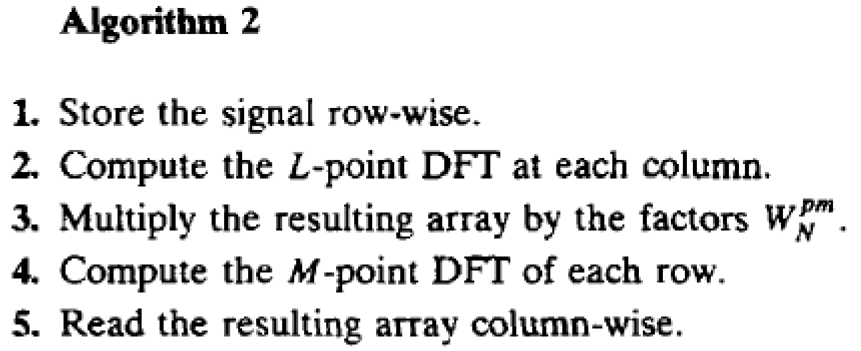
The development of computationally efficient algorithms for the DFT is made possible if we adopt a divide-and-conquer approach. This approach is based on the decomposition of an N-point DFT into successively smaller DFT. This basic approach leads to a family o f computationally efficient algorithm s know n collectively as FFT algorithms.

T o illustrate the basic notions, let us consider the computation of an N point DFT , where *N* can be factored as a product of two integers, that is, *N* = *L M*



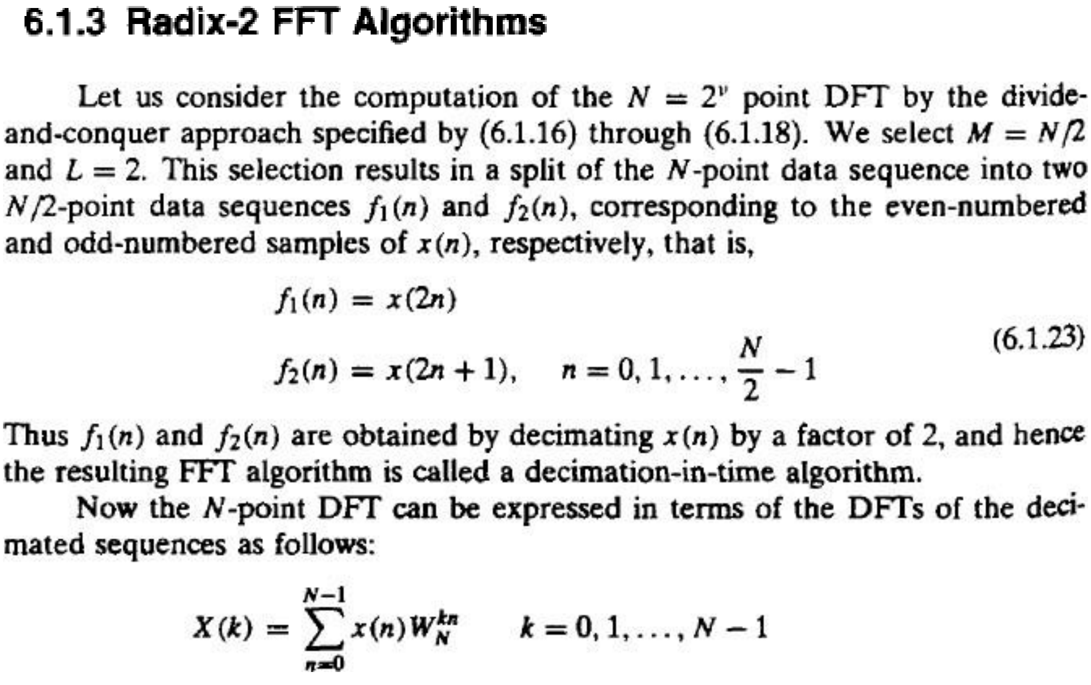
Algorithm 1

1. Store the signal column-wise.
2. Compute the M-point DFT of each row.
3. Multiply the resulting array by the phase factors W_N^{lq}.
4. Compute the L-point DFT of each column
5. Read the resulting array row-wise.



Algorithm 2

1. Store the signal row-wise.
2. Compute the L-point DFT at each column.
3. Multiply the resulting array by the factors W_N^{pm}.
4. Compute the M-point DFT of each row.
5. Read the resulting array column-wise.



6.1.3 Radix-2 FFT Algorithms

Let us consider the computation of the $N = 2^\nu$ point DFT by the divide-and-conquer approach specified by (6.1.16) through (6.1.18). We select $M = N/2$ and $L = 2$. This selection results in a split of the N-point data sequence into two $N/2$-point data sequences $f_1(n)$ and $f_2(n)$, corresponding to the even-numbered and odd-numbered samples of $x(n)$, respectively, that is,

$$\begin{aligned} f_1(n) &= x(2n) \\ f_2(n) &= x(2n+1), \qquad n = 0, 1, \dots, \frac{N}{2} - 1 \end{aligned} \tag{6.1.23}$$

Thus $f_1(n)$ and $f_2(n)$ are obtained by decimating $x(n)$ by a factor of 2, and hence the resulting FFT algorithm is called a decimation-in-time algorithm.

Now the N-point DFT can be expressed in terms of the DFTs of the decimated sequences as follows:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad k = 0, 1, \dots, N-1$$

$$= \sum_{n \text{ even}} x(n) W_N^{kn} + \sum_{n \text{ odd}} x(n) W_N^{kn} \tag{6.1.24}$$

$$= \sum_{m=0}^{(N/2)-1} x(2m) W_N^{2mk} + \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{k(2m+1)}$$

But $W_N^2 = W_{N/2}$. With this substitution, (6.1.24) can be expressed as

$$\begin{aligned} X(k) &= \sum_{m=0}^{(N/2)-1} f_1(m) W_{N/2}^{km} + W_N^k \sum_{m=0}^{(N/2)-1} f_2(m) W_{N/2}^{km} \\ &= F_1(k) + W_N^k F_2(k) \qquad k = 0, 1, \ldots, N-1 \end{aligned} \tag{6.1.25}$$

where $F_1(k)$ and $F_2(k)$ are the $N/2$-point DFTs of the sequences $f_1(m)$ and $f_2(m)$, respectively.

Since $F_1(k)$ and $F_2(k)$ are periodic, with period $N/2$, we have $F_1(k + N/2) = F_1(k)$ and $F_2(k + N/2) = F_2(k)$. In addition, the factor $W_N^{k+N/2} = -W_N^k$. Hence (6.1.25) can be expressed as

$$X(k) = F_1(k) + W_N^k F_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \tag{6.1.26}$$

$$X\left(k + \frac{N}{2}\right) = F_1(k) - W_N^k F_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \tag{6.1.27}$$

To be consistent with our previous notation, we may define

$$G_1(k) = F_1(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

$$G_2(k) = W_N^k F_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

Then the DFT $X(k)$ may be expressed as

$$\begin{aligned} X(k) &= G_1(k) + G_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \\ X(k + \frac{N}{2}) &= G_1(k) - G_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \end{aligned} \tag{6.1.28}$$

$N/4$-point sequences

$$\begin{aligned} v_{11}(n) &= f_1(2n) \qquad n = 0, 1, \ldots, \frac{N}{4} - 1 \\ v_{12}(n) &= f_1(2n+1) \quad n = 0, 1, \ldots, \frac{N}{4} - 1 \end{aligned} \tag{6.1.29}$$

and $f_2(n)$ would yield

$$\begin{aligned} v_{21}(n) &= f_2(2n) \qquad n = 0, 1, \ldots, \frac{N}{4} - 1 \\ v_{22}(n) &= f_2(2n+1) \quad n = 0, 1, \ldots, \frac{N}{4} - 1 \end{aligned} \tag{6.1.30}$$

By computing $N/4$-point DFTs, we would obtain the $N/2$-point DFTs $F_1(k)$ and $F_2(k)$ from the relations

$$\begin{aligned} F_1(k) &= V_{11}(k) + W_{N/2}^k V_{12}(k) \quad k = 0, 1, \ldots, \frac{N}{4} - 1 \\ F_1\left(k + \frac{N}{4}\right) &= V_{11}(k) - W_{N/2}^k V_{12}(k) \quad k = 0, 1, \ldots, \frac{N}{4} - 1 \end{aligned} \tag{6.1.31}$$

$$\begin{aligned} F_2(k) &= V_{21}(k) + W_{N/2}^k V_{22}(k) \quad k = 0, 1, \ldots, \frac{N}{4} - 1 \\ F_2\left(k + \frac{N}{4}\right) &= V_{21}(k) - W_{N/2}^k V_{22}(k) \quad k = 0, \ldots, \frac{N}{4} - 1 \end{aligned} \tag{6.1.32}$$

where the $\{V_{ij}(k)\}$ are the $N/4$-point DFTs of the sequences $\{v_{ij}(n)\}$.

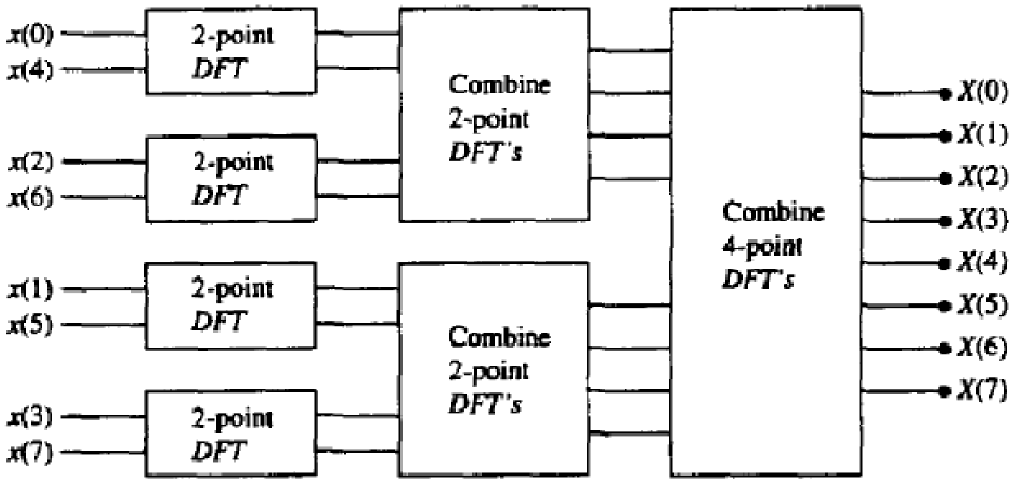


Figure 6.5 Three stages in the computation of an $N = 8$-point DFT.

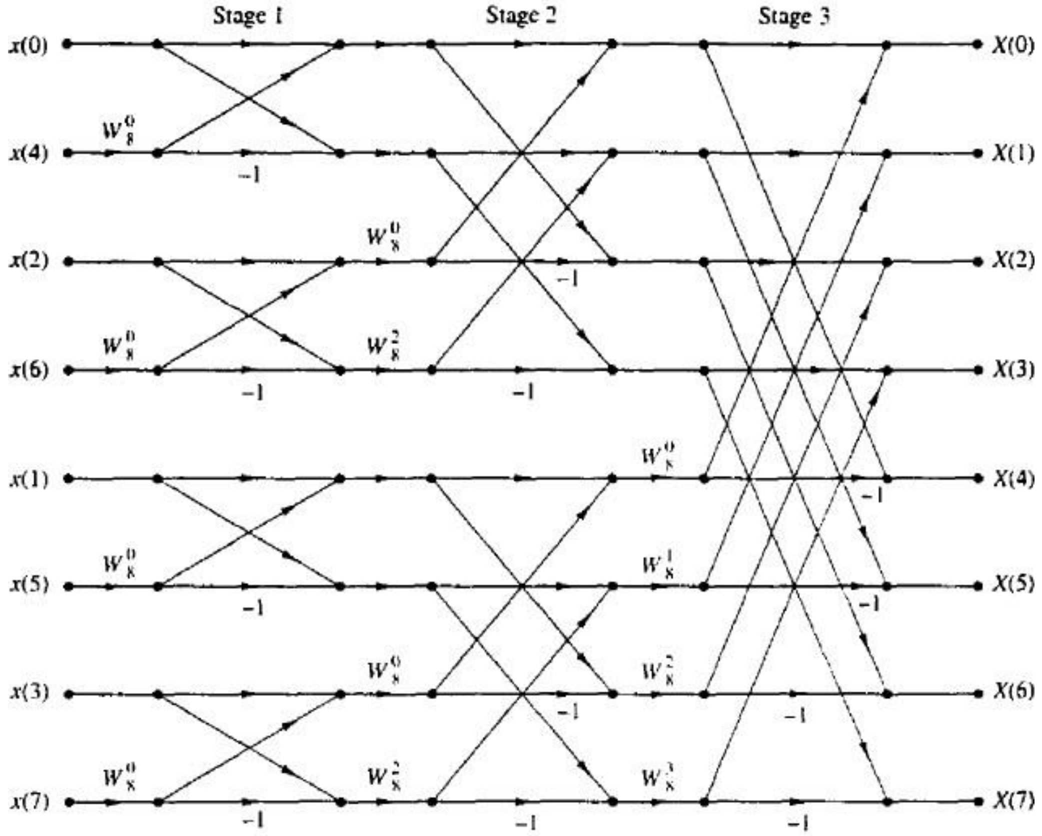


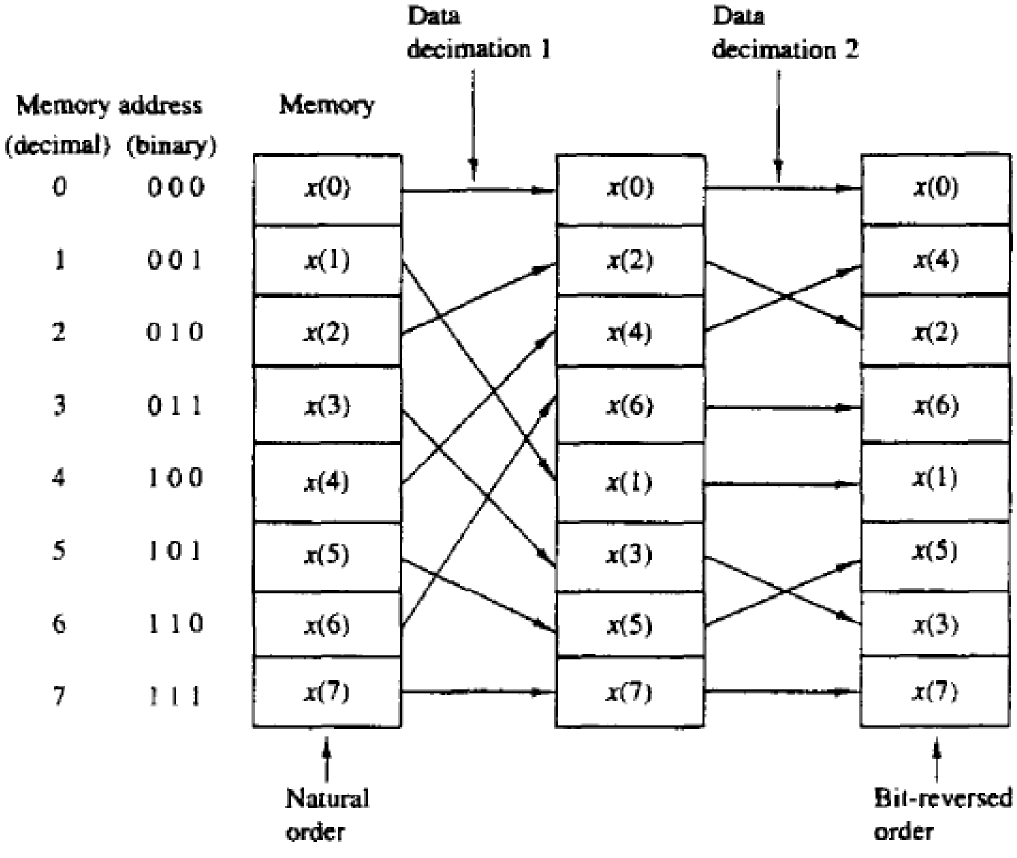
Figure 6.6 Eight-point decimation-in-time FFT algorithm.

Another important radix-2 FFT algorithm, called the decimation-in-frequency algorithm, is obtained by using the divide-and-conquer approach described in Section 6.1.2 with the choice of $M = 2$ and $L = N/2$. This choice of parameters implies a column-wise storage of the input data sequence. To derive the algorithm, we begin by splitting the DFT formula into two summations, one of which involves the sum over the first $N/2$ data points and the second sum involves the last $N/2$ data points. Thus we obtain

$$\begin{aligned} X(k) &= \sum_{n=0}^{(N/2)-1} x(n)W_N^{kn} + \sum_{n=N/2}^{N-1} x(n)W_N^{kn} \\ &= \sum_{n=0}^{(N/2)-1} x(n)W_N^{kn} + W_N^{Nk/2} \sum_{n=0}^{(N/2)-1} x\left(n+\frac{N}{2}\right)W_N^{kn} \end{aligned} \tag{6.1.33}$$

Since $W_N^{kN/2} = (-1)^k$, the expression (6.1.33) can be rewritten as

$$X(k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + (-1)^k x\left(n+\frac{N}{2}\right)\right] W_N^{kn} \tag{6.1.34}$$



(a)

Now, let us split (decimate) $X(k)$ into the even- and odd-numbered samples. Thus we obtain

$$X(2k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + x\left(n + \frac{N}{2}\right)\right] W_{N/2}^{kn} \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \qquad (6.1.35)$$

and

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} \left\{\left[x(n) - x\left(n + \frac{N}{2}\right)\right] W_N^n\right\} W_{N/2}^{kn} \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \qquad (6.1.36)$$

where we have used the fact that $W_N^2 = W_{N/2}$.

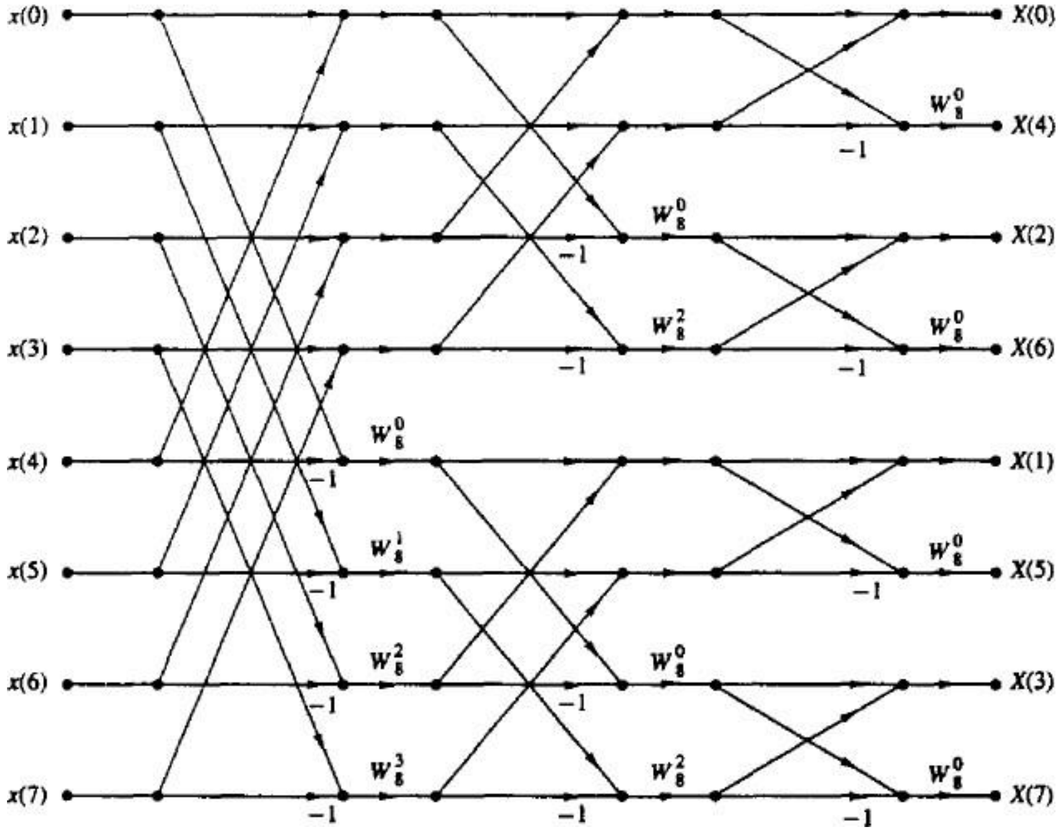
If we define the $N/2$-point sequences $g_1(n)$ and $g_2(n)$ as

$$\begin{aligned} g_1(n) &= x(n) + x\left(n + \frac{N}{2}\right) \\ g_2(n) &= \left[x(n) - x\left(n + \frac{N}{2}\right)\right] W_N^n \qquad n = 0, 1, 2, \ldots, \frac{N}{2} - 1 \end{aligned} \qquad (6.1.37)$$

then

$$\begin{aligned} X(2k) &= \sum_{n=0}^{(N/2)-1} g_1(n) W_{N/2}^{kn} \\ X(2k+1) &= \sum_{n=0}^{(N/2)-1} g_2(n) W_{N/2}^{kn} \end{aligned} \qquad (6.1.38)$$

We observe from Fig. 6.11, that the input data $x(n)$ occurs in natural order, but the output DFT occurs in bit-reversed order. We also note that the computations are performed in place. However, it is possible to reconfigure the decimation-in-frequency algorithm so that the input sequence occurs in bit-reversed order while the output DFT occurs in normal order. Furthermore, if we abandon the requirement that the computations be done in place, it is also possible to have both the input data and the output DFT in normal order.



6.1.4 Radix-4 FFT Algorithms

When the number of data points N in the DFT is a power of 4 (i.e., $N = 4^{\nu}$), we can, of course, always use a radix-2 algorithm for the computation. However, for this case, it is more efficient computationally to employ a radix-4 FFT algorithm.

Let us begin by describing a radix-4 decimation-in-time FFT algorithm, which is obtained by selecting $L = 4$ and $M = N/4$ in the divide-and-conquer approach described in Section 6.1.2. For this choice of L and M, we have $l, p = 0, 1, 2, 3$; $m, q = 0, 1, \ldots, N/4 - 1$; $n = 4m + l$; and $k = (N/4)p + q$. Thus we split or decimate the N-point input sequence into four subsequences, $x(4n)$, $x(4n+1)$, $x(4n+2)$, $x(4n+3)$, $n = 0, 1, \ldots, N/4 - 1$.

By applying (6.1.15) we obtain

$$X(p, q) = \sum_{l=0}^{3} \left[W_N^{lq} F(l, q) \right] W_4^{lp} \qquad p = 0, 1, 2, 3 \tag{6.1.39}$$

where $F(l, q)$ is given by (6.1.16), that is,

$$F(l, q) = \sum_{m=0}^{(N/4)-1} x(l, m) W_{N/4}^{mq} \qquad \begin{array}{l} l = 0, 1, 2, 3, \\ q = 0, 1, 2, \ldots, \dfrac{N}{4} - 1 \end{array} \tag{6.1.40}$$

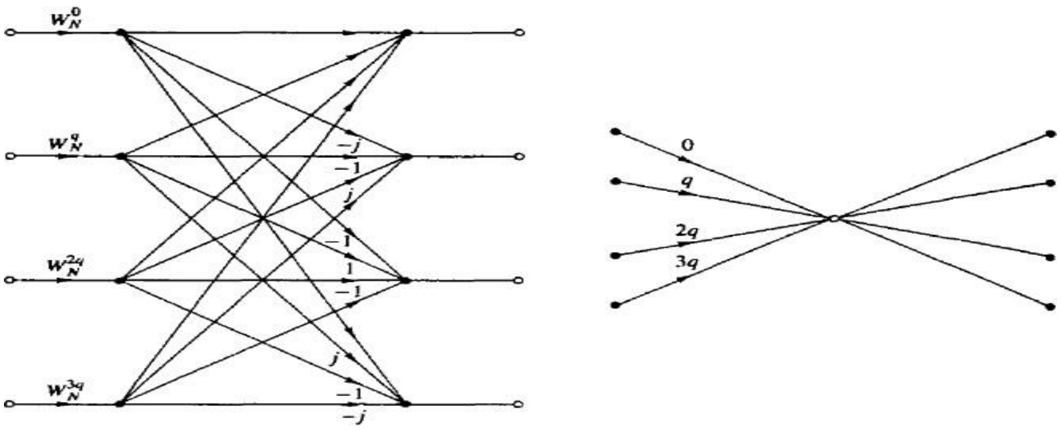
and

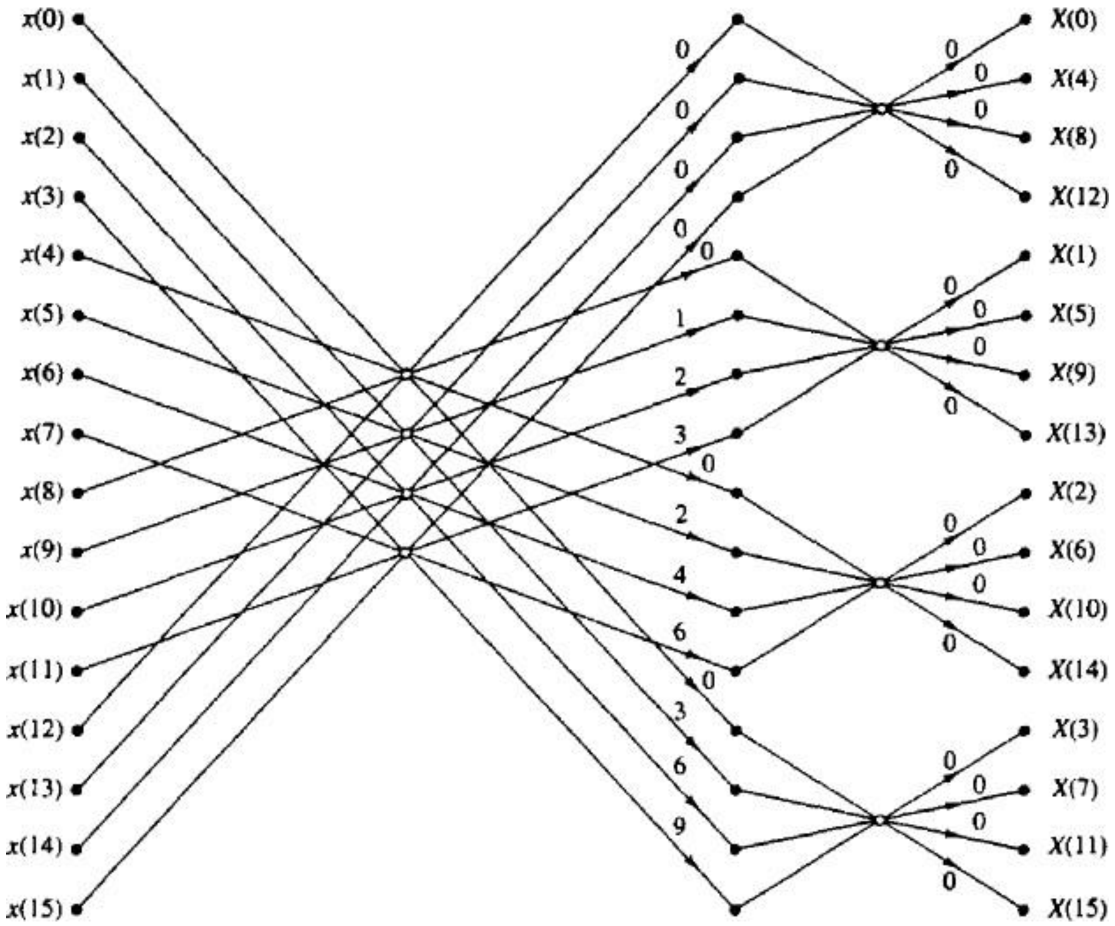
$$x(l, m) = x(4m + l) \tag{6.1.41}$$

$$X(p, q) = X\left(\frac{N}{4}p + q\right) \tag{6.1.42}$$

Thus, the four $N/4$-point DFTs obtained from (6.1.40) are combined according to (6.1.39) to yield the N-point DFT. The expression in (6.1.39) for combining the $N/4$-point DFTs defines a radix-4 decimation-in-time butterfly, which can be expressed in matrix form as

$$\begin{bmatrix} X(0, q) \\ X(1, q) \\ X(2, q) \\ X(3, q) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} W_N^0 F(0, q) \\ W_N^q F(1, q) \\ W_N^{2q} F(2, q) \\ W_N^{3q} F(3, q) \end{bmatrix} \tag{6.1.43}$$





$$X(4k) = \sum_{n=0}^{N/4-1} \left[x(n) + x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) + x\left(n + \frac{3N}{4}\right) \right] W_N^0 W_{N/4}^{kn}$$

$$X(4k+1) = \sum_{n=0}^{N/4-1} \left[x(n) - jx\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) + jx\left(n + \frac{3N}{4}\right) \right] W_N^{n} W_{N/4}^{kn}$$

$$X(4k+2) = \sum_{n=0}^{N/4-1} \left[x(n) - x\left(n + \frac{N}{4}\right) + x\left(n + \frac{N}{2}\right) - x\left(n + \frac{3N}{4}\right) \right] W_N^{2n} W_{N/4}^{kn}$$

$$X(4k+3) = \sum_{n=0}^{N/4-1} \left[x(n) + jx\left(n + \frac{N}{4}\right) - x\left(n + \frac{N}{2}\right) - jx\left(n + \frac{3N}{4}\right) \right] W_N^{3n} W_{N/4}^{kn}$$

6.1.5 Split-Radix FFT Algorithms

An inspection of the radix-2 decimation-in-frequency flowgraph shown in Fig. 6.11 indicates that the even-numbered points of the DFT can be computed independently of the odd-numbered points. This suggests the possibility of using different computational methods for independent parts of the algorithm with the objective of reducing the number of computations. The split-radix FFT (SRFFT) algorithms exploit this idea by using both a radix-2 and a radix-4 decomposition in the same FFT algorithm.

We illustrate this approach with a decimation-in-frequency SRFFT algorithm due to Duhamel (1986). First, we recall that in the radix-2 decimation-in-frequency FFT algorithm, the even-numbered samples of the N-point DFT are given as

$$X(2k) = \sum_{n=0}^{N/2-1} \left[x(n) + x\left(n + \frac{N}{2}\right) \right] W_{N/2}^{nk} \qquad k = 0, 1, \ldots, \frac{N}{2} - 1 \qquad (6.1.55)$$

Note that these DFT points can be obtained from an $N/2$-point DFT without any additional multiplications. Consequently, a radix-2 suffices for this computation.

The odd-numbered samples $\{X(2k+1)\}$ of the DFT require the premultiplication of the input sequence with the twiddle factors W_N^n. For these samples a radix-4 decomposition produces some computational efficiency because the four-point DFT has the largest multiplication-free butterfly. Indeed, it can be shown that using a radix greater than 4, does not result in a significant reduction in computational complexity.

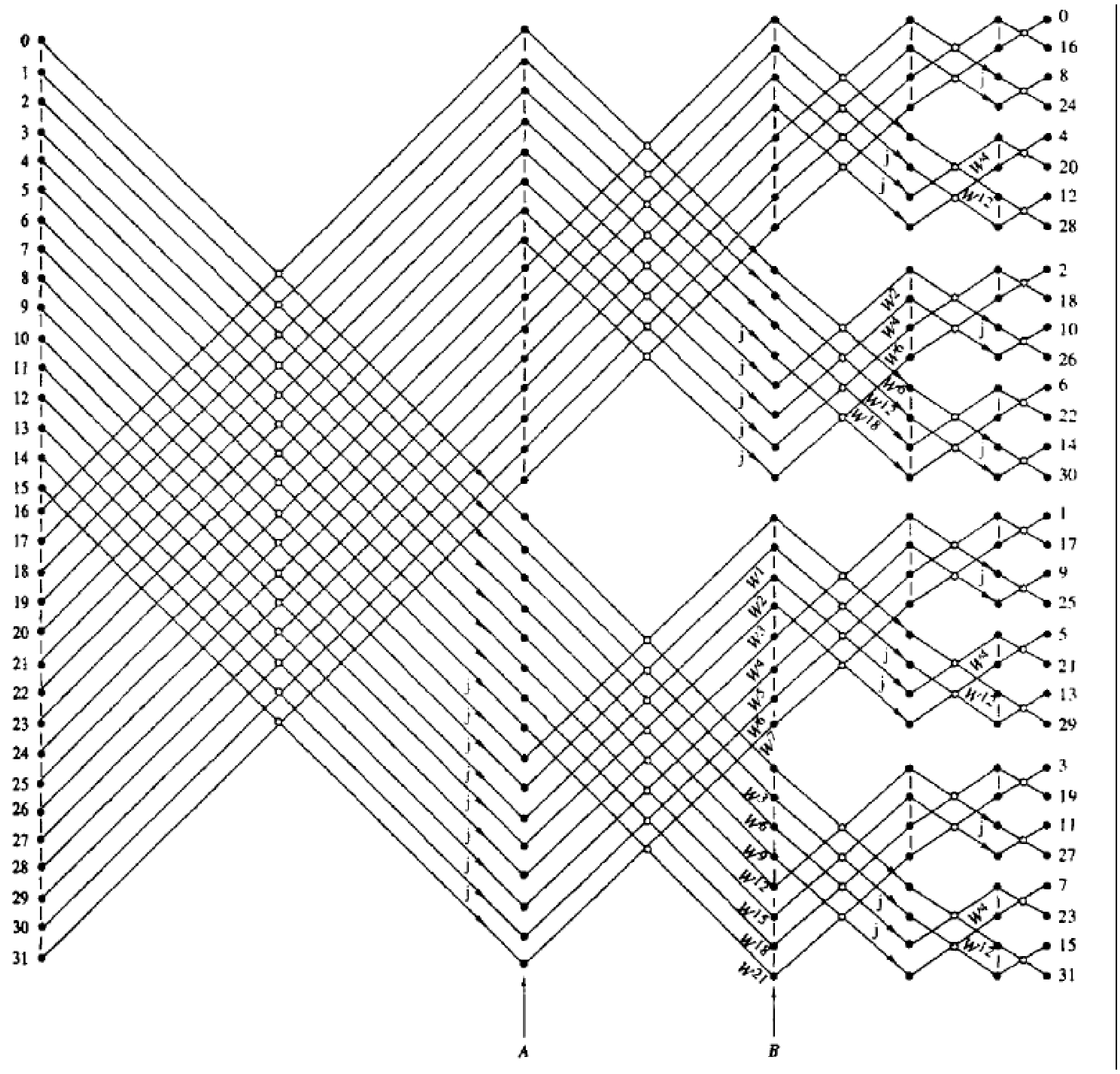
If we use a radix-4 decimation-in-frequency FFT algorithm for the odd-numbered samples of the N-point DFT, we obtain the following $N/4$-point DFTs:

$$\begin{aligned} X(4k+1) = & \sum_{n=0}^{N/4-1} \{[x(n) - x(n+N/2)] \qquad (6.1.56) \\ & - j[x(n+N/4) - x(n+3N/4)]\} W_N^n W_{N/4}^{kn} \end{aligned}$$

$$\begin{aligned} X(4k+3) = & \sum_{n=0}^{N/4-1} \{[x(n) - x(n+N/2)] \qquad (6.1.57) \\ & + j[x(n+N/4) - x(n+3N/4)]\} W_N^{3n} W_{N/4}^{kn} \end{aligned}$$

Thus the N-point DFT is decomposed into one $N/2$-point DFT without additional twiddle factors and two $N/4$-point DFTs with twiddle factors. The N-point DFT is obtained by successive use of these decompositions up to the last stage. Thus we obtain a decimation-in-frequency SRFFT algorithm.

Figure 6.15 shows the flow graph for an in-place 32-point decimation-in-frequency SRFFT algorithm. At stage A of the computation for $N = 32$, the



An additional factor of 2 savings in storage of twiddle factors can be obtained by introducing a 90° phase offset at the mid point of each twiddle array , which can be removed if necessary at the ouput of the SRFFT computation. The incorporation of this improvement into the SRFFT results in an other algorithm also due to price called the PFFT algorithm.

Implementation of FFT Algorithms

Now that w e has described the basic radix-2 and radix -4 F FT algorithm s, let us consider some of the implementation issues. Our remarks apply directly to radix-2 algorithms, although similar comments may be made about radix-4 and higher-radix algorithms.

Basically, the radix-2 FFT algorithm consists of taking two data points at a time from memory, performing the butterfly computations and returning the resulting numbers to memory. This procedure is repeated many times ($(N \log_2 N)/2$ times) in the computation of an N-point DFT.

The butterfly computations require the twiddle factors $\{W_N^k\}$ at various stages in either natural or bit-reversed order. In an efficient implementation of the algorithm, the phase factors are computed once and stored in a table, either in normal order or in bit-reversed order, depending on the specific implementation of the algorithm.

Memory requirement is another factor that must be considered. If the computations are performed in place, the number of memory locations required is $2N$ since the numbers are complex. However, we can instead double the memory to 4N, thus simplifying the indexing and control operations in the FFT algorithms. In this case we simply alternate in the use of the two sets of memory locations from one stage of the FFT algorithm to the other. Doubling of the memory also allows us to have both the input sequence and the output sequence in normal order.

Finally, we note that the emphasis in our discussion of FFT algorithms was on radix-2, radix-4, and split-radix algorithms. These are by far the most widely used in practice. When the number of data points is not a power of 2 or 4, it is a simple matter to pad the sequence $x(n)$ with zeros such that $N = 2^{\nu}$ or $N = 4^{\nu}$.

The measure of complexity for FFT algorithms that we have emphasized is the required number of arithmetic operations (multiplications and additions). Although this is a very important benchmark for computational complexity, there are other issues to be considered in practical implementation of FFT algorithms. These include the architecture of the processor, the available instruction set, the data structures for storing twiddle factors, and other considerations.

For general-purpose computers, where the cost of the numerical operations dominate, radix-2, radix-4, and split-radix FFT algorithms are good candidates. However, in the case of special-purpose digital signal processors, featuring single-cycle multiply-and-accumulate operation, bit-reversed addressing, and a high degree of instruction parallelism, the structural regularity of the algorithm is equally important as arithmetic complexity. Hence for DSP processors, radix-2 or radix-4 decimation-in-frequency FFT algorithms are preferable in terms of speed and accuracy. The irregular structure of the SRFFT may render it less suitable for implementation on digital signal processors. Structural regularity is also important in the implementation of FFT algorithms on vector processors, multiprocessors, and in VLSI. Interprocessor communication is an important consideration in such implementations on parallel processors.

In conclusion, we have presented several important considerations in the implementation of FFT algorithms. Advances in digital signal processing technology, in hardware and software, will continue to influence the choice among FFT algorithms for various practical applications.

APPLICATIONS OF FFT ALGORITHMS

The FFT algorithms described in the preceding section find application in a variety of areas, including linear filtering, correlation, and spectrum analysis. Basically, the FFT algorithm is used as an efficient means to compute the DFT and the IDFT.

In this section we consider the use of the FFT algorithm in linear filtering and in the computation of the crosscorrelation of two sequences. The use of the FFT in spectrum analysis is considered in Chapter 12. In addition we illustrate how to enhance the efficiency of the FFT algorithm by forming complex-valued sequences from real-valued sequences prior to the computation of the DFT.

6.2.1 Efficient Computation of the DFT of Two Real Sequences

The FFT algorithm is designed to perform complex multiplications and additions, even though the input data may be real valued. The basic reason for this situation is

Suppose that $x_1(n)$ and $x_2(n)$ are two real-valued sequences of length N, and let $x(n)$ be a complex-valued sequence defined as

$$x(n) = x_1(n) + jx_2(n) \qquad 0 \le n \le N-1 \tag{6.2.1}$$

The DFT operation is linear and hence the DFT of $x(n)$ can be expressed as

$$X(k) = X_1(k) + jX_2(k) \tag{6.2.2}$$

The sequences $x_1(n)$ and $x_2(n)$ can be expressed in terms of $x(n)$ as follows:

$$x_1(n) = \frac{x(n) + x^*(n)}{2} \tag{6.2.3}$$

$$x_2(n) = \frac{x(n) - x^*(n)}{2j} \tag{6.2.4}$$

Hence the DFTs of $x_1(n)$ and $x_2(n)$ are

$$X_1(k) = \frac{1}{2}\{DFT[x(n)] + DFT[x^*(n)]\} \tag{6.2.5}$$

$$X_2(k) = \frac{1}{2j}\{DFT[x(n)] - DFT[x^*(n)]\} \tag{6.2.6}$$

Recall that the DFT of $x^*(n)$ is $X^*(N-k)$. Therefore,

$$X_1(k) = \frac{1}{2}[X(k) + X^*(N-k)] \tag{6.2.7}$$

$$X_2(k) = \frac{1}{j2}[X(k) - X^*(N-k)] \tag{6.2.8}$$

6.2.2 Efficient Computation of the DFT of a 2*N*-Point Real Sequence

Suppose that $g(n)$ is a real-valued sequence of $2N$ points. We now demonstrate how to obtain the $2N$-point DFT of $g(n)$ from computation of one N-point DFT involving complex-valued data. First, we define

$$\begin{aligned} x_1(n) &= g(2n) \\ x_2(n) &= g(2n+1) \end{aligned} \tag{6.2.9}$$

Let $x(n)$ be the N-point complex-valued sequence

$$x(n) = x_1(n) + jx_2(n) \tag{6.2.10}$$

From the results of the preceding section, we have

$$\begin{aligned} X_1(k) &= \frac{1}{2}[X(k) + X^*(N-k)] \\ X_2(k) &= \frac{1}{2j}[X(k) - X^*(N-k)] \end{aligned} \tag{6.2.11}$$

Finally, we must express the $2N$-point DFT in terms of the two N-point DFTs, $X_1(k)$ and $X_2(k)$. To accomplish this, we proceed as in the decimation-in-time FFT algorithm, namely,

$$\begin{aligned} G(k) &= \sum_{n=0}^{N-1} g(2n) W_{2N}^{2nk} + \sum_{n=0}^{N-1} g(2n+1) W_{2N}^{(2n+1)k} \\ &= \sum_{n=0}^{N-1} x_1(n) W_N^{nk} + W_{2N}^k \sum_{n=0}^{N-1} x_2(n) W_N^{nk} \end{aligned}$$

Consequently,

$$\begin{aligned} G(k) &= X_1(k) + W_2^k N X_2(k) \qquad k = 0, 1, \ldots, N-1 \\ G(k+N) &= X_1(k) - W_2^k N X_2(k) \qquad k = 0, 1, \ldots, N-1 \end{aligned} \tag{6.2.12}$$

Thus we have computed the DFT of a $2N$-point real sequence from one N-point DFT and some additional computation as indicated by (6.2.11) and (6.2.12).

6.2.3 Use of the FFT Algorithm in Linear Filtering and Correlation

An important application of the FFT algorithm is in FIR linear filtering of long data sequences. In Chapter 5 we described two methods, the overlap-add and the overlap-save methods for filtering a long data sequence with an FIR filter, based on the use of the DFT. In this section we consider the use of these two methods in conjunction with the FFT algorithm for computing the DFT and the IDFT.

Let $h(n)$, $0 \leq n \leq M-1$, be the unit sample response of the FIR filter and let $x(n)$ denote the input data sequence. The block size of the FFT algorithm is N, where $N = L + M - 1$ and L is the number of new data samples being processed by the filter. We assume that for any given value of M, the number L of data samples is selected so that N is a power of 2. For purposes of this discussion, we consider only radix-2 FFT algorithms.

The N-point DFT of $h(n)$, which is padded by $L-1$ zeros, is denoted as $H(k)$. This computation is performed once via the FFT and the resulting N complex numbers are stored. To be specific we assume that the decimation-in-frequency

FFT algorithm is used to compute $H(k)$. This yields $H(k)$ in bit-reversed order, which is the way it is stored in memory.

In the overlap-save method, the first $M-1$ data points of each data block are the last $M-1$ data points of the previous data block. Each data block contains L new data points, such that $N = L + M - 1$. The N-point DFT of each data block is performed by the FFT algorithm. If the decimation-in-frequency algorithm is employed, the input data block requires no shuffling and the values of the DFT occur in bit-reversed order. Since this is exactly the order of $H(k)$, we can multiply the DFT of the data, say $X_m(k)$, with $H(k)$ and thus the result

$$Y_m(k) = H(k)X_m(k)$$

is also in bit-reversed order.

The inverse DFT (IDFT) can be computed by use of an FFT algorithm that takes the input in bit-reversed order and produces an output in normal order. Thus there is no need to shuffle any block of data either in computing the DFT or the IDFT.

If the overlap-add method is used to perform the linear filtering, the computational method using the FFT algorithm is basically the same. The only difference is that the N-point data blocks consist of L new data points and $M-1$ additional zeros. After the IDFT is computed for each data block, the N-point filtered blocks are overlapped as indicated in Section 5.3.2, and the $M-1$ overlapping data points between successive output records are added together.

6.3.1 The Goertzel Algorithm

The Goertzel algorithm exploits the periodicity of the phase factors $\{W_N^k\}$ and allows us to express the computation of the DFT as a linear filtering operation. Since $W_N^{-kN} = 1$, we can multiply the DFT by this factor. Thus

$$X(k) = W_N^{-kN} \sum_{m=0}^{N-1} x(m) W_N^{km} = \sum_{m=0}^{N-1} x(m) W_N^{-k(N-m)} \qquad (6.3.1)$$

We note that (6.3.1) is in the form of a convolution. Indeed, if we define the sequence $y_k(n)$ as

$$y_k(n) = \sum_{m=0}^{N-1} x(m) W_N^{-k(n-m)} \qquad (6.3.2)$$

then it is clear that $y_k(n)$ is the convolution of the finite-duration input sequence $x(n)$ of length N with a filter that has an impulse response

$$h_k(n) = W_N^{-kn} u(n) \qquad (6.3.3)$$

The output of this filter at $n = N$ yields the value of the DFT at the frequency $\omega_k = 2\pi k/N$. That is,

$$X(k) = y_k(n)|_{n=N} \qquad (6.3.4)$$

as can be verified by comparing (6.3.1) with (6.3.2).

The filter with impulse response $h_k(n)$ has the system function

$$H_k(z) = \frac{1}{1 - W_N^{-k} z^{-1}} \qquad (6.3.5)$$

This filter has a pole on the unit circle at the frequency $\omega_k = 2\pi k/N$. Thus, the entire DFT can be computed by passing the block of input data into a parallel bank of N single-pole filters (resonators), where each filter has a pole at the corresponding frequency of the DFT.

Instead of performing the computation of the DFT as in (6.3.2), via convolution, we can use the difference equation corresponding to the filter given by (6.3.5) to compute $y_k(n)$ recursively. Thus we have

$$y_k(n) = W_N^{-k} y_k(n-1) + x(n) \qquad y_k(-1) = 0 \tag{6.3.6}$$

The desired output is $X(k) = y_k(N)$, for $k = 0, 1, \ldots, N-1$. To perform this computation, we can compute once and store the phase factors W_N^{-k}.

The complex multiplications and additions inherent in (6.3.6) can be avoided by combining the pairs of resonators possessing complex-conjugate poles. This leads to two-pole filters with system functions of the form

$$H_k(z) = \frac{1 - W_N^k z^{-1}}{1 - 2\cos(2\pi k/N)z^{-1} + z^{-2}} \tag{6.3.7}$$

6.3.2 The Chirp-z Transform Algorithm

The DFT of an N-point data sequence x(n) has been viewed as the z-transform of $x(n)$ evaluated at N equally spaced points on the unit circle. It has also been viewed as N equally spaced samples of the Fourier transform of the data sequence $x(n)$. In this section we consider the evaluation of $X(z)$ on other contours in the z-plane, including the unit circle.

Suppose that we wish to compute the values of the z-transform of $x(n)$ at a set of points $\{z_k\}$. Then,

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n} \qquad k = 0, 1, \ldots, L-1 \tag{6.3.10}$$

For example, if the contour is a circle of radius r and the z_k are N equally spaced points, then

$$\begin{aligned} z_k &= re^{j2\pi kn/N} \qquad k = 0, 1, 2, \ldots, N-1 \\ X(z_k) &= \sum_{n=0}^{N-1} [x(n) r^{-n}] e^{-j2\pi kn/N} \qquad k = 0, 1, 2, \ldots, N-1 \end{aligned} \tag{6.3.11}$$

In this case the FFT algorithm can be applied on the modified sequence $x(n)r^{-n}$.

More generally, suppose that the points z_k in the z-plane fall on an arc which begins at some point

$$z_0 = r_0 e^{j\theta_0}$$

and spirals either in toward the origin or out away from the origin such that the points $\{z_k\}$ are defined as

$$z_k = r_0 e^{j\theta_0} (R_0 e^{j\phi_0})^k \qquad k = 0, 1, \ldots, L-1 \tag{6.3.12}$$

When points $\{z_k\}$ in (6.3.12) are substituted into the expression for the z-transform, we obtain

$$\begin{aligned} X(z_k) &= \sum_{n=0}^{N-1} x(n) z_k^{-n} \\ &= \sum_{n=0}^{N-1} x(n)(r_0 e^{j\theta_0})^{-n} V^{-nk} \end{aligned} \tag{6.3.13}$$

where, by definition,

$$V = R_0 e^{j\phi_0} \tag{6.3.14}$$

We can express (6.3.13) in the form of a convolution, by noting that

$$nk = \tfrac{1}{2}[n^2 + k^2 - (k-n)^2] \tag{6.3.15}$$

Substitution of (6.3.15) into (6.3.13) yields

$$X(z_k) = V^{-k^2/2} \sum_{n=0}^{N-1} [x(n)(r_0 e^{j\theta_0})^{-n} V^{-n^2/2}] V^{(k-n)^2/2} \tag{6.3.16}$$

Let us define a new sequence $g(n)$ as

$$g(n) = x(n)(r_0 e^{j\theta_0})^{-n} V^{-n^2/2} \tag{6.3.17}$$

Then (6.3.16) can be expressed as

$$X(z_k) = V^{-k^2/2} \sum_{n=0}^{N-1} g(n) V^{(k-n)^2/2} \tag{6.3.18}$$

The summation in (6.3.18) can be interpreted as the convolution of the sequence $g(n)$ with the impulse response $h(n)$ of a filter, where

$$h(n) = V^{n^2/2} \tag{6.3.19}$$

Consequently, (6.3.18) may be expressed as

$$\begin{aligned} X(z_k) &= V^{-k^2/2} y(k) \\ &= \frac{y(k)}{h(k)} \qquad k = 0, 1, \ldots, L-1 \end{aligned} \tag{6.3.20}$$

where $y(k)$ is the output of the filter

$$y(k) = \sum_{n=0}^{N-1} g(n) h(k-n) \qquad k = 0, 1, \ldots, L-1 \tag{6.3.21}$$

QUANTIZATION EFFECTS IN THE COMPUTATION OF THE DFT*

As we have observed in our previous discussions, the DFT plays an important role in many digital signal processing applications, including FIR filtering, the computation of the correlation between signals, and spectral analysis. For this reason it is important for us to know the effect of quantization errors in its computation. In particular, we shall consider the effect of round-off errors due to the multiplications performed in the DFT with fixed-point arithmetic.

6.4.1 Quantization Errors in the Direct Computation of the DFT

Given a finite-duration sequence $\{x(n)\}$, $0 \leq n \leq N-1$, the DFT of $\{x(n)\}$ is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad k = 0, 1, \ldots, N-1 \tag{6.4.1}$$

where $W_N = e^{-j2\pi/N}$. We assume that in general, $\{x(n)\}$ is a complex-valued sequence. We also assume that the real and imaginary components of $\{x(n)\}$ and $\{W_N^{kn}\}$ are represented by b bits. Consequently, the computation of the product $x(n)W_N^{kn}$ requires four real multiplications. Each real multiplication is rounded from $2b$ bits to b bits, and hence there are four quantization errors for each complex-valued multiplication.

In the direct computation of the DFT, there are N complex-valued multiplications for each point in the DFT. Therefore, the total number of real multiplications in the computation of a single point in the DFT is $4N$. Consequently, there are $4N$ quantization errors.

Let us evaluate the variance of the quantization errors in a fixed-point computation of the DFT. First, we make the following assumptions about the statistical properties of the quantization errors.

1. The quantization errors due to rounding are uniformly distributed random variables in the range $(-\Delta/2, \Delta/2)$ where $\Delta = 2^{-b}$.
2. The $4N$ quantization errors are mutually uncorrelated.
3. The $4N$ quantization errors are uncorrelated with the sequence $\{x(n)\}$.

Since each of the quantization errors has a variance

$$\sigma_e^2 = \frac{\Delta^2}{12} = \frac{2^{-2b}}{12} \tag{6.4.2}$$

the variance of the quantization errors from the $4N$ multiplications is

$$\begin{aligned} \sigma_q^2 &= 4N\sigma_e^2 \\ &= \frac{N}{3} \cdot 2^{-2b} \end{aligned} \tag{6.4.3}$$

as

$$\sigma_q^2 = \frac{2^{-2(b-\nu/2)}}{3} \tag{6.4.4}$$

This expression implies that every fourfold increase in the size N of the DFT requires an additional bit in computational precision to offset the additional quantization errors.

To prevent overflow, the input sequence to the DFT requires scaling. Clearly, an upper bound on $|X(k)|$ is

$$|X(k)| \leq \sum_{n=0}^{N-1} |x(n)| \tag{6.4.5}$$

If the dynamic range in addition is $(-1, 1)$, then $|X(k)| < 1$ requires that

$$\sum_{n=0}^{N-1} |x(n)| < 1 \tag{6.4.6}$$

If $|x(n)|$ is initially scaled such that $|x(n)| < 1$ for all n, then each point in the sequence can be divided by N to ensure that (6.4.6) is satisfied.

The scaling implied by (6.4.6) is extremely severe. For example, suppose that the signal sequence $\{x(n)\}$ is white and, after scaling, each value $|x(n)|$ of the sequence is uniformly distributed in the range $(-1/N, 1/N)$. Then the variance of the signal sequence is

$$\sigma_x^2 = \frac{(2/N)^2}{12} = \frac{1}{3N^2} \tag{6.4.7}$$

and the variance of the output DFT coefficients $|X(k)|$ is

$$\begin{aligned} \sigma_X^2 &= N\sigma_x^2 \\ &= \frac{1}{3N} \end{aligned} \tag{6.4.8}$$

Thus the signal-to-noise power ratio is

$$\frac{\sigma_X^2}{\sigma_q^2} = \frac{2^{2b}}{N^2} \tag{6.4.9}$$

We observe that the scaling is responsible for reducing the SNR by N and the combination of scaling and quantization errors result in a total reduction that is proportional to N^2. Hence scaling the input sequence $\{x(n)\}$ to satisfy (6.4.6) imposes a severe penalty on the signal-to-noise ratio in the DFT.

UNIT-2&3

STRUCTURES OF FIR AND IIR SYSTEMS

STRUCTURES FOR THE REALIZATION OF DISCRETE-TIME SYSTEMS

The major factors that influence our choice o f a specific realization are computational complexity, memory requirements, and finite-word-length effects in the computations.

STRUCTURES FOR FIR SYSTEMS

In general, an FIR system is described by the difference equation

$$y(n) = \sum_{k=0}^{M-1} b_k x(n-k) \qquad (7.2.1)$$

or, equivalently, by the system function

$$H(z) = \sum_{k=0}^{M-1} b_k z^{-k} \qquad (7.2.2)$$

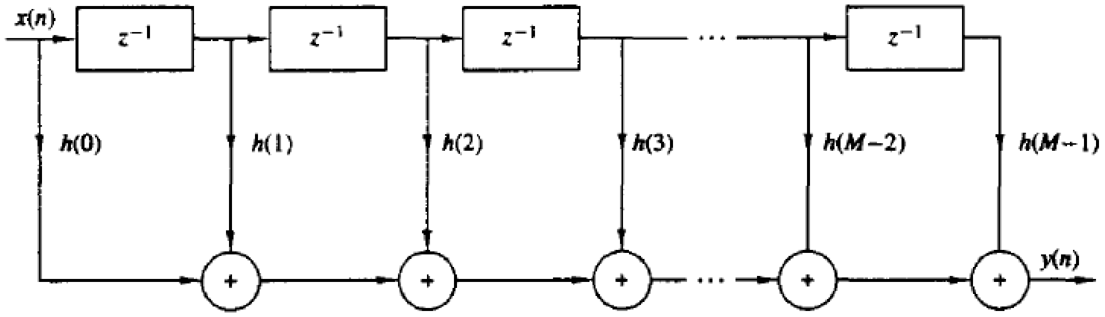
Furthermore, the unit sample response of the FIR system is identical to the coefficients $\{b_k\}$, that is,

$$h(n) = \begin{cases} b_n, & 0 \le n \le M-1 \\ 0, & \text{otherwise} \end{cases}$$

Direct-Form Structure

The direct form realization follows immediately from the non recursive difference equation given below

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k)$$



Cascade-Form Structures

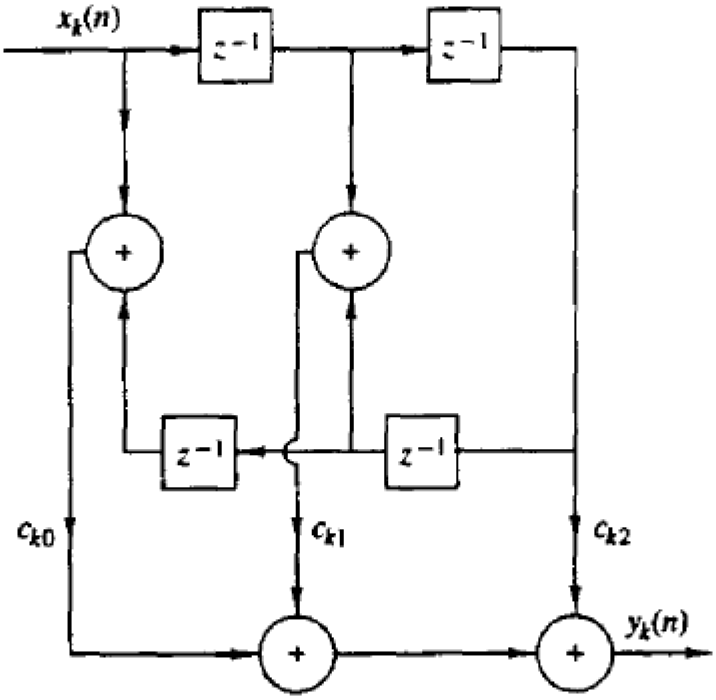
The cascaded realization follows naturally system function given by equation. It is simple matter to factor H(z) into second order FIR system so that

$$H(z) = \prod_{k=1}^{K} H_k(z)$$

where

$$H_k(z) = b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2} \qquad k = 1, 2, \dots, K$$

$$H_k(z) = c_{k0}(1 - z_k z^{-1})(1 - z_k^* z^{-1})(1 - z^{-1}/z_k)(1 - z^{-1}/z_k^*)$$
$$= c_{k0} + c_{k1}z^{-1} + c_{k2}z^{-2} + c_{k1}z^{-3} + z^{-4}$$



Frequency-Sampling Structures

The frequency-sampling realization is an alternative structure for an FIR filter in which the parameters that characterize the filter are the values o f the desired frequency response instead of the impulse response *h(n)*. To derive the frequency sampling structure, we specify the desired frequency response at a set o f equally spaced frequencies, namely

$$\omega_k = \frac{2\pi}{M}(k + \alpha) \qquad k = 0, 1, \ldots, \frac{M-1}{2} \quad M \text{ odd}$$
$$k = 0, 1, \ldots, \frac{M}{2} - 1 \quad M \text{ even}$$
$$\alpha = 0 \text{ or } \tfrac{1}{2}$$

The frequency response of the system is given by

$$H(\omega) = \sum_{n=0}^{M-1} h(n)e^{-j\omega n}$$

$$H(k+\alpha) = H\left(\frac{2\pi}{M}(k+\alpha)\right)$$

$$= \sum_{n=0}^{M-1} h(n)e^{-j2\pi(k+\alpha)n/M} \qquad k = 0, 1, \ldots, M-1$$

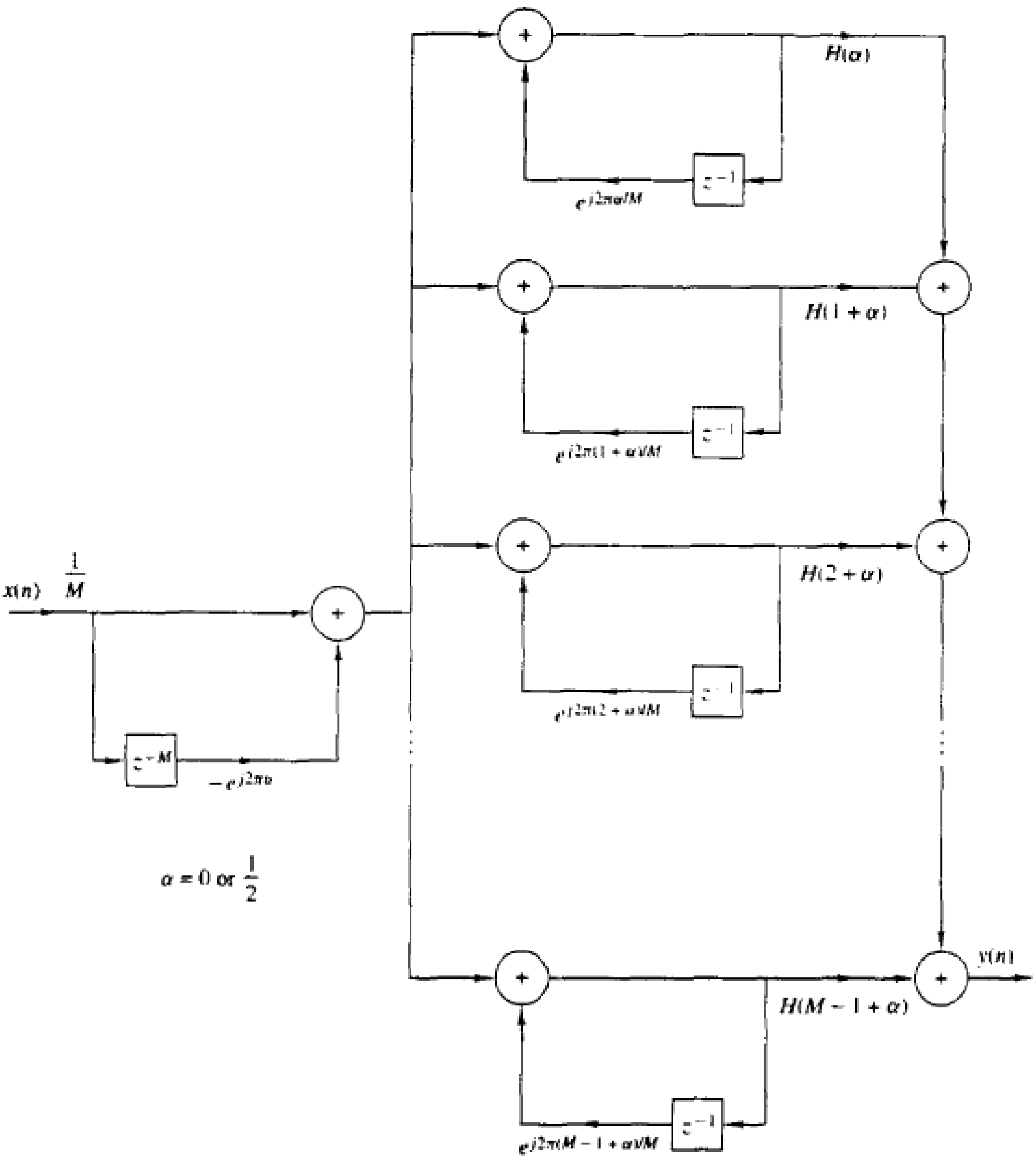
$$h(n) = \frac{1}{M}\sum_{k=0}^{M-1} H(k+\alpha)e^{j2\pi(k+\alpha)n/M} \qquad n = 0, 1, \ldots, M-1$$

$$H(z) = \sum_{n=0}^{M-1} h(n)z^{-n}$$

$$= \sum_{n=0}^{M-1}\left[\frac{1}{M}\sum_{k=0}^{M-1} H(k+\alpha)e^{j2\pi(k+\alpha)n/M}\right]z^{-n}$$

$$H(z) = \sum_{k=0}^{M-1} H(k+\alpha)\left[\frac{1}{M}\sum_{n=0}^{M-1}(e^{j2\pi(k+\alpha)/M}z^{-1})^{n}\right]$$

$$= \frac{1-z^{-M}e^{j2\pi\alpha}}{M}\sum_{k=0}^{M-1}\frac{H(k+\alpha)}{1-e^{j2\pi(k+\alpha)/M}z^{-1}}$$



Lattice Structure

In this section w e introduce another F IR filter structure, called the lattice filter or Lattice realization. Lattice filters are used extensively in digital speech processing And in the implementation of adaptive filters. Let us begin the development by considering a sequence of FIR filters with system functions

$$H_m(z) = A_m(z) \qquad m = 0, 1, 2, \dots, M-1 \tag{7.2.17}$$

where, by definition, $A_m(z)$ is the polynomial

$$A_m(z) = 1 + \sum_{k=1}^{m} \alpha_m(k) z^{-k} \qquad m \geq 1 \tag{7.2.18}$$

and $A_0(z) = 1$. The unit sample response of the mth filter is $h_m(0) = 1$ and $h_m(k) = \alpha_m(k)$, $k = 1, 2, \dots, m$. The subscript m on the polynomial $A_m(z)$ denotes the degree of the polynomial. For mathematical convenience, we define $\alpha_m(0) = 1$.

If $\{x(n)\}$ is the input sequence to the filter $A_m(z)$ and $\{y(n)\}$ is the output sequence, we have

$$y(n) = x(n) + \sum_{k=1}^{m} \alpha_m(k) x(n-k) \tag{7.2.19}$$

Next, let us consider an FIR filter for which $m = 2$. In this case the output from a direct-form structure is

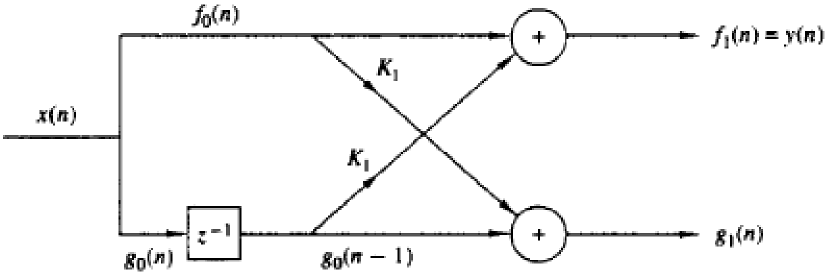
$$y(n) = x(n) + \alpha_2(1)x(n-1) + \alpha_2(2)x(n-2) \tag{7.2.22}$$

By cascading two lattice stages as shown in Fig. 7.10, it is possible to obtain the same output as (7.2.22). Indeed, the output from the first stage is

$$\begin{aligned} f_1(n) &= x(n) + K_1 x(n-1) \\ g_1(n) &= K_1 x(n) + x(n-1) \end{aligned} \tag{7.2.23}$$

The output from the second stage is

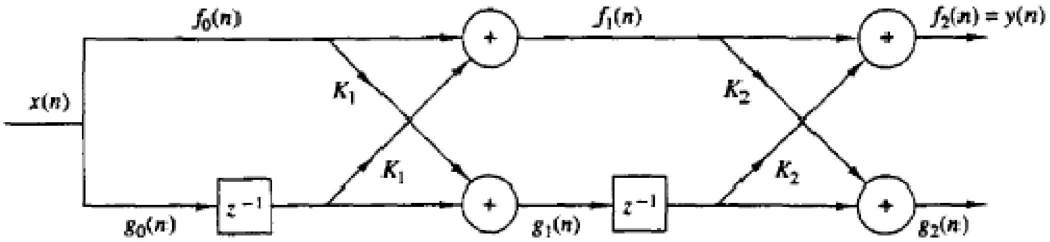
$$\begin{aligned} f_2(n) &= f_1(n) + K_2 g_1(n-1) \\ g_2(n) &= K_2 f_1(n) + g_1(n-1) \end{aligned} \tag{7.2.24}$$



$f_0(n) = g_0(n) = x(n)$

$f_1(n) = f_0(n) + K_1 g_0(n-1) = x(n) + K_1 x(n-1)$

$g_1(n) = K_1 f_0(n) + g_0(n-1) = K_1 x(n) + x(n-1)$



$$\begin{aligned} f_2(n) &= x(n) + K_1 x(n-1) + K_2[K_1 x(n-1) + x(n-2)] \\ &= x(n) + K_1(1+K_2)x(n-1) + K_2 x(n-2) \end{aligned}$$

The general form of lattice structure for m stage is given by'

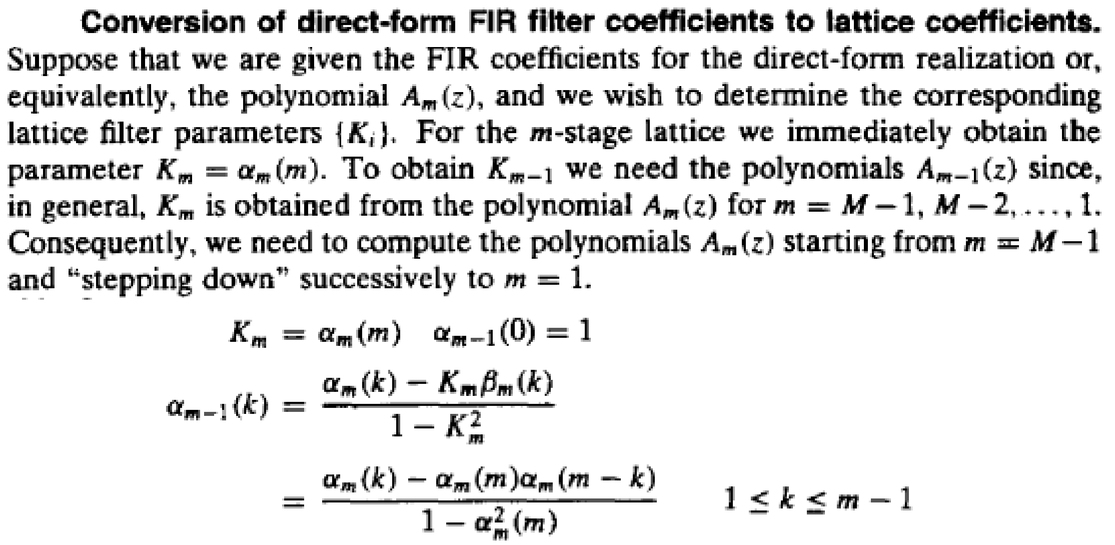
$$\begin{aligned} f_0(n) &= g_0(n) = x(n) \\ f_m(n) &= f_{m-1}(n) + K_m g_{m-1}(n-1) \qquad m = 1, 2, \ldots, M-1 \\ g_m(n) &= K_m f_{m-1}(n) + g_{m-1}(n-1) \qquad m = 1, 2, \ldots, M-1 \end{aligned}$$

Conversion of lattice coefficients to direct-form filter coefficients. The direct-form FIR filter coefficients $\{\alpha_m(k)\}$ can be obtained from the lattice coefficients $\{K_i\}$ by using the following relations:

$$A_0(z) = B_0(z) = 1 \tag{7.2.47}$$

$$A_m(z) = A_{m-1}(z) + K_m z^{-1} B_{m-1}(z) \qquad m = 1, 2, \ldots, M-1 \tag{7.2.48}$$

$$B_m(z) = z^{-m} A_m(z^{-1}) \qquad m = 1, 2, \ldots, M-1 \tag{7.2.49}$$



Conversion of direct-form FIR filter coefficients to lattice coefficients. Suppose that we are given the FIR coefficients for the direct-form realization or, equivalently, the polynomial $A_m(z)$, and we wish to determine the corresponding lattice filter parameters $\{K_i\}$. For the m-stage lattice we immediately obtain the parameter $K_m = \alpha_m(m)$. To obtain K_{m-1} we need the polynomials $A_{m-1}(z)$ since, in general, K_m is obtained from the polynomial $A_m(z)$ for $m = M-1, M-2, \ldots, 1$. Consequently, we need to compute the polynomials $A_m(z)$ starting from $m = M-1$ and "stepping down" successively to $m = 1$.

$$K_m = \alpha_m(m) \quad \alpha_{m-1}(0) = 1$$

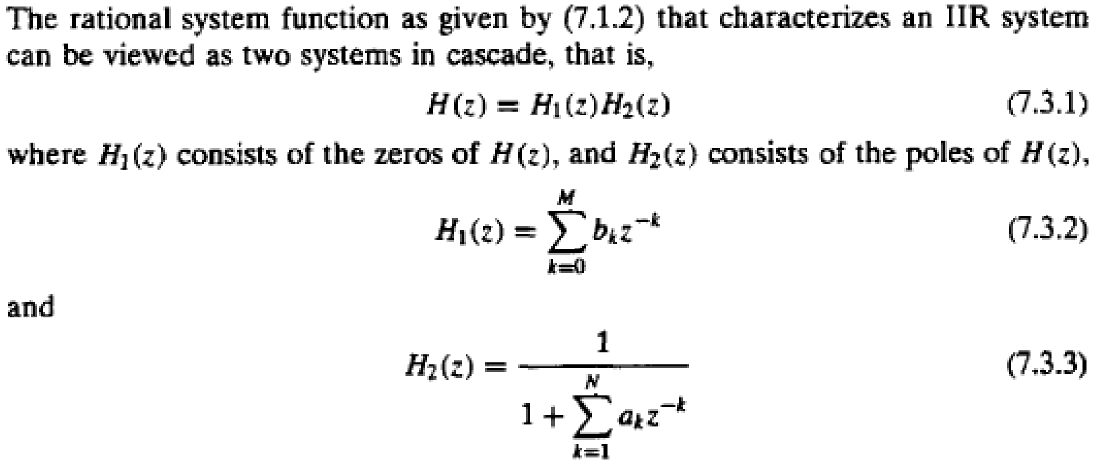
$$\alpha_{m-1}(k) = \frac{\alpha_m(k) - K_m\beta_m(k)}{1 - K_m^2}$$

$$= \frac{\alpha_m(k) - \alpha_m(m)\alpha_m(m-k)}{1 - \alpha_m^2(m)} \qquad 1 \leq k \leq m-1$$

STRUCTURES FOR IIR SYSTEMS

In this section we consider different IIR system s structures described by the difference equation given by the system function. Just as in the case o f FIR system s, there are several types o f structures or realizations, including direct-form structures, cascade-form structures, lattice structures, and lattice-ladder structures. In addition, IIR systems lend them selves to a parallel form realization. We begin by describing two direct-form realizations.

DIRECT FORM STRUCTURES:



The rational system function as given by (7.1.2) that characterizes an IIR system can be viewed as two systems in cascade, that is,

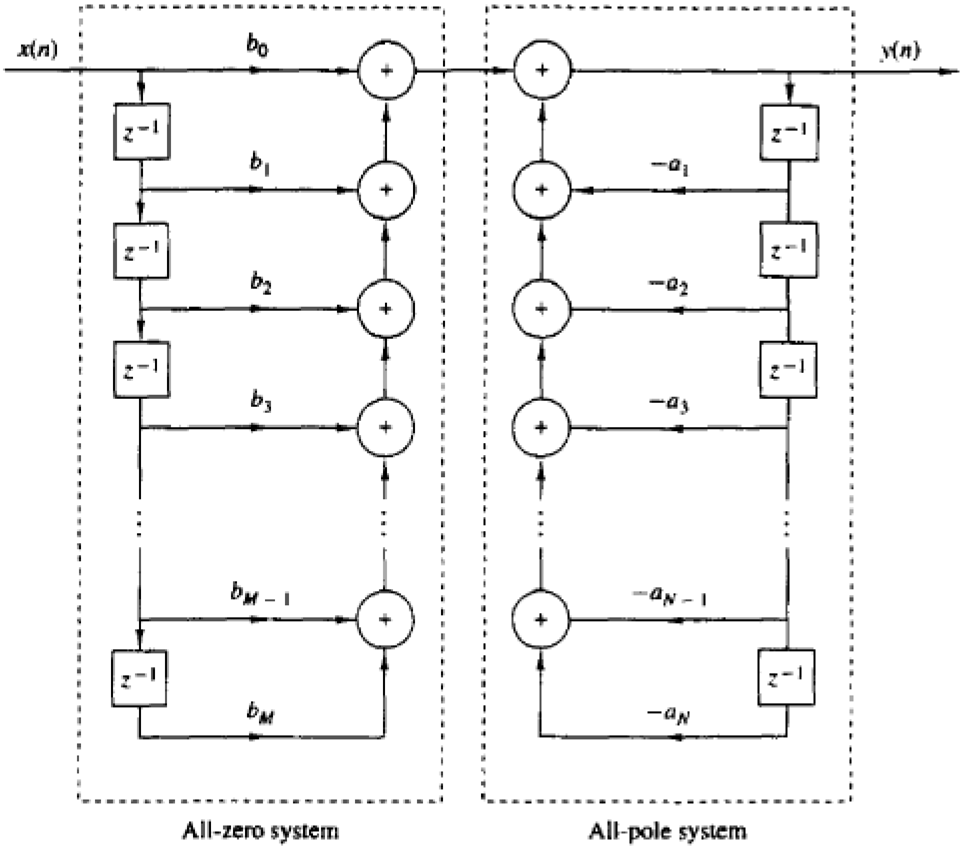
$$H(z) = H_1(z)H_2(z) \tag{7.3.1}$$

where $H_1(z)$ consists of the zeros of $H(z)$, and $H_2(z)$ consists of the poles of $H(z)$,

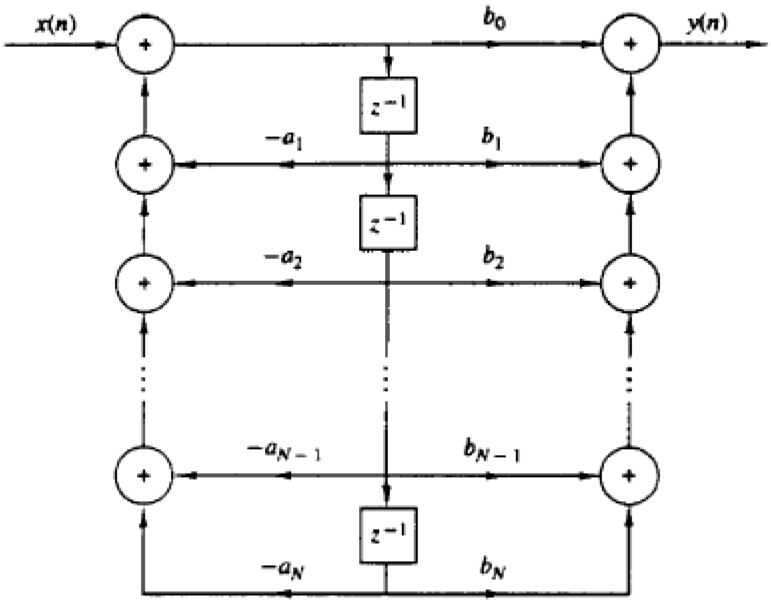
$$H_1(z) = \sum_{k=0}^{M} b_k z^{-k} \tag{7.3.2}$$

and

$$H_2(z) = \frac{1}{1 + \sum_{k=1}^{N} a_k z^{-k}} \tag{7.3.3}$$

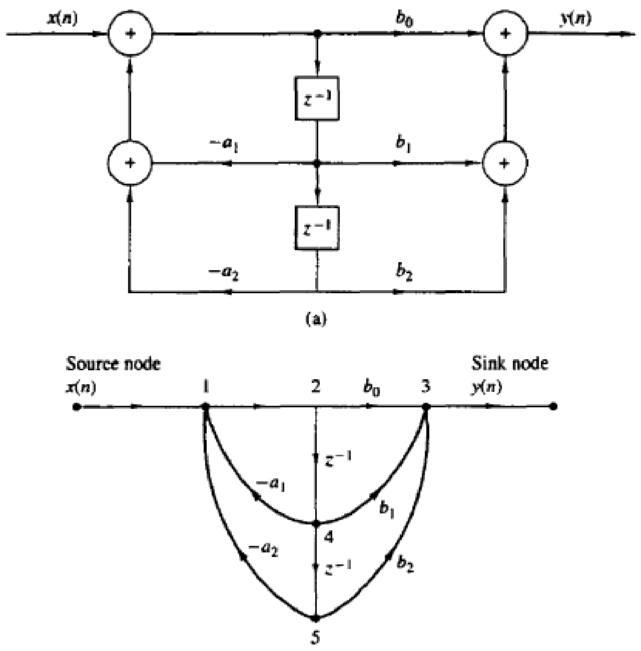


DIRECT FORM II



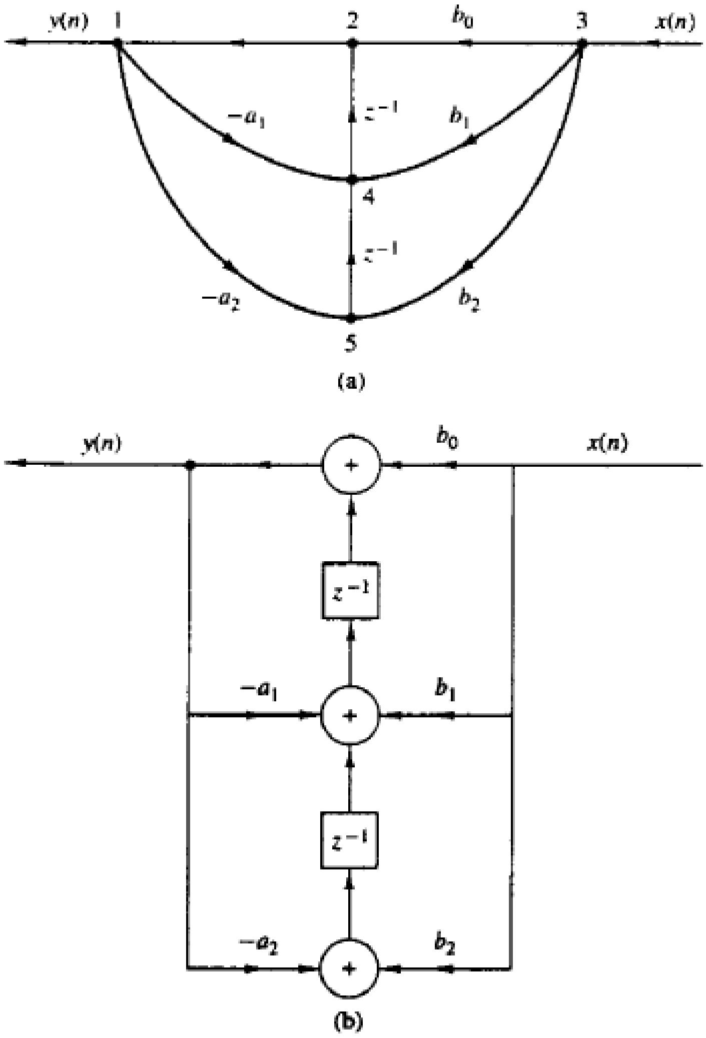
Signal Flow Graphs and Transposed Structures

A signal flow graph provides an alternative, N but equivalent, graphical representation to a block diagram structure that we have been using to illustrate various system realizations. T he basic elements o f a flow graph are branches and nodes. A signal flow graph is basically a set o f directed branches that connect at nodes. By definition, the signal out of a branch is equal to the branch gain (system function) times the signal into the branch. Furthermore, the signal at anode o f a flow graph is equal to the sum o f the signals from all branches connecting to the node.



(a)

(b)



(a)

(b)

Cascade-Form Structures

Let us consider a high-order IIR system with system function given by equation. Without loss of generality we assume that $N > M$. The system can be factored into a cascade of second-order subsystems, such that $H(z)$ can be expressed as

$$H(z) = \prod_{k=1}^{K} H_k(z)$$

where K is the integer part of $(N + 1)/2$. $H_k(z)$ has the general form

$$H_k(z) = \frac{b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}}$$

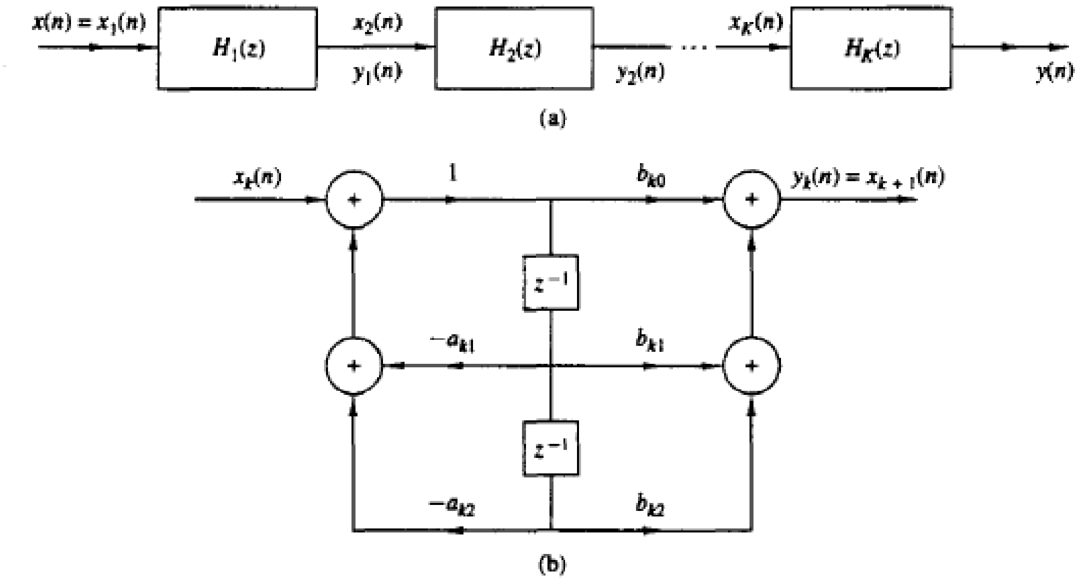
The general form of the cascade structure is illustrated in Fig. 7.19. If we use the direct form II structure for each of the subsystems, the computational algorithm for realizing the IIR system with system function $H(z)$ is described by the following set of equations.

$$y_0(n) = x(n) \qquad (7.3.16)$$

$$w_k(n) = -a_{k1}w_k(n-1) - a_{k2}w_k(n-2) + y_{k-1}(n) \qquad k = 1, 2, \ldots, K \qquad (7.3.17)$$

$$y_k(n) = b_{k0}w_k(n) + b_{k1}w_k(n-1) + b_{k2}w_k(n-2) \qquad k = 1, 2, \ldots, K \qquad (7.3.18)$$

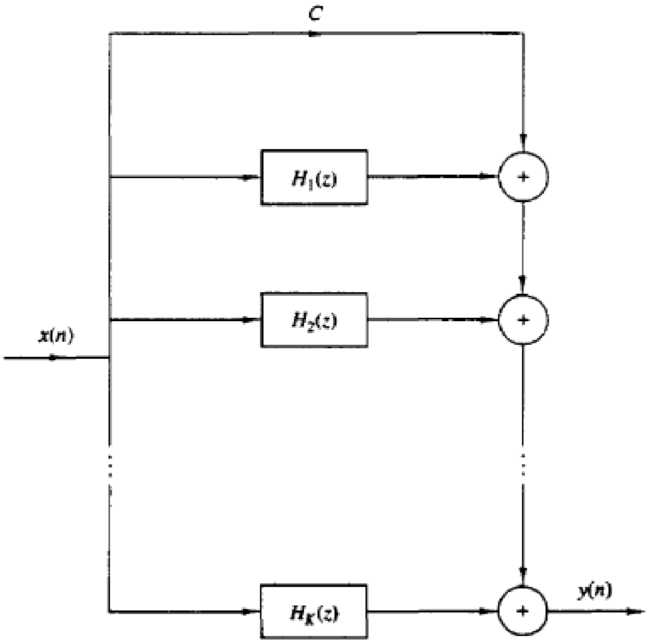
$$y(n) = y_K(n) \qquad (7.3.19)$$



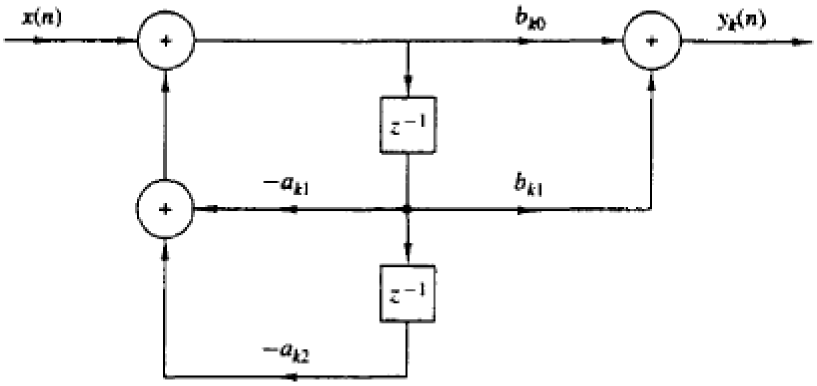
Parallel-Form Structures

A parallel-form realization o f an IIR system can be obtained by performing a partial-fraction expansion o f *H(z)* . Without loss o f generality, w e again assume that *N* > *M* and that the poles are distinct. Then, by performing a partial-fraction expansion o f *H(z)*, we obtain the result

$$H(z) = C + \sum_{k=1}^{N} \frac{A_k}{1 - p_k z^{-1}}$$



The realization of second order form is given by



The general form of parallel form of structure is f\given by

$$w_k(n) = -a_{k1}w_k(n-1) - a_{k2}w_k(n-2) + x(n) \qquad k = 1, 2, \ldots, K$$

$$y_k(n) = b_{k0}w_k(n) + b_{k1}w_k(n-1) \qquad k = 1, 2, \ldots, K$$

$$y(n) = Cx(n) + \sum_{k=1}^{K} y_k(n)$$

Lattice and Lattice-Ladder Structures for IIR Systems

In Section 7.2.4 we developed a lattice filter structure that is equivalent to an FIR system. In this section we extend the development to IIR systems.

Let us begin with an all-pole system with system function

$$H(z) = \frac{1}{1 + \sum_{k=1}^{N} a_N(k)z^{-k}} = \frac{1}{A_N(z)} \tag{7.3.26}$$

The direct form realization of this system is illustrated in Fig. 7.23. The difference equation for this IIR system is

$$y(n) = -\sum_{k=1}^{N} a_N(k)y(n-k) + x(n) \tag{7.3.27}$$

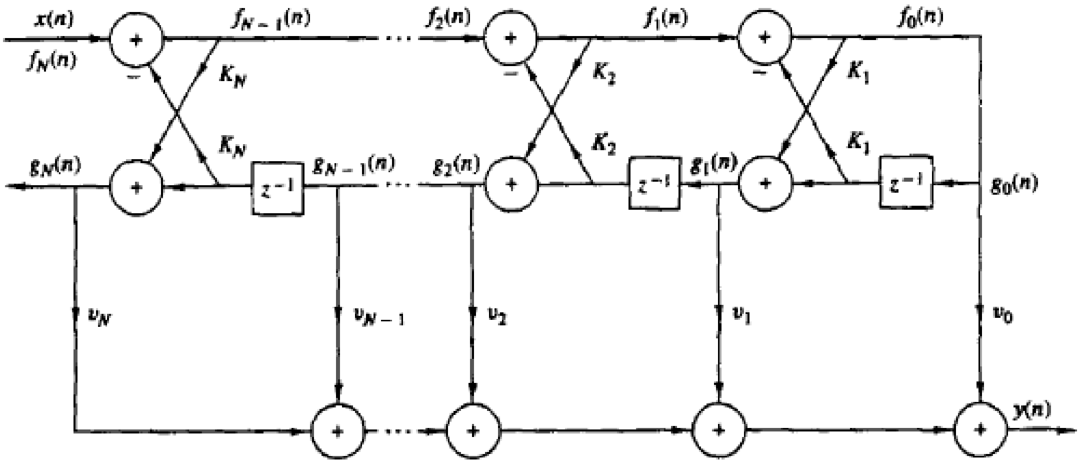
It is interesting to note that if we interchange the roles of input and output [i.e., interchange $x(n)$ with $y(n)$ in (7.3.27)], we obtain

$$x(n) = -\sum_{k=1}^{N} a_N(k)x(n-k) + y(n)$$

or, equivalently,

$$y(n) = x(n) + \sum_{k=1}^{N} a_N(k)x(n-k) \tag{7.3.28}$$

We note that the equation in (7.3.28) describes an FIR system having the system function $H(z) = A_N(z)$, while the system described by the difference equation in (7.3.27) represents an IIR system with system function $H(z) = 1/A_N(z)$.



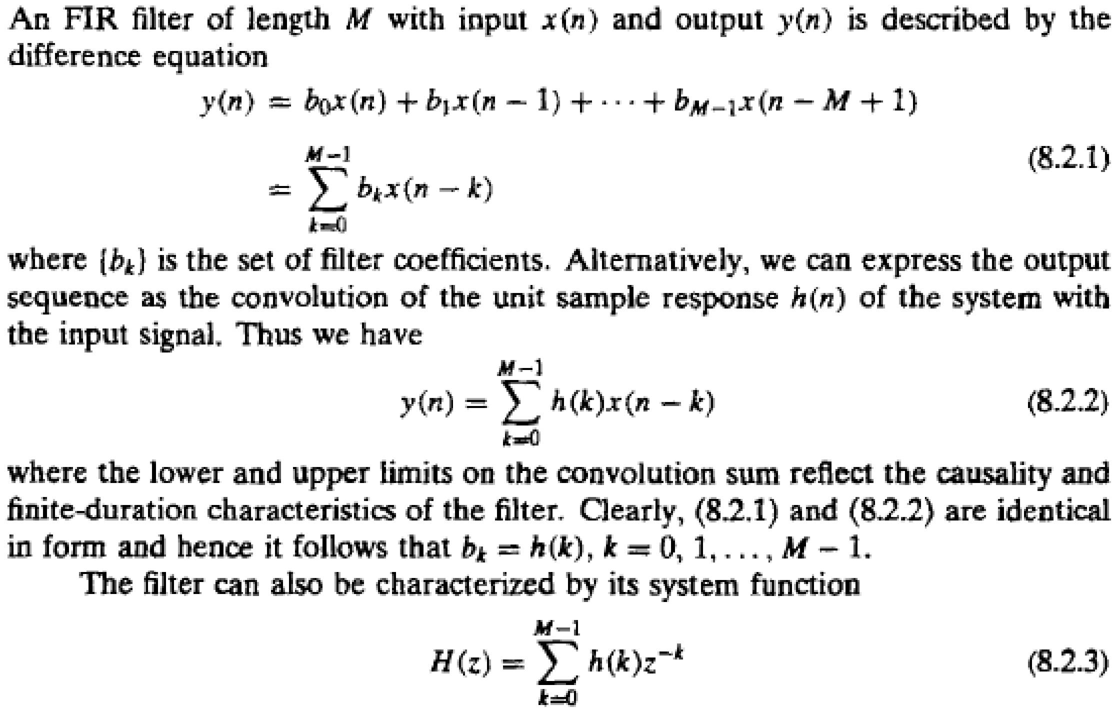
IIR AND FIR FILTERS

The transfer function is obtained by taking Z transform of finite sample impulse response. The filters designed by using finite samples of impulse response are called FIR filters.

Some of the advantages of FIR filter are linear phase, both recursive and non recursive, stable and round off noise can be made smaller.

Some of the disadvantages of FIR filters are large amount of processing is required and non integral delay may lead to problems.

DESIGN OF FIR FILTERS



An FIR filter of length M with input $x(n)$ and output $y(n)$ is described by the difference equation

$$\begin{aligned} y(n) &= b_0 x(n) + b_1 x(n-1) + \cdots + b_{M-1} x(n-M+1) \\ &= \sum_{k=0}^{M-1} b_k x(n-k) \end{aligned} \tag{8.2.1}$$

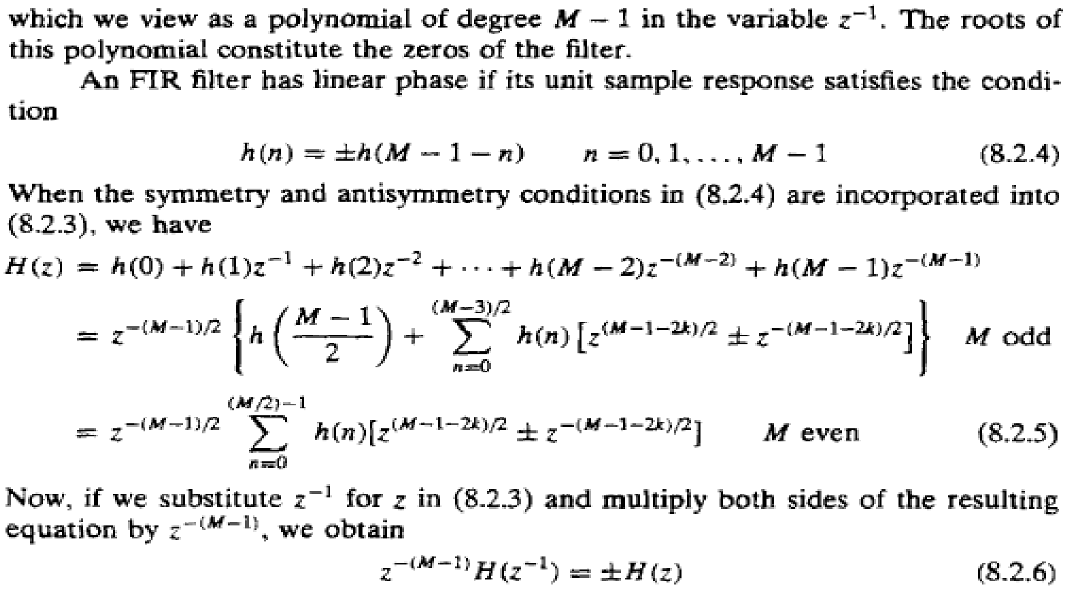
where $\{b_k\}$ is the set of filter coefficients. Alternatively, we can express the output sequence as the convolution of the unit sample response $h(n)$ of the system with the input signal. Thus we have

$$y(n) = \sum_{k=0}^{M-1} h(k) x(n-k) \tag{8.2.2}$$

where the lower and upper limits on the convolution sum reflect the causality and finite-duration characteristics of the filter. Clearly, (8.2.1) and (8.2.2) are identical in form and hence it follows that $b_k = h(k)$, $k = 0, 1, \ldots, M-1$.

The filter can also be characterized by its system function

$$H(z) = \sum_{k=0}^{M-1} h(k) z^{-k} \tag{8.2.3}$$



which we view as a polynomial of degree $M-1$ in the variable z^{-1}. The roots of this polynomial constitute the zeros of the filter.

An FIR filter has linear phase if its unit sample response satisfies the condition

$$h(n) = \pm h(M-1-n) \qquad n = 0, 1, \ldots, M-1 \tag{8.2.4}$$

When the symmetry and antisymmetry conditions in (8.2.4) are incorporated into (8.2.3), we have

$$\begin{aligned} H(z) &= h(0) + h(1)z^{-1} + h(2)z^{-2} + \cdots + h(M-2)z^{-(M-2)} + h(M-1)z^{-(M-1)} \\ &= z^{-(M-1)/2} \left\{ h\left(\frac{M-1}{2}\right) + \sum_{n=0}^{(M-3)/2} h(n) \left[z^{(M-1-2k)/2} \pm z^{-(M-1-2k)/2} \right] \right\} \quad M \text{ odd} \\ &= z^{-(M-1)/2} \sum_{n=0}^{(M/2)-1} h(n) [z^{(M-1-2k)/2} \pm z^{-(M-1-2k)/2}] \qquad M \text{ even} \end{aligned} \tag{8.2.5}$$

Now, if we substitute z^{-1} for z in (8.2.3) and multiply both sides of the resulting equation by $z^{-(M-1)}$, we obtain

$$z^{-(M-1)} H(z^{-1}) = \pm H(z) \tag{8.2.6}$$

When $h(n) = h(M-1-n)$, $H(\omega)$ can be expressed as

$$H(\omega) = H_r(\omega)e^{-j\omega(M-1)/2} \tag{8.2.7}$$

where $H_r(\omega)$ is a real function of ω and can be expressed as

$$H_r(\omega) = h\left(\frac{M-1}{2}\right) + 2\sum_{n=0}^{(M-3)/2} h(n)\cos\omega\left(\frac{M-1}{2}-n\right) \qquad M \text{ odd} \tag{8.2.8}$$

$$H_r(\omega) = 2\sum_{n=0}^{(M/2)-1} h(n)\cos\omega\left(\frac{M-1}{2}-n\right) \qquad M \text{ even} \tag{8.2.9}$$

The phase characteristic of the filter for both M odd and M even is

$$\Theta(\omega) = \begin{cases} -\omega\left(\dfrac{M-1}{2}\right), & \text{if } H_r(\omega) > 0 \\ -\omega\left(\dfrac{M-1}{2}\right) + \pi, & \text{if } H_r(\omega) < 0 \end{cases} \tag{8.2.10}$$

When

$$h(n) = -h(M-1-n)$$

the unit sample response is *antisymmetric*. For M odd, the center point of the antisymmetric $h(n)$ is $n = (M-1)/2$. Consequently,

$$h\left(\frac{M-1}{2}\right) = 0$$

It is straightforward to show that the frequency response of an FIR filter with an antisymmetric unit sample response can be expressed as

$$H(\omega) = H_r(\omega)e^{j[-\omega(M-1)/2+\pi/2]} \tag{8.2.11}$$

where

$$H_r(\omega) = 2\sum_{n=0}^{(M-3)/2} h(n)\sin\omega\left(\frac{M-1}{2}-n\right) \qquad M \text{ odd} \tag{8.2.12}$$

$$H_r(\omega) = 2\sum_{n=0}^{(M/2)-1} h(n)\sin\omega\left(\frac{M-1}{2}-n\right) \qquad M \text{ even} \tag{8.2.13}$$

The phase characteristic of the filter for both M odd and M even is

$$\Theta(\omega) = \begin{cases} \dfrac{\pi}{2} - \omega\left(\dfrac{M-1}{2}\right), & \text{if } H_r(\omega) > 0 \\ \dfrac{3\pi}{2} - \omega\left(\dfrac{M-1}{2}\right), & \text{if } H_r(\omega) < 0 \end{cases} \tag{8.2.14}$$

The choice of a symmetric or antisymmetric unit sample response depends on the application. As we shall see later, a symmetric unit sample response is suitable for some applications, while an antisymmetric unit sample response is more suitable for other applications. For example, if $h(n) = -h(M-1-n)$ and M is odd, (8.2.12) implies that $H_r(0) = 0$ and $H_r(\pi) = 0$. Consequently, (8.2.12) is not suitable as either a lowpass filter or a highpass filter. Similarly, the antisymmetric unit sample response with M even also results in $H_r(0) = 0$, as can be easily verified from (8.2.13). Consequently, we would not use the antisymmetric condition in the design of a lowpass linear-phase FIR filter. On the other hand, the symmetry condition $h(n) = h(M-1-n)$ yields a linear-phase FIR filter with a nonzero response at $\omega = 0$, if desired, that is,

$$H_r(0) = h\left(\frac{M-1}{2}\right) + 2\sum_{n=0}^{(M-3)/2} h(n), \qquad M \text{ odd} \tag{8.2.15}$$

$$H_r(0) = 2\sum_{n=0}^{(M/2)-1} h(n), \qquad M \text{ even} \tag{8.2.16}$$

8.2.2 Design of Linear-Phase FIR Filters Using Windows

In this method we begin with the desired frequency response specification $H_d(\omega)$ and determine the corresponding unit sample response $h_d(n)$. Indeed, $h_d(n)$ is related to $H_d(\omega)$ by the Fourier transform relation

$$H_d(\omega) = \sum_{n=0}^{\infty} h_d(n)e^{-j\omega n} \tag{8.2.17}$$

where

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(\omega)e^{j\omega n}d\omega \tag{8.2.18}$$

Thus, given $H_d(\omega)$, we can determine the unit sample response $h_d(n)$ by evaluating the integral in (8.2.18).

In general, the unit sample response $h_d(n)$ obtained from (8.2.17) is infinite in duration and must be truncated at some point, say at $n = M - 1$, to yield an FIR filter of length M. Truncation of $h_d(n)$ to a length $M - 1$ is equivalent to multiplying $h_d(n)$ by a "rectangular window," defined as

$$w(n) = \begin{cases} 1, & n = 0, 1, \ldots, M-1 \\ 0, & \text{otherwise} \end{cases} \tag{8.2.19}$$

Thus the unit sample response of the FIR filter becomes

$$\begin{aligned} h(n) &= h_d(n)w(n) \\ &= \begin{cases} h_d(n), & n = 0, 1, \ldots, M-1 \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{8.2.20}$$

It is instructive to consider the effect of the window function on the desired frequency response $H_d(\omega)$. Recall that multiplication of the window function $w(n)$ with $h_d(n)$ is equivalent to convolution of $H_d(\omega)$ with $W(\omega)$, where $W(\omega)$ is the frequency-domain representation (Fourier transform) of the window function, that is,

$$W(\omega) = \sum_{n=0}^{M-1} w(n)e^{-j\omega n} \tag{8.2.21}$$

Thus the convolution of $H_d(\omega)$ with $W(\omega)$ yields the frequency response of the (truncated) FIR filter. That is,

$$H(\omega) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(\nu)W(\omega - \nu)d\nu \tag{8.2.22}$$

The Fourier transform of the rectangular window is

$$\begin{aligned} W(\omega) &= \sum_{n=0}^{M-1} e^{-j\omega n} \\ &= \frac{1 - e^{-j\omega M}}{1 - e^{-j\omega}} = e^{-j\omega(M-1)/2}\frac{\sin(\omega M/2)}{\sin(\omega/2)} \end{aligned} \tag{8.2.23}$$

This window function has a magnitude response

$$|W(\omega)| = \frac{|\sin(\omega M/2)|}{|\sin(\omega/2)|} \qquad -\pi \le \omega \le \pi \tag{8.2.24}$$

and a piecewise linear phase

$$\Theta(\omega) = \begin{cases} -\omega\left(\dfrac{M-1}{2}\right), & \text{when } \sin(\omega M/2) \ge 0 \\ -\omega\left(\dfrac{M-1}{2}\right) + \pi, & \text{when } \sin(\omega M/2) < 0 \end{cases} \tag{8.2.25}$$

The magnitude response of the window function is illustrated in Fig. 8.4 for $M = 31$ and 61. The width of the main lobe [width is measured to the first zero of $W(\omega)$]

Name of window	Time-domain sequence, $h(n), 0 \leq n \leq M-1$
Bartlett (triangular)	$1 - \dfrac{2\left\lvert n - \dfrac{M-1}{2} \right\rvert}{M-1}$
Blackman	$0.42 - 0.5 \cos \dfrac{2\pi n}{M-1} + 0.08 \cos \dfrac{4\pi n}{M-1}$
Hamming	$0.54 - 0.46 \cos \dfrac{2\pi n}{M-1}$
Hanning	$\dfrac{1}{2}\left(1 - \cos \dfrac{2\pi n}{M-1}\right)$
Kaiser	$\dfrac{I_0\left[\alpha\sqrt{\left(\dfrac{M-1}{2}\right)^2 - \left(n - \dfrac{M-1}{2}\right)^2}\right]}{I_0\left[\alpha\left(\dfrac{M-1}{2}\right)\right]}$
Lanczos	$\left\{\dfrac{\sin\left[2\pi\left(n - \dfrac{M-1}{2}\right)\Big/(M-1)\right]}{2\pi\left(n - \dfrac{M-1}{2}\right)\Big/\left(\dfrac{M-1}{2}\right)}\right\}^L \qquad L > 0$
Tukey	$1, \left\lvert n - \dfrac{M-1}{2} \right\rvert \leq \alpha \dfrac{M-1}{2} \qquad 0 < \alpha < 1$ $\dfrac{1}{2}\left[1 + \cos\left(\dfrac{n - (1+\alpha)(M-1)/2}{(1-\alpha)(M-1)/2}\pi\right)\right]$ $\alpha(M-1)/2 \leq \left\lvert n - \dfrac{M-1}{2} \right\rvert \leq \dfrac{M-1}{2}$

8.2.3 Design of Linear-Phase FIR Filters by the Frequency-Sampling Method

In the frequency sampling method for FIR filter design, we specify the desired frequency response $H_d(\omega)$ at a set of equally spaced frequencies, namely

$$\begin{aligned} \omega_k = \tfrac{2\pi}{M}(k + \alpha) \qquad & k = 0, 1, \ldots, \frac{M-1}{2} \quad M \text{ odd} \\ & k = 0, 1, \ldots, \frac{M}{2} - 1 \quad M \text{ even} \\ & \alpha = 0 \quad \text{or} \quad \tfrac{1}{2} \end{aligned} \tag{8.2.30}$$

and solve for the unit sample response $h(n)$ of the FIR filter from these equally spaced frequency specifications. To reduce sidelobes, it is desirable to optimize the frequency specification in the transition band of the filter. This optimization can be accomplished numerically on a digital computer by means of linear programming techniques as shown by Rabiner et al. (1970).

In this section we exploit a basic symmetry property of the sampled frequency response function to simplify the computations. Let us begin with the desired frequency response of the FIR filter, which is [for simplicity, we drop the subscript in $H_d(\omega)$],

$$H(\omega) = \sum_{n=0}^{M-1} h(n) e^{-j\omega n} \tag{8.2.31}$$

Suppose that we specify the frequency response of the filter at the frequencies given by (8.2.30). Then from (8.2.31) we obtain

$$H(k+\alpha) \equiv H\left(\frac{2\pi}{M}(k+\alpha)\right)$$

$$H(k+\alpha) \equiv \sum_{n=0}^{M-1} h(n) e^{-j2\pi(k+\alpha)n/M} \qquad k = 0, 1, \ldots, M-1 \tag{8.2.32}$$

It is a simple matter to invert (8.2.32) and express $h(n)$ in terms of $H(k+\alpha)$. If we multiply both sides of (8.2.32) by the exponential, $\exp(j2\pi km/M)$, $m = 0, 1, \ldots, M-1$, and sum over $k = 0, 1, \ldots, M-1$, the right-hand side of (8.2.32) reduces to $Mh(m)\exp(-j2\pi\alpha m/M)$. Thus we obtain

$$h(n) = \frac{1}{M}\sum_{k=0}^{M-1} H(k+\alpha) e^{j2\pi(k+\alpha)n/M} \qquad n = 0, 1, \ldots, M-1 \tag{8.2.33}$$

The relationship in (8.2.33) allows us to compute the values of the unit sample response $h(n)$ from the specification of the frequency samples $H(k+\alpha)$, $k = 0, 1, \ldots, M-1$. Note that when $\alpha = 0$, (8.2.32) reduces to the discrete Fourier transform (DFT) of the sequence $\{h(n)\}$ and (8.2.33) reduces to the inverse DFT (IDFT).

Since $\{h(n)\}$ is real, we can easily show that the frequency samples $\{H(k+\alpha)\}$ satisfy the symmetry condition

$$H(k+\alpha) = H^*(M-k-\alpha) \tag{8.2.34}$$

This symmetry condition, along with the symmetry conditions for $\{h(n)\}$, can be used to reduce the frequency specifications from M points to $(M+1)/2$ points for M odd and $M/2$ points for M even. Thus the linear equations for determining $\{h(n)\}$ from $\{H(k+\alpha)\}$ are considerably simplified.

In particular, if (8.2.11) is sampled at the frequencies $\omega_k = 2\pi(k+\alpha)/M$, $k = 0, 1, \ldots, M-1$, we obtain

$$H(k+\alpha) = H_r\left(\frac{2\pi}{M}(k+\alpha)\right) e^{j[\beta\pi/2 - 2\pi(k+\alpha)(M-1)/2M]} \tag{8.2.35}$$

where $\beta = 0$ when $\{h(n)\}$ is symmetric and $\beta = 1$ when $\{h(n)\}$ is antisymmetric. A simplication occurs by defining a set of real frequency samples $\{G(k+m)\}$

$$G(k+\alpha) = (-1)^k H_r\left(\frac{2\pi}{M}(k+\alpha)\right) \qquad k = 0, 1, \ldots, M-1 \tag{8.2.36}$$

We use (8.2.36) in (8.2.35) to eliminate $H_r(\omega_k)$. Thus we obtain

$$H(k+\alpha) = G(k+\alpha)e^{j\pi k}e^{j[\beta\pi/2 - 2\pi(k+\alpha)(M-1)/2M]} \tag{8.2.37}$$

Now the symmetry condition for $H(k+\alpha)$ given in (8.2.34) translates into a corresponding symmetry condition for $G(k+\alpha)$, which can be exploited by substituting into (8.2.33), to simplify the expressions for the FIR filter impulse response $\{h(n)\}$ for the four cases $\alpha = 0$, $\alpha = \frac{1}{2}$, $\beta = 0$, and $\beta = 1$. The results are summarized in Table 8.3. The detailed derivations are left as exercises for the reader.

8.2.4 Design of Optimum Equiripple Linear-Phase FIR Filters

The window method and the frequency-sampling method are relatively simple techniques for designing linear-phase FIR filters. However, they also possess some minor disadvantages, described in Section 8.2.6, which may render them undesirable for some applications. A major problem is the lack of precise control of the critical frequencies such as ω_p and ω_s.

The filter design method described in this section is formulated as a Chebyshev approximation problem. It is viewed as an optimum design criterion in the sense that the weighted approximation error between the desired frequency response and the actual frequency response is spread evenly across the passband and evenly across the stopband of the filter minimizing the maximum error. The resulting filter designs have ripples in both the passband and the stopband.

To describe the design procedure, let us consider the design of a lowpass filter with passband edge frequency ω_p and stopband edge frequency ω_s. From the general specifications given in Fig. 8.2, in the passband, the filter frequency response satisfies the condition

$$1 - \delta_1 \le H_r(\omega) \le 1 + \delta_1 \qquad |\omega| \le \omega_p \tag{8.2.43}$$

Similarly, in the stopband, the filter frequency response is specified to fall between the limits $\pm\delta_2$, that is,

$$-\delta_2 \le H_r(\omega) \le \delta_2 \qquad |\omega| > \omega_s \tag{8.2.44}$$

Thus δ_1 represents the ripple in the passband and δ_2 represents the attenuation or ripple in the stopband. The remaining filter parameter is M, the filter length or the number of filter coefficients.

Case 1: Symmetric unit sample response $h(n) = h(M-1-n)$ and M Odd. In this case, the real-valued frequency response characteristic $H_r(\omega)$ is

$$H_r(\omega) = h\left(\frac{M-1}{2}\right) + 2\sum_{n=0}^{(M-3)/2} h(n)\cos\omega\left(\frac{M-1}{2} - n\right) \tag{8.2.45}$$

If we let $k = (M-1)/2 - n$ and define a new set of filter parameters $\{a(k)\}$ as

$$a(k) = \begin{cases} h\left(\dfrac{M-1}{2}\right), & k = 0 \\ 2h\left(\dfrac{M-1}{2} - k\right), & k = 1, 2, \ldots, \dfrac{M-1}{2} \end{cases} \tag{8.2.46}$$

then (8.2.45) reduces to the compact form

$$H_r(\omega) = \sum_{k=0}^{(M-1)/2} a(k)\cos\omega k \tag{8.2.47}$$

Case 2: Symmetric unit sample response $h(n) = h(M-1-n)$ and M Even. In this case, $H_r(\omega)$ is expressed as

$$H_r(\omega) = 2\sum_{n=0}^{(M/2)-1} h(n)\cos\omega\left(\frac{M-1}{2} - n\right) \tag{8.2.48}$$

Again, we change the summation index from n to $k = M/2 - n$ and define a new set of filter parameters $\{b(k)\}$ as

$$b(k) = 2h\left(\frac{M}{2} - k\right), k = 1, 2, \ldots, M/2 \tag{8.2.49}$$

With these substitutions (8.2.48) becomes

$$H_r(\omega) = \sum_{k=1}^{M/2} b(k)\cos\omega\left(k - \frac{1}{2}\right) \tag{8.2.50}$$

In carrying out the optimization, it is convenient to rearrange (8.2.50) further into the form

$$H_r(\omega) = \cos\frac{\omega}{2}\sum_{k=0}^{(M/2)-1} \tilde{b}(k)\cos\omega k \tag{8.2.51}$$

where the coefficients $\{\tilde{b}(k)\}$ are linearly related to the coefficients $\{b(k)\}$. In fact, it can be shown that the relationship is

$$\begin{aligned} \tilde{b}(0) &= \tfrac{1}{2}b(1) \\ \tilde{b}(k) &= 2b(k) - \tilde{b}(k-1) \qquad k = 1, 2, 3, \ldots, \frac{M}{2} - 2 \\ \tilde{b}\left(\frac{M}{2} - 1\right) &= 2b\left(\frac{M}{2}\right) \end{aligned} \tag{8.2.52}$$

Case 3: Antisymmetric unit sample response $h(n) = -h(M-1-n)$ and M Odd. The real-valued frequency response characteristic $H_r(\omega)$ for this case is

$$H_r(\omega) = 2 \sum_{n=0}^{(M-3)/2} h(n) \sin \omega \left(\frac{M-1}{2} - n \right) \tag{8.2.53}$$

If we change the summation in (8.2.53) from n to $k = (M-1)/2 - n$ and define a new set of filter parameters $\{c(k)\}$ as

$$c(k) = 2h\left(\frac{M-1}{2} - k \right) \qquad k = 1, 2, \ldots, (M-1)/2 \tag{8.2.54}$$

then (8.2.53) becomes

$$H_r(\omega) = \sum_{k=1}^{(M-1)/2} c(k) \sin \omega k \tag{8.2.55}$$

As in the previous case, it is convenient to rearrange (8.2.55) into the form

$$H_r(\omega) = \sin \omega \sum_{k=0}^{(M-3)/2} \tilde{c}(k) \cos \omega k \tag{8.2.56}$$

Case 4: Antisymmetric unit sample response $h(n) = -h(M-1-n)$ and M Even. In this case, the real-valued frequency response characteristic $H_r(\omega)$ is

$$H_r(\omega) = 2 \sum_{n=0}^{(M/2)-1} h(n) \sin \omega \left(\frac{M-1}{2} - n \right) \tag{8.2.58}$$

A change in the summation index from n to $k = M/2-n$ combined with a definition of a new set of filter coefficients $\{d(k)\}$, related to $\{h(n)\}$ according to

$$d(k) = 2h\left(\frac{M}{2} - k \right) \qquad k = 1, 2, \ldots, \frac{M}{2} \tag{8.2.59}$$

results in the expression

$$H_r(\omega) = \sum_{k=1}^{M/2} d(k) \sin \omega \left(k - \frac{1}{2} \right) \tag{8.2.60}$$

As in the previous two cases, we find it convenient to rearrange (8.2.60) into the form

$$H_r(\omega) = \sin \frac{\omega}{2} \sum_{k=0}^{(M/2)-1} \tilde{d}(k) \cos \omega k \tag{8.2.61}$$

Filter type	$Q(\omega)$	$P(\omega)$
$h(n) = h(M-1-n)$ M odd (case 1)	1	$\sum_{k=0}^{(M-1)/2} a(k)\cos\omega k$
$h(n) = h(M-1-n)$ M even (case 2)	$\cos\frac{\omega}{2}$	$\sum_{k=0}^{(M/2)-1} \tilde{b}(k)\cos\omega k$
$h(n) = -h(M-1-n)$ M odd (case 3)	$\sin\omega$	$\sum_{k=0}^{(M-3)/2} \tilde{c}(k)\cos\omega k$
$h(n) = -h(M-1-n)$ M even (case 4)	$\sin\frac{\omega}{2}$	$\sum_{k=0}^{(M/2)-1} \tilde{d}(k)\cos\omega k$

IIR FILTER DESIGN

DESIGN OF IIR FILTERS FROM ANALOG FILTERS

Just as in the design of FIR filters, there are several methods that can be used to design digital filters having an infinite-duration unit sample response. The techniques described in this section are all based on converting an analog filter into a digital filter. Analog filter design is a mature and well developed field, so it is not surprising that we begin the design of a digital filter in the analog domain and then convert the design into the digital domain.

An analog filter can be described by its system function.

$$H_a(s) = \frac{B(s)}{A(s)} = \frac{\sum_{k=0}^{M} \beta_k s^k}{\sum_{k=0}^{N} \alpha_k s^k} \tag{8.3.1}$$

where $\{\alpha_k\}$ and $\{\beta_k\}$ are the filter coefficients, or by its impulse response, which is related to $H_a(s)$ by the Laplace transform

$$H_a(s) = \int_{-\infty}^{\infty} h(t)e^{-st}\,dt \tag{8.3.2}$$

Alternatively, the analog filter having the rational system function $H(s)$ given in (8.3.1), can be described by the linear constant-coefficient differential equation

$$\sum_{k=0}^{N} \alpha_k \frac{d^k y(t)}{dt^k} = \sum_{k=0}^{M} \beta_k \frac{d^k x(t)}{dt^k} \tag{8.3.3}$$

where $x(t)$ denotes the input signal and $y(t)$ denotes the output of the filter.

Each of these three equivalent characterizations of an analog filter leads to alternative methods for converting the filter into the digital domain, as will be described in Sections 8.3.1 through 8.3.4. We recall that an analog linear time-invariant system with system function $H(s)$ is stable if all its poles lie in the left half of the s-plane. Consequently, if the conversion technique is to be effective, it should possess the following desirable properties:

1. The $j\Omega$ axis in the s-plane should map into the unit circle in the z-plane. Thus there will be a direct relationship between the two frequency variables in the two domains.

2. The left-half plane (LHP) of the s-plane should map into the inside of the unit circle in the z-plane. Thus a stable analog filter will be converted to a stable digital filter.

We mentioned in the preceding section that physically realizable and stable IIR filters cannot have linear phase. Recall that a linear-phase filter must have a system function that satisfies the condition

$$H(z) = \pm z^{-N} H(z^{-1}) \tag{8.3.4}$$

where z^{-N} represents a delay of N units of time. But if this were the case, the filter would have a mirror-image pole outside the unit circle for every pole inside the unit circle. Hence the filter would be unstable. Consequently, a causal and stable IIR filter cannot have linear phase.

If the restriction on physical realizability is removed, it is possible to obtain a linear-phase IIR filter, at least in principle. This approach involves performing a time reversal of the input signal $x(n)$, passing $x(-n)$ through a digital filter $H(z)$, time-reversing the output of $H(z)$, and finally, passing the result through $H(z)$ again. This signal processing is computationally cumbersome and appears to offer no advantages over linear-phase FIR filters. Consequently, when an application requires a linear-phase filter, it should be an FIR filter.

In the design of IIR filters, we shall specify the desired filter characteristics for the magnitude response only. This does not mean that we consider the phase response unimportant. Since the magnitude and phase characteristics are related, as indicated in Section 8.1, we specify the desired magnitude characteristics and accept the phase response that is obtained from the design methodology.

8.3.1 IIR Filter Design by Approximation of Derivatives

One of the simplest methods for converting an analog filter into a digital filter is to approximate the differential equation in (8.3.3) by an equivalent difference equation. This approach is often used to solve a linear constant-coefficient differential equation numerically on a digital computer.

For the derivative $dy(t)/dt$ at time $t = nT$, we substitute the *backward difference* $[y(nT) - y(nT - 1)]/T$. Thus

$$\begin{aligned} \left.\frac{dy(t)}{dt}\right|_{t=nT} &= \frac{y(nT) - y(nT - T)}{T} \\ &= \frac{y(n) - y(n-1)}{T} \end{aligned} \tag{8.3.5}$$

where T represents the sampling interval and $y(n) \equiv y(nT)$. The analog differentiator with output $dy(t)/dt$ has the system function $H(s) = s$, while the digital system that produces the output $[y(n) - y(n-1)]/T$ has the system function $H(z) = (1 - z^{-1})/T$. Consequently, as shown in Fig. 8.29, the frequency-domain

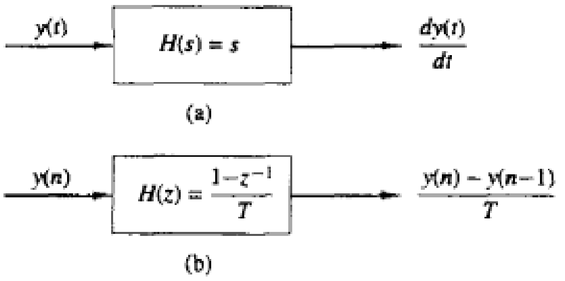


Figure 8.29 Substitution of the backward difference for the derivative implies the mapping $s = (1 - z^{-1})/T$.

equivalent for the relationship in (8.3.5) is

$$s = \frac{1 - z^{-1}}{T} \tag{8.3.6}$$

The second derivative $d^2y(t)/dt^2$ is replaced by the second difference, which is derived as follows:

$$\begin{aligned}\left.\frac{d^2y(t)}{dt^2}\right|_{t=nT} &= \frac{d}{dt}\left[\frac{dy(t)}{dt}\right]_{t=nT}\\ &= \frac{[y(nT) - y(nT - T)]/T - [y(nT - T) - y(nT - 2T)]/T}{T}\\ &= \frac{y(n) - 2y(n-1) + y(n-2)}{T^2}\end{aligned} \tag{8.3.7}$$

In the frequency domain, (8.3.7) is equivalent to

$$s^2 = \frac{1 - 2z^{-1} + z^{-2}}{T^2} = \left(\frac{1 - z^{-1}}{T}\right)^2 \tag{8.3.8}$$

It easily follows from the discussion that the substitution for the kth derivative of $y(t)$ results in the equivalent frequency-domain relationship

$$s^k = \left(\frac{1 - z^{-1}}{T}\right)^k \tag{8.3.9}$$

Consequently, the system function for the digital IIR filter obtained as a result of the approximation of the derivatives by finite differences is

$$H(z) = H_a(s)|_{s=(1-z^{-1})/T} \tag{8.3.10}$$

where $H_a(s)$ is the system function of the analog filter characterized by the differential equation given in (8.3.3).

Let us investigate the implications of the mapping from the s-plane to the z-plane as given by (8.3.6) or, equivalently,

$$z = \frac{1}{1 - sT} \tag{8.3.11}$$

If we substitute $s = j\Omega$ in (8.2.11), we find that

$$z = \frac{1}{1 - j\Omega T}$$

$$= \frac{1}{1+\Omega^2 T^2} + j\frac{\Omega T}{1+\Omega^2 T^2} \tag{8.3.12}$$

As Ω varies from $-\infty$ to ∞, the corresponding locus of points in the z-plane is a circle of radius $\frac{1}{2}$ and with center at $z = \frac{1}{2}$, as illustrated in Fig. 8.30.

It is easily demonstrated that the mapping in (8.3.11) takes points in the LHP of the s-plane into corresponding points inside this circle in the z-plane and points in the RHP of the s-plane are mapped into points outside this circle. Consequently, this mapping has the desirable property that a stable analog filter is transformed into a stable digital filter. However, the possible location of the poles of the digital filter are confined to relatively small frequencies and, as a consequence, the mapping is restricted to the design of lowpass filters and bandpass filters having relatively small resonant frequencies. It is not possible, for example, to transform a highpass analog filter into a corresponding highpass digital filter.

In an attempt to overcome the limitations in the mapping given above, more complex substitutions for the derivatives have been proposed. In particular, an Lth-order difference of the form

$$\left.\frac{dy(t)}{dt}\right|_{t=nT} = \frac{1}{T}\sum_{k=1}^{L}\alpha_k \frac{y(nT+kT)-y(nT-kT)}{T} \tag{8.3.13}$$

has been proposed, where $\{\alpha_k\}$ are a set of parameters that can be selected to optimize the approximation. The resulting mapping between the s-plane and the z-plane is now

$$s = \frac{1}{T}\sum_{k=1}^{L}\alpha_k (z^k - z^{-k}) \tag{8.3.14}$$

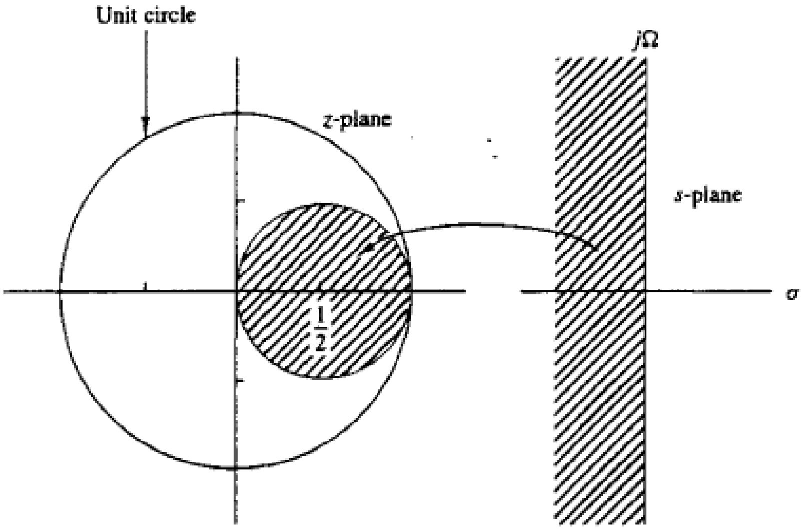


Figure 8.30 The mapping $s = (1 - z^{-1})/T$ takes LHP in the s-plane into points inside the circle of radius $\frac{1}{2}$ and center $z = \frac{1}{2}$ in the z-plane.

When $z = e^{j\omega}$, we have

$$s = j\frac{2}{T}\sum_{k=1}^{L} \alpha_k \sin \omega k \tag{8.3.15}$$

which is purely imaginary. Thus

$$\Omega = \frac{2}{T}\sum_{k=1}^{L} \alpha_k \sin \omega k \tag{8.3.16}$$

is the resulting mapping between the two frequency variables. By proper choice of the coefficients $\{\alpha_k\}$ it is possible to map the $j\Omega$-axis into the unit circle. Furthermore, points in the LHP in s can be mapped into points inside the unit circle in z.

Despite achieving the two desirable characteristics with the mapping of (8.3.16), the problem of selecting the set of coefficients $\{\alpha_k\}$ remains. In general, this is a difficult problem. Since simpler techniques exist for converting analog filters into IIR digital filters, we shall not emphasize the use of the Lth-order difference as a substitute for the derivative.

8.3.2 IIR Filter Design by Impulse Invariance

In the impulse invariance method, our objective is to design an IIR filter having a unit sample response $h(n)$ that is the sampled version of the impulse response of the analog filter. That is,

$$h(n) \equiv h(nT) \qquad n = 0, 1, 2, \ldots \tag{8.3.17}$$

where T is the sampling interval.

To examine the implications of (8.3.17), we refer back to Section 4.2.9. Recall that when a continuous time signal $x_a(t)$ with spectrum $X_a(F)$ is sampled at a rate $F_s = 1/T$ samples per second, the spectrum of the sampled signal is the periodic repetition of the scaled spectrum $F_s X_a(F)$ with period F_s. Specifically, the relationship is

$$X(f) = F_s \sum_{k=-\infty}^{\infty} X_a[(f - k)F_s] \tag{8.3.18}$$

where $f = F/F_s$ is the normalized frequency. Aliasing occurs if the sampling rate F_s is less than twice the highest frequency contained in $X_a(F)$.

Expressed in the context of sampling the impulse response of an analog filter with frequency response $H_a(F)$, the digital filter with unit sample response $h(n) \equiv h_a(nT)$ has the frequency response

$$H(f) = F_s \sum_{k=-\infty}^{\infty} H_a[(f - k)F_s] \tag{8.3.19}$$

or, equivalently,

$$H(\omega) = F_s \sum_{k=-\infty}^{\infty} H_a[(\omega - 2\pi k)F_s] \tag{8.3.20}$$

or

$$H(\Omega T) = \frac{1}{T} \sum_{k=-\infty}^{\infty} H_a\left(\Omega - \frac{2\pi k}{T}\right) \qquad (8.3.21)$$

Figure 8.31 depicts the frequency response of a lowpass analog filter and the frequency response of the corresponding digital filter.

It is clear that the digital filter with frequency response $H(\omega)$ has the frequency response characteristics of the corresponding analog filter if the sampling interval T is selected sufficiently small to completely avoid or at least minimize the effects of aliasing. It is also clear that the impulse invariance method is inappropriate for designing highpass filters due the to spectrum aliasing that results from the sampling process.

To investigate the mapping of points between the z-plane and the s-plane implied by the sampling process, we rely on a generalization of (8.3.21) which relates the z-transform of $h(n)$ to the Laplace transform of $h_a(t)$. This relationship is

$$H(z)|_{z=e^{sT}} = \frac{1}{T} \sum_{k=-\infty}^{\infty} H_a\left(s - j\frac{2\pi k}{T}\right) \qquad (8.3.22)$$

where

$$H(z) = \sum_{n=0}^{\infty} h(n)z^{-n}$$

$$H(z)|_{z=e^{sT}} = \sum_{n=0}^{\infty} h(n)e^{-sTn} \tag{8.3.23}$$

Note that when $s = j\Omega$, (8.3.22) reduces to (8.3.21), where the factor of j in $H_a(\Omega)$ is suppressed in our notation.

Let us consider the mapping of points from the s-plane to the z-plane implied by the relation

$$z = e^{sT} \tag{8.3.24}$$

If we substitute $s = \sigma + j\Omega$ and express the complex variable z in polar form as $z = re^{j\omega}$, (8.3.24) becomes

$$re^{j\omega} = e^{\sigma T} e^{j\Omega T}$$

Clearly, we must have

$$\begin{aligned} r &= e^{\sigma T} \\ \omega &= \Omega T \end{aligned} \tag{8.3.25}$$

Consequently, $\sigma < 0$ implies that $0 < r < 1$ and $\sigma > 0$ implies that $r > 1$. When $\sigma = 0$, we have $r = 1$. Therefore, the LHP in s is mapped inside the unit circle in z and the RHP in s is mapped outside the unit circle in z.

Also, the $j\Omega$-axis is mapped into the unit circle in z as indicated above. However, the mapping of the $j\Omega$-axis into the unit circle is not one-to-one. Since ω is unique over the range $(-\pi, \pi)$, the mapping $\omega = \Omega T$ implies that the interval $-\pi/T \leq \Omega \leq \pi/T$ maps into the corresponding values of $-\pi \leq \omega \leq \pi$. Furthermore, the frequency interval $\pi/T \leq \Omega \leq 3\pi/T$ also maps into the interval $-\pi \leq \omega \leq \pi$ and, in general, so does the interval $(2k-1)\pi/T \leq \Omega \leq (2k+1)\pi/T$, when k is an integer. Thus the mapping from the analog frequency Ω to the frequency variable ω in the digital domain is many-to-one, which simply reflects the effects of aliasing due to sampling. Figure 8.32 illustrates the mapping from the s-plane to the z-plane for the relation in (8.3.24).

To explore further the effect of the impulse invariance design method on the characteristics of the resulting filter, let us express the system function of the analog filter in partial-fraction form. On the assumption that the poles of the analog filter are distinct, we can write

$$H_a(s) = \sum_{k=1}^{N} \frac{c_k}{s - p_k} \tag{8.3.26}$$

where $\{p_k\}$ are the poles of the analog filter and $\{c_k\}$ are the coefficients in the partial-fraction expansion. Consequently,

$$h_a(t) = \sum_{k=1}^{N} c_k e^{p_k t} \qquad t \geq 0 \tag{8.3.27}$$

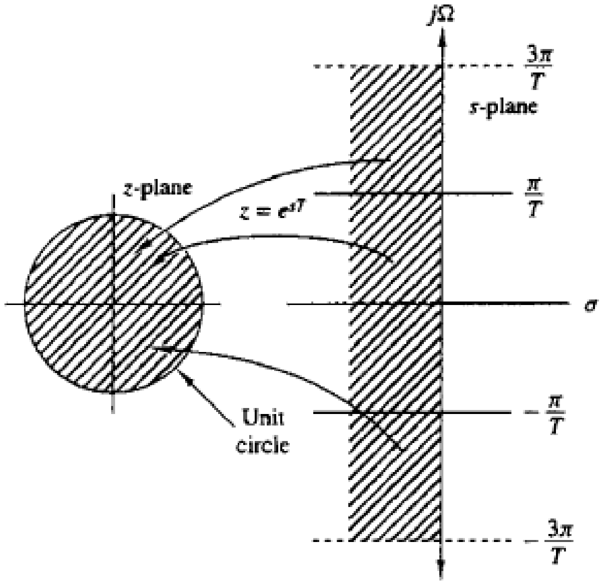


Figure 8.32 The mapping of $z = e^{sT}$ maps strips of width $2\pi/T$ (for $\sigma < 0$) in the s-plane into points in the unit circle in the z-plane.

If we sample $h_a(t)$ periodically at $t = nT$, we have

$$h(n) = h_a(nT)$$

$$= \sum_{k=1}^{N} c_k e^{p_k T n} \tag{8.3.28}$$

Now, with the substitution of (8.3.28), the system function of the resulting digital IIR filter becomes

$$H(z) = \sum_{n=0}^{\infty} h(n) z^{-n}$$

$$= \sum_{n=0}^{\infty} \left(\sum_{k=1}^{N} c_k e^{p_k T n} \right) z^{-n}$$

$$= \sum_{k=1}^{N} c_k \sum_{n=0}^{\infty} (e^{p_k T} z^{-1})^n \tag{8.3.29}$$

The inner sum in (8.3.29) converges because $p_k < 0$ and yields

$$\sum_{n=0}^{\infty} (e^{p_k T} z^{-1})^n = \frac{1}{1 - e^{p_k T} z^{-1}} \tag{8.3.30}$$

Therefore, the system function of the digital filter is

$$H(z) = \sum_{k=1}^{N} \frac{c_k}{1 - e^{p_k T} z^{-1}} \tag{8.3.31}$$

We observe that the digital filter has poles at

$$z_k = e^{p_k T} \qquad k = 1, 2, \ldots, N \tag{8.3.32}$$

8.3.3 IIR Filter Design by the Bilinear Transformation

The IIR filter design techniques described in the preceding two sections have a severe limitation in that they are appropriate only for lowpass filters and a limited class of bandpass filters.

In this section we describe a mapping from the s-plane to the z-plane, called the bilinear transformation, that overcomes the limitation of the other two design methods described previously. The bilinear transformation is a conformal mapping that transforms the $j\Omega$-axis into the unit circle in the z-plane only once, thus avoiding aliasing of frequency components. Furthermore, all points in the LHP of s are mapped inside the unit circle in the z-plane and all points in the RHP of s are mapped into corresponding points outside the unit circle in the z-plane.

The bilinear transformation can be linked to the trapezoidal formula for numerical integration. For example, let us consider an analog linear filter with system function

$$H(s) = \frac{b}{s+a} \tag{8.3.33}$$

This system is also characterized by the differential equation

$$\frac{dy(t)}{dt} + ay(t) = bx(t) \tag{8.3.34}$$

Instead of substituting a finite difference for the derivative, suppose that we integrate the derivative and approximate the integral by the trapezoidal formula. Thus

$$y(t) = \int_{t_0}^{t} y'(\tau)d\tau + y(t_0) \tag{8.3.35}$$

where $y'(t)$ denotes the derivative of $y(t)$. The approximation of the integral in (8.3.35) by the trapezoidal formula at $t = nT$ and $t_0 = nT - T$ yields

$$y(nT) = \frac{T}{2}[y'(nT) + y'(nT - T)] + y(nT - T) \tag{8.3.36}$$

Now the differential equation in (8.3.34) evaluated at $t = nT$ yields

$$y'(nT) = -ay(nT) + bx(nT) \tag{8.3.37}$$

We use (8.3.37) to substitute for the derivative in (8.3.36) and thus obtain a difference equation for the equivalent discrete-time system. With $y(n) \equiv y(nT)$ and $x(n) \equiv x(nT)$, we obtain the result

$$\left(1 + \frac{aT}{2}\right) y(n) - \left(1 - \frac{aT}{2}\right) y(n-1) = \frac{bT}{2}[x(n) + x(n-1)] \tag{8.3.38}$$

The z-transform of this difference equation is

$$\left(1 + \frac{aT}{2}\right) Y(z) - \left(1 - \frac{aT}{2}\right) z^{-1}Y(z) = \frac{bT}{2}(1 + z^{-1})X(z)$$

Consequently, the system function of the equivalent digital filter is

$$H(z) = \frac{Y(z)}{X(z)} = \frac{(bT/2)(1 + z^{-1})}{1 + aT/2 - (1 - aT/2)z^{-1}}$$

or, equivalently,

$$H(z) = \frac{b}{\frac{2}{T}\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right) + a} \tag{8.3.39}$$

Clearly, the mapping from the s-plane to the z-plane is

$$s = \frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right) \tag{8.3.40}$$

This is called the *bilinear transformation*.

Although our derivation of the bilinear transformation was performed for a first-order differential equation, it holds, in general, for an Nth-order differential equation.

To investigate the characteristics of the bilinear transformation, let

$$z = re^{j\omega}$$

$$s = \sigma + j\Omega$$

Then (8.3.40) can be expressed as

$$\begin{aligned} s &= \frac{2}{T}\frac{z-1}{z+1} \\ &= \frac{2}{T}\frac{re^{j\omega}-1}{re^{j\omega}+1} \\ &= \frac{2}{T}\left(\frac{r^2-1}{1+r^2+2r\cos\omega} + j\frac{2r\sin\omega}{1+r^2+2r\cos\omega}\right) \end{aligned}$$

Consequently,

$$\sigma = \frac{2}{T}\frac{r^2-1}{1+r^2+2r\cos\omega} \tag{8.3.41}$$

$$\Omega = \frac{2}{T}\frac{2r\sin\omega}{1+r^2+2r\cos\omega} \tag{8.3.42}$$

First, we note that if $r < 1$, then $\sigma < 0$, and if $r > 1$, then $\sigma > 0$. Consequently, the LHP in s maps into the inside of the unit circle in the z-plane and the RHP in s maps into the outside of the unit circle. When $r = 1$, then $\sigma = 0$ and

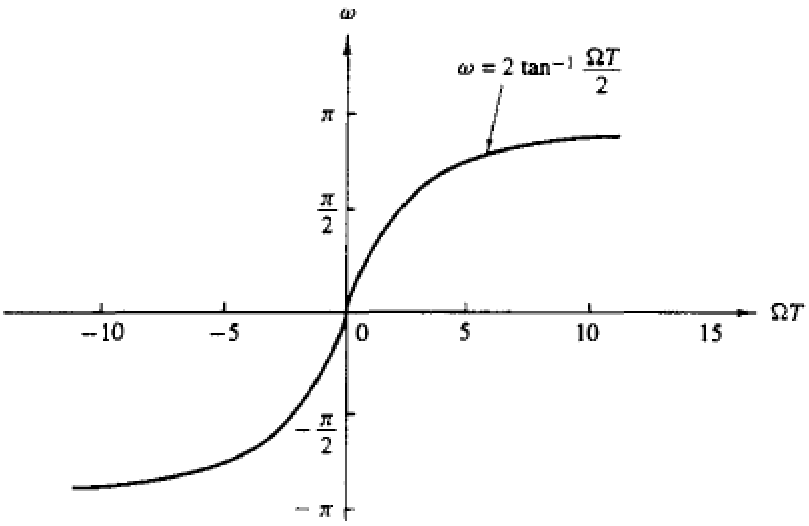
$$\begin{aligned} \Omega &= \frac{2}{T}\frac{\sin\omega}{1+\cos\omega} \\ &= \frac{2}{T}\tan\frac{\omega}{2} \end{aligned} \tag{8.3.43}$$

or, equivalently,

$$\omega = 2\tan^{-1}\frac{\Omega T}{2} \tag{8.3.44}$$

The relationship in (8.3.44) between the frequency variables in the two domains is illustrated in Fig. 8.36. We observe that the entire range in Ω is mapped only once into the range $-\pi \le \omega \le \pi$. However, the mapping is highly nonlinear. We observe a frequency compression or *frequency warping*, as it is usually called, due to the nonlinearity of the arctangent function.

It is also interesting to note that the bilinear transformation maps the point $s = \infty$ into the point $z = -1$. Consequently, the single-pole lowpass filter in



Chebyshev filters. There are two types of Chebyshev filters. Type I Chebyshev filters are all-pole filters that exhibit equiripple behavior in the passband and a monotonic characteristic in the stopband. On the other hand, the family of type II Chebyshev filters contains both poles and zeros and exhibits a

monotonic behavior in the passband and an equiripple behavior in the stopband. The zeros of this class of filters lie on the imaginary axis in the s-plane.

The magnitude squared of the frequency response characteristic of a type I Chebyshev filter is given as

$$|H(\Omega)|^2 = \frac{1}{1+\epsilon^2 T_N^2(\Omega/\Omega_p)} \tag{8.3.51}$$

where ϵ is a parameter of the filter related to the ripple in the passband and $T_N(x)$ is the Nth-order Chebyshev polynomial defined as

$$T_N(x) = \begin{cases} \cos(N\cos^{-1}x), & |x| \le 1 \\ \cosh(N\cosh^{-1}x), & |x| > 1 \end{cases} \tag{8.3.52}$$

The Chebyshev polynomials can be generated by the recursive equation

$$T_{N+1}(x) = 2xT_N(x) - T_{N-1}(x) \qquad N = 1, 2, \ldots \tag{8.3.53}$$

where $T_0(x) = 1$ and $T_1(x) = x$. From (8.3.53) we obtain $T_2(x) = 2x^2 - 1$, $T_3(x) = 4x^3 - 3x$, and so on.

Some of the properties of these polynomials are as follows:

1. $|T_N(x)| \le 1$ for all $|x| \le 1$.
2. $T_N(1) = 1$ for all N.
3. All the roots of the polynomial $T_N(x)$ occur in the interval $-1 \le x \le 1$.

The filter parameter ϵ is related to the ripple in the passband, as illustrated in Fig. 8.39, for N odd and N even. For N odd, $T_N(0) = 0$ and hence $|H(0)|^2 = 1$. On the other hand, for N even, $T_N(0) = 1$ and hence $|H(0)|^2 = 1/(1+\epsilon^2)$. At the band edge frequency $\Omega = \Omega_p$, we have $T_N(1) = 1$, so that

$$\frac{1}{\sqrt{1+\epsilon^2}} = 1 - \delta_1$$

or, equivalently,

$$\epsilon^2 = \frac{1}{(1-\delta_1)^2} - 1 \tag{8.3.54}$$

where δ_1 is the value of the passband ripple.

The poles of a type I Chebyshev filter lie on an ellipse in the s-plane with major axis

$$r_1 = \Omega_p \frac{\beta^2 + 1}{2\beta} \tag{8.3.55}$$

and minor axis

$$r_2 = \Omega_p \frac{\beta^2 - 1}{2\beta} \tag{8.3.56}$$

where β is related to ϵ according to the equation

$$\beta = \left[\frac{\sqrt{1+\epsilon^2} + 1}{\epsilon}\right]^{1/N} \tag{8.3.57}$$

The pole locations are most easily determined for a filter of order N by first locating the poles for an equivalent Nth-order Butterworth filter that lie on circles of radius r_1 or radius r_2, as illustrated in Fig. 8.40. If we denote the angular positions of the poles of the Butterworth filter as

$$\phi_k = \frac{\pi}{2} + \frac{(2k+1)\pi}{2N} \qquad k = 0, 1, 2, \ldots, N-1 \tag{8.3.58}$$

then the positions of the poles for the Chebyshev filter lie on the ellipse at the coordinates (x_k, y_k), $k = 0, 1, \ldots, N-1$, where

$$\begin{aligned} x_k &= r_2 \cos\phi_k, \qquad k = 0, 1, \ldots, N-1 \\ y_k &= r_1 \sin\phi_k, \qquad k = 0, 1, \ldots, N-1 \end{aligned} \tag{8.3.59}$$

A type II Chebyshev filter contains zeros as well as poles. The magnitude squared of its frequency response is given as

$$|H(\Omega)|^2 = \frac{1}{1 + \epsilon^2[T_N^2(\Omega_s/\Omega_p)/T_N^2(\Omega_s/\Omega)]} \tag{8.3.60}$$

where $T_N(x)$ is, again, the Nth-order Chebyshev polynomial and Ω_s is the stopband frequency as illustrated in Fig. 8.41. The zeros are located on the imaginary axis at the points

$$s_k = j\frac{\Omega_s}{\sin\phi_k} \qquad k = 0, 1, \ldots, N-1 \tag{8.3.61}$$

The poles are located at the points (v_k, w_k), where

$$v_k = \frac{\Omega_s x_k}{\sqrt{x_k^2 + y_k^2}} \qquad k = 0, 1, \ldots, N-1 \tag{8.3.62}$$

$$w_k = \frac{\Omega_s y_k}{\sqrt{x_k^2 + y_k^2}} \qquad k = 0, 1, \ldots, N-1 \tag{8.3.63}$$

where $\{x_k\}$ and $\{y_k\}$ are defined in (8.3.59) with β now related to the ripple in the stopband through the equation

$$\beta = \left[\frac{1 + \sqrt{1 - \delta_2^2}}{\delta_2}\right]^{1/N} \tag{8.3.64}$$

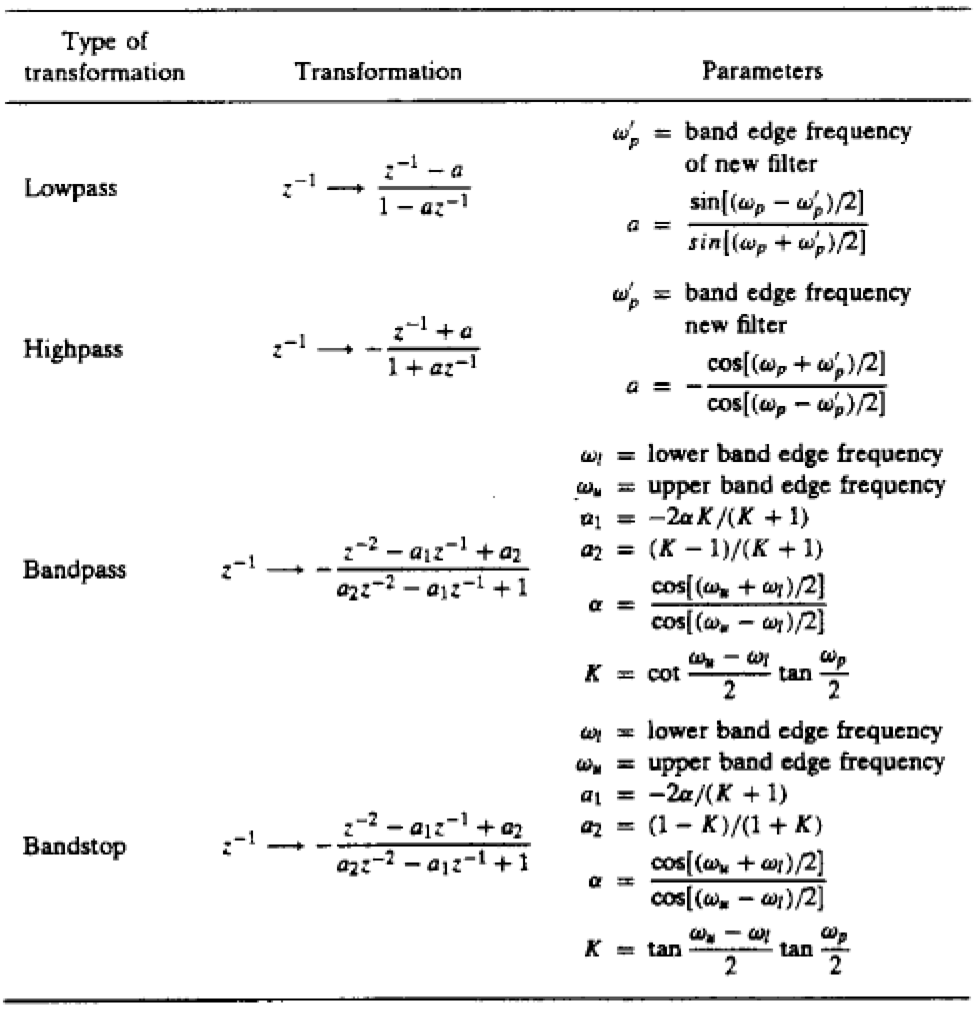
$$\begin{aligned} N &= \frac{\log\left[\left(\sqrt{1 - \delta_2^2} + \sqrt{1 - \delta_2^2(1 + \epsilon^2)}\right) / \epsilon\delta_2\right]}{\log\left[(\Omega_s/\Omega_p) + \sqrt{(\Omega_s/\Omega_p)^2 - 1}\right]} \\ &= \frac{\cosh^{-1}(\delta/\epsilon)}{\cosh^{-1}(\Omega_s/\Omega_p)} \end{aligned}$$

where, by definition, $\delta_2 = 1/\sqrt{1 + \delta^2}$.

Frequency Transformations in the Analog Domain

Type of transformation	Transformation	Band edge frequencies of new filter
Lowpass	$s \longrightarrow \dfrac{\Omega_p}{\Omega'_p} s$	Ω'_p
Highpass	$s \longrightarrow \dfrac{\Omega_p \Omega'_p}{s}$	Ω'_p
Bandpass	$s \longrightarrow \Omega_p \dfrac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)}$	Ω_l, Ω_u
Bandstop	$s \longrightarrow \Omega_p \dfrac{s(\Omega_u - \Omega_c)}{s^2 + \Omega_u \Omega_l}$	Ω_l, Ω_u

Frequency Transformations in the Digital Domain



Type of transformation	Transformation	Parameters
Lowpass	$z^{-1} \longrightarrow \frac{z^{-1}-a}{1-az^{-1}}$	ω_p' = band edge frequency of new filter $a = \frac{\sin[(\omega_p-\omega_p')/2]}{sin[(\omega_p+\omega_p')/2]}$
Highpass	$z^{-1} \longrightarrow -\frac{z^{-1}+a}{1+az^{-1}}$	ω_p' = band edge frequency new filter $a = -\frac{\cos[(\omega_p+\omega_p')/2]}{\cos[(\omega_p-\omega_p')/2]}$
Bandpass	$z^{-1} \longrightarrow -\frac{z^{-2}-a_1z^{-1}+a_2}{a_2z^{-2}-a_1z^{-1}+1}$	ω_l = lower band edge frequency ω_u = upper band edge frequency $a_1 = -2\alpha K/(K+1)$ $a_2 = (K-1)/(K+1)$ $\alpha = \frac{\cos[(\omega_u+\omega_l)/2]}{\cos[(\omega_u-\omega_l)/2]}$ $K = \cot\frac{\omega_u-\omega_l}{2}\tan\frac{\omega_p}{2}$
Bandstop	$z^{-1} \longrightarrow -\frac{z^{-2}-a_1z^{-1}+a_2}{a_2z^{-2}-a_1z^{-1}+1}$	ω_l = lower band edge frequency ω_u = upper band edge frequency $a_1 = -2\alpha/(K+1)$ $a_2 = (1-K)/(1+K)$ $\alpha = \frac{\cos[(\omega_u+\omega_l)/2]}{\cos[(\omega_u-\omega_l)/2]}$ $K = \tan\frac{\omega_u-\omega_l}{2}\tan\frac{\omega_p}{2}$

****

UNIT - IV

FINITE WORD LENGTH EFFECTS IN DIGITAL FILTER

Finite Word length Effects:

- In the design of FIR Filters, The filter coefficients are determined by the system transfer functions. These filter co-efficient are quantized/truncated while implementing DSP System because of finite length registers.
- Only Finite numbers of bits are used to perform arithmetic operations. Typical word length is 16 bits, 24 bits, 32 bits etc.
- This finite word length introduces an error which can affect the performance of the DSP system.
- The main errors are
  1. Input quantization error
  2. Co-efficient quantization error
  3. Overflow & round off error (Product Quantization error)
- The effect of error introduced by a signal process depend upon number of factors including the
  1. Type of arithmetic
  2. Quality of input signal
  3. Type of algorithm implemented

1. **Input quantization error**
   - The conversion of continuous-time input signal into digital value produces an error which is known as input quantization error.
   - This error arises due to the representation of the input signal by a fixed number of digits in A/D conversion process.
2. **Co-efficient quantization error**
   - The filter coefficients are compared to infinite precision. If they are quantized the frequency response of the resulting filter may differ from the desired frequency response.
     i.e poles of the desired filter may change leading to instability.
3. **Product Quantization error**
   - It arises at the output of the multiplier
   - When a ‘b’ bit data is multiplied with another ‘b’ bit coefficient the product (‘2b’ bits) should be stored in ‘b’ bits register. The multiplier Output must be rounded or truncated to ‘b’ bits. This known as overflow and round off error.

****

Types of number representation:

There are two common forms that are used to represent the numbers in a digital or any other digital hardware.

1. Fixed point representation
2. Floating point representation

*** Explain the various formulas of the fixed point representation of binary numbers.**

1. **Fixed point representation**

- In the fixed point arithmetic, the position of the binary point is fixed. The bit to the right represents the fractional part of the number and to those to the left represents the integer part.

- For example, the binary number 01.1100 has the value 1.75 in decimal.

  $(0*2^{1)} + (1*2^{0}) + (1*2^{-1}) + (1*2^{-2}) + (0*2^{-3}) = 1.75$

  In general, we can represent the fixed point number ‘N’ to any desired accuracy by the series

$$N = \sum_{i=n_i}^{n_2} C_i r^i$$

  Where, r is called as radix.

- If r=10, the representation is known as decimal representation having numbers from 0 to 9. In this representation the number

$$30.285 = \sum_{i=-3}^{1_2} C_i 10^i$$

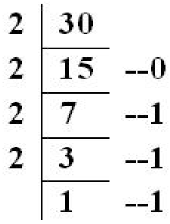
$$= (3*10^{1}) + (0*10^{0}) + (2*10^{-1}) + (8*10^{-2}) + (5*10^{-3})$$

- If r=2, the representation is known as binary representation with two numbers 0 to 1.
- For example, the binary number

$$110.010 = (1*2^{2}) + (1*2^{1}) + (0*2^{0}) + (0*2^{-1}) + (1*2^{-2}) + (0*2^{-3}) = 6.25$$

Examples:

Convert the decimal number 30.275 to binary form



2	30	
2	15	--0
2	7	--1
2	3	--1
	1	--1

0.275 * 2	➔0.55	➔0
0.55 * 2	➔1.10	➔1
0.10 * 2	➔0.20	➔0
0.20 * 2	➔0.40	➔0
0.40 * 2	➔0.80	➔0
0.80 * 2	➔1.60	➔1
0.60 * 2	➔1.20	➔1
0.20 * 2	➔0.40	➔0

$(30.275)_{10}$ = $(11110.01000110)_{2}$

In fixed point arithmetic =, the negative numbers are represented by 3 forms.

1. Sign-magnitude form
2. One’s complement form
3. Two’s complement form

1.1 Sign-magnitude form:

- Here an additional bit called sign bit is added as MSB.
  - If this bit is zero → It is a positive number
  - If this bit is one → It is a positive number
- For example
  - 1.75 is represented as 01.110000.
  - -1.75 is represented as 11.110000

1.2 One’s complement form:

- Here the positive number is represented same as that in sign magnitude form.
- But the negative number is obtained by complementing all the bits of the positive number
- For eg: the decimal number -0.875 can be represented as
  - $(0.875)_{10}=(0.111000)_{2}$
  - $(-0.875)_{10}=(1.000111)_{2}$

0.111000
↓ ↓↓↓↓↓↓ (Complement each bit)
1.000111

1.3 Two's complement form:

- Here the positive numbers are represented as same in sign magnitude and one's complement form.
- The negative numbers are obtained by complementing all the bits of the positive number and adding one to the least significant bit

$(0.875)_{10}=(0.111000)_2$

↓ ↓↓↓↓↓↓ (Complement each bit)

1.000111

+ 1

1.001000

$(-0.875)_{10}=(1.001000)_2$

Examples:

- **Find the sign magnitude, 1's complement, 2's complement for the given numbers.**

1. $-\frac{7}{32}$

2. $-\frac{7}{8}$

3. $+\frac{7}{8}$

1. $-\frac{7}{32}$

0.21875 * 2 ➔0.43750 ➔0
0.43750 * 2 ➔0.87500 ➔0
0.87500* 2 ➔ 1.750000 ➔1
0.75* 2 ➔1.50 ➔1
0.50* 2 ➔1.00 ➔1

$-\frac{7}{32}=(-0.21875)_{10} \quad =(1.00111)_2$

Sign magnitude form = 1.00111
1's complement form = 1.11000
2's complement form = 1.11001

2. $-\frac{7}{8}$

0.875 * 2 ➔1.75 ➔1
0.750 * 2 ➔1.500 ➔1
0.500 * 2 ➔1.000 ➔1

$-\frac{7}{8} =(-0.875)_{10} \quad =(0.111)_2$

Sign magnitude form = 0.111
1's complement form = 1.000
2's complement form = 1.001

3. $+\frac{7}{8}$

Sign magnitude form = 0.111
1's complement form = 0.111
2's complement form = 0.111

Addition of two fixed point numbers:

- **Add $(0.5)_{10} + (0.125)_{10}$**

$(0.5)_{10} \quad = \quad (0.100)_2$

$(0.125)_{10} \quad = \quad \underline{(0.001)_2}$

$(0.101)_2 = (0.625)_{10}$

- Addition of two fixed point numbers causes an overflow.

  For example

  $(0.100)_2$

  $\underline{(0.101)_2}$

  $(1.001)_2 = (-0.125)_{10}$ in sign magnitude form

Subtraction of two fixed point numbers:

- Subtraction of two numbers can be easily performed easily by using two's complement representation.
- **Subtract 0.25 from 0.5**

  $0.25 * 2 \rightarrow 0.50 \quad \rightarrow 0$ Sign magnitude form = $(0.010)_2$

  $0.50 * 2 \rightarrow 1.00 \quad \rightarrow 1$ 1's complement form = $(1.101)_2$

  $0.00 * 2 \rightarrow 0.00 \quad \rightarrow 0$ 2's complement form = $(1.110)_2$

  $(0.5)_{10} = \quad (0.100)_2$

  $-(0.25)_{10} = \quad \underline{(1.110)_2}$ → Two's complement of -0.25

  $(10.010)_2$

  Here the carry is generated after the addition. Neglect the carry bit to get the result in decimal.

  $(0.010)_2 = (0.25)_{10}$

- **Subtract 0.5 from 0.25**

  $0.5 * 2 \rightarrow 1.00 \quad \rightarrow 1$ Sign magnitude form = $(0.100)_2$

  $0.00 * 2 \rightarrow 0.00 \quad \rightarrow 0$ 1's complement form = $(1.011)_2$

  $0.00 * 2 \rightarrow 0.00 \quad \rightarrow 0$ 2's complement form = $(1.100)_2$

  $(0.25)_{10} \quad = \quad (0.010)_2$

  $-(0.5)_{10} \quad = \quad \underline{(1.100)_2}$

  $(1.110)_2$

  Here the carry is not generated after the addition. So the result is negative.

Multiplication in fixed point arithmetic:

- Here the sign magnitude components are separated.
- The magnitudes of the numbers are multiplied. Then the sign of the product is determined and applied to the result.
- In the fixed point arithmetic, multiplication of two fractions results in a fraction.
- For multiplications with fractions, overflow can never occur.
- Eg:

  $(0.1001)_2 * (0.0011)_2 \quad = \quad (0.00011011)_2$

2. Floating point representation

- Here, a number 'x' is represented by

  $X=M.r^e$

  Where, M → Mantissa which requires a sign bit for representing positive number and negative numbers.

  R → base (or) radix

  e → exponent which require an additional and it may be either positive or negative.

  - For eg, 278 can be represented in floating point representation.

    $$278 \quad = \quad \frac{278 \ X \ 1000}{1000} \quad = \quad 0.278*10_3$$

    0.278 → Mantissa (M)

    10 → base (or) radix (r)

    3 → exponents (e)

- Similarly, to represent a binary floating point number
  X=M.2^e in which the fractional part of a number should fall (or) lie in the range of 1/2 to 1.

  5 $= \frac{5 \text{ X } 8}{8}$ 0.625 X2^3

  Mantissa (M) = 0.625
  Base (or) radix (r) = 2
  Exponent (e) = 3

- Some decimal numbers and their floating point representations are given below:

  4.5 → 0.5625 X 2^3 =0.1001 X 2^{011}
  1.5 → 0.75 X 2^1 =0.1100 X 2^{001}
  6.5 → 0.8125 X 2^3 =0.1100 X 2^{011}
  0.625 → 0.625 X 2^0 =0.1010 X 2^{000}

- Negative floating point numbers are generally represented by considering the mantissa as a fixed point number. The sign of the floating point number is obtained from the first bit of mantissa.
- To represent floating point in multiplication

  Consider $X_1 = M_1 r^{e_1}$

  $X_2 = M_2 r^{e_2}$

  $X_1 X_2 = (M_1 * M_2) r^{(e_1 + e_2)}$

Example

Given $X_1 = 3.5*10^{-12}$, $X_2 = 4.75*10^6$. Find the product $X_1 X_2$

X=(3.5 X 4.75) $10^{(-12+6)}$

= $(16.625)10^{-6}$ → in decimal

In binary: $(1.5)_{10}$ X $(1.25)_{10}$ = $(2^1 0.75)$ X $(2^1 0.625)$

= 2^{001} X 0.1100 X 2^{001} X 0.1010

= 2^{010} X 0.01111

Addition and subtraction:

- Here the exponent of a smaller number is adjusted until it matches the exponent of a larger number.
- Then, the mantissa are added or subtracted
- The resulting representation is rescaled so that its mantissa lies in the range 0.5 to 1.
- Eg: **Add $(3.0)_{10}$ & $(0.125)_{10}$**

  $(3.0)_{10}$ =2^{010} X 0.1100 = $r^{e_1} \text{X} M_1$

  $(0.125)_{10}$ =2^{000} X 0.0010 = $r^{e_2} \text{X} M_2$

  Now adjust e_2 Such that e_1=e_2

  $(0.125)_{10}$ =2^{010} X 0.0000100

  Addition →2^{010} (0.110000 + 0.0000100) →2^{010} X 0.110010

  Subraction →2^{010} X 1.001101

Compare floating point with fixed point arithmetic.

Sl.No	Fixed point arithmetic	Floating point arithmetic
1	Fast operation	Slow operation
2	Relatively economical	More expensive because of costlier hardware
3	Small dynamic range	Increased Dynamic range
4	Round off errors occurs only for addition	Round off errors can occur with addition and multiplication
5	Overflow occur in addition	Overflow does not arise
6	Used in small computers	Used in large general purpose computers.

Quantization:

***Discuss the various methods of quantization.**

***Derive the expression for rounding and truncation errors**

*** Discuss in detail about Quantization error that occurs due to finite word length of registers.**

The common methods of quantization are

1. Truncation
2. Rounding

1. Truncation

- The abrupt termination of given number having a large string of bits (or)
- Truncation is a process of discarding all bits less significant than the LSB that is retained.
- Suppose if we truncate the following binary number from 8 bits to 4 bits, we obtain
  - 0.00110011 to 0.0011
    (8 bits) (4 bits)
  - 1.01001001 to 1.0100
    (8 bits) (4 bits)
- When we truncate the number, the signal value is approximated by the highest quantization level that is not greater than the signal.

2. Rounding (or) Round off

- Rounding is the process of reducing the size of a binary number to finite word size of 'b' bits such that the rounded b-bit number is closest to the original unquantised number.

Error Due to truncation and rounding:

- While storing (or) computation on a number we face registers length problems. Hence given number is quantized to truncation (or) round off.
  i.e. Number of bits in the original number is reduced register length.

Truncation error in sign magnitude form:

- Consider a 5 bit number which has value of
  $0.11001_2 \rightarrow (0.7815)_{10}$
- This 5 bit number is truncated to a 4 bit number
  $0.1100_2 \rightarrow (0.75)_{10}$
  i.e. 5 bit number →0.11001 has 'l' bits
  4 bit number →0.1100 has 'b' bits
- Truncation error, e_t = 0.1100 – 0.11001
  = -0.00001 → $(-0.03125)_{10}$
- Here original length is 'l' bits. (l=5). The truncated length is 'b' bits.
- The truncation error, $e_t = 2^{-b}-2^{-l}$
  $= -(2^{-l}-2^{-b})$
  $e_t = -(2^{-5}-2^{-4}) = -2^{-1}$
- The truncation error for a positive number is
  $-\left(2^{-b}-2^{-l}\right) \le e_t \le 0$ → Non causal
- The truncation error for a negative number is
  $0 \le e_t \le \left(2^{-b}-2^{-l}\right)$ → Causal

Truncation error in two's complement:

- For a positive number, the truncation results in a smaller number and hence remains same as in the case of sign magnitude form.
- For a negative number, the truncation produces negative error in two's complement

$$-\left(2^{-b}-2^{-l}\right) \le e_t \le \left(2^{-b}-2^{-l}\right)$$

Round off error (Error due to rounding):

- Let us consider a number with original length as '5' bits and round off length as '4' bits.

$$0.11001 \xrightarrow{\text{Round off to}} 0.1101$$

- Now error due to rounding $e_r = \frac{2^{-b} - 2^{-l}}{2}$

  Where b→Number of bits to the right of binary point after rounding
  L→Number of bits to the right of binary point before rounding

- Rounding off error for positive Number:

$$-\frac{2^{-b} - 2^{-l}}{2} \le e_r \le 0$$

- Rounding off error for negative Number:

$$0 \le e_r \le \frac{2^{-b} - 2^{-l}}{2}$$

- For two's complement

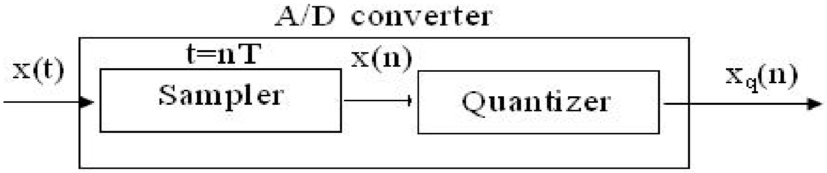
$$-\frac{2^{-b} - 2^{-l}}{2} \le e_r \le \frac{2^{-b} - 2^{-l}}{2}$$

****

Quantization Noise:

***Derive the expression for signal to quantization noise ratio**

***What is called Quantization Noise? Derive the expression for quantization noise power.**



- The analog signal is converted into digital signal by ADC
- At first, the signal x(t) is sampled at regular intervals t=nT, where n=0,1,2… to create sequence x(n). This is done by a sampler.
- Then the numeric equivalent of each sample x(n) is expressed by a finite number of bits giving the sequence $x_q(n)$
- The difference signal $e(n)= x_q(n)- x(n)$ is called quantization noise (or) A/D conversion noise.
- Let us assume a sinusoidal signal varying between +1 & -1 having a dynamic range 2
- ADC employs (b+1) bits including sign bit. In this case, the number of levels available for quantizing x(n) is 2^{b+1}.
- The interval between the successive levels is

  $q = \frac{2}{2^{b+1}} = 2^{-b}$

  Where q → quantization step size

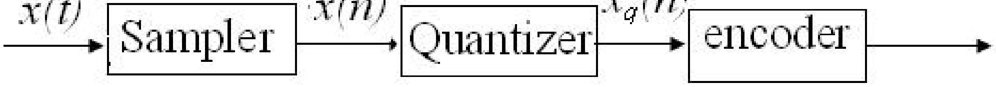
  If b=3 bits, then $q=2^{-3}=0.125$

****

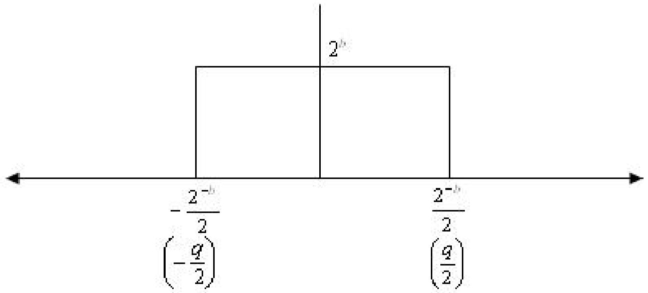
Quantization Noise power:

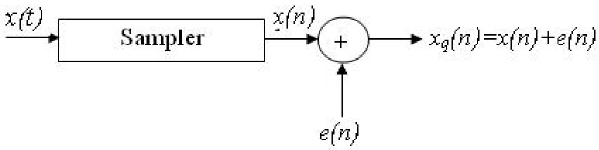
<u>Input Quantization error:</u>

***Derive the equation for quantization noise power (or) Steady state Input Noise Power.**



Probability density function for round off error in A/D conversion is





If rounding is used for quantization, which is bounded by $-\frac{q}{2} \le e(n) \le \frac{q}{2}$, then the error lies between $-\frac{q}{2}$ to $\frac{q}{2}$ with equal probability, where q→ quantization step size.

Properties of analog to digital conversion error, e(n):

1. The error sequence e(n) is a sample sequence of a stationary random process.
2. The error sequence is uncorrelated with x(n) and other signals in the system.
3. The error is a white noise process with uniform amplitude probability distribution over the range of quantization error.

The variance of e(n) is given by

$$\pm_e^2 = E\left[e^2(n)\right] - E^2\left[e(n)\right] \text{------------------------------>(1)}$$

Where $E\left[e^2(n)\right]$→Average of e^2(n)

$E\left[e(n)\right]$→Mean value of e(n).

For rounding, e(n) lies between $-\frac{q}{2}$ and $\frac{q}{2}$ with equal probability

$$E\left[e^2(n)\right] = \int_{-\infty}^{\infty} e^2(n)p(e)de \text{------------------------------>(2)}$$

$$p(e) = \frac{1}{q}, -\frac{q}{2} \le e(n) \le \frac{q}{2} \text{------------------------------>(3)}$$

Substituting (3) in (2)

$$E\left[e^2(n)\right] = \int_{-\frac{q}{2}}^{\frac{q}{2}} e^2(n)\frac{1}{q}de$$

$$E\left[e^2(n)\right] = \frac{1}{q}\int_{-\frac{q}{2}}^{\frac{q}{2}} e^2(n)de \text{------------------------------->(4)}$$

$$E\left[e(n)\right] = 0$$

$$E^2\left[e(n)\right] = 0 \text{--->(5)}$$

Substituting (4) and (5) in (1)

$$\pm_e^2 = \frac{1}{q}\int_{-\frac{q}{2}}^{\frac{q}{2}} e^2(n)de - 0$$

$$= \frac{1}{q}\left[\frac{e^3}{3}\right]_{-\frac{q}{2}}^{\frac{q}{2}}$$

$$= \frac{1}{3q}\left[\left(\frac{q}{2}\right)^3 - \left(-\frac{q}{2}\right)^3\right]$$

$$= \frac{1}{3q}\left[\left(\frac{q^3}{8}\right) - \left(-\frac{q^3}{8}\right)\right]$$

$$= \frac{1}{3q}\left[\left(\frac{q^3}{8}\right) + \left(\frac{q^3}{8}\right)\right]$$

$$= \frac{1}{3q}\left[\frac{2q^3}{8}\right]$$

$$\pm_e^2 = \frac{q^2}{12} \text{ --->(6)}$$

In general, $\frac{1}{2^b} = 2^{-b} = q$ --->(7)

$$\pm_e^2 = \frac{\left(2^{-b}\right)^2}{12}$$

$$\pm_e^2 = \frac{2^{-2b}}{12} \text{ --->(8)}$$

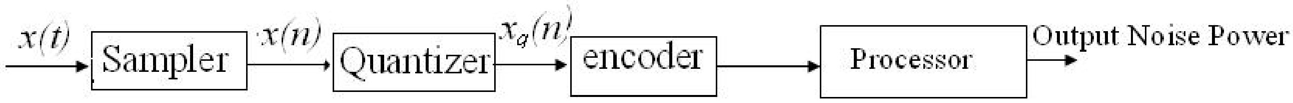
Equation (8) is known as the steady state noise power due to input quantization.

$q = \frac{R}{2^b}$ → in two's complement representation.

$q = \frac{R}{2^b - 1}$ → in sign magnitude (or) one's complement representation.

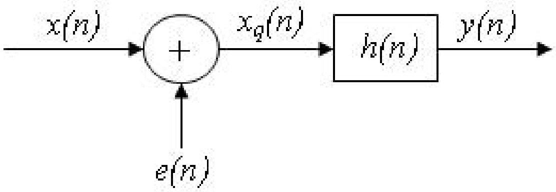
R → Range of analog signal to be quantized.

Steady state Output Noise power:



After quantization, we have noise power $\pm_e^2$ as input noise power. Therefore, Output noise power of system is given by

$$\pm_{eo}^2 = \pm_e^2\left[\sum_{n=0}^{\infty} h^2(n)\right] \text{ ------------------------------------>(9)}$$



where h(n) → impulse response of the system.

Let error E(n) be output noise power due to quantization

Error $E(n) = e(n) * h(n)$

$$= \sum_{k=0}^{\infty} h(n)e(n-k)$$

The variance of error E(n) is called output noise power, $\pm_e^2$.

By using Parseval’s theorem,

$$\pm_{eo}^2 \quad = \pm_e^2 \sum_{n=0}^{\infty} h^2(n)$$

$$= \pm_e^2 \frac{1}{2\circledR j}\oint H(Z)H\left(Z^{-1}\right)\frac{dZ}{z}$$

Where the closed contour integration is evaluated using the method of residue by taking only the poles that lie inside the unit circle.

Z transform of h(n), $\quad H(Z) = \sum_{n=0}^{\infty} h(n)z^{-n}$

Z transform of h^2(n) = Z[h^2(n)] $\quad = \sum_{n=0}^{\infty} h^2(n)z^{-n} = \sum_{n=0}^{\infty} h(n)h(n)z^{-n}$ -------------------->(10)

By Inverse Z transform, $\quad h(n) = \frac{1}{2\circledR j}\oint H(Z)Z^{n-1}dZ$ -------------------------------->(11)

Substituting (11) in (10)

$$\sum_{n=0}^{\infty} h^2(n)z^{-n} = \sum_{n=0}^{\infty} \frac{1}{2\circledR j}\oint H(Z)Z^{n-1}dZ \;\; h(n)z^{-n}$$

$$= \frac{1}{2\circledR j}\oint H(Z)\left[\sum_{n=0}^{\infty} h(n)Z^{-1}\right]dZ$$

$$\sum_{n=0}^{\infty} h^2(n) = \frac{1}{2\circledR j}\oint H(Z)\left[\sum_{n=0}^{\infty} h(n)Z^{-1}\right]\frac{dZ}{Z^{-n}}$$

$$= \frac{1}{2\circledR j}\oint H(Z)\left[\sum_{n=0}^{\infty} h(n)\left(Z^{-n}\right)^{-1}Z^{-1}dZ\right]$$

$$\sum_{n=0}^{\infty} h^2(n) = \frac{1}{2\circledR j}\oint H(Z)H\left(Z^{-1}\right)\frac{dZ}{Z} \text{ ----------------------------------->(12)}$$

Substituting (12) in (9)

$$\pm_{eo}^2 = \pm_e^2\left[\frac{1}{2\circledR j}\oint H(Z)H\left(Z^{-1}\right)Z^{-1}dZ\right]$$

Problem:

The output signal of an A/D converter is passed through a first order low pass filter, with transfer function given by

$H(z) = \dfrac{(1-a)z}{z-a}$ *for* $0 < a < 1$. **Find the steady state output noise power due to quantization at the output of the digital filter. [Nov/Dec-2015]**

Solution:

$$\pm_e^2 = \pm_e^2 \frac{1}{2\circledR j}\oint_c H(z)H(z^{-1})z^{-1}dz$$

Given $\quad H(z) = \dfrac{(1-a)z}{(z-a)} \qquad H(z^{-1}) = \dfrac{(1-a)z^{-1}}{\left(z^{-1}-a\right)}$

Substituti ng $H(z)$ *and* $H(z^{-1})$ *in equation (1), we have*

$$\pm_e^2 = \frac{\pm_e^2}{2\circledR j}\oint_c \frac{(1-a)z}{(z-a)}\frac{(1-a)z^{-1}}{\left(z^{-1}-a\right)}z^{-1}dz = \frac{\pm_e^2}{2\circledR j}\oint_c \frac{(1-a)^2}{(z-a)}\frac{dz}{\left(z^{-1}-a\right)}z^{-1}$$

$$= \pm_e{}^2 \left[residue\ of\ H(z)\,H(z^{-1})z^{-1} at\ z = a\ + residue\ of\ H(z)\,H(z^{-1})z^{-1} at\ z = \frac{1}{a} \right]$$

$$= \pm_e{}^2 \left[(z-a)\frac{(1-a)^2 z^{-1}}{(z-a)(z^{-1}-a)} + 0 \right]$$

$$= \pm_e{}^2 \left[\frac{(1-a)^2}{(z^{-1}-a)} \right] = \pm_e{}^2 \left[\frac{(1-a)}{(1+a)} \right]$$

Where, $\pm_e{}^2 = \frac{2^{-2b}}{12}$

****

Find the steady state variance of the noise in the output due to quantization of input for the first order filter.

$$y(n) = a\,y(n-1) + x(n)$$

Solution:

The impulse response for the above filter is given by $h(n) = a^n u(n)$

$$\pm_\in^2 = \pm_e^2 \sum_{k=0}^{\sim} h^2(n)$$

$$= \pm_e^2 \sum_{k=0}^{\sim} a^{2n}$$

$$= \pm_e^2 \left[1 + a^2 + a^4 + \ldots.\infty\right]$$

$$= \pm_e^2 \frac{1}{1-a^2}$$

$$= \frac{2^{-2b}}{12}\left[\frac{1}{1-a^2}\right] \qquad (or)$$

Taking Z-transform on both sides we have

$$Y(z) = az^{-1}Y(z) + X(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-az^{-1}} = \frac{z}{z-a}$$

$$H(z^{-1}) = \frac{z^{-1}}{z^{-1}-a}$$

We know

$$\pm_\in^2 = \pm_e^2 \frac{1}{2\circledR\, j}\int_c H(z)H(z^{-1})z^{-1}dz$$

Substituting $H(z)$ *and* $H(z^{-1})$ *values in the above equation we get*

$$\pm_\in^2 = \pm_e^2 \frac{1}{2\circledR\, j}\int_c \frac{z}{z-a}\,\frac{z^{-1}}{z^{-1}-a}\,z^{-1}dz$$

$$\pm_\in^2 = \pm_e^2 \frac{1}{2\circledR\, j}\int_c \frac{z^{-1}}{(z-a)(z^{-1}-a)}dz$$

$$= \pm_e^2 \left[\begin{array}{l} residue\ of\ \dfrac{z^{-1}}{(z-a)(z^{-1}-a)}\ at\ z = a \\ +residue\ of\ \dfrac{z^{-1}}{(z-a)(z^{-1}-a)}\ at\ z = 1/a \end{array} \right]$$

$$= \pm_e^2 \left[(z-a)\frac{z^{-1}}{(z-a)(z^{-1}-a)}\bigg|_{z=a} \right]$$

$$= \pm_e^2 \frac{a^{-1}}{a^{-1}-a} = \pm_e^2 \frac{1}{1-a^2}$$

****

The output of the A/D converter is applied to a digital filter with the system function

$$H(Z) = \frac{0.45Z}{Z-0.72}$$

Find the output noise power of the digital filter, when the input signal is quantized to 7 bits.

Given:

$$H(Z) = \frac{0.45Z}{Z-0.72}$$

Solution:

$$\begin{aligned}
H(Z)H\left(Z^{-1}\right)Z^{-1} &= \frac{0.45Z}{Z-0.72} \times \frac{0.45Z^{-1}}{Z^{-1}-0.72} \times Z^{-1} \\
&= \frac{0.45^2\, Z^{-1}}{(Z-0.72)\left(\frac{1}{Z}-0.72\right)} \\
&= \frac{0.2025Z^{-1}}{(Z-0.72)\left(\frac{1-0.72Z}{Z}\right)} \\
&= \frac{0.2025Z^{-1}Z}{(Z-0.72)\left(Z-\frac{1}{0.72}\right)} \\
&= \frac{-0.28125}{(Z-0.72)(Z-1.3889)}
\end{aligned}$$

Now the poles of $H(Z)H(Z^{-1})Z^{-1}$ are $p_1=0.72$, $p_2=1.3889$

Output noise power due to input quantization

$$\begin{aligned}
\pm_{eo}^2 &= \pm_e^2 \left[\frac{1}{2®j} \oint H(Z)H\left(Z^{-1}\right)Z^{-1}dZ\right] \\
&= \pm_e^2 \sum_{i=1}^{N} \operatorname{Re} s\left[H(Z)H\left(Z^{-1}\right)Z^{-1}\right]\Big|_{z=p_i} \\
&= \pm_e^2 \sum_{i=1}^{N} \operatorname{Re} s\left[H(Z)H\left(Z^{-1}\right)Z^{-1}\right]\Big|_{z=p_i}
\end{aligned}$$

Where $p_1, p_2, \ldots . p_n$ are the poles of $H(Z)H(Z^{-1})\, Z^{-1}$ that lies inside the unit circle in z-plane.

$$\begin{aligned}
\pm_{eo}^2 &= \pm_e^2 \times (Z-0.72) \times \frac{-0.28125}{(Z-0.72)(Z-1.3889)}\Big|_{Z=0.72} \\
&= \pm_e^2 \times \frac{-0.28125}{0.72-1.3889} \\
&= 0.4205 \pm_e^2
\end{aligned}$$

****

Consider the transfer function $H(z) = H_1(z)H_2(z)$ **where** $H_1(z) = \frac{1}{1-a_1z^{-1}}$ *and* $H_2(z) = \frac{1}{1-a_2z^{-1}}$

Find the output round off noise power. Assume $\sim_1 = 0.5\,and\,\sim_2 = 0.6$ and find output round off noise power.

Solution:

The round off noise model for $H(z) = H_1(z)H_2(z)$ is given by,

From the realization we can find that the noise transfer function seen by noise source e$_1$(n) is H(z), where,

$$H(z) = \frac{1}{\left(1-a_1z^{-1}\right)\left(1-a_2z^{-1}\right)} ------------------------(1)$$

Whereas, the noise transfer function seen by $e_2(n)$ is,

$$H_2(z) = \frac{1}{\left(1-a_2z^{-1}\right)} ----------------------------(2)$$

The total steady state noise variance can be obtained, we have

$$\pm_0{}^2 = \pm_{01}{}^2 + \pm_{02}{}^2 -----------------------------(3)$$

$$\pm_{01}{}^2 = \frac{1}{2\text{®}j}\oint_c H(z)H(z^{-1})z^{-1}dz$$

$$= \pm_e{}^2\frac{1}{2\text{®}j}\oint_c \frac{1}{1-a_1z^{-1}}\frac{1}{1-a_2z^{-1}}\frac{1}{1-a_1z}\frac{1}{1-a_2z}z^{-1}dz$$

$$= \pm_e{}^2\left[\sum of\ residue\,of\ H(z)H(z^{-1})z^{-1}\ at\ po1es\ z = a_1, z = a_2, z = \frac{1}{a_1}\,and\ z = \frac{1}{a_2}\right]$$

If a1 and a2 are less than the poles z=1/a$_1$ and z=1/a$_2$ lies outside of the circle $|z| = 1$. So, the residue of H(z) H(z^{-1}) z^{-1}at z=1/a$_1$ and z=1/a$_2$ are zero. Consequently we have,

$$\pm_{01}{}^2 = \left[\sum of\ residue\,of\ H(z)H(z^{-1})z^{-1}\ at\ po1es\ z = a_1, z = a_2\right]$$

$$= \left[(z-a_1)\frac{z^{-1}}{\left(1-a_1z^{-1}\right)\left(1-a_2z^{-1}\right)\left(1-a_1z\right)\left(1-a_2z\right)}\bigg|_{z=a_1} + (z-a_2)\frac{z^{-1}}{\left(1-a_1z^{-1}\right)\left(1-a_2z^{-1}\right)\left(1-a_1z\right)\left(1-a_2z\right)}\bigg|_{z=a_2}\right]$$

$$= \pm_e{}^2\left[\frac{1}{\left(1-\dfrac{a_2}{a_1}\right)\left(1-a_2{}^2\right)\left(1-a_1a_2\right)} + \frac{1}{\left(1-\dfrac{a_2}{a_1}\right)\left(1-a_1a_2\right)\left(1-a_2{}^2\right)}\right]$$

$$\pm_{01}{}^2 = \pm_e{}^2\left[\frac{a_1}{a_1-a_2}.\frac{1}{1-a_1{}^2}.\frac{1}{1-a_1a_2} + \frac{a_2}{a_2-a_1}.\frac{1}{1-a_2{}^2}.\frac{1}{1-a_1a_2}\right]---------------(4)$$

In the same way,

$$\pm_{02}{}^2 = \frac{\pm_e{}^2}{2\text{®}j}\oint_c H_2(z)H_2(z^{-1})z^{-1}dz$$

$$= \frac{\pm_e{}^2}{2\text{®}j}\oint_c \frac{1}{1-a_2z^{-1}}\frac{1}{1-a_2z}z^{-1}dz$$

$$= \pm_e{}^2\left[(z-a_2)\frac{z^{-1}}{\left(1-a_2z^{-1}\right)\left(1-a_2z\right)}\bigg|_{z=a_2}\right]$$

$$= \pm_e{}^2\left[\left(z-a_2z^{-1}\right)\frac{z^{-1}}{\left(1-a_2z^{-1}\right)\left(1-a_2z\right)}\bigg|_{z=a_2}\right]$$

$$= \pm_e{}^2\left[\frac{1}{1-a_2{}^2}\right]---------------------------------------(5)$$

$$\pm_0{}^2 = \pm_e{}^2\left[\frac{1}{1-a_2{}^2} + \frac{a_1}{a_1-a_2}.\frac{1}{1-a_1{}^2}.\frac{1}{1-a_1a_2} + \frac{a_2}{a_2-a_1}.\frac{1}{1-a_2{}^2}.\frac{1}{1-a_1a_2}\right]$$

$$= \pm_e{}^2\left[\frac{1}{1-a_2{}^2} + \frac{a_1\left(1-a_2{}^2\right)-a_2{}^2\left(1-a_1{}^2\right)}{\left(1-a_1{}^2\right)\left(1-a_2{}^2\right)\left(1-a_1a_2\right)\left(a_1-a_2\right)}\right]$$

$$= \pm_e{}^2\left[\frac{1}{1-a_2{}^2} + \frac{\left(a_1-a_2\right)\left(1+a_1a_2\right)}{\left(1-a_1{}^2\right)\left(1-a_2{}^2\right)\left(1-a_1a_2\right)\left(a_1-a_2\right)}\right]$$

$$= \frac{2^{-2b}}{12}\left[\frac{1}{1-a_2{}^2} + \frac{\left(1+a_1a_2\right)}{\left(1-a_1{}^2\right)\left(1-a_2{}^2\right)\left(1-a_1a_2\right)}\right]$$

The steady state noise power for $a_1 = 0.5, a_2 = 0.6$ is given by

$$= \frac{2^{-2b}}{12}\left[\frac{1}{1-(0.6)^2} + \frac{1+(0.5)(0.6)}{\left(1-(0.5)^2\right)\left(1-(0.6)^2\right)\left(1-0.6(0.5)\right)}\right]$$

$$= \frac{2^{-2b}}{12}(5.4315)$$

****

Draw the quantization noise model for a second order system $H(z) = \dfrac{1}{1-2r\cos¯ z^{-1}+r^2z^{-2}}$ **and find the steady state output noise variance.**

Solution:

Given:

$$H(z) = \frac{1}{1-2r\cos¯ z^{-1}+r^2z^{-2}}$$

The quantization noise model is,

we know, $\pm_0{}^2 = \pm_{01}{}^2 + \pm_{02}{}^2$

Both noise sources see the same transfer function

$$H(z) = \frac{1}{1-2r\cos¯ z^{-1}+r^2z^{-2}}$$

The impulse response of the transfer function is given by

$$h(n) = r^n\,\frac{\sin(n+1)¯}{\sin¯}\,u(n)$$

Now the steady state output noise variance is,

$$\pm_0{}^2 = \pm_{01}{}^2 + \pm_{02}{}^2$$

But $\pm_{01}{}^2 = \pm_{02}{}^2 = \pm_e{}^2\sum_{n=-\infty}^{\infty}h^2(n), \textit{which gives us}$

$$\pm_0{}^2 = 2.\frac{2^{-2b}}{12}\sum_{n=0}^{\infty} r^{2n}\,\frac{\sin^2(n+1)\text{¯}}{\sin^{2}\text{¯}}$$

$$= 2.\frac{2^{-2b}}{12}\,\frac{1}{2\sin^{2}\text{¯}}\sum_{n=0}^{\infty} r^{2n}\left[1-\cos 2(n+1)\text{¯}\right] \qquad \therefore \mathit{cos2}\text{¯} = \mathit{1}-\mathit{2sin}^{2\text{¯}}$$

$$= \frac{2^{-2b}}{6}\,\frac{1}{2\sin^{2}\text{¯}}\left[\sum_{n=0}^{\infty} r^{2n} - \sum_{n=0}^{\infty} r^{2n}\cos 2(n+1)\text{¯}\right]$$

$$= \frac{2^{-2b}}{6}\,\frac{1}{2\sin^{2}\text{¯}}\left[\frac{1}{1-r^2} - \frac{1}{2}\left(\sum_{n=0}^{\infty} r^{2n}e^{j2(n+1)\text{¯}} + \sum_{n=0}^{\infty} r^{2n}e^{-j2(n+1)\text{¯}}\right)\right]$$

$$= \frac{2^{-2b}}{6}\,\frac{1}{2\sin^{2}\text{¯}}\left[\frac{1}{1-r^2} - \frac{1}{2}\left(\frac{e^{j2\text{¯}}}{1-r^2e^{2j\text{¯}}} + \frac{e^{-j2\text{¯}}}{1-r^2e^{-2j\text{¯}}}\right)\right]$$

$$= \frac{2^{-2b}}{6}\,\frac{1}{2\sin^{2}\text{¯}}\left[\frac{1}{1-r^2} - \frac{\cos 2\text{¯} - r^2}{1-2r^2\cos 2\text{¯} + r^4}\right]$$

$$= \frac{2^{-2b}}{6}\,\frac{1}{2\sin^{2}\text{¯}}\left[\frac{(1+r)^2(1-\cos 2\text{¯})}{(1-r^2)(1-2r^2\cos 2\text{¯} + r^4)}\right]$$

$$= \frac{2^{-2b}}{6}\,\frac{(1+r)^2}{(1-r^2)(1-2r^2\cos 2\text{¯} + r^4)}$$

****

Co-efficient quantization error

- We know that the IIR Filter is characterized by the system function

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

- After quantizing ,

$$[H(Z)]_q = \frac{\sum_{k=0}^{M} [b_k]_q z^{-k}}{1+\sum_{k=1}^{N} [a_k]_q z^{-k}}$$

Where $[a_k]_q = a_k + \Delta a_k$

$[b_k]_q = b_k + \Delta b_k$

- The quantization of filter coefficients alters the positions of the poles and zeros in z-plane.
  1. If the poles of desired filter lie close to the unit circle, then the quantized filter poles may lie outside the unit circle leading into instability of filter.
  2. Deviation in poles and zeros also lead to deviation in frequency response.

****

Consider a second order IIR filter with $H(z) = \frac{1.0}{(1-0.5z^{-1})(1-0.45z^{-1})}$ find the effect on quantization on pole locations of the given system function in direct form and in cascade form. Take b=3bits. [Apr/May-10] [Nov/Dec-11]

Solution:

Given that,

$$H(z) = \frac{1.0}{(1-0.5z^{-1})(1-0.45z^{-1})}$$

$$H(z) = \frac{1}{z^{-1}(z-0.5z^{-1})z^{-1}(z-0.5)}$$

$$= \frac{z^{2}}{(z-0.5)(z-0.45)}$$

The roots of the denominator of H(z) are the original poles of H(z). let the original poles of H(z) be p_1 and p_2.

Here p_1=0.5 and p_2=0.45

Direct form I:

$$H(z) = \frac{1.0}{(1-0.5z^{-1})(1-0.45z^{-1})}$$

$$H(z) = \frac{1}{1-0.5z^{-1}-0.45z^{-1}+0.225z-2}$$

$$= \frac{1}{1-0.95z^{-1}+0.225z^{-2}}$$

Let us quantize the coefficients by truncation.

$$.95_{10} \xrightarrow[\text{Binary}]{\text{Convert to}} .1111_2 \xrightarrow[\text{3-bits}]{\text{Truncate to}} .111_2 \xrightarrow[\text{decimal}]{\text{Convert to}} .875_{10}$$

$$.225_{10} \xrightarrow[\text{Binary}]{\text{Convert to}} .0011_2 \xrightarrow[\text{3-bits}]{\text{Truncate to}} .001_2 \xrightarrow[\text{decimal}]{\text{Convert to}} .125_{10}$$

Let $\overline{H}(z)$ be the transfer function of the IIR system after quantizing the coefficients.

$$\overline{H}(z) = \frac{1}{1-0.875z^{-1}+0.125z^{-2}}$$

$$let\ \overline{H}(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-0.875z^{-1}+0.125z^{-2}}$$

On cross multiplying the above equation we get,

$$Y(z)-0.875z^{-1}Y(z)+0.125z^{-2}Y(z) = X(z)$$

$$Y(z) = X(z)+0.875z^{-1}Y(z)-0.125z^{-2}Y(z)$$

<u>Cascade form</u>:

Given that

$$H(z) = \frac{1.0}{(1-0.5z^{-1})(1-0.45z^{-1})}$$

In cascade realization the system can be realized as cascade of first order sections.

$H(z)=H_1(z)+H_2(z)$

Where,

$$H_1(z) = \frac{1}{1-0.5z^{-1}}\ and\ \ H_2(z) = \frac{1}{1-0.45z^{-1}}$$

Let us quantize the coefficients of $H_1(z)$ and $H_2(z)$ by truncation.

$$.5_{10} \xrightarrow[\text{Binary}]{\text{Convert to}} .1000_2 \xrightarrow[\text{3-bits}]{\text{Truncate to}} .100_2 \xrightarrow[\text{decimal}]{\text{Convert to}} .5_{10}$$

$$.45_{10} \xrightarrow[\text{Binary}]{\text{Convert to}} .0111_2 \xrightarrow[\text{3-bits}]{\text{Convert to}} .011_2 \xrightarrow[\text{decimal}]{\text{Convert to}} .375_{10}$$

let , $\overline{H_1}(z)$ and $\overline{H_2}(z)$ be the transfer function of the first-order sections after quantizing the coefficients.

$$\overline{H_1}(\mathrm{z})=\frac{1}{1-0.5z^{-1}}$$

$$\overline{H_2}(\mathrm{z})=\frac{1}{1-0.375z^{-1}}$$

$$let,\overline{H_1}(\mathrm{z})=\frac{Y_1(\mathrm{z})}{X(\mathrm{z})}=\frac{1}{1-0.5z^{-1}}$$

$$Y_1(\mathrm{z})-0.5\,\mathrm{z}^{-1}\,Y_1(\mathrm{z})=\mathrm{X(z)}$$

$$Y_1(\mathrm{z})=X(\mathrm{z})+0.5\,\mathrm{z}^{-1}\,Y_1(\mathrm{z})$$

$$\mathrm{let},\overline{H_2}(\mathrm{z})=\frac{Y(\mathrm{z})}{Y_1(\mathrm{z})}=\frac{1}{1-0.375z^{-1}}$$

on cross multiplying the above equation we get,

$$Y(\mathrm{z})-0.375\,\mathrm{z}^{-1}\,Y(\mathrm{z})=\mathrm{Y}_1(\mathrm{z})$$

$$Y(\mathrm{z})=\mathrm{Y}_1(\mathrm{z})+0.375\,\mathrm{z}^{-1}\mathrm{Y(z)}$$

****

Round off effects and overflow in digital filter:

***Explain in detail about round off effects in digital filters.**

- The presence of one or more quantizer in the realization of a digital filter results in a non-linear device. i.e. recursive digital filter may exhibit undesirable oscillations in its output
- In the finite arithmetic operations, some registers may overflow if the input signal level becomes large.
- These overflow represents non-linear distortion leading to limit cycle oscillations
- There are two types of limit cycle oscillations which includes
    1. Zero input limit cycle oscillations (Low amplitude compared to overflow limit cycle oscillations)
    2. Over flow limit cycle oscillations.

<u>Zero input limit cycle oscillations</u>

- The arithmetic operations produces oscillations even when the input is zero or some non zero constant values. Such oscillations are called zero input limit cycle oscillations.

<u>Overflow limit cycle oscillations</u>

- The limit cycle occurs due to the overflow of adder is known as overflow limit cycle oscillations.

<u>Dead Band:</u>

The limit cycle occurs as a result of quantization effect in multiplication. The amplitude of the output during a limit cycle is confined to a range of values called the dead band of the filter**.**

$$|y(n-1)|\le\frac{\frac{2^{-b}}{2}}{(1-|a|)}$$

Consider a first order filter

$$y(n)=ay(n-1)+x(n);\qquad n>0$$

After rounding the product

$$y_q(n)=Q[a*y(n-1)]+x(n);$$

The round off error

$$-\frac{2^{-b}}{2}\le e_r\le\frac{2^{-b}}{2}$$

where, e_r →difference between the quantized value and the actual value.

$$Q[ay(n-1)-ay(n-1)]\le\frac{2^{-b}}{2}$$

The dead band of the filter for the limit cycle oscillations are

$$Q[ay(n-1)]=\begin{cases}y(n-1) & a>0\\ -y(n-1) & a<0\end{cases}$$

$$|y(n-1)| - a|y(n-1)| \le \frac{2^{-b}}{2}$$

$$y(n-1)\,(1-|a|) \le \frac{2^{-b}}{2}$$

Dead band of the filter, $|y(n-1)| \le \dfrac{\frac{2^{-b}}{2}}{(1-|a|)}$

****

Problem: Consider a 1st order FIR system equation $y(n) = x(n) + ay(n-1)$ **with**

$$x(n) = \begin{cases} 0.875 & ,\ n = 0 \\ 0 & ,\ otherwise \end{cases}$$

Find the limit cycle effect and the dead band. Assume b=4 and a=0.95. (Nov/Dec-12)(Nov/Dec-15) [May/June-2016]

Solution:

Given:

$$x(n) = \begin{cases} 0.875 & ,\ n = 0 \\ 0 & ,\ otherwise \end{cases}$$

$$Dead\ band = \frac{2^{-b}}{2(1-|a|)} = \frac{2^{-4}}{2(1-|0.95|)} = 0.625$$

$$y(n) = x(n) + 0.95y(n-1)$$

n	***x(n)***	$y(n-1)$	$ay(n-1)$	$Q[ay(n-1)]$ *(round off to 4-bits)*	$y(n) = x(n) + Q[ay(n-1)]$
0	0.875	0	0	0.0000	y(0)=0.875
1	0	0.875	$0.875*0.95$ $=(0.83125)_{10}$ $=(0.11010)_2$	$=(0.1101)_2$ $=0.8125$	y(1)=0.8125
2	0	0.8125	$0.8125*0.95$ $=(0.77187)_{10}$ $=(0.110001)_2$	$=(0.1100)_2$ $=0.75$	$y(2) = 0.75$
3	0	0.75	$0.75*0.95$ $=(0.7125)_{10}$ $=(0.1011011)_2$	$=(0.1011)_2$ $=0.6875$	$y(3) = 0.6875$
4	0	0.6875	$0.6875*0.95$ $=(0.653125)_{10}$ $=(0.101001)_2$	$=(0.1010)_2$ $=0.625$	$y(4) = 0.625$
5	0	0.625	$0.625*0.95$ $=(0.59375)_{10}$ $=(0.10011)_2$	$=(0.1010)_2$ $=0.625$	$y(5) = 0.625$
6	0	0.625	$0.625*0.95$ $=(0.59375)_{10}$ $=(0.10011)_2$	$=(0.1010)_2$ $=0.625$	$y(6) = 0.625$

Conclusion:

The dead band of the filter is 0.625. When $n \geq 5$ the output remains constant at 0.625 causing limit cycle oscillations.

****

Overflow Limit cycle oscillations:

***What are called overflow oscillations? How it can be prevented?**

- We know that the limit cycle oscillation is caused by rounding the result of multiplication.
- The limit cycle occurs due to the overflow of adder is known as overflow limit cycle oscillations.\
- Several types of limit cycle oscillations are caused by addition, which makes the filter output oscilate between maximum and minimum amplitudes.
- Let us consider 2 positive numbers n_1 & n_2

  n_1=0.111→7/8

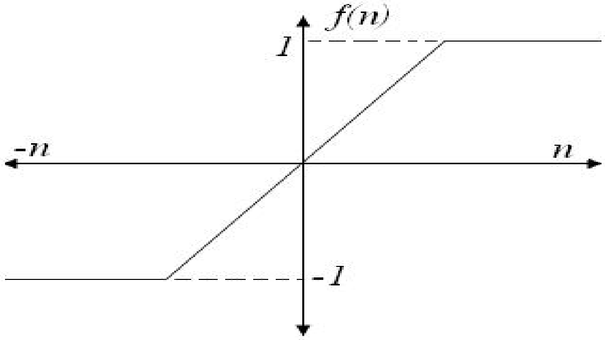
  n_2=0.110→6/8

  n_1 + n2=1.101→-5/8 in sign magnitude form.

  The sum is wrongly interpreted as a negative number.
- The transfer characteristics of an saturation adder is shown in fig below

  where n → The input to the adder

  f(n) → The corresponding output



Saturation adder transfer characteristics

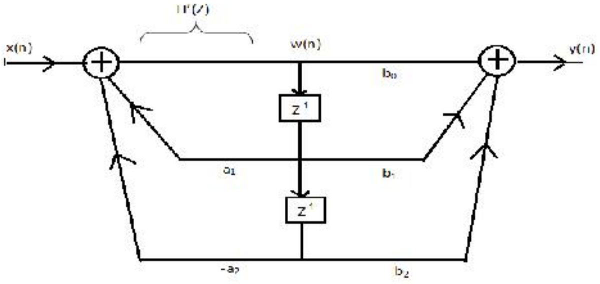
- From the transfer characteristics, we find that when overflow occurs, the sum of adder is set equal to the maximum value.

****

Signal Scaling:

Explain how reduction of round-off errors is achieved in digital filters.

- Saturation arithmetic eliminates limit cycles due to overflow, but it causes undeniable signal distortion due to the non linearity of the clipper.
- In order to limit the amount of non linear distortion, it is important to scale input signal and unit sample response between input and any internal summing node in the system to avoid overflow.



Realization of a second order IIR Filter

- Let us consider a second order IIR filter as shown in the above figure. Here a scale factor S_0 is introduced between the input x(n) and the adder 1 to prevent overflow at the output of adder 1.

- Now the overall input-output transfer function is

Now the transfer function

$$H(z) = S_0 \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$= S_0 \frac{N(z)}{D(z)}$$

From figure

$$H^{'}(z) = \frac{W(z)}{X(z)} = \frac{S_0}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{S_0}{D(z)}$$

$$W(z) = \frac{S_0 X(z)}{D(z)} = S_0\, \mathrm{S}(z)\, \mathrm{x(Z)}$$

$$\text{Where}\, S(z) = \frac{1}{D(z)}$$

we have

$$w(n) = \frac{S_0}{2\text{®}} \int S(e^{j\text{¯}}) X(e^{j\text{¯}})(e^{jn\text{¯}})\, \mathrm{d}\text{¯}$$

$$\mathrm{w(n)}\ ^{2} = \frac{S_0^2}{2\text{®}^{\,2}} \left| \int S(e^{j\text{¯}}) X(e^{j\text{¯}})(e^{jn\text{¯}})\, \mathrm{d}\text{¯} \right|^2$$

Using Schwartz inequality

$$\mathrm{w(n)}\ ^{2} \le S_0^2 \left[\int_{2\text{®}} \left| S(e^{j\text{¯}}) \right|^2 \mathrm{d}\text{¯} \right] \left[\int_{2\text{®}} \left| X(e^{j\text{¯}}) \right|^2 \mathrm{d}\text{¯} \right]$$

Applying parsevals theorem

$$\mathrm{w(n)}\ ^{2} \le S_0^2 \sum_{n=0}^{\infty} x^2(n) \frac{1}{2\text{®}} \int_{2\text{®}} \left| S(e^{j\text{¯}}) \right|^2 \mathrm{d}\text{¯}$$

$$if\;\; z = e^{j\text{¯}} \;\;\; then\;\; dz = \; je^{j\text{¯}}\;\; d\text{¯}$$

which gives

$$d\text{¯} = \frac{dz}{jz}$$

By substituting all values

$$\mathrm{w(n)}\ ^{2} \le S_0^2 \sum_{n=0}^{\infty} x^2(n) \frac{1}{2\text{®}\, j} \int_c \left| S(z) \right|^2 \mathrm{z}^{-1} \mathrm{d}\, z$$

$$\mathrm{w(n)}\ ^{2} \le S_0^2 \sum_{n=0}^{\infty} x^2(n) \frac{1}{2\text{®}\, j} \int_c S(z) S(z^{-1})\, \mathrm{z}^{-1} \mathrm{d}\, z$$

$$w^2(n) \le \sum_{n=0}^{\infty} x^2(n) \;\; when$$

$$S_0^2 \frac{1}{2\text{®}\, j} \int_c S(z) S(z^{-1})\, \mathrm{d}\, z = 1$$

Which gives us,

$$S_0^2 = \frac{1}{\frac{1}{2® j}\int_c S(z)S(z^{-1})\,\mathrm{z}^{-1}\mathrm{d}\,z}$$

$$= \frac{1}{\frac{1}{2® j}\int_c \frac{z^{-1}dz}{D(z)D(z^{-1})}}$$

$$S_0^2 = \frac{1}{I}$$

Where I=

$$\frac{1}{2\pi j}\int_c \frac{z^{-1}dz}{D(z)D(z^{-1})}$$

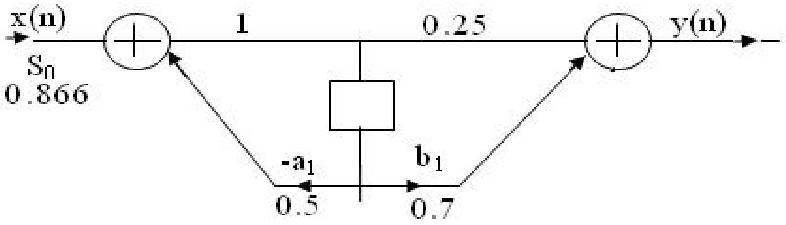
Note:

- Because of the process of scaling, the overflow is eliminated. Here so is the scaling factor for the first stage.
- Scaling factor for the second stage = S_{01} and it is given by $S_{01}^2 = \frac{1}{S_0^2 I_2}$

  Where $I_2 = \frac{1}{2®j}\oint_c \frac{H_1(Z)H_1(Z^{-1})Z^{-1}}{D_2(Z)D_2(Z^{-1})}dZ$

****

For the given transfer function, $H(Z) = \frac{0.25 + 0.7Z^{-1}}{1 - 0.5Z^{-1}}$**, find scaling factor so as to avoid overflow in the adder '1' of the filter.**



Given:

D(Z) = 1-0.5Z^{-1}

D(Z^{-1}) = 1-0.5 Z

Solution:

$$I = \frac{1}{2®j}\oint_c \frac{1}{D(Z)D(Z^{-1})}\frac{dZ}{Z}$$

$$= \frac{1}{2®j}\oint_c \frac{1}{(1-0.5Z^{-1})(1-0.5Z)}\frac{dZ}{Z}$$

$$= \frac{1}{2®j}\oint_c \frac{Z}{(Z-0.5)}\frac{1}{(1-0.5Z)}\frac{dZ}{Z}$$

Residue of $\left.\frac{Z}{(Z-0.5)}\frac{1}{(1-0.5Z)}\right|_{Z=0.5} + 0$

I=1.3333

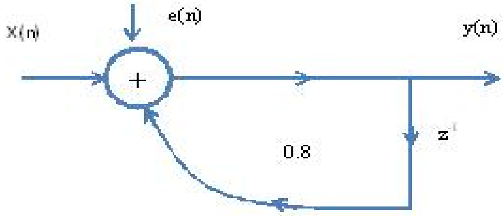
$$S_0 = \frac{1}{\sqrt{I}}$$

$$S_0 = \frac{1}{\sqrt{1.333}}$$

$$= 0.866$$

****

Consider the recursive filter shown in fig. The input x(n) has a range of values of ±100V, represented by 8 bits. Compute the variance of output due to A/D conversion process. (6)



Solution:

Given the range is ±100V

The difference equation of the system is given by $y(n) = 0.8y(n-1) + x(n)$, whose impulse response h(n) can be obtained as

$$h(n) = (0.8)^n u(n)$$

$$quantization\ step\ size = \frac{range\,of\ \ the\ signal}{No.of\ quantization\ levels}$$

$$= \frac{200}{2^8}$$

$$= 0.78125$$

$Variance\ of\ \ the\ error\ signal$

$$\pm_e^2 = \frac{q^2}{12}$$

$$= \frac{(0.78125)^2}{12}$$

$$\pm_e^2 = 0.05086$$

$Variance\ of\ output$

$$\pm_{\notin}^2 = \pm_e^2 \sum_{n=0}^{\sim} h^2(n)$$

$$= (0.05086)\sum_{n=0}^{\sim}(0.8)^{2n}$$

$$= \frac{0.05086}{1-(0.8)^2} = 0.14128$$

****

The input to the system y(n)=0.999y(n-1)+x(n) is applied to an ADC. What is the power produced by the quantization noise at the output of the filter if the input is quantized to a) 8 bits b)16 bits. May-07

Solution:

y(n)=0.999y(n-1)+x(n)

Taking z-transform on both sides

Y(z)=0.999z^{-1}Y(z)+X(z)

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-0.999z^{-1}}$$

$$H(z)H(z^{-1})z^{-1} = (\frac{z}{z-0.999})(\frac{z^{-1}}{z^{-1}-0.999})z^{-1}$$

$$= \frac{z^{-1}}{(z-0.999)(-0.999)(z-\frac{1}{0.999})}$$

$$= \frac{-0.001}{(z-0.999)(z-0.001)}$$

$$\left.\begin{matrix} \textit{output noise power due} \\ \textit{to input quantization} \end{matrix}\right\} \pm^{2}_{eoi} = \pm^{2}_{e} \frac{1}{2\text{®}\ j} \int_{c} H(z)H(z^{-1})z^{-1}dz$$

$$= \pm^{2}_{e} \sum_{i=1}^{N} \mathrm{Re}\,s\left[H(z)H(z^{-1})z^{-1} \right]\Big|_{z=p_i}$$

$$= \pm^{2}_{e} \sum_{i=1}^{N} \left[(z = p_i)H(z)H(z^{-1})z^{-1} \right]\Big|_{z=p_i}$$

Where $p_1, p_2, \ldots\ldots p_N$ are poles of $H(z)H(z^{-1})z^{-1}$, that lies inside the unit circle in z-plane.

$$\sigma_{eoi^2} = \sigma_{e^2}(z-0.999)(\frac{0.001}{(z-0.999)(z-0.001)})\Big|_{z=0.999}$$

$$= \sigma_{e^2} 500.25$$

a) b+1=8 bits(Assuming including sign bit)

$$\pm^{2}_{\pounds} = \frac{2^{2(7)}}{12}(500.25) = 2.544\times10^{-3}$$

b) b+1=16 bits

$$\pm^{2}_{\pounds} = \frac{2^{2(15)}}{12}(500.25) = 3.882\times10^{-8}$$

****

Find the effect of coefficient quantization on pole locations of the given second order IIR system, when it is realized in direct form I and in cascade form. Assume a word length of 4 bits through truncation.

$$H(z) = \frac{1}{1-0.9z^{-1}+0.2z^{-2}}$$

Solution:

<u>Direct form I</u>

Let b=4 bits including a sign bit

$(0.9)_{10} = (0.111001\ldots)_2$

$Integer \ \ part$

$\frac{0.9\times 2}{1.8}$

$\mapsto \qquad 1 \qquad \downarrow$

$\frac{0.8\times 2}{1.6}$

$\mapsto \qquad 1$

$\frac{0.6\times 2}{1.2} \qquad 1$

$\mapsto$

$\frac{0.2\times 2}{0.4}$

$\mapsto \qquad 0$

$\frac{0.4\times 2}{0.8}$

$\mapsto \qquad 0$

$\frac{0.8\times 2}{1.6}$

$\mapsto \qquad 1$

After truncation we get

$(0.111)_2=(0.875)_{10}$

$(0.2)_{10}=(0.00110\ldots)_2$

$(0.2)_{10} = \frac{0.2\times 2}{0.4}$

$\mapsto \qquad 0 \qquad \downarrow$

$\frac{0.4\times 2}{0.8}$

$\mapsto \qquad 0$

$\frac{0.8\times 2}{1.6}$

$\mapsto \qquad 1$

$\frac{0.6\times 2}{1.2}$

$\mapsto \qquad 1$

$\frac{0.2\times 2}{0.4}$

$\mapsto \qquad 0$

After truncation we get

$(0.001)_2=(0.125)_{10}$

The system function after coefficient quantization is

$$H(z) = \frac{1}{1-0.875z^{-1}+0.125z^{-2}}$$

Now the pole locations are given by

$z_1=0.695$

$z_2=0.178$

If we compare the Poles of H(z) and $\overline{H}(z)$ we can observe that the poles of $\overline{H}(z)$ deviate very much from the original poles.

<u>Cascade form</u>

$$H(z) = \frac{1}{1-0.5z^{-1}(1-0.4z^{-1})}$$

$(0.5)_{10} = (0.1000)_2$

After truncation we get

$(0.100)_2=(0.5)_{10}$

After truncation we get

$(0.011)_2=(0.375)_{10}$

$$
\begin{array}{lll}
(0.4)_{10} = \dfrac{0.4\times 2}{0.8} & & \\
\mapsto & 0 & \downarrow \\
\dfrac{0.8\times 2}{1.6} & & \\
\mapsto & 0 & \\
\dfrac{0.6\times 2}{1.2} & & \\
\mapsto & 1 & \\
\dfrac{0.2\times 2}{0.4} & & \\
\mapsto & 1 & \\
\dfrac{0.4\times 2}{0.8} & & \\
\mapsto & 0 &
\end{array}
$$

$(0.4)_{10}=(0.01100\ldots)_2$

The system function after coefficient quantization is

$$H(z) = \frac{1}{(1-0.5z^{-1})(1-0.375z^{-1})}$$

The pole locations are given by

$z_1=0.5$

$z_2=0.375$

on comparing the poles of the cascade system with original poles we can say that one of the poles is same and other pole is very close to original pole.

****

A LTI system is characterized by the difference equation y(n)=0.68y(n-1)+0.5x(n). The input signal x(n) has a range of -5V to +5V, represented by 8-bits. Find the quantization step size, variance of the error signal and variance of the quantization noise at the output.

Solution:

Given

Range R=-5V to +5V = 5-(-5) =10

Size of binary, B= 8 bits (including sign bit)

Quantization step size,

$$q = \frac{R}{2^8} = \frac{10}{2^8} = 0.0390625$$

$$\text{var}\,iance\,of\ error\ signal, \pm_e^2 = \frac{q^2}{12} = \frac{0.0390625^2}{12} = 1.27116*10^{-4}$$

The difference equation governing the LTI system is

Y (n) =0.68y (n-1) +0.15x (n)

On taking z transform of above equation we get

$$Y(z) = 0.68z^{-1}Y(z) + 0.15X(z)$$

$$Y(z) - 0.68z^{-1}Y(z) = 0.15X(z)$$

$$Y(z)[1 - 0.68z^{-1}] = 0.15X(z)$$

$$\frac{Y(z)}{X(z)} = \frac{0.15}{1 - 0.68z^{-1}}$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{0.15}{1 - 0.68z^{-1}}$$

$$H(z)H(z^{-1})z^{-1} = \frac{0.15}{1 - 0.68z^{-1}} * \frac{0.15}{1 - 0.68z} * z^{-1}$$

$$H(z)H(z^{-1})z^{-1} = \frac{0.225z^{-1}}{\left(1 - \frac{0.68}{z}\right)(-0.68)\left(z - \frac{1}{0.68}\right)}$$

$$H(z)H(z^{-1})z^{-1} = \frac{-0.0331z^{-1}}{\left(\frac{z - 0.68}{z}\right)(z - 1.4706)} = \frac{-0.0331z^{-1}}{(z - 0.68)(z - 1.4706)}$$

Now, poles of H (z) H (z^{-1}) z^{-1} are p_1=0.68, p_2=1.4706

Here, p_1=0.68 is the only pole that lies inside the unit circle in z-plane

Variance of the input quantization noise at the output.

$$\pm^2_{eoi} = \pm^2_e \frac{1}{2\circledR\, j} \int_c H(z)H(z^{-1})z^{-1}dz$$

$$\pm^2_{eoi} = \pm^2_e \sum_{i=1}^{N} \left[\mathrm{Res}\, H(z)H(z^{-1})z^{-1} \right] \Big|_{z=pi}$$

$$\pm^2_{eoi} = \pm^2_e \sum_{i=1}^{N} \left[(z - p_i)H(z)H(z^{-1})z^{-1} \right] \Big|_{z=pi}$$

$$\pm^2_{eoi} = \pm^2_e (z - 0.68) * \frac{-0.0331}{(z - 0.68)(z - 1.4706)} \Big|_{z=0.68}$$

$$\pm^2_{eoi} = \pm^2_e * \frac{-0.0331}{(0.68 - 1.4706)} \Big| = 0.0419\pm^2_e$$

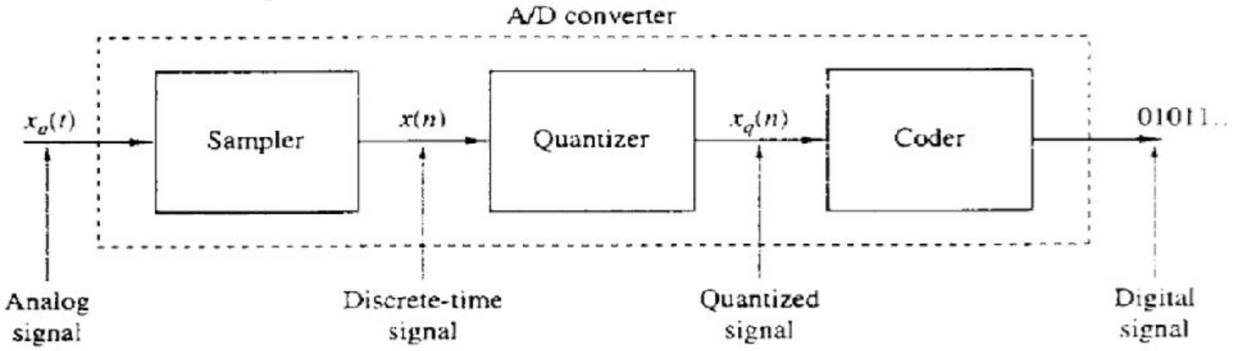
$$\pm^2_{eoi} = 0.0419 * 1.2716 * 10^{-4}$$

$$\pm^2_{eoi} = 5.328 * 10^{-6}$$

Analog to digital conversion:

10. Explain the ADC and DAC in detail.

A/D conversion has three process.



Basic parts of an analog-to digital (A/D) converter

1. Sampling

- Sampling is the conversion of a continuous- tome signal into a discrete-time signal obtained by taking the samples of continuous-time signal at discrete instants.
- Thus if $x_a(t)$ is the input to the sampler, the output is $x_a(nT) \equiv x(n)$, where T is called the sampling interval.

2. Quantisation

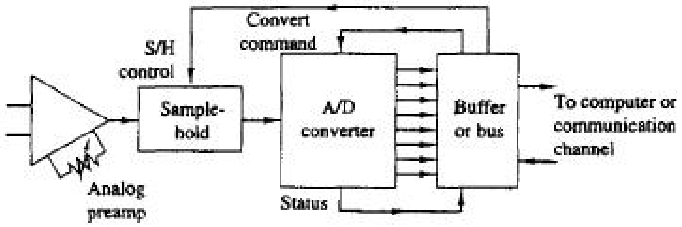
- The process of converting a discrete-time continuous amplitude signal into digital signal is called quantization.
- The value of each signal sample is represented by a value selected from a finite set of possible values.
- The difference between the unquantised sample x(n) and the quantized output $x_q(n)$ is called the quantization error or quantization noise.

$$e_q(n)= x_q(n)-x(n)$$

- To eliminate the excess bits either discard them by the process of truncation or discard them by rounding the resulting number by the process of rounding.
- The values allowed in the digital signals are called the quantization levels
- The distance Δ between two successive quantization levels is called the quantization step size or resolution.
- The quality of the output of the A/D converter is measured by the signal-to-quantization noise ratio.

3. Coding

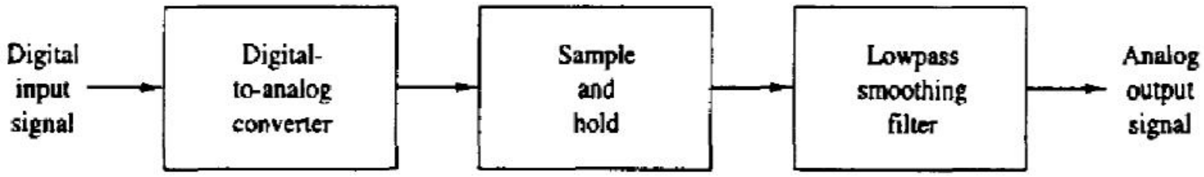
- In the coding process, each discrete value $x_q(n)$ is represented by a b-bit binary sequence.



Block diagram of basic elements of an A/D Converter

Digital to analog conversion:

- To convert a digital signal into an analog signal, digital to analog converters are used.



Basic operations in converting a digital signal into an analog signal

- The D/A converter accepts, at its input, electrical signals that corresponds to a binary word, and produces an output voltage or current that is proportional to the value of the binary word.
- The task of D/A converter is to interpolate between samples.
- The sampling theorem specifies the optimum interpolation for a band limited signal.
- The simplest D/A converter is the zero order hold which holds constant value of sample until the next one is received.
- Additional improvement can be obtained by using linear interpolation to connect successive samples with straight line segment.
- Better interpolation can be achieved y using more sophisticated higher order interpolation techniques.
- Suboptimum interpolation techniques result in passing frequencies above the folding frequency. Such frequency components are undesirable and are removed by passing the output of the interpolator through a proper analog filter which is called as post filter or smoothing filter.
- Thus D/A conversion usually involve a suboptimum interpolator followed by a post filter.

****

UNIT 5
MULTIRATE DIGITAL SIGNAL PROCESSING

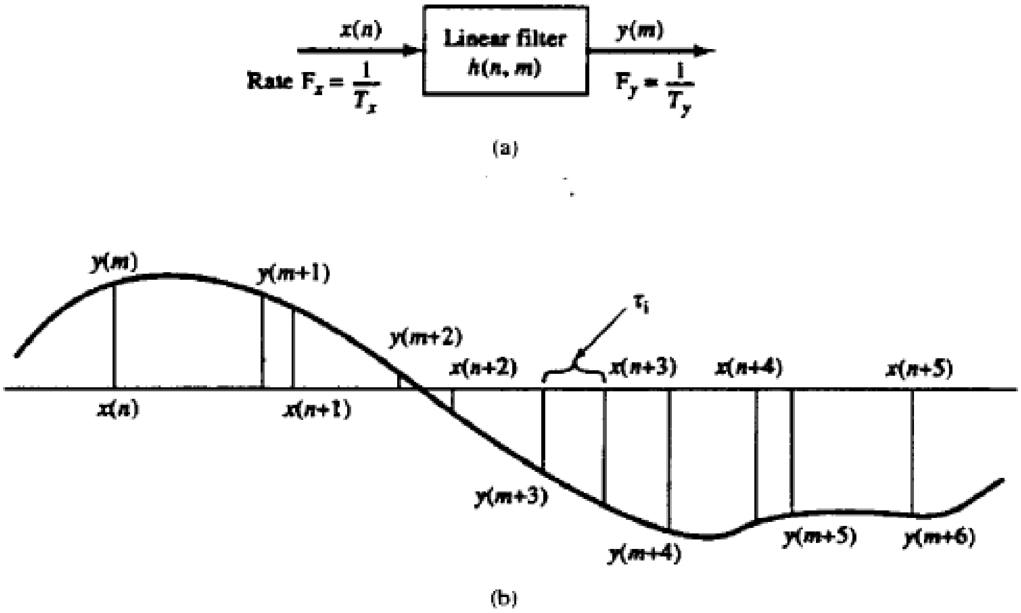
INTRODUCTION

The process of sampling rate conversion in the digital domain can be viewed as a linear filtering operation, as illustrated in Fig. 10.1(a). The input signal $x(n)$ is characterized by the sampling rate $F_x = 1/T_x$ and the output signal $y(m)$ is characterized by the sampling rate $F_y = 1/T_y$, where T_x and T_y are the corresponding sampling intervals. In the main part of our treatment, the ratio F_y/F_x is constrained to be rational,

$$\frac{F_y}{F_x} = \frac{I}{D}$$

where D and I are relatively prime integers. We shall show that the linear filter is characterized by a time-variant impulse response, denoted as $h(n, m)$. Hence the input $x(n)$ and the output $y(m)$ are related by the convolution summation for time-variant systems.

The sampling rate conversion process can also be understood from the point of view of digital resampling of the same analog signal. Let $x(t)$ be the analog signal that is sampled at the first rate F_x to generate $x(n)$. The goal of rate conversion is to obtain another sequence $y(m)$ directly from $x(n)$, which is equal to the sampled values of $x(t)$ at a second rate F_y. As is depicted in Fig. 10.1(b), $y(m)$ is a time-shifted version of $x(n)$. Such a time shift can be



(a)

(b)

DECIMATION BY A FACTOR D

Let us assume that the signal $x(n)$ with spectrum $X(\omega)$ is to be downsampled by an integer factor D. The spectrum $X(\omega)$ is assumed to be nonzero in the frequency interval $0 \le |\omega| \le \pi$ or, equivalently, $|F| \le F_x/2$. We know that if we reduce the sampling rate simply by selecting every Dth value of $x(n)$, the resulting signal will be an aliased version of $x(n)$, with a folding frequency of $F_x/2D$. To avoid aliasing, we must first reduce the bandwidth of $x(n)$ to $F_{\max} = F_x/2D$ or, equivalently, to $\omega_{\max} = \pi/D$. Then we may downsample by D and thus avoid aliasing.

The decimation process is illustrated in Fig. 10.2. The input sequence $x(n)$ is passed through a lowpass filter, characterized by the impulse response $h(n)$ and a frequency response $H_D(\omega)$, which ideally satisfies the condition

$$H_D(\omega) = \begin{cases} 1, & |\omega| \le \pi/D \\ 0, & \text{otherwise} \end{cases} \tag{10.2.1}$$

Thus the filter eliminates the spectrum of $X(\omega)$ in the range $\pi/D < \omega < \pi$. Of course, the implication is that only the frequency components of $x(n)$ in the range $|\omega| \le \pi/D$ are of interest in further processing of the signal.

The output of the filter is a sequence $v(n)$ given as

$$v(n) = \sum_{k=0}^{\infty} h(k)x(n-k) \tag{10.2.2}$$

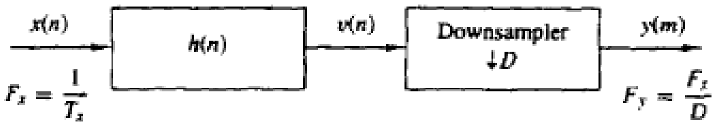


Figure 10.2 Decimation by a factor D.

which is then downsampled by the factor D to produce $y(m)$. Thus

$$\begin{aligned} y(m) &= v(mD) \\ &= \sum_{k=0}^{\infty} h(k)x(mD-k) \end{aligned} \tag{10.2.3}$$

Although the filtering operation on $x(n)$ is linear and time invariant, the downsampling operation in combination with the filtering results in a time-variant system. This is easily verified. Given the fact that $x(n)$ produces $y(m)$, we note that $x(n-n_0)$ does not imply $y(n-n_0)$ unless n_0 is a multiple of D. Consequently, the overall linear operation (linear filtering followed by downsampling) on $x(n)$ is not time invariant.

The frequency-domain characteristics of the output sequence $y(m)$ can be obtained by relating the spectrum of $y(\mathrm{m})$ to the spectrum of the input sequence $x(n)$. First, it is convenient to define a sequence $\tilde{v}(n)$ as

$$\tilde{v}(n) = \begin{cases} v(n), & n = 0, \pm D, \pm 2D, \ldots \\ 0, & \text{otherwise} \end{cases} \tag{10.2.4}$$

Clearly, $\tilde{v}(n)$ can be viewed as a sequence obtained by multiplying $v(n)$ with a periodic train of impulses $p(n)$, with period D, as illustrated in Fig. 10.3. The discrete Fourier series representation of $p(n)$ is

$$p(n) = \frac{1}{D}\sum_{k=0}^{D-1} e^{j2\pi kn/D} \tag{10.2.5}$$

Hence

$$\tilde{v}(n) = v(n)p(n) \tag{10.2.6}$$

and

$$y(m) = \tilde{v}(mD) = v(mD)p(mD) = v(mD) \tag{10.2.7}$$

Now the z-transform of the output sequence $y(m)$ is

$$\begin{aligned} Y(z) &= \sum_{m=-\infty}^{\infty} y(m)z^{-m} \\ &= \sum_{m=-\infty}^{\infty} \tilde{v}(mD)z^{-m} \\ Y(z) &= \sum_{m=-\infty}^{\infty} \tilde{v}(m)z^{-m/D} \end{aligned} \tag{10.2.8}$$

where the last step follows from the fact that $\tilde{v}(m) = 0$, except at multiples of D. By making use of the relations in (10.2.5) and (10.2.6) in (10.2.8), we obtain

$$
\begin{aligned}
Y(z) &= \sum_{m=-\infty}^{\infty} v(m)\left[\frac{1}{D}\sum_{k=0}^{D-1} e^{j2\pi mk/D}\right] z^{-m/D} \\
&= \frac{1}{D}\sum_{k=0}^{D-1}\sum_{m=-\infty}^{\infty} v(m)(e^{-j2\pi k/D} z^{1/D})^{-m} \\
&= \frac{1}{D}\sum_{k=0}^{D-1} V(e^{-j2\pi k/D} z^{1/D}) \\
&= \frac{1}{D}\sum_{k=0}^{D-1} H_D(e^{-j2\pi k/D} z^{1/D}) X(e^{-j2\pi k/D} z^{1/D})
\end{aligned}
\tag{10.2.9}
$$

where the last step follows from the fact that $V(z) = H_D(z)X(z)$.

By evaluating $Y(z)$ in the unit circle, we obtain the spectrum of the output signal $y(m)$. Since the rate of $y(m)$ is $F_y = 1/T_y$, the frequency variable, which we denote as ω_y, is in radians and is relative to the sampling rate F_y,

$$\omega_y = \frac{2\pi F}{F_y} = 2\pi F T_y \tag{10.2.10}$$

Since the sampling rates are related by the expression

$$F_y = \frac{F_x}{D} \tag{10.2.11}$$

it follows that the frequency variables ω_y and

$$\omega_x = \frac{2\pi F}{F_x} = 2\pi F T_x \tag{10.2.12}$$

are related by

$$\omega_y = D\omega_x \tag{10.2.13}$$

Thus, as expected, the frequency range $0 \leq |\omega_x| \leq \pi/D$ is stretched into the corresponding frequency range $0 \leq |\omega_y| \leq \pi$ by the downsampling process.

We conclude that the spectrum $Y(\omega_y)$, which is obtained by evaluating (10.2.9) on the unit circle, can be expressed as

$$Y(\omega_y) = \frac{1}{D}\sum_{k=0}^{D-1} H_D\left(\frac{\omega_y - 2\pi k}{D}\right) X\left(\frac{\omega_y - 2\pi k}{D}\right) \tag{10.2.14}$$

With a properly designed filter $H_D(\omega)$, the aliasing is eliminated and, consequently, all but the first term in (10.2.14) vanish. Hence

$$
\begin{aligned}
Y(\omega_y) &= \frac{1}{D} H_D\left(\frac{\omega_y}{D}\right) X\left(\frac{\omega_y}{D}\right) \\
&= \frac{1}{D} X\left(\frac{\omega_y}{D}\right)
\end{aligned}
\tag{10.2.15}
$$

for $0 \leq |\omega_y| \leq \pi$. The spectra for the sequences $x(n)$, $v(n)$, and $y(m)$ are illustrated in Fig. 10.4.

INTERPOLATION BY A FACTOR I

An increase in the sampling rate by an integer factor of I can be accomplished by interpolating $I-1$ new samples between successive values of the signal. The interpolation process can be accomplished in a variety of ways. We shall describe a process that preserves the spectral shape of the signal sequence $x(n)$.

Let $v(m)$ denote a sequence with a rate $F_y = IF_x$, which is obtained from $x(n)$ by adding $I-1$ zeros between successive values of $x(n)$. Thus

$$v(m) = \begin{cases} x(m/I), & m = 0, \pm I, \pm 2I, \dots \\ 0, & \text{otherwise} \end{cases} \tag{10.3.1}$$

and its sampling rate is identical to the rate of $y(m)$. This sequence has a z-transform

$$\begin{aligned} V(z) &= \sum_{m=-\infty}^{\infty} v(m) z^{-m} \\ &= \sum_{m=-\infty}^{\infty} x(m) z^{-mI} \\ &= X(z^I) \end{aligned} \tag{10.3.2}$$

The corresponding spectrum of $v(m)$ is obtained by evaluating (10.3.2) on the unit circle. Thus

$$V(\omega_y) = X(\omega_y I) \tag{10.3.3}$$

where ω_y denotes the frequency variable relative to the new sampling rate F_y (i.e., $\omega_y = 2\pi F/F_y$). Now the relationship between sampling rates is $F_y = IF_x$ and hence, the frequency variables ω_x and ω_y are related according to the formula

$$\omega_y = \frac{\omega_x}{I} \tag{10.3.4}$$

The spectra $X(\omega_x)$ and $V(\omega_y)$ are illustrated in Fig. 10.5. We observe that the sampling rate increase, obtained by the addition of $I-1$ zero samples between successive values of $x(n)$, results in a signal whose spectrum $V(\omega_y)$ is an I-fold periodic repetition of the input signal spectrum $X(\omega_x)$.

Since only the frequency components of $x(n)$ in the range $0 \le \omega_y \le \pi/I$ are unique, the images of $X(\omega)$ above $\omega_y = \pi/I$ should be rejected by passing the sequence $v(m)$ through a lowpass filter with frequency response $H_I(\omega_y)$ that ideally has the characteristic

$$H_I(\omega_y) = \begin{cases} C, & 0 \le |\omega_y| \le \pi/I \\ 0, & \text{otherwise} \end{cases} \tag{10.3.5}$$

where C is a scale factor required to properly normalize the output sequence $y(m)$. Consequently, the output spectrum is

$$Y(\omega_y) = \begin{cases} CX(\omega_y I), & 0 \le |\omega_y| \le \pi/I \\ 0, & \text{otherwise} \end{cases} \tag{10.3.6}$$

The scale factor C is selected so that the output $y(m) = x(m/I)$ for $m = 0$, $\pm I$, $+2I, \ldots$. For mathematical convenience, we select the point $m = 0$. Thus

$$\begin{aligned} y(0) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} Y(\omega_y)d\omega_y \\ &= \frac{C}{2\pi}\int_{-\pi/I}^{\pi/I} X(\omega_y I)d\omega_y \end{aligned} \tag{10.3.7}$$

Since $\omega_y = \omega_x/I$, (10.3.7) can be expressed as

$$\begin{aligned} y(0) &= \frac{C}{I}\frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega_x)d\omega_x \\ &= \frac{C}{I}x(0) \end{aligned} \tag{10.3.8}$$

Therefore, $C = I$ is the desired normalization factor.

Finally, we indicate that the output sequence $y(m)$ can be expressed as a convolution of the sequence $v(n)$ with the unit sample response $h(n)$ of the lowpass filter. Thus

$$y(m) = \sum_{k=-\infty}^{\infty} h(m-k)v(k) \tag{10.3.9}$$

Since $v(k) = 0$ except at multiples of I, where $v(kI) = x(k)$, (10.3.9) becomes

$$y(m) = \sum_{k=-\infty}^{\infty} h(m-kI)x(k) \tag{10.3.10}$$

SAMPLING RATE CONVERSION BY A RATIONAL FACTOR *I/D*

Having discussed the special cases of decimation (downsampling by a factor D) and interpolation (upsampling by a factor I), we now consider the general case of sampling rate conversion by a rational factor I/D. Basically, we can achieve this sampling rate conversion by first performing interpolation by the factor I and then decimating the output of the interpolator by the factor D. In other words, a sampling rate conversion by the rational factor I/D is accomplished by cascading an interpolator with a decimator, as illustrated in Fig. 10.6.

We emphasize that the importance of performing the interpolation first and the decimation second, is to preserve the desired spectral characteristics of $x(n)$. Furthermore, with the cascade configuration illustrated in Fig. 10.6, the two filters with impulse response $\{h_u(l)\}$ and $\{h_d(l)\}$ are operated at the same rate, namely IF_x and hence can be combined into a single lowpass filter with impulse response $h(l)$ as illustrated in Fig. 10.7. The frequency response $H(\omega_v)$ of the combined filter must incorporate the filtering operations for both interpolation and decimation, and hence it should ideally possess the frequency response characteristic

$$H(\omega_v) = \begin{cases} I, & 0 \le |\omega_v| \le \min(\pi/D, \pi/I) \\ 0, & \text{otherwise} \end{cases} \tag{10.4.1}$$

where $\omega_v = 2\pi F/F_v = 2\pi F/I F_x = \omega_x/I$.

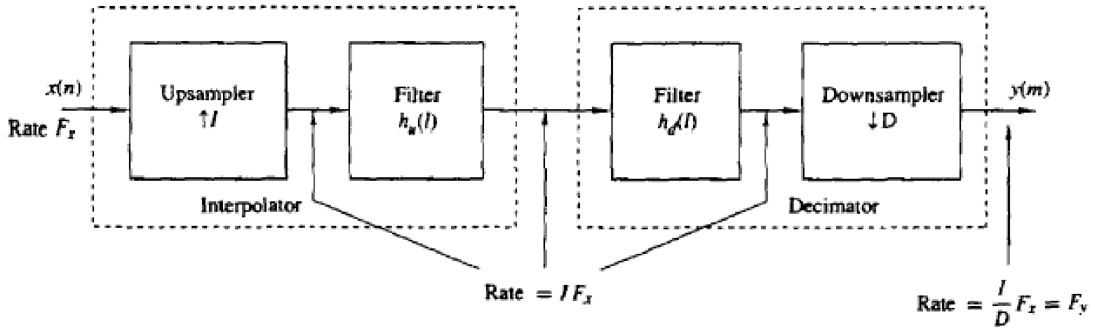


Figure 10.6 Method for sampling rate conversion by a factor I/D.

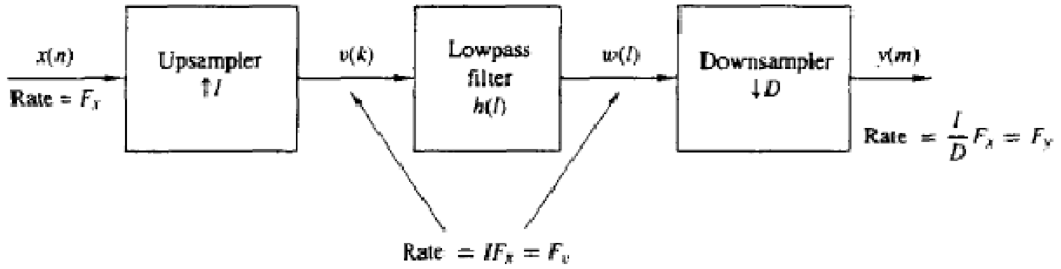


Figure 10.7 Method for sampling rate conversion by a factor I/D.

In the time domain, the output of the upsampler is the sequence

$$v(l) = \begin{cases} x(l/I), & l = 0, \pm I, \pm 2I, \ldots \\ 0, & \text{otherwise} \end{cases} \tag{10.4.2}$$

and the output of the linear time-invariant filter is

$$\begin{aligned} w(l) &= \sum_{k=-\infty}^{\infty} h(l-k)v(k) \\ &= \sum_{k=-\infty}^{\infty} h(l-kI)x(k) \end{aligned} \tag{10.4.3}$$

Finally, the output of the sampling rate converter is the sequence $\{y(m)\}$, which is obtained by downsampling the sequence $\{w(l)\}$ by a factor of D. Thus

$$\begin{aligned} y(m) &= w(mD) \\ &= \sum_{k=-\infty}^{\infty} h(mD-kI)x(k) \end{aligned} \tag{10.4.4}$$

It is illuminating to express (10.4.4) in a different form by making a change in variable. Let

$$k = \left\lfloor \frac{mD}{I} \right\rfloor - n \qquad (10.4.5)$$

where the notation $\lfloor r \rfloor$ denotes the largest integer contained in r. With this change in variable, (10.4.4) becomes

$$y(m) = \sum_{n=-\infty}^{\infty} h\left(mD - \left\lfloor \frac{mD}{I} \right\rfloor I + nI\right) x\left(\left\lfloor \frac{mD}{I} \right\rfloor - n\right) \qquad (10.4.6)$$

We note that

$$\begin{aligned} mD - \left\lfloor \frac{mD}{I} \right\rfloor I &= mD \qquad \text{modulo } I \\ &= (mD)_I \end{aligned}$$

Consequently, (10.4.6) can be expressed as

$$y(m) = \sum_{n=-\infty}^{\infty} h(nI + (mD)_I) x\left(\left\lfloor \frac{mD}{I} \right\rfloor - n\right) \qquad (10.4.7)$$

It is apparent from this form that the output $y(m)$ is obtained by passing the input sequence $x(n)$ through a time-variant filter with impulse response

$$g(n, m) = h(nI + (mD)_I) \qquad -\infty < m, n < \infty \qquad (10.4.8)$$

where $h(k)$ is the impulse response of the time-invariant lowpass filter operating at the sampling rate IF_x. We further observe, that for any integer k,

$$\begin{aligned} g(n, m + kI) &= h(nI + (mD + kDI)_I) \\ &= h(nI + (mD)_I) \qquad (10.4.9) \\ &= g(n, m) \end{aligned}$$

Hence $g(n, m)$ is periodic in the variable m with period I.

The frequency-domain relationships can be obtained by combining the results of the interpolation and decimation processes. Thus the spectrum at the output of the linear filter with impulse response $h(l)$ is

$$\begin{aligned} V(\omega_v) &= H(\omega_v) X(\omega_v I) \\ &= \begin{cases} IX(\omega_v I), & 0 \le |\omega_v| \le \min(\pi/D, \pi/I) \\ 0, & \text{otherwise} \end{cases} \end{aligned} \qquad (10.4.10)$$

The spectrum of the output sequence $y(m)$, obtained by decimating the sequence $v(n)$ by a factor of D, is

$$Y(\omega_y) = \frac{1}{D} \sum_{k=0}^{D-1} V\left(\frac{\omega_y - 2\pi k}{D}\right) \qquad (10.4.11)$$

where $\omega_y = D\omega_v$. Since the linear filter prevents aliasing as implied by (10.4.10), the spectrum of the output sequence given by (10.4.11) reduces to

$$Y(\omega_y) = \begin{cases} \dfrac{I}{D} X\left(\dfrac{\omega_y}{D}\right), & 0 \le |\omega_y| \le \min\left(\pi, \dfrac{\pi D}{I}\right) \\ 0, & \text{otherwise} \end{cases} \qquad (10.4.12)$$

SAMPLING-RATE CONVERSION BY AN ARBITRARY FACTOR

In the previous sections of this chapter, we have shown how to perform sampling rate conversion exactly by a rational number I/D. In some applications, it is either inefficient or, sometimes impossible to use such an exact rate conversion scheme. We first consider the following two cases.

Case 1. We need to perform rate conversion by the rational number I/D, where I is a large integer (e.g., $I/D = 1023/511$). Although we can achieve exact rate conversion by this number, we would need a polyphase filter with 1023 subfilters. Such an exact implementation is obviously inefficient in memory usage because we need to store a large number of filter coefficients.

Case 2. In some applications, the exact conversion rate is not known when we design the rate converter, or the rate is continuously changing during the conversion process. For example, we may encounter the situation where the input and output samples are controlled by two independent clocks. Even though it is still possible to define a nominal conversion rate that is a rational number, the actual rate would be slightly different, depending on the frequency difference between the two clocks. Obviously, it is not possible to design an exact rate converter in this case.

To implement sampling rate conversion for applications similar to these cases, we resort to nonexact rate conversion schemes. Unavoidably, a nonexact scheme will introduce some distortion in the converted output signal. (It should be noted that distortion exists even in an exact rational rate converter because the polyphase filter is never ideal.) Such a converter will be adequate, as long as the total distortion does not exceed the specification required in the application.

Depending on the application requirements and implementation constraints, we can use first-order, second-order, or higher-order approximations. We shall describe first-order and second-order approximation methods and provide an analysis of the resulting timing errors.

10.8.1 First-Order Approximation

Let us denote the arbitrary conversion rate by r and suppose that the input to the rate converter is the sequence $\{x(n)\}$. We need to generate a sequence of output samples separated in time by T_x/r, where T_x is the sample interval for $\{x(n)\}$. By constructing a polyphase filter with a large number of subfilters as just described, we can approximate such a sequence with a nonuniformly spaced sequence. Without loss of generality, we can express $1/r$ as

$$\frac{1}{r} = \frac{k}{I} + \beta$$

where k and I are positive integers and β is a number in the range

$$0 < \beta < \frac{1}{I}$$

Consequently, $1/r$ is bounded from above and below as

$$\frac{k}{I} < \frac{1}{r} < \frac{k+1}{I}$$

I corresponds to the interpolation factor, which will be determined to satisfy the specification on the amount of tolerable distortion introduced by rate conversion. I is also equal to the number of polyphase filters.

For example, suppose that $r = 2.2$ and that we have determined, as we will demonstrate, that $I = 6$ polyphase filters are required to meet the distortion specification. Then

$$\frac{k}{I} \equiv \frac{2}{6} < \frac{1}{r} < \frac{3}{6} \equiv \frac{k+1}{I}$$

so that $k = 2$. The time spacing between samples of the interpolated sequence is T_x/I. However, the desired conversion rate $r = 2.2$ for $I = 6$ corresponds to a decimation factor of 2.727, which falls between $k = 2$ and $k = 3$. In the first-order approximation, we achieve the desired decimation rate by selecting the output sample from the polyphase filter closest in time to the desired sampling time. This is illustrated in Fig. 10.27 for $I = 6$.

In general, to perform rate conversion by a factor r, we employ a polyphase filter to perform interpolation and therefore to increase the frequency of the original sequence of a factor of I. The time spacing between the samples of the interpolated sequence is equal to T_x/I. If the ideal sampling time of the mth sample, $y(m)$, of the desired output sequence is between the sampling times of two samples of the interpolated sequence, we select the sample closer to $y(m)$ as its approximation.

Let us assume that the mth selected sample is generated by the (i_m)th subfilter using the input samples $x(n), x(n-1), \ldots, x(n-K+1)$ in the delay line. The normalized sampling time error (i.e., the time difference between the selected sampling time and the desired sampling time normalized by T_x) is denoted by t_m. The sign of t_m is positive if the desired sampling time leads the selected sampling time, and negative otherwise. It is easy to show that $|t_m| \leq 0.5/I$. The normalized time advance from the mth output $y(m)$ to the $(m+1)$st output $y(m+1)$ is equal to $(1/r) + t_m$.

To compute the next output, we first determine a number closest to $i_m/I + 1/r + t_m + k_m/I$ that is of the form $l_{m-1} + i_{m+1}/I$, where both l_{m+1} and i_{m+1} are integers and $i_{m+1} < I$. Then, the $(m+1)$st output $y(m+1)$ is computed using the (i_{m+1})th subfilter after shifting the signal in the delay line by l_{m+1} input samples. The normalized timing error for the $(m+1)$th sample is $t_{m+1} = (i_m/I + 1/r + t_m) - (l_{m+1} + i_{m+1}/I)$. It is saved for the computation of the next output sample.

By increasing the number of subfilters used, we can arbitrarily increase the conversion accuracy. However, we also require more memory to store the large number of filter coefficients. Hence it is desirable to use as few subfilters as possible while keeping the distortion in the converted signal below the specification. The distortion introduced due to the sampling-time approximation is most conveniently evaluated in the frequency domain.

Suppose that the input data sequence $\{x(n)\}$ has a flat spectrum from $-\omega_x$ to ω_x, where $\omega_x < \pi$, with a magnitude A. Its total power can be computed using Parseval's theorem, namely,

$$P_s = \frac{1}{2\pi} \int_{-\omega_x}^{\omega_x} |X(\omega)|^2 d\omega = \frac{A^2 \omega_x}{\pi} \qquad (10.8.1)$$

APPLICATIONS OF MULTIRATE SIGNAL PROCESSING

There are numerous practical applications of multirate signal processing. In this section we describe a few of these applications.

10.9.1 Design of Phase Shifters

Suppose that we wish to design a network that delays the signal $x(n)$ by a fraction of a sample. Let us assume that the delay is a rational fraction of a sampling interval T_x [i.e., $d = (k/I)T_x$, where k and I are relatively prime positive integers]. In the frequency domain, the delay corresponds to a linear phase shift of the form

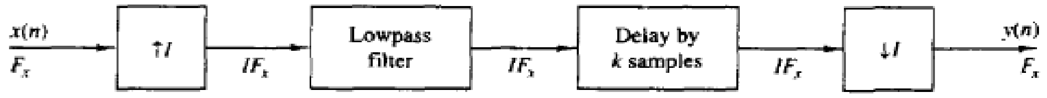


Figure 10.29 Method for generating a delay in a discrete-time signal.

$$\Theta(\omega) = -\frac{k\omega}{I} \qquad (10.9.1)$$

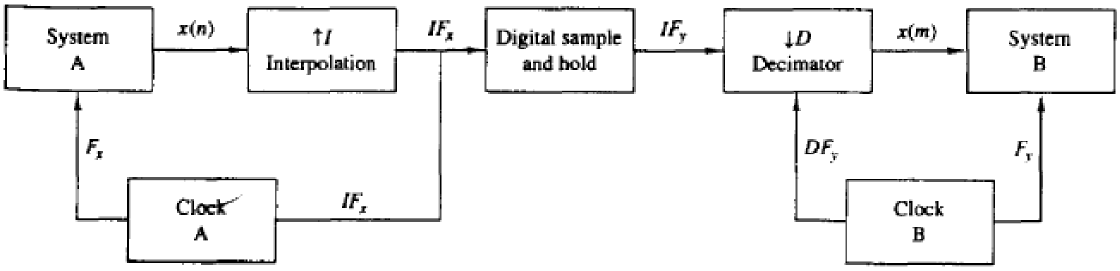
The design of an all-pass linear-phase filter is relatively difficult. However, we can use the methods of sample-rate conversion to achieve a delay of $(k/I)T_x$, exactly, without introducing any significant distortion in the signal. To be specific, let us consider the system shown in Fig. 10.29. The sampling rate is increased by a factor I using a standard interpolator. The lowpass filter eliminates the images in the spectrum of the interpolated signal, and its output is delayed by k samples at the sampling rate IF_x. The delayed signal is decimated by a factor $D = I$. Thus we have achieved the desired delay of $(k/I)T_x$.

10.9.2 Interfacing of Digital Systems with Different Sampling Rates

In practice we frequently encounter the problem of interfacing two digital systems that are controlled by independently operating clocks. An analog solution to this problem is to convert the signal from the first system to analog form and then resample it at the input to the second system using the clock in this system. However, a simpler approach is one where the interfacing is done by a digital method using the basic sample-rate conversion methods described in this chapter.

To be specific, let us consider interfacing the two systems with independent clocks as shown in Fig. 10.31. The output of system A at rate F_x is fed to an interpolator which increases the sampling rate by I. The output of the interpolator is fed at the rate IF_x to a digital sample-and-hold which serves as the interface to system B at the high sampling rate IF_x. Signals from the digital sample-and-hold are read out into system B at the clock rate DF_y of system B. Thus the output rate from the sample-and-hold is not synchronized with the input rate.

In the special case where $D = I$ and the two clock rates are comparable but not identical, some samples at the output of the sample-and-hold may be repeated or dropped at times. The amount of signal distortion resulting from this method can be kept small if the interpolator/decimator factor is large. By using linear interpolation in place of the digital sample-and-hold, as we described in Section 10.8, we can further reduce the distortion and thus reduce the size of the interpolator factor.



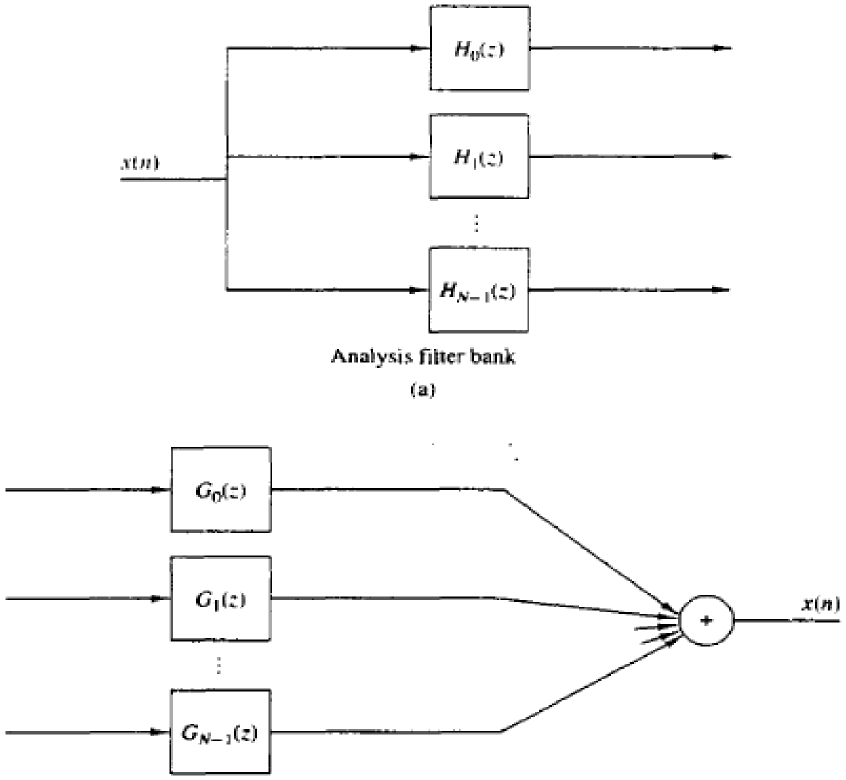
10.9.3 Implementation of Narrowband Lowpass Filters

In Section 10.6 we demonstrated that a multistage implementation of sampling-rate conversion often provides for a more efficient realization, especially when the filter specifications are very tight (e.g., a narrow passband and a narrow transition band). Under similar conditions, a lowpass, linear-phase FIR filter may be more efficiently implemented in a multistage decimator-interpolator configuration. To be more specific, we can employ a multistage implementation of a decimator of size D, followed by a multistage implementation of an interpolator of size I, where $I = D$.

10.9.4 Implementation of Digital Filter Banks

Filter banks are generally categorized as two types, *analysis filter banks* and *synthesis filter banks*. An analysis filter bank consists of a set of filters, with system functions $\{H_z(k)\}$, arranged in a parallel bank as illustrated in Fig. 10.32a. The frequency response characteristics of this filter bank splits the signal into a corresponding number of subbands. On the other hand, a synthesis filter bank consists of a set of filters with system functions $\{G_k(z)\}$, arranged as shown in Fig. 10.32b, with corresponding inputs $\{y_k(n)\}$. The outputs of the filters are summed to form the synthesized signal $\{x(n)\}$.

Filter banks are often used for performing spectrum analysis and signal synthesis. When a filter bank is employed in the computation of the discrete Fourier



Analysis filter bank
(a)

Synthesis filter bank
(b)

transform (DFT) of a sequence $\{x(n)\}$, the filter bank is called a DFT filter bank. An analysis filter bank consisting of N filters $\{H_k(z), k = 0, 1, \ldots, N-1\}$ is called a uniform DFT filter bank if $H_k(z)$, $k = 1, 2, \ldots, N-1$, are derived from a prototype filter $H_0(z)$, where

$$H_k(\omega) = H_0\left(\omega - \frac{2\pi k}{N}\right) \qquad k = 1, 2, \ldots, N-1 \tag{10.9.2}$$

Hence the frequency response characteristics of the filters $\{H_k(z), k = 0, 1, \ldots, N-1\}$ are simply obtained by uniformly shifting the frequency response of the prototype filter by multiples of $2\pi/N$. In the time domain the filters are characterized by their impulse responses, which can be expressed as

$$h_k(n) = h_0(n)e^{j2\pi nk/N} \qquad k = 0, 1, \ldots, N-1 \tag{10.9.3}$$

where $\{h_0(n)\}$ is the impulse response of the prototype filter.

The uniform DFT analysis filter bank can be realized as shown in Fig. 10.33a, where the frequency components in the sequence $\{x(n)\}$ are translated in frequency to lowpass by multiplying $x(n)$ with the complex exponentials $\exp(-j2\pi nk/N)$, $k = 1, \ldots, N-1$, and the resulting product signals are passed through a lowpass filter with impulse response $\{h_0(n)\}$. Since the output of the lowpass filter is relatively narrow in bandwidth, the signal can be decimated by a factor $D \leq N$. The resulting decimated output signal can be expressed as

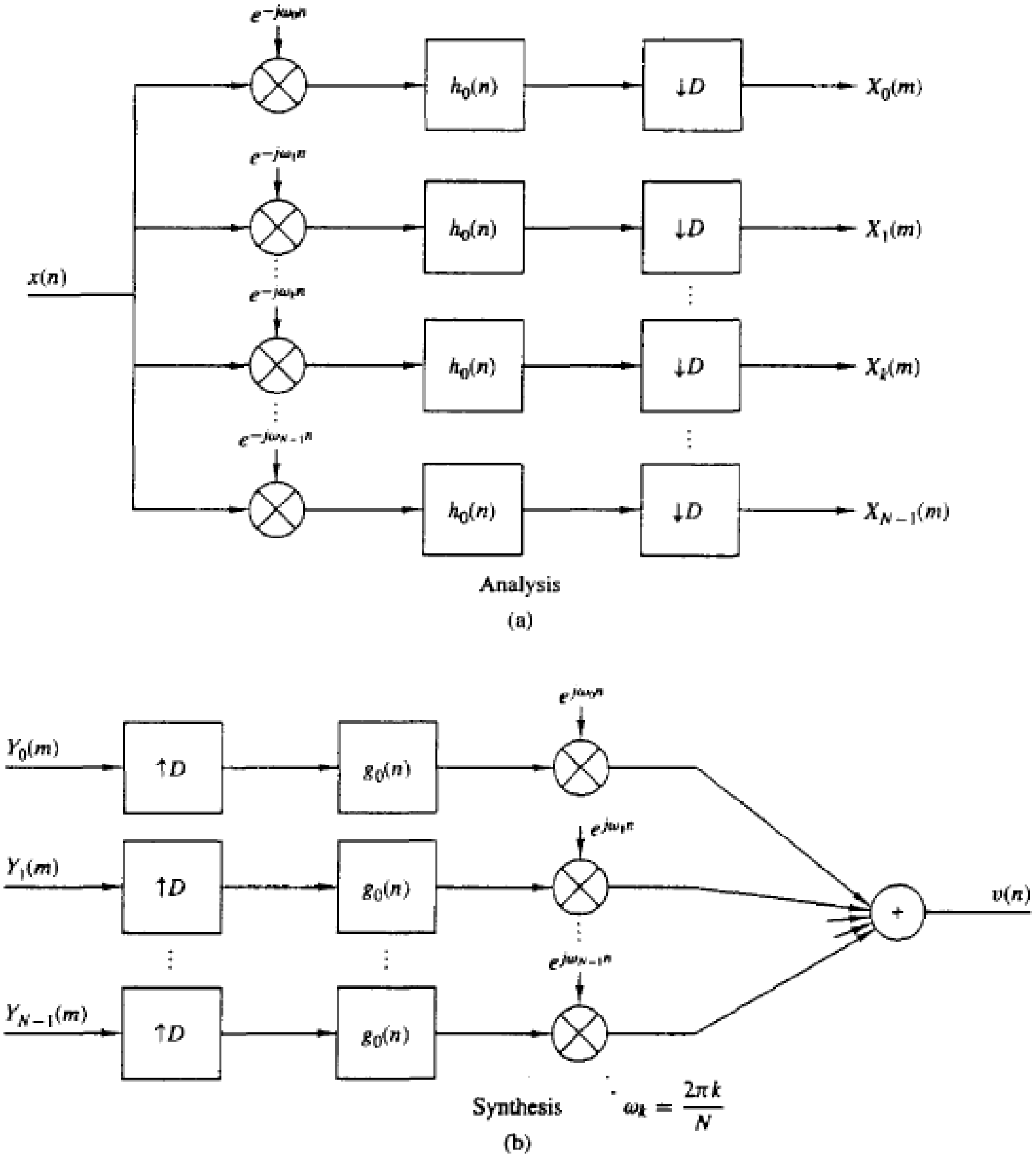
$$X_k(m) = \sum_n h_0(mD - n)x(n)e^{-j2\pi nk/N} \qquad \begin{array}{l} k = 0, 1, \ldots, N-1 \\ m = 0, 1, \ldots \end{array} \tag{10.9.4}$$

where $\{X_k(m)\}$ are samples of the DFT at frequencies $\omega_k = 2\pi k/N$.

The corresponding synthesis filter for each element in the filter bank can be viewed as shown in Fig. 10.33b, where the input signal sequences $\{Y_k(m), k = 0, 1, \ldots, N-1\}$ are upsampled by a factor of $I = D$, filtered to remove the images, and translated in frequency by multiplication by the complex exponentials $\{\exp(j2\pi nk/N), k = 0, 1, \ldots, N-1\}$. The resulting frequency-translated signals from the N filters are then summed. Thus we obtain the sequence

$$\begin{aligned} v(n) &= \frac{1}{N}\sum_{k=0}^{N-1} e^{j2\pi nk/N}\left[\sum_m Y_k(m)g_0(n - mI)\right] \\ &= \sum_m g_0(n - mI)\left[\frac{1}{N}\sum_{k=0}^{N-1} Y_k(m)e^{j2\pi nk/N}\right] \\ &= \sum_m g_0(n - mI)y_n(m) \end{aligned} \tag{10.9.5}$$

where the factor $1/N$ is a normalization factor, $\{y_n(m)\}$ represent samples of the inverse DFT sequence corresponding to $\{Y_k(m)\}$, $\{g_0(n)\}$ is the impulse response of the interpolation filter, and $I = D$.



Analysis

(a)

Synthesis

(b)

$\omega_k = \dfrac{2\pi k}{N}$

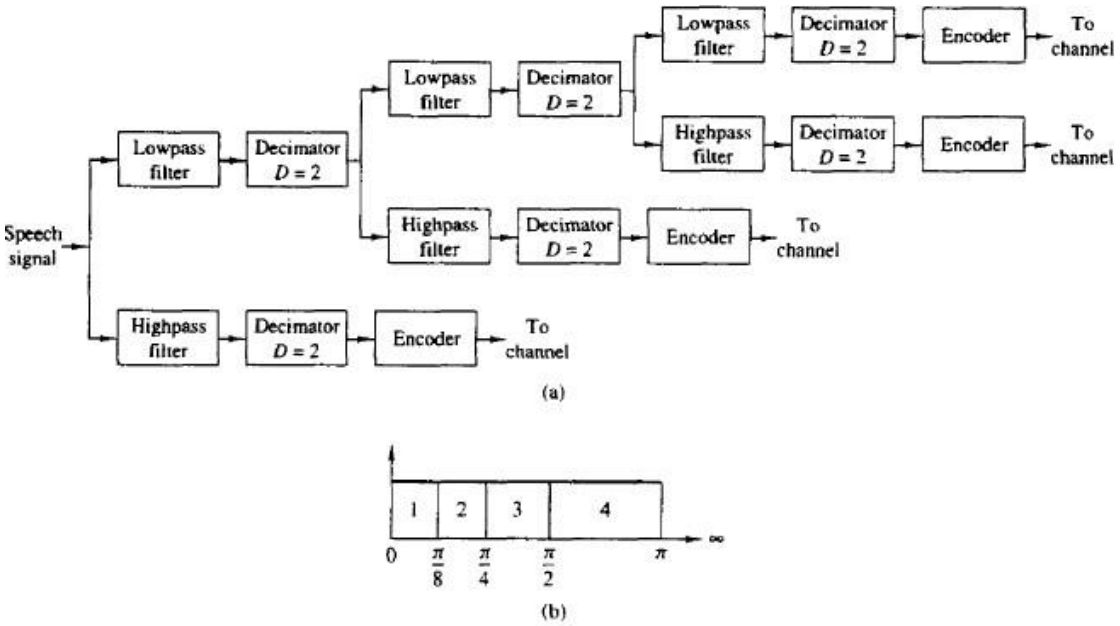
Figure 10.33 A uniform DFT filter bank.

10.9.5 Subband Coding of Speech Signals

A variety of techniques have been developed to efficiently represent speech signals in digital form for either transmission or storage. Since most of the speech energy is contained in the lower frequencies, we would like to encode the lower-frequency band with more bits than the high-frequency band. Subband coding is a method, where the speech signal is subdivided into several frequency bands and each band is digitally encoded separately.

An example of a frequency subdivision is shown in Fig. 10.37a. Let us assume that the speech signal is sampled at a rate F_s samples per second. The first frequency subdivision splits the signal spectrum into two equal-width segments, a lowpass signal ($0 \leq F \leq F_s/4$) and a highpass signal ($F_s/4 \leq F \leq F_s/2$). The second frequency subdivision splits the lowpass signal from the first stage into two equal bands, a lowpass signal ($0 < F \leq F_s/8$) and a highpass signal ($F_s/8 \leq F \leq F_s/4$). Finally, the third frequency subdivision splits the lowpass signal from the second stage into two equal bandwidth signals. Thus the signal is subdivided into four frequency bands, covering three octaves, as shown in Fig. 10.37b.

Decimation by a factor of 2 is performed after frequency subdivision. By allocating a different number of bits per sample to the signal in the four subbands, we can achieve a reduction in the bit rate of the digitalized speech signal.



(a)

(b)

Figure 10.37 Block diagram of a subband speech coder.

Filter design is particularly important in achieving good performance in subband coding. Aliasing resulting from decimation of the subband signals must be negligible. It is clear that we cannot use brickwall filter characteristics as shown in Fig. 10.38a, since such filters are physically unrealizable. A particularly practical solution to the aliasing problem is to use *quadrature mirror filters* (QMF), which have the frequency response characteristics shown in Fig. 10.38b. These filters are described in the following section.

The synthesis method for the subband encoded speech signal is basically the reverse of the encoding process. The signals in adjacent lowpass and highpass frequency bands are interpolated, filtered, and combined as shown in Fig. 10.39. A pair of QMF is used in the signal synthesis for each octave of the signal.

Subband coding is also an effective method to achieve data compression in image signal processing. By combining subband coding with vector quantization for each subband signal, Safranek et al. (1988) have obtained coded images with approximately $\frac{1}{2}$ bit per pixel, compared with 8 bits per pixel for the uncoded image.

In general, subband coding of signals is an effective method for achieving bandwidth compression in a digital representation of the signal, when the signal energy is concentrated in a particular region of the frequency band. Multirate signal processing notions provide efficient implementations of the subband encoder.

10.9.6 Quadrature Mirror Filters

The basic building block in applications of quadrature mirror filters (QMF) is the two-channel QMF bank shown in Fig. 10.40. This is a multirate digital filter structure that employs two decimators in the "signal analysis" section and two interpolators in the "signal synthesis" section. The lowpass and highpass filters in the analysis section have impulse responses $h_0(n)$ and $h_1(n)$, respectively. Similarly, the lowpass and highpass filters contained in the synthesis section have impulse responses $g_0(n)$ and $g_1(n)$, respectively.

The Fourier transforms of the signals at the outputs of the two decimators are

$$\begin{aligned} X_{a0}(\omega) &= \frac{1}{2}\left[X\left(\frac{\omega}{2}\right)H_0\left(\frac{\omega}{2}\right) + X\left(\frac{\omega-2\pi}{2}\right)H_0\left(\frac{\omega-2\pi}{2}\right)\right] \\ X_{a1}(\omega) &= \frac{1}{2}\left[X\left(\frac{\omega}{2}\right)H_1\left(\frac{\omega}{2}\right) + X\left(\frac{\omega-2\pi}{2}\right)H_1\left(\frac{\omega-2\pi}{2}\right)\right] \end{aligned} \tag{10.9.17}$$

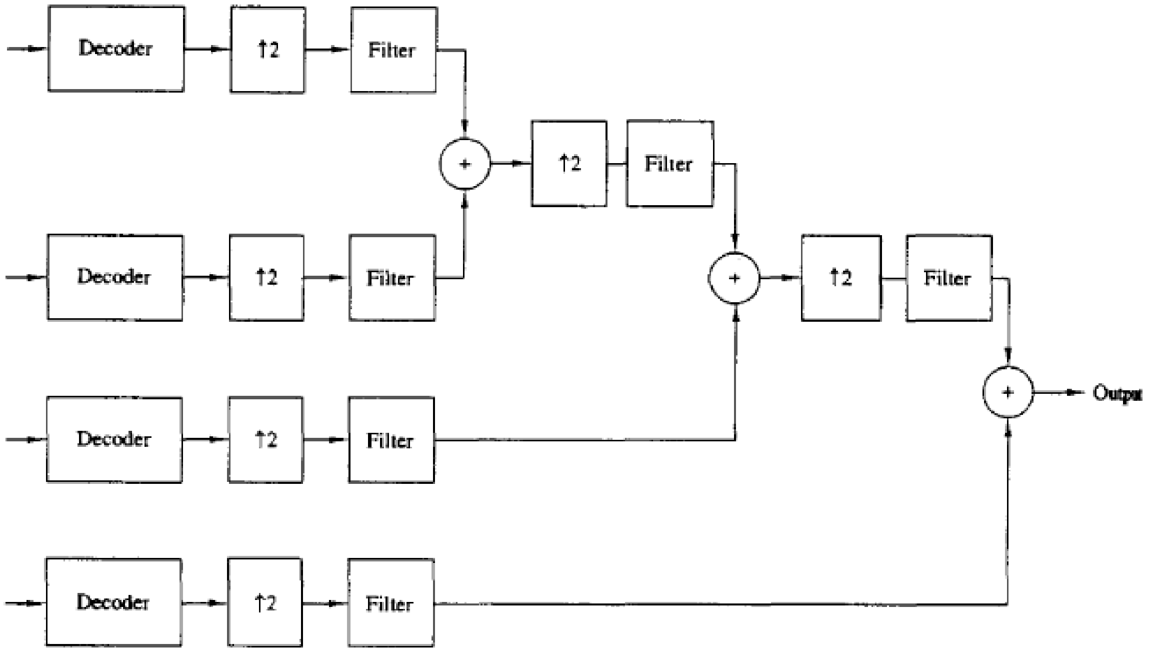


Figure 10.39 Synthesis of subband-encoded signals.

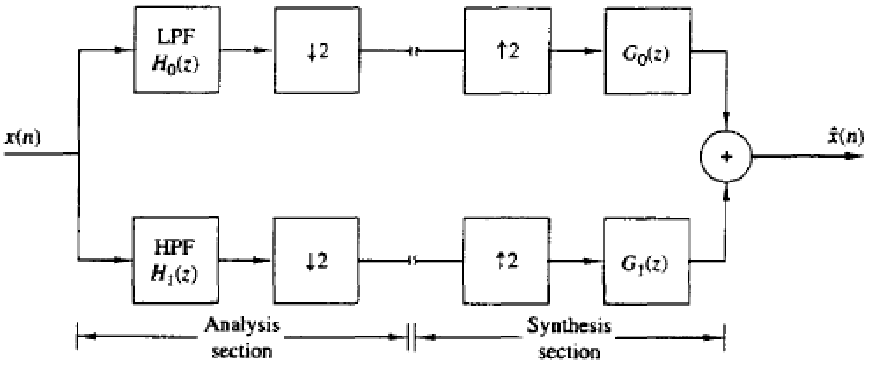


Figure 10.40 Two-channel QMF bank.

If $X_{s0}(\omega)$ and $X_{s1}(\omega)$ represent the two inputs to the synthesis section, the output is simply

$$\hat{X}(\omega) = X_{s0}(2\omega)G_0(\omega) + X_{s1}(2\omega)G_1(\omega)$$

10.9.7 Transmultiplexers

Another application of multirate signal processing is in the design and implementation of digital transmultiplexers which are devices for converting between time-division-multiplexed (TDM) signals and frequency-division-multiplexed (FDM) signals.

In a transmultiplexer for TDM-to-FDM conversion, the input signal $\{x(n)\}$ is a time-division multiplexed signal consisting of L signals, which are separated by a commutator switch. Each of these L signals are then modulated on different carrier frequencies to obtain an FDM signal for transmission. In a transmultiplexer for FDM-to-TDM conversion, the composite signal is separated by filtering into the L signal components which are then time-division multiplexed.

In telephony, single-sideband transmission is used with channels spaced at a nominal 4-kHz bandwidth. Twelve channels are usually stacked in frequency to form a basic group channel, with a bandwidth of 48 kHz. Larger bandwidth FDM signals are formed by frequency translation of multiple groups into adjacent frequency bands. We shall confine our discussion to digital transmultiplexers for 12-channel FDM and TDM signals.

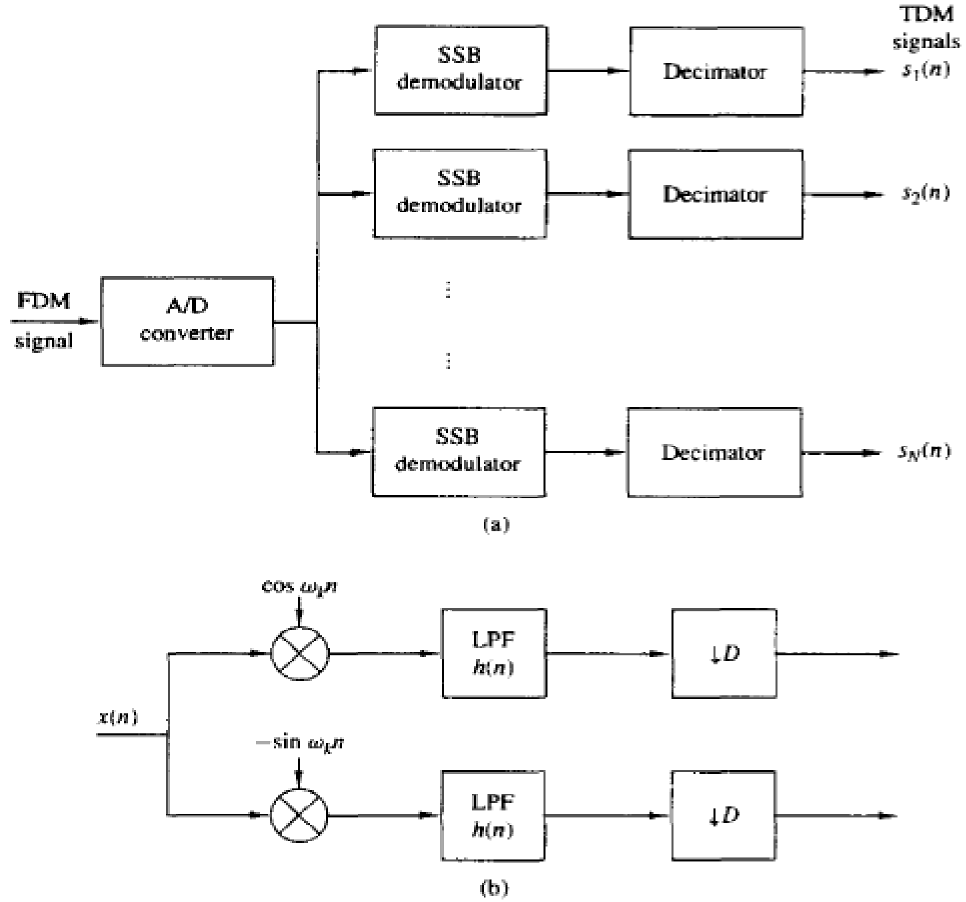


Figure 10.44 Block diagram of FDM-to-TDM transmultiplexer.

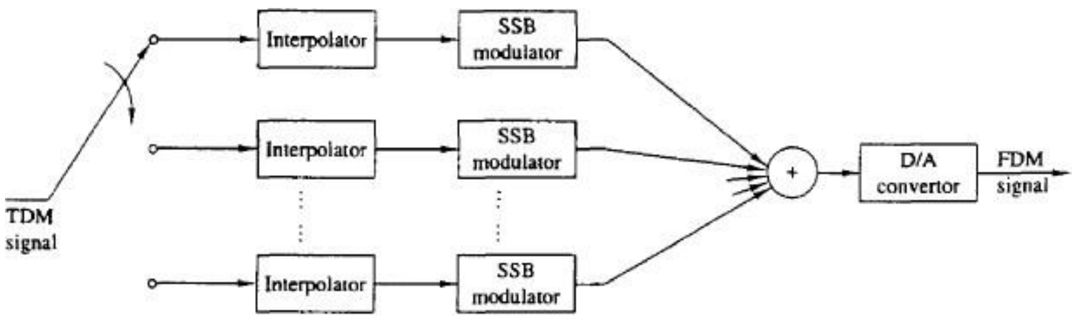


Figure 10.45 Block diagram of TDM-to-FDM transmultiplexer.

10.9.8 Oversampling A/D and D/A Conversion

Our treatment of oversampling A/D and D/A converters in Chapter 9 provides another example of multirate signal processing. Recall that an oversampling A/D converter is implemented by a cascade of an analog sigma-delta modulator (SDM) followed by a digital antialiasing decimation filter and a digital highpass filter as shown in Fig. 10.46. The analog SDM produces a 1-bit per sample output at a very high sampling rate. This 1-bit per sample output is passed through a digital lowpass filter, which provides a high-precision (multiple-bit) output that is decimated to a lower sampling rate. This output is then passed to a digital highpass filter that serves to attenuate the quantization noise at the lower frequencies.

The reverse operations take place in an oversampling D/A converter, as shown in Fig. 10.47. As illustrated in this figure, the digital signal is passed through a highpass filter whose output is fed to a digital interpolator (upsampler and anti-imaging filter). This high-sampling-rate signal is the input to the digital SDM that provides a high-sampling-rate 1-bit per sample output. The 1-bit per sample output is then converted to an analog signal by lowpass filtering and further smoothing with analog filters.

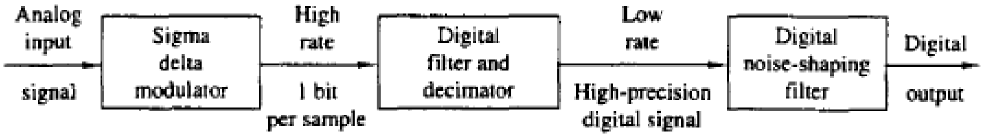


Figure 10.46 Diagram of oversampling A/D converter

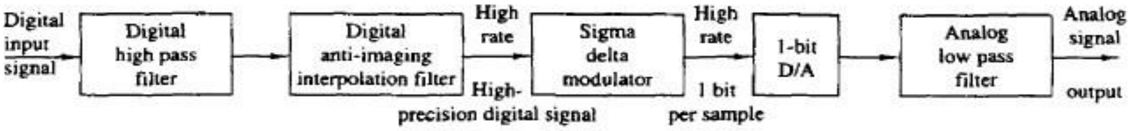


Figure 10.47 Diagram of oversampling D/A converter

Figure 10.48 illustrates the block diagram of a commercial (Analog Devices ADSP-28 msp02) codec (encoder and decoder) for voice-band signals based on sigma-delta A/D and D/A converters and analog front-end circuits needed as an interface to the analog voice-band signals. The nominal sampling rate (after decimation) is 8 kHz and the sampling rate of the SDM is 1 MHz. The codec has a 65-dB SNR and harmonic distortion performance.

****

OUTCOMES:

At the end of the course, the student should be able to

- Apply DFT and FFT algorithms for the analysis of digital signals and systems.
- Design FIR filters for various applications.
- Design IIR filters for various applications.
- Characterize the effects of finite precision representation on digital filters.

TEXT BOOKS:

1. John G.Proakis and Dimitris G. Manolakis, “Digital signal processing - Principles, algorithms and applications”, Pearson education / Prentice hall, Fourth edition, 2007.

REFERENCES:

1. Sanjay K.Mithra, “Digital signal processing - A Computer based approach”, Tata McGraw-Hill, 2007.

2. M.H.Hayes, “Digital signal processing”, Schum’s outlines, Tata McGraw Hill, 2007.

3. A.V.Oppenheim, R. W. Schafer and J. R. Buck, “Discrete-time signal processing”, Pearson, 2004.

4. I.C.Ifeachor and B.W.Jervis, “Digital signal processing – A practical approach”, Pearson 2002

5. L.R. Rabiner and B. Gold, “Theory and application of digital signal processing”, Prentice Hall, 1992.