

Figure 8.1.15 Length 32 split-radix FFT algorithms from paper by Duhamel (1986); reprinted with permission from the IEEE.

we decompose the computation into an eight-point, radix-2 DFT and two four-point, radix-4 DFTs. Thus at stage B, the top eight points constitute the sequence (with $N = 16$)

$$g'_0(n) = g_0(n) + g_0(n + N/2), \quad 0 \leq n \leq 7 \quad (8.1.61)$$

and the next eight points constitute the two four-point sequences $g'_1(n)$ and $jg'_2(n)$, where

$$g'_1(n) = g_0(n) - g_0(n + N/2), \quad 0 \leq n \leq 3 \quad (8.1.62)$$

$$g'_2(n) = g_0(n + N/4) - g_0(n + 3N/4), \quad 0 \leq n \leq 3$$

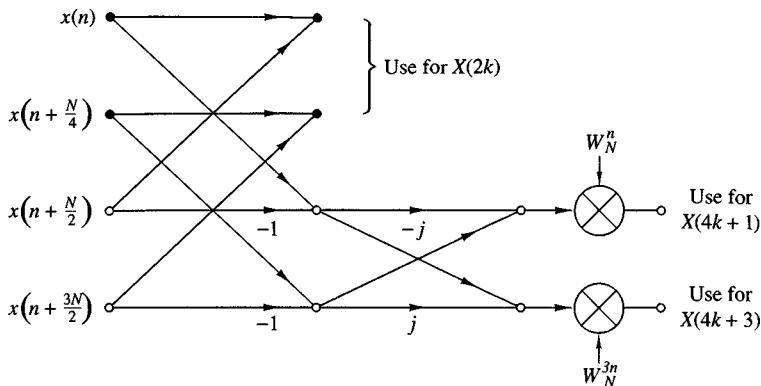


Figure 8.1.16 Butterfly for SRFFT algorithm.

The bottom 16 points of stage B are in the form of two eight-point DFTs. Hence each eight-point DFT is decomposed into a four-point, radix-2 DFT and a four-point, radix-4 DFT. In the final stage, the computations involve the combination of two-point sequences.

Table 8.2 presents a comparison of the number of *nontrivial* real multiplications and additions required to perform an N -point DFT with complex-valued data, using a radix-2, radix-4, radix-8, and a split-radix FFT. Note that the SRFFT algorithm requires the lowest number of multiplication and additions. For this reason, it is preferable in many practical applications.

Another type of SRFFT algorithm has been developed by Price (1990). Its relation to Duhamel's algorithm described previously can be seen by noting that the radix-4 DFT terms $X(4k+1)$ and $X(4k+3)$ involve the $N/4$ -point DFTs of the sequences $[g_1(n) - jg_2(n)]W_N^n$ and $[g_1(n) + jg_2(n)]W_N^{3n}$, respectively. In effect, the sequences $g_1(n)$ and $g_2(n)$ are multiplied by the factor (vector) $(1, -j) = (1, W_{32}^8)$.

TABLE 8.2 Number of Nontrivial Real Multiplications and Additions to Compute an N -point Complex DFT

N	Real Multiplications				Real Additions			
	Radix 2	Radix 4	Radix 8	Split Radix	Radix 2	Radix 4	Radix 8	Split Radix
16	24	20		20	152	148		148
32	88			68	408			388
64	264	208	204	196	1,032	976	972	964
128	712			516	2,504			2,308
256	1,800	1,392		1,284	5,896	5,488		5,380
512	4,360		3,204	3,076	13,566		12,420	12,292
1,024	10,248	7,856		7,172	30,728	28,336		27,652

Source: Extracted from Duhamel (1986).

and by W_N^n for the computation of $X(4k + 1)$, while the computation of $X(4k + 3)$ involves the factor $(1, j) = (1, W_{32}^{-8})$ and W_N^{3n} . Instead, one can rearrange the computation so that the factor for $X(4k + 3)$ is $(-j, -1) = -(W_{32}^{-8}, 1)$. As a result of this phase rotation, the phase factors in the computation of $X(4k + 3)$ become exactly the same as those for $X(4k + 1)$, except that they occur in mirror image order. For example, at stage B of Fig. 8.1.15, the phase factors $W^{21}, W^{18}, \dots, W^3$ are replaced by W^1, W^2, \dots, W^7 , respectively. This mirror-image symmetry occurs at every subsequent stage of the algorithm. As a consequence, the number of phase factors that must be computed and stored is reduced by a factor of 2 in comparison to Duhamel's algorithm. The resulting algorithm is called the "mirror" FFT (MFFT) algorithm.

An additional factor-of-2 savings in storage of phase factors can be obtained by introducing a 90° phase offset at the midpoint of each factor array, which can be removed if necessary at the output of the SRFFT computation. The incorporation of this improvement into the SRFFT (or the MFFT) results in another algorithm, also due to Price (1990), called the "phase" FFT (PFFT) algorithm.

8.1.6 Implementation of FFT Algorithms

Now that we have described the basic radix-2 and radix-4 FFT algorithms, 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 phase 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.

There are a number of other implementation issues regarding indexing, bit reversal, and the degree of parallelism in the computations. To a large extent, these issues are a function of the specific algorithm and the type of implementation, namely, a hardware or software implementation. In implementations based on a fixed-point arithmetic, or floating-point arithmetic on small machines, there is also the issue of round-off errors in the computation. This topic is considered in Section 8.4.

Although the FFT algorithms described previously were presented in the context of computing the DFT efficiently, they can also be used to compute the IDFT, which is

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk} \quad (8.1.63)$$

The only difference between the two transforms is the normalization factor $1/N$ and the sign of the phase factor W_N . Consequently, an FFT algorithm for computing the DFT can be converted to an FFT algorithm for computing the IDFT by changing the sign on all the phase factors and dividing the final output of the algorithm by N .

In fact, if we take the decimation-in-time algorithm that we described in Section 8.1.3, reverse the direction of the flow graph, change the sign on the phase factors, interchange the output and input, and finally, divide the output by N , we obtain a decimation-in-frequency FFT algorithm for computing the IDFT. On the other hand, if we begin with the decimation-in-frequency FFT algorithm described in Section 8.1.3 and repeat the changes described above, we obtain a decimation-in-time FFT algorithm for computing the IDFT. Thus it is a simple matter to devise FFT algorithms for computing the IDFT.

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 $N = 2^v$ or $N = 4^v$.

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 phase factors, and other considerations.

For general-purpose computers, where the cost of the numerical operations dominates, 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 as 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.

8.2 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 estimation is considered in Chapter 14. 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.

8.2.1 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 that the phase factors are complex and hence, after the first stage of the algorithm, all variables are basically complex valued.

In view of the fact that the algorithm can handle complex-valued input sequences, we can exploit this capability in the computation of the DFT of two real-valued sequences.

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

$$x(n) = x_1(n) + jx_2(n), \quad 0 \leq n \leq N - 1 \quad (8.2.1)$$

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

$$X(k) = X_1(k) + jX_2(k) \quad (8.2.2)$$

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

$$x_1(n) = \frac{x(n) + x^*(n)}{2} \quad (8.2.3)$$

$$x_2(n) = \frac{x(n) - x^*(n)}{2j} \quad (8.2.4)$$

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

$$X_1(k) = \frac{1}{2} \{ \text{DFT}[x(n)] + \text{DFT}[x^*(n)] \} \quad (8.2.5)$$

$$X_2(k) = \frac{1}{2j} \{ \text{DFT}[x(n)] - \text{DFT}[x^*(n)] \} \quad (8.2.6)$$

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

$$X_1(k) = \frac{1}{2} [X(k) + X^*(N - k)] \quad (8.2.7)$$

$$X_2(k) = \frac{1}{j2} [X(k) - X^*(N - k)] \quad (8.2.8)$$

Thus, by performing a single DFT on the complex-valued sequence $x(n)$, we have obtained the DFT of the two real sequences with only a small amount of additional computation that is involved in computing $X_1(k)$ and $X_2(k)$ from $X(k)$ by use of (8.2.7) and (8.2.8).

8.2.2 Efficient Computation of the DFT of a $2N$ -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

$$\begin{aligned}x_1(n) &= g(2n) \\x_2(n) &= g(2n + 1)\end{aligned}\tag{8.2.9}$$

Thus we have subdivided the $2N$ -point real sequence into two N -point real sequences. Now we can apply the method described in the preceding section.

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

$$x(n) = x_1(n) + jx_2(n)\tag{8.2.10}$$

From the results of the preceding section, we have

$$\begin{aligned}X_1(k) &= \frac{1}{2}[X(k) + X^*(N - k)] \\X_2(k) &= \frac{1}{2j}[X(k) - X^*(N - k)]\end{aligned}\tag{8.2.11}$$

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,

$$\begin{aligned}G(k) &= X_1(k) + W_2^k N X_2(k), & k = 0, 1, \dots, N - 1 \\G(k + N) &= X_1(k) - W_2^k N X_2(k), & k = 0, 1, \dots, N - 1\end{aligned}\tag{8.2.12}$$

Thus we have computed the DFT of a $2N$ -point real sequence from one N -point DFT and some additional computation as indicated by (8.2.11) and (8.2.12).

8.2.3 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 7 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 in computing either 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 7.3.2, and the $M - 1$ overlapping data points between successive output records are added together.

Let us assess the computational complexity of the FFT method for linear filtering. For this purpose, the one-time computation of $H(k)$ is insignificant and can be ignored. Each FFT requires $(N/2)\log_2 N$ complex multiplications and $N\log_2 N$ additions. Since the FFT is performed twice, once for the DFT and once for the IDFT, the computational burden is $N\log_2 N$ complex multiplications and $2N\log_2 N$ additions. There are also N complex multiplications and $N - 1$ additions required to compute $Y_m(k)$. Therefore, we have $(N\log_2 2N)/L$ complex multiplications per output data point and approximately $(2N\log_2 2N)/L$ additions per output data point.

The overlap-add method requires an incremental increase of $(M - 1)/L$ in the number of additions.

By way of comparison, a direct-form realization of the FIR filter involves M real multiplications per output point if the filter is not linear phase, and $M/2$ if it is linear phase (symmetric). Also, the number of additions is $M - 1$ per output point (see Sec. 10.2).

It is interesting to compare the efficiency of the FFT algorithm with the direct form realization of the FIR filter. Let us focus on the number of multiplications, which are more time consuming than additions. Suppose that $M = 128 = 2^7$ and $N = 2^\nu$. Then the number of complex multiplications per output point for an FFT size of $N = 2^\nu$ is

$$\begin{aligned} c(\nu) &= \frac{N \log_2 2N}{L} = \frac{2^\nu(\nu + 1)}{N - M + 1} \\ &\approx \frac{2^\nu(\nu + 1)}{2^\nu - 2^7} \end{aligned}$$

The values of $c(\nu)$ for different values of ν are given in Table 8.3. We observe that there is an optimum value of ν which minimizes $c(\nu)$. For the FIR filter of size $M = 128$, the optimum occurs at $\nu = 10$.

We should emphasize that $c(\nu)$ represents the number of complex multiplications for the FFT-based method. The number of real multiplications is four times this number. However, even if the FIR filter has linear phase (see Sec. 10.2), the number of computations per output point is still less with the FFT-based method. Furthermore, the efficiency of the FFT method can be improved by computing the DFT of two successive data blocks simultaneously, according to the method just described. Consequently, the FFT-based method is indeed superior from a computational point of view when the filter length is relatively large.

The computation of the cross correlation between two sequences by means of the FFT algorithm is similar to the linear FIR filtering problem just described. In practical applications involving crosscorrelation, at least one of the sequences has finite duration and is akin to the impulse response of the FIR filter. The second sequence may be a long sequence which contains the desired sequence corrupted by additive noise. Hence the second sequence is akin to the input to the FIR filter.

TABLE 8.3 Computational Complexity

Size of FFT $\nu = \log_2 N$	$c(\nu)$	Number of Complex Multiplications per Output Point
9		13.3
10		12.6
11		12.8
12		13.4
14		15.1

By time reversing the first sequence and computing its DFT, we have reduced the cross correlation to an equivalent convolution problem (i.e., a linear FIR filtering problem). Therefore, the methodology we developed for linear FIR filtering by use of the FFT applies directly.

8.3 A Linear Filtering Approach to Computation of the DFT

The FFT algorithm takes N points of input data and produces an output sequence of N points corresponding to the DFT of the input data. As we have shown, the radix-2 FFT algorithm performs the computation of the DFT in $(N/2) \log_2 N$ multiplications and $N \log_2 N$ additions for an N -point sequence.

There are some applications where only a selected number of values of the DFT are desired, but the entire DFT is not required. In such a case, the FFT algorithm may no longer be more efficient than a direct computation of the desired values of the DFT. In fact, when the desired number of values of the DFT is less than $\log_2 N$, a direct computation of the desired values is more efficient.

The direct computation of the DFT can be formulated as a linear filtering operation on the input data sequence. As we will demonstrate, the linear filter takes the form of a parallel bank of resonators where each resonator selects one of the frequencies $\omega_k = 2\pi k/N$, $k = 0, 1, \dots, N - 1$, corresponding to the N frequencies in the DFT.

There are other applications in which we require the evaluation of the z -transform of a finite-duration sequence at points other than the unit circle. If the set of desired points in the z -plane possesses some regularity, it is possible to also express the computation of the z -transform as a linear filtering operation. In this connection, we introduce another algorithm, called the chirp- z transform algorithm, which is suitable for evaluating the z -transform of a set of data on a variety of contours in the z -plane. This algorithm is also formulated as a linear filtering of a set of input data. As a consequence, the FFT algorithm can be used to compute the chirp- z transform and thus to evaluate the z -transform at various contours in the z -plane, including the unit circle.

8.3.1 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

$$X(k) = W_N^{-kN} \sum_{m=0}^{N-1} x(m) W_N^{km} = \sum_{m=0}^{N-1} x(m) W_N^{-k(N-m)} \quad (8.3.1)$$

We note that (8.3.1) is in the form of a convolution. Indeed, if we define the sequence $y_k(n)$ as

$$y_k(n) = \sum_{m=0}^{N-1} x(m) W_N^{-k(n-m)} \quad (8.3.2)$$

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

$$h_k(n) = W_N^{-kn} u(n) \quad (8.3.3)$$

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,

$$X(k) = y_k(n)|_{n=N} \quad (8.3.4)$$

as can be verified by comparing (8.3.1) with (8.3.2).

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

$$H_k(z) = \frac{1}{1 - W_N^{-k} z^{-1}} \quad (8.3.5)$$

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 (8.3.2), via convolution, we can use the difference equation corresponding to the filter given by (8.3.5) to compute $y_k(n)$ recursively. Thus we have

$$y_k(n) = W_N^{-k} y_k(n-1) + x(n), \quad y_k(-1) = 0 \quad (8.3.6)$$

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

The complex multiplications and additions inherent in (8.3.6) 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

$$H_k(z) = \frac{1 - W_N^k z^{-1}}{1 - 2 \cos(2\pi k/N) z^{-1} + z^{-2}} \quad (8.3.7)$$

The direct form II realization of the system illustrated in Fig. 8.3.1 is described by the difference equations

$$v_k(n) = 2 \cos \frac{2\pi k}{N} v_k(n-1) - v_k(n-2) + x(n) \quad (8.3.8)$$

$$y_k(n) = v_k(n) - W_N^k v_k(n-1) \quad (8.3.9)$$

with initial conditions $v_k(-1) = v_k(-2) = 0$.

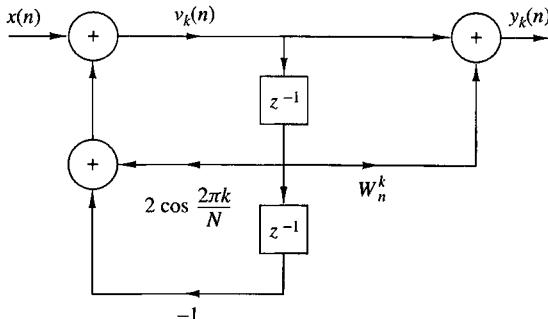


Figure 8.3.1
Direct form II realization
of two-pole resonator for
computing the DFT.

The recursive relation in (8.3.8) is iterated for $n = 0, 1, \dots, N$, but the equation in (8.3.9) is computed only once at time $n = N$. Each iteration requires one real multiplication and two additions. Consequently, for a real input sequence $x(n)$, this algorithm requires $N + 1$ real multiplications to yield not only $X(k)$ but also, due to symmetry, the value of $X(N - k)$.

The Goertzel algorithm is particularly attractive when the DFT is to be computed at a relatively small number M of values, where $M \leq \log_2 N$. Otherwise, the FFT algorithm is a more efficient method.

8.3.2 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,

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n}, \quad k = 0, 1, \dots, L-1 \quad (8.3.10)$$

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

$$\begin{aligned} z_k &= r e^{j2\pi kn/N}, \quad k = 0, 1, 2, \dots, N-1 \\ X(z_k) &= \sum_{n=0}^{N-1} [x(n)r^{-n}] e^{-j2\pi kn/N}, \quad k = 0, 1, 2, \dots, N-1 \end{aligned} \quad (8.3.11)$$

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

$$z_k = r_0 e^{j\theta_0} (R_0 e^{j\phi_0})^k, \quad k = 0, 1, \dots, L - 1 \quad (8.3.12)$$

Note that if $R_0 < 1$, the points fall on a contour that spirals toward the origin, and if $R_0 > 1$, the contour spirals away from the origin. If $R_0 = 1$, the contour is a circular arc of radius r_0 . If $r_0 = 1$ and $R_0 = 1$, the contour is an arc of the unit circle. The latter contour would allow us to compute the frequency content of the sequence $x(n)$ at a dense set of L frequencies in the range covered by the arc without having to compute a large DFT, that is, a DFT of the sequence $x(n)$ padded with many zeros to obtain the desired resolution in frequency. Finally, if $r_0 = R_0 = 1$, $\theta_0 = 0$, $\phi_0 = 2\pi/N$, and $L = N$, the contour is the entire unit circle and the frequencies are those of the DFT. The various contours are illustrated in Fig. 8.3.2.

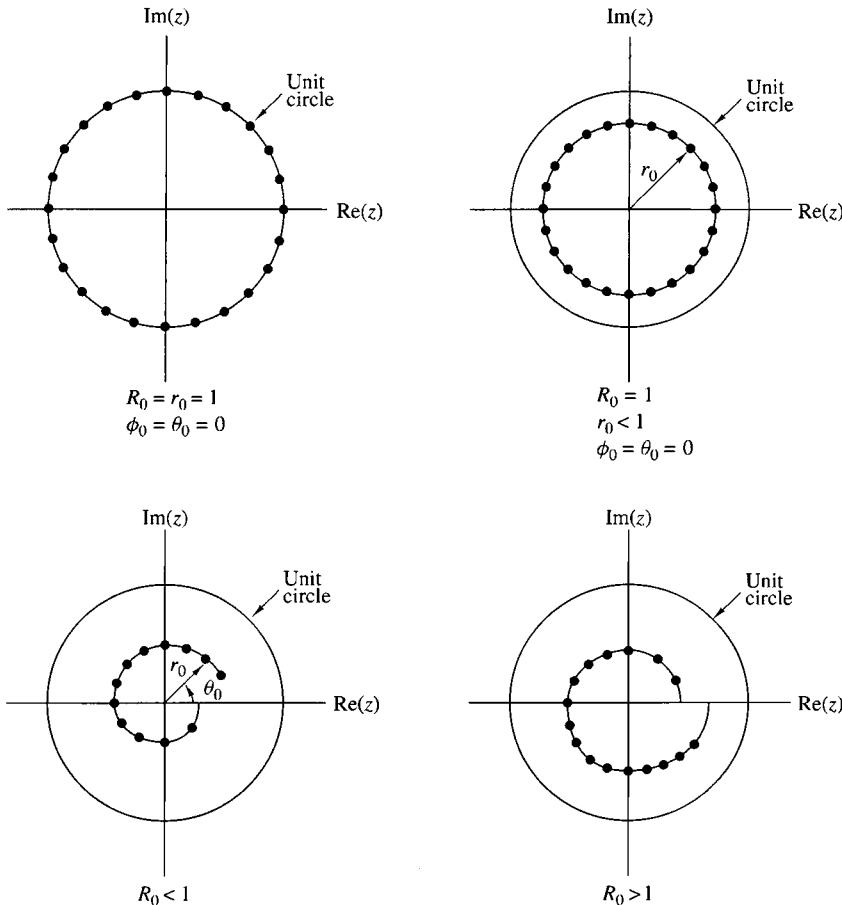


Figure 8.3.2 Some examples of contours on which we may evaluate the z -transform.

When points $\{z_k\}$ in (8.3.12) are substituted into the expression for the z -transform, we obtain

$$\begin{aligned} X(z_k) &= \sum_{n=0}^{N-1} x(n) z_k^{-n} \\ &= \sum_{n=0}^{N-1} x(n) (r_0 e^{j\theta_0})^{-n} V^{-nk} \end{aligned} \quad (8.3.13)$$

where, by definition,

$$V = R_0 e^{j\phi_0} \quad (8.3.14)$$

We can express (8.3.13) in the form of a convolution, by noting that

$$nk = \frac{1}{2}[n^2 + k^2 - (k - n)^2] \quad (8.3.15)$$

Substitution of (8.3.15) into (8.3.13) yields

$$X(z_k) = V^{-k^2/2} \sum_{n=0}^{N-1} [x(n) (r_0 e^{j\theta_0})^{-n} V^{-n^2/2}] V^{(k-n)^2/2} \quad (8.3.16)$$

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

$$g(n) = x(n) (r_0 e^{j\theta_0})^{-n} V^{-n^2/2} \quad (8.3.17)$$

Then (8.3.16) can be expressed as

$$X(z_k) = V^{-k^2/2} \sum_{n=0}^{N-1} g(n) V^{(k-n)^2/2} \quad (8.3.18)$$

The summation in (8.3.18) can be interpreted as the convolution of the sequence $g(n)$ with the impulse response $h(n)$ of a filter, where

$$h(n) = V^{n^2/2} \quad (8.3.19)$$

Consequently, (8.3.18) may be expressed as

$$\begin{aligned} X(z_k) &= V^{-k^2/2} y(k) \\ &= \frac{y(k)}{h(k)}, \quad k = 0, 1, \dots, L-1 \end{aligned} \quad (8.3.20)$$

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

$$y(k) = \sum_{n=0}^{N-1} g(n) h(k-n), \quad k = 0, 1, \dots, L-1 \quad (8.3.21)$$

We observe that both $h(n)$ and $g(n)$ are complex-valued sequences.

The sequence $h(n)$ with $R_0 = 1$ has the form of a complex exponential with argument $\omega n = n^2\phi_0/2 = (n\phi_0/2)n$. The quantity $n\phi_0/2$ represents the frequency of the complex exponential signal, which increases linearly with time. Such signals are used in radar systems and are called *chirp signals*. Hence the z -transform evaluated as in (8.3.18) is called the *chirp-z transform*.

The linear convolution in (8.3.21) is most efficiently done by use of the FFT algorithm. The sequence $g(n)$ is of length N . However, $h(n)$ has infinite duration. Fortunately, only a portion $h(n)$ is required to compute the L values of $X(z)$.

Since we will compute the convolution in (8.3.21) via the FFT, let us consider the circular convolution of the N -point sequence $g(n)$ with an M -point section of $h(n)$, where $M > N$. In such a case, we know that the first $N - 1$ points contain aliasing and that the remaining $M - N + 1$ points are identical to the result that would be obtained from a linear convolution of $h(n)$ with $g(n)$. In view of this, we should select a DFT of size

$$M = L + N - 1$$

which would yield L valid points and $N - 1$ points corrupted by aliasing.

The section of $h(n)$ that is needed for this computation corresponds to the values of $h(n)$ for $-(N - 1) \leq n \leq (L - 1)$, which is of length $M = L + N - 1$, as observed from (8.3.21). Let us define the sequence $h_1(n)$ of length M as

$$h_1(n) = h(n - N + 1), \quad n = 0, 1, \dots, M - 1 \quad (8.3.22)$$

and compute its M -point DFT via the FFT algorithm to obtain $H_1(k)$. From $x(n)$ we compute $g(n)$ as specified by (8.3.17), pad $g(n)$ with $L - 1$ zeros, and compute its M -point DFT to yield $G(k)$. The IDFT of the product $Y_1(k) = G(k)H_1(k)$ yields the M -point sequence $y_1(n)$, $n = 0, 1, \dots, M - 1$. The first $N - 1$ points of $y_1(n)$ are corrupted by aliasing and are discarded. The desired values are $y_1(n)$ for $N - 1 \leq n \leq M - 1$, which correspond to the range $0 \leq n \leq L - 1$ in (8.3.21), that is,

$$y(n) = y_1(n + N - 1), \quad n = 0, 1, \dots, L - 1 \quad (8.3.23)$$

Alternatively, we can define a sequence $h_2(n)$ as

$$h_2(n) = \begin{cases} h(n), & 0 \leq n \leq L - 1 \\ h(n - N - L + 1), & L \leq n \leq M - 1 \end{cases} \quad (8.3.24)$$

The M -point DFT of $h_2(n)$ yields $H_2(k)$, which when multiplied by $G(k)$ yields $Y_2(k) = G(k)H_2(k)$. The IDFT of $Y_2(k)$ yields the sequence $y_2(n)$ for $0 \leq n \leq M - 1$. Now the desired values of $y_2(n)$ are in the range $0 \leq n \leq L - 1$, that is,

$$y(n) = y_2(n), \quad n = 0, 1, \dots, L - 1 \quad (8.3.25)$$

Finally, the complex values $X(z_k)$ are computed by dividing $y(k)$ by $h(k)$, $k = 0, 1, \dots, L - 1$, as specified by (8.3.20).

In general, the computational complexity of the chirp- z transform algorithm described above is of the order of $M \log_2 M$ complex multiplications, where $M = N + L - 1$. This number should be compared with the product, $N \cdot L$, the number of computations required by direct evaluation of the z -transform. Clearly, if L is small, direct computation is more efficient. However, if L is large, then the chirp- z transform algorithm is more efficient.

The chirp- z transform method has been implemented in hardware to compute the DFT of signals. For the computation of the DFT, we select $r_0 = R_0 = 1$, $\theta_0 = 0$, $\phi_0 = 2\pi/N$, and $L = N$. In this case

$$\begin{aligned} V^{-n^2/2} &= e^{-j\pi n^2/N} \\ &= \cos \frac{\pi n^2}{N} - j \sin \frac{\pi n^2}{N} \end{aligned} \quad (8.3.26)$$

The chirp filter with impulse response

$$\begin{aligned} h(n) &= V^{n^2/2} \\ &= \cos \frac{\pi n^2}{N} + j \sin \frac{\pi n^2}{N} \\ &= h_r(n) + j h_i(n) \end{aligned} \quad (8.3.27)$$

has been implemented as a pair of FIR filters with coefficients $h_r(n)$ and $h_i(n)$, respectively. Both *surface acoustic wave* (SAW) devices and *charge coupled devices*

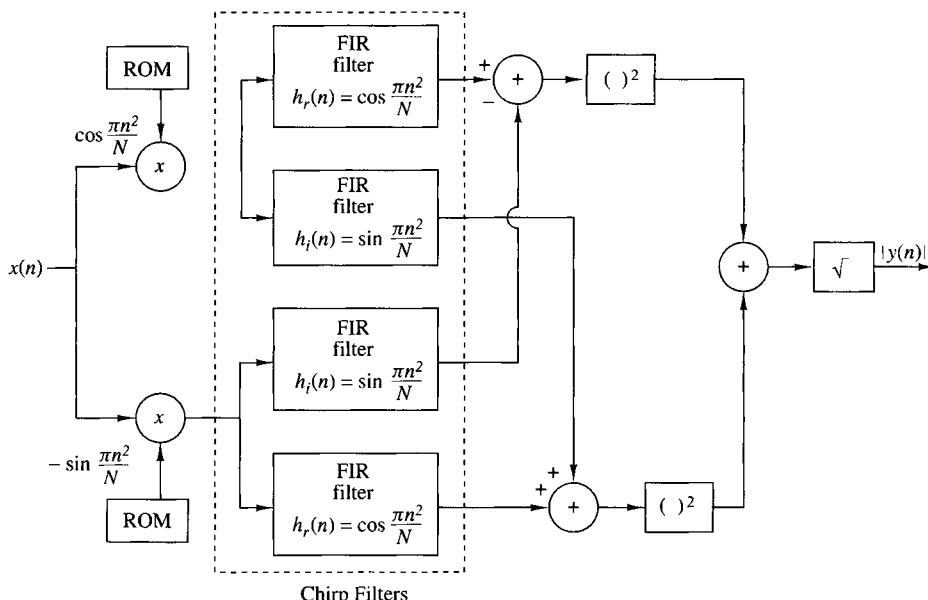


Figure 8.3.3 Block diagram illustrating the implementation of the chirp- z transform for computing the DFT (magnitude only).

(CCD) have been used in practice for the FIR filters. The cosine and sine sequences given in (8.3.26) needed for the premultiplications and postmultiplications are usually stored in a read-only memory (ROM). Furthermore, we note that if only the magnitude of the DFT is desired, the postmultiplications are unnecessary. In this case,

$$|X(z_k)| = |y(k)|, \quad k = 0, 1, \dots, N - 1 \quad (8.3.28)$$

as illustrated in Fig. 8.3.3. Thus the linear FIR filtering approach using the chirp- z transform has been implemented for the computation of the DFT.

8.4 Quantization Effects in the Computation of the DFT¹

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.

The model that we shall adopt for characterizing round-off errors in multiplication is the additive white noise model that we use in the statistical analysis of round-off errors in IIR and FIR filters (see Fig. 9.6.8). Although the statistical analysis is performed for rounding, the analysis can be easily modified to apply to truncation in two's-complement arithmetic (see Sec. 9.4.3).

Of particular interest is the analysis of round-off errors in the computation of the DFT via the FFT algorithm. However, we shall first establish a benchmark by determining the round-off errors in the direct computation of the DFT.

8.4.1 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

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \quad k = 0, 1, \dots, N - 1 \quad (8.4.1)$$

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.

¹ It is recommended that the reader review Section 9.5 prior to reading this section.

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

$$\sigma_e^2 = \frac{\Delta^2}{12} = \frac{2^{-2b}}{12} \quad (8.4.2)$$

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

$$\begin{aligned} \sigma_q^2 &= 4N\sigma_e^2 \\ &= \frac{N}{3} \cdot 2^{-2b} \end{aligned} \quad (8.4.3)$$

Hence the variance of the quantization error is proportional to the size of DFT. Note that when N is a power of 2 (i.e., $N = 2^v$), the variance can be expressed as

$$\sigma_q^2 = \frac{2^{-2(b-v/2)}}{3} \quad (8.4.4)$$

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

$$|X(k)| \leq \sum_{n=0}^{N-1} |x(n)| \quad (8.4.5)$$

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

$$\sum_{n=0}^{N-1} |x(n)| < 1 \quad (8.4.6)$$

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 (8.4.6) is satisfied.

The scaling implied by (8.4.6) 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

$$\sigma_x^2 = \frac{(2/N)^2}{12} = \frac{1}{3N^2} \quad (8.4.7)$$

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

$$\begin{aligned}\sigma_X^2 &= N\sigma_x^2 \\ &= \frac{1}{3N}\end{aligned}\tag{8.4.8}$$

Thus the signal-to-noise power ratio is

$$\frac{\sigma_X^2}{\sigma_q^2} = \frac{2^{2b}}{N^2}\tag{8.4.9}$$

We observe that the scaling is responsible for reducing the SNR by N and the combination of scaling and quantization errors results in a total reduction that is proportional to N^2 . Hence scaling the input sequence $\{x(n)\}$ to satisfy (8.4.6) imposes a severe penalty on the signal-to-noise ratio in the DFT.

EXAMPLE 8.4.1

Use (8.4.9) to determine the number of bits required to compute the DFT of a 1024-point sequence with an SNR of 30 dB.

Solution. The size of the sequence is $N = 2^{10}$. Hence the SNR is

$$10 \log_{10} \frac{\sigma_X^2}{\sigma_q^2} = 10 \log_{10} 2^{2b-20}$$

For an SNR of 30 dB, we have

$$3(2b - 20) = 30$$

$$b = 15 \text{ bits}$$

Note that the 15 bits is the precision for both multiplication and addition.

Instead of scaling the input sequence $\{x(n)\}$, suppose we simply require that $|x(n)| < 1$. Then we must provide a sufficiently large dynamic range for addition such that $|X(k)| < N$. In such a case, the variance of the sequence $\{|x(n)|\}$ is $\sigma_x^2 = \frac{1}{3}$, and hence the variance of $|X(k)|$ is

$$\sigma_X^2 = N\sigma_x^2 = \frac{N}{3}\tag{8.4.10}$$

Consequently, the SNR is

$$\frac{\sigma_X^2}{\sigma_q^2} = 2^{2b}\tag{8.4.11}$$

If we repeat the computation in Example 8.4.1, we find that the number of bits required to achieve an SNR of 30 dB is $b = 5$ bits. However, we need an additional 10 bits for the accumulator (the adder) to accommodate the increase in the dynamic range for addition. Although we did not achieve any reduction in the dynamic range for addition, we have managed to reduce the precision in multiplication from 15 bits to 5 bits, which is highly significant.

8.4.2 Quantization Errors in FFT Algorithms

As we have shown, the FFT algorithms require significantly fewer multiplications than the direct computation of the DFT. In view of this we might conclude that the computation of the DFT via an FFT algorithm will result in smaller quantization errors. Unfortunately, that is not the case, as we will demonstrate.

Let us consider the use of fixed-point arithmetic in the computation of a radix-2 FFT algorithm. To be specific, we select the radix-2, decimation-in-time algorithm illustrated in Fig. 8.4.1 for the case $N = 8$. The results on quantization errors that we obtain for this radix-2 FFT algorithm are typical of the results obtained with other radix-2 and higher radix algorithms.

We observe that each butterfly computation involves one complex-valued multiplication or, equivalently, four real multiplications. We ignore the fact that some butterflies contain a trivial multiplication by ± 1 . If we consider the butterflies that affect the computation of any one value of the DFT, we find that, in general, there are $N/2$ in the first stage of the FFT, $N/4$ in the second stage, $N/8$ in the third stage, and so on, until the last stage, where there is only one. Consequently, the number of

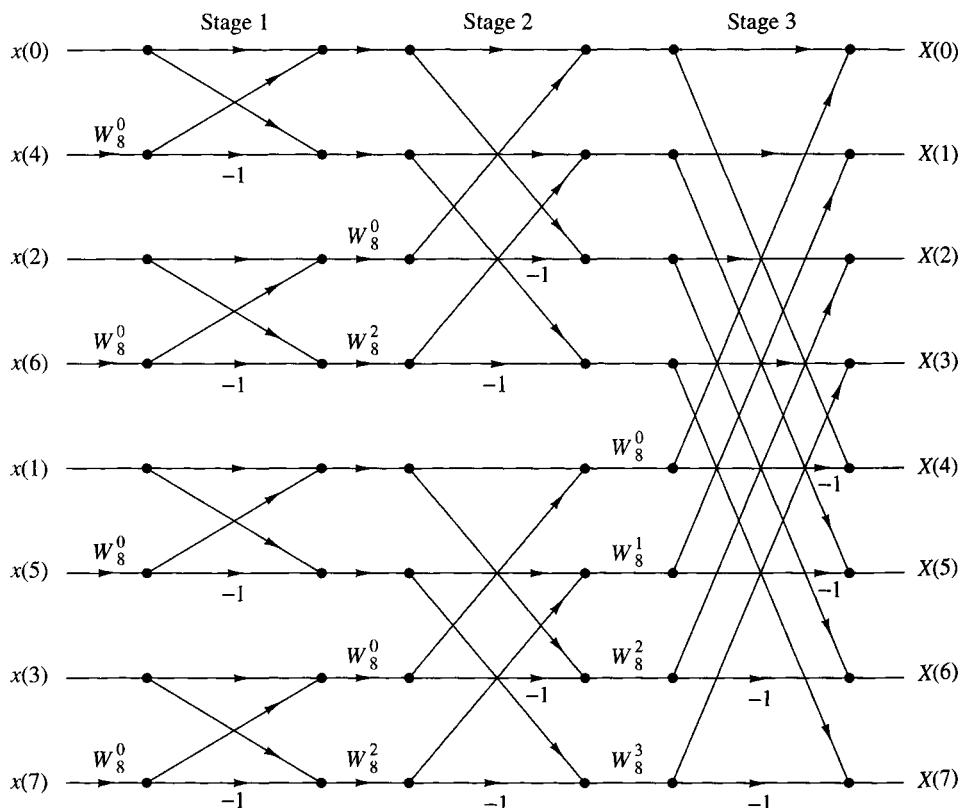


Figure 8.4.1 Decimation-in-time FFT algorithm.

butterflies per output point is

$$\begin{aligned} 2^{v-1} + 2^{v-2} + \cdots + 2 + 1 &= 2^{v-1} \left[1 + \left(\frac{1}{2}\right) + \cdots + \left(\frac{1}{2}\right)^{v-1} \right] \\ &= 2^v \left[1 - \left(\frac{1}{2}\right)^v \right] = N - 1 \end{aligned} \quad (8.4.12)$$

For example, the butterflies that affect the computation of $X(3)$ in the eight-point FFT algorithm of Fig. 8.4.1 are illustrated in Fig. 8.4.2.

The quantization errors introduced in each butterfly propagate to the output. Note that the quantization errors introduced in the first stage propagate through $(v - 1)$ stages, those introduced in the second stage propagate through $(v - 2)$ stages, and so on. As these quantization errors propagate through a number of subsequent stages, they are phase shifted (phase rotated) by the phase factors W_N^{kn} . These phase rotations do not change the statistical properties of the quantization errors and, in particular, the variance of each quantization error remains invariant.

If we assume that the quantization errors in each butterfly are uncorrelated with the errors in other butterflies, then there are $4(N - 1)$ errors that affect the output of each point of the FFT. Consequently, the variance of the total quantization error at the output is

$$\sigma_q^2 = 4(N - 1) \frac{\Delta^2}{12} \approx \frac{N\Delta^2}{3} \quad (8.4.13)$$

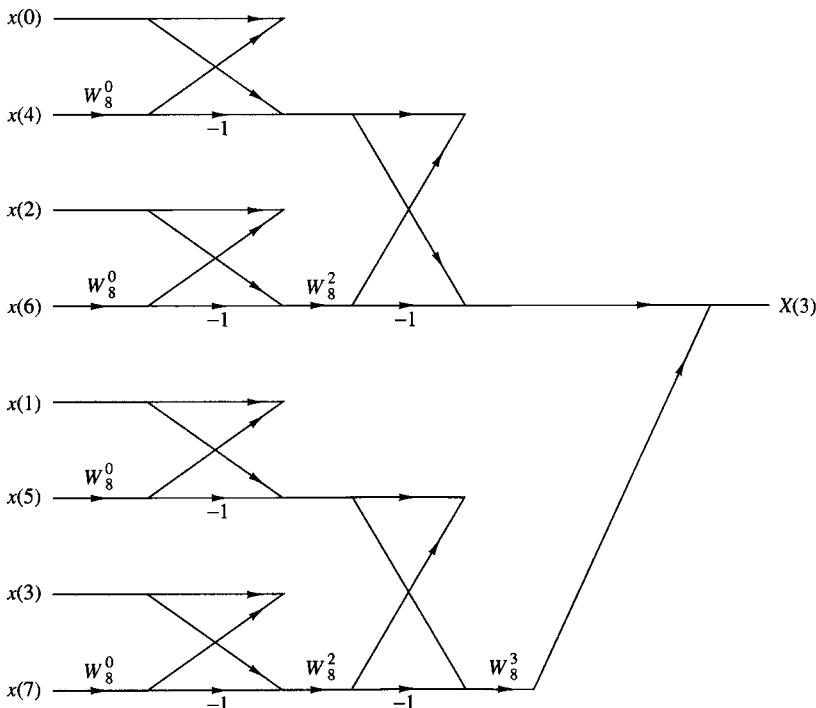


Figure 8.4.2 Butterflies that affect the computation of $X(3)$.

where $\Delta = 2^{-b}$. Hence

$$\sigma_q^2 = \frac{N}{3} \cdot 2^{-2b} \quad (8.4.14)$$

This is exactly the same result that we obtained for the direct computation of the DFT.

The result in (8.4.14) should not be surprising. In fact, the FFT algorithm does not reduce the number of multiplications required to compute a single point of the DFT. It does, however, exploit the periodicities in W_N^{kn} and thus reduces the number of multiplications in the computation of the entire block of N points in the DFT.

As in the case of the direct computation of the DFT, we must scale the input sequence to prevent overflow. Recall that if $|x(n)| < 1/N$, $0 \leq n \leq N - 1$, then $|X(k)| < 1$ for $0 \leq k \leq N - 1$. Thus overflow is avoided. With this scaling, the relations in (8.4.7), (8.4.8), and (8.4.9), obtained previously for the direct computation of the DFT, apply to the FFT algorithm as well. Consequently, the same SNR is obtained for the FFT.

Since the FFT algorithm consists of a sequence of stages, where each stage contains butterflies that involve pairs of points, it is possible to devise a different scaling strategy that is not as severe as dividing each input point by N . This alternative scaling strategy is motivated by the observation that the intermediate values $|X_n(k)|$ in the $n = 1, 2, \dots, v$ stages of the FFT algorithm satisfy the conditions (see Problem 8.35)

$$\begin{aligned} \max[|X_{n+1}(k)|, |X_{n+1}(l)|] &\geq \max[|X_n(k)|, |X_n(l)|] \\ \max[|X_{n+1}(k)|, |X_{n+1}(l)|] &\leq 2\max[|X_n(k)|, |X_n(l)|] \end{aligned} \quad (8.4.15)$$

In view of these relations, we can distribute the total scaling of $1/N$ into each of the stages of the FFT algorithm. In particular, if $|x(n)| < 1$, we apply a scale factor of $\frac{1}{2}$ in the first stage so that $|x(n)| < \frac{1}{2}$. Then the output of each subsequent stage in the FFT algorithm is scaled by $\frac{1}{2}$, so that after v stages we have achieved an overall scale factor of $(\frac{1}{2})^v = 1/N$. Thus overflow in the computation of the DFT is avoided.

This scaling procedure does not affect the signal level at the output of the FFT algorithm, but it significantly reduces the variance of the quantization errors at the output. Specifically, each factor of $\frac{1}{2}$ reduces the variance of a quantization error term by a factor of $\frac{1}{4}$. Thus the $4(N/2)$ quantization errors introduced in the first stage are reduced in variance by $(\frac{1}{4})^{v-1}$, the $4(N/4)$ quantization errors introduced in the second stage are reduced in variance by $(\frac{1}{4})^{v-2}$, and so on. Consequently, the total variance of the quantization errors at the output of the FFT algorithm is

$$\begin{aligned} \sigma_q^2 &= \frac{\Delta^2}{12} \left\{ 4 \left(\frac{N}{2} \right) \left(\frac{1}{4} \right)^{v-1} + 4 \left(\frac{N}{4} \right) \left(\frac{1}{4} \right)^{v-2} + 4 \left(\frac{N}{8} \right) \left(\frac{1}{4} \right)^{v-3} + \cdots + 4 \right\} \\ &= \frac{\Delta^2}{3} \left\{ \left(\frac{1}{2} \right)^{v-1} + \left(\frac{1}{2} \right)^{v-2} + \cdots + \frac{1}{2} + 1 \right\} \\ &= \frac{2\Delta^2}{3} \left[1 - \left(\frac{1}{2} \right)^v \right] \approx \frac{2}{3} \cdot 2^{-2b} \end{aligned} \quad (8.4.16)$$

where the factor $(\frac{1}{2})^v$ is negligible.

We now observe that (8.4.16) is no longer proportional to N . On the other hand, the signal has the variance $\sigma_x^2 = 1/3N$, as given in (8.4.8). Hence the SNR is

$$\begin{aligned}\frac{\sigma_x^2}{\sigma_q^2} &= \frac{1}{2N} \cdot 2^{2b} \\ &= 2^{2b-v-1}\end{aligned}\tag{8.4.17}$$

Thus, by distributing the scaling of $1/N$ uniformly throughout the FFT algorithm, we have achieved an SNR that is inversely proportional to N instead of N^2 .

EXAMPLE 8.4.2

Determine the number of bits required to compute an FFT of 1024 points with an SNR of 30 dB when the scaling is distributed as described above.

Solution. The size of the FFT is $N = 2^{10}$. Hence the SNR according to (8.4.17) is

$$\begin{aligned}10 \log_{10} 2^{2b-v-1} &= 30 \\ 3(2b - 11) &= 30 \\ b &= \frac{21}{2} \text{(11 bits)}\end{aligned}$$

This can be compared with the 15 bits required if all the scaling is performed in the first stage of the FFT algorithm.

8.5 Summary and References

The focus of this chapter was on the efficient computation of the DFT. We demonstrated that by taking advantage of the symmetry and periodicity properties of the exponential factors W_N^{kn} , we can reduce the number of complex multiplications needed to compute the DFT from N^2 to $N \log_2 N$ when N is a power of 2. As we indicated, any sequence can be augmented with zeros, such that $N = 2^v$.

For decades, FFT-type algorithms were of interest to mathematicians who were concerned with computing values of Fourier series by hand. However, it was not until Cooley and Tukey (1965) published their well-known paper that the impact and significance of the efficient computation of the DFT was recognized. Since then the Cooley–Tukey FFT algorithm and its various forms, for example, the algorithms of Singleton (1967, 1969), have had a tremendous influence on the use of the DFT in convolution, correlation, and spectrum analysis. For a historical perspective on the FFT algorithm, the reader is referred to the paper by Cooley et al. (1967).

The split-radix FFT (SRFFT) algorithm described in Section 8.1.5 is due to Duhamel and Hollmann (1984, 1986). The “mirror” FFT (MFFT) and “phase” FFT (PFFT) algorithms were described to the authors by R. Price. The exploitation of symmetry properties in the data to reduce the computation time is described in a paper by Swarztrauber (1986).

Over the years, a number of tutorial papers have been published on FFT algorithms. We cite the early papers by Brigham and Morrow (1967), Cochran et al. (1967), Bergland (1969), and Cooley et al. (1967, 1969).

The recognition that the DFT can be arranged and computed as a linear convolution is also highly significant. Goertzel (1968) indicated that the DFT can be computed via linear filtering, although the computational savings of this approach is rather modest, as we have observed. More significant is the work of Bluestein (1970), who demonstrated that the computation of the DFT can be formulated as a chirp linear filtering operation. This work led to the development of the chirp-z transform algorithm by Rabiner et al. (1969).

In addition to the FFT algorithms described in this chapter, there are other efficient algorithms for computing the DFT, some of which further reduce the number of multiplications, but usually require more additions. Of particular importance is an algorithm due to Rader and Brenner (1976), the class of prime factor algorithms, such as the Good algorithm (1971), and the Winograd algorithm (1976, 1978). For a description of these and related algorithms, the reader may refer to the text by Blahut (1985).

Problems

- 8.1** Show that each of the numbers

$$e^{j(2\pi/N)k}, \quad 0 \leq k \leq N - 1$$

corresponds to an N th root of unity. Plot these numbers as phasors in the complex plane and illustrate, by means of this figure, the orthogonality property

$$\sum_{n=0}^{N-1} e^{j(2\pi/N)kn} e^{-j(2\pi/N)ln} = \begin{cases} N, & \text{if } k = l \\ 0, & \text{if } k \neq l \end{cases}$$

- 8.2 (a)** Show that the phase factors can be computed recursively by

$$W_N^{ql} = W_N^q W_N^{q(l-1)}$$

(b) Perform this computation once using single-precision floating-point arithmetic and once using only four significant digits. Note the deterioration due to the accumulation of round-off errors in the latter case.

(c) Show how the results in part (b) can be improved by resetting the result to the correct value $-j$, each time $ql = N/4$.

- 8.3** Let $x(n)$ be a real-valued N -point ($N = 2^v$) sequence. Develop a method to compute an N -point DFT $X'(k)$, which contains only the odd harmonics [i.e., $X'(k) = 0$ if k is even] by using only a real $N/2$ -point DFT.

- 8.4** A designer has available a number of eight-point FFT chips. Show explicitly how he should interconnect three such chips in order to compute a 24-point DFT.

- 8.5** The z -transform of the sequence $x(n) = u(n) - u(n - 7)$ is sampled at five points on the unit circle as follows:

$$x(k) = X(z)|_{z=e^{j2\pi k/5}}, \quad k = 0, 1, 2, 3, 4$$

Determine the inverse DFT $x'(n)$ of $X(k)$. Compare it with $x(n)$ and explain the results.

- 8.6** Consider a finite-duration sequence $x(n)$, $0 \leq n \leq 7$, with z -transform $X(z)$. We wish to compute $X(z)$ at the following set of values:

$$z_k = 0.8e^{j[(2\pi k/8) + (\pi/8)]}, \quad 0 \leq k \leq 7$$

- (a) Sketch the points $\{z_k\}$ in the complex plane.
 (b) Determine a sequence $s(n)$ such that its DFT provides the desired samples of $X(z)$.

- 8.7** Derive the radix-2 decimation-in-time FFT algorithm given by (8.1.26) and (8.1.27) as a special case of the more general algorithmic procedure given by (8.1.16) through (8.1.18).

- 8.8** Compute the eight-point DFT of the sequence

$$x(n) = \begin{cases} 1, & 0 \leq n \leq 7 \\ 0, & \text{otherwise} \end{cases}$$

by using the decimation-in-frequency FFT algorithm described in the text.

- 8.9** Derive the signal flow graph for the $N = 16$ -point, radix-4 decimation-in-time FFT algorithm in which the input sequence is in normal order and the computations are done in place.
8.10 Derive the signal flow graph for the $N = 16$ -point, radix-4 decimation-in-frequency FFT algorithm in which the input sequence is in digit-reversed order and the output DFT is in normal order.
8.11 Compute the eight-point DFT of the sequence

$$x(n) = \left\{ \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, 0, 0, 0 \right\}$$

using the in-place radix-2 decimation-in-time and radix-2 decimation-in-frequency algorithms. Follow exactly the corresponding signal flow graphs and keep track of all the intermediate quantities by putting them on the diagrams.

- 8.12** Compute the 16-point DFT of the sequence

$$x(n) = \cos \frac{\pi}{2} n, \quad 0 \leq n \leq 15$$

using the radix-4 decimation-in-time algorithm.

- 8.13** Consider the eight-point decimation-in-time (DIT) flow graph in Fig. 8.1.6.
 (a) What is the gain of the “signal path” that goes from $x(7)$ to $X(2)$?
 (b) How many paths lead from the input to a given output sample? Is this true for every output sample?
 (c) Compute $X(3)$ using the operations dictated by this flow graph.

- 8.14** Draw the flow graph for the decimation-in-frequency (DIF) SRFFT algorithm for $N = 16$. What is the number of nontrivial multiplications?
- 8.15** Derive the algorithm and draw the $N = 8$ flow graph for the DIT SRFFT algorithm. Compare your flow graph with the DIF radix-2 FFT flow graph shown in Fig. 8.1.11.
- 8.16** Show that the product of two complex numbers $(a+jb)$ and $(c+jd)$ can be performed with three real multiplications and five additions using the algorithm

$$x_R = (a - b)d + (c - d)a$$

$$x_I = (a - b)d + (c + d)b$$

where

$$x = x_R + jx_I = (a + jb)(c + jd)$$

- 8.17** Explain how the DFT can be used to compute N equispaced samples of the z -transform of an N -point sequence, on a circle of radius r .
- 8.18** A real-valued N -point sequence $x(n)$ is called DFT bandlimited if its DFT $X(k) = 0$ for $k_0 \leq k \leq N - k_0$. We insert $(L - 1)N$ zeros in the middle of $X(k)$ to obtain the following LN -point DFT:

$$X'(k) = \begin{cases} X(k), & 0 \leq k \leq k_0 - 1 \\ 0, & k_0 \leq k \leq LN - k_0 \\ X(k + N - LN), & LN - k_0 + 1 \leq k \leq LN - 1 \end{cases}$$

Show that

$$Lx'(Ln) = x(n), \quad 0 \leq n \leq N - 1$$

where

$$x'(n) \xrightarrow[LN]{DFT} X'(k)$$

Explain the meaning of this type of processing by working out an example with $N = 4$, $L = 1$, and $X(k) = \{1, 0, 0, 1\}$.

- 8.19** Let $X(k)$ be the N -point DFT of the sequence $x(n)$, $0 \leq n \leq N - 1$. What is the N -point DFT of the sequence $s(n) = X(n)$, $0 \leq n \leq N - 1$?
- 8.20** Let $X(k)$ be the N -point DFT of the sequence $x(n)$, $0 \leq n \leq N - 1$. We define a $2N$ -point sequence $y(n)$ as

$$y(n) = \begin{cases} x\left(\frac{n}{2}\right), & n \text{ even} \\ 0, & n \text{ odd} \end{cases}$$

Express the $2N$ -point DFT of $y(n)$ in terms of $X(k)$.

- 8.21 (a)** Determine the z -transform $W(z)$ of the Hanning window
 $w(n) = (1 - \cos \frac{2\pi n}{N-1}) / 2$.
- (b)** Determine a formula to compute the N -point DFT $X_w(k)$ of the signal $x_w(n) = w(n)x(n)$, $0 \leq n \leq N - 1$, from the N -point DFT $X(k)$ of the signal $x(n)$.

- 8.22** Create a DFT coefficient table that uses only $N/4$ memory locations to store the first quadrant of the sine sequence (assume N even).
- 8.23** Determine the computational burden of the algorithm given by (8.2.12) and compare it with the computational burden required in the $2N$ -point DFT of $g(n)$. Assume that the FFT algorithm is a radix-2 algorithm.
- 8.24** Consider an IIR system described by the difference equation

$$y(n) = - \sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k)$$

Describe a procedure that computes the frequency response $H\left(\frac{2\pi}{N}k\right)$, $k = 0, 1, \dots, N-1$ using the FFT algorithm ($N = 2^v$).

- 8.25** Develop a radix-3 decimation-in-time FFT algorithm for $N = 3^v$ and draw the corresponding flow graph for $N = 9$. What is the number of required complex multiplications? Can the operations be performed in place?
- 8.26** Repeat Problem 8.25 for the DIF case.
- 8.27** *FFT input and output pruning* In many applications we wish to compute only a few points M of the N -point DFT of a finite-duration sequence of length L (i.e., $M \ll N$ and $L \ll N$).
- (a) Draw the flow graph of the radix-2 DIF FFT algorithm for $N = 16$ and eliminate [i.e., prune] all signal paths that originate from zero inputs assuming that only $x(0)$ and $x(1)$ are nonzero.
 - (b) Repeat part (a) for the radix-2 DIT algorithm.
 - (c) Which algorithm is better if we wish to compute all points of the DFT? What happens if we want to compute only the points $X(0)$, $X(1)$, $X(2)$, and $X(3)$? Establish a rule to choose between DIT and DIF pruning depending on the values of M and L .
 - (d) Give an estimate of saving in computations in terms of M , L , and N .
- 8.28** *Parallel computation of the DFT* Suppose that we wish to compute an $N = 2^p 2^v$ -point DFT using 2^p digital signal processors (DSPs). For simplicity we assume that $p = v = 2$. In this case each DSP carries out all the computations that are necessary to compute 2^v DFT points.
- (a) Using the radix-2 DIF flow graph, show that to avoid data shuffling, the entire sequence $x(n)$ should be loaded to the memory of each DSP.
 - (b) Identify and redraw the portion of the flow graph that is executed by the DSP that computes the DFT samples $X(2)$, $X(10)$, $X(6)$, and $X(14)$.
 - (c) Show that, if we use $M = 2^p$ DSPs, the computation speed-up S is given by

$$S = M \frac{\log_2 N}{\log_2 N - \log_2 M + 2(M-1)}$$

- 8.29** Develop an inverse radix-2 DIT FFT algorithm starting with the definition. Draw the flow graph for computation and compare with the corresponding flow graph for the direct FFT. Can the IFFT flow graph be obtained from the one for the direct FFT?
- 8.30** Repeat Problem 8.29 for the DIF case.
- 8.31** Show that an FFT on data with Hermitian symmetry can be derived by reversing the flow graph of an FFT for real data.
- 8.32** Determine the system function $H(z)$ and the difference equation for the system that uses the Goertzel algorithm to compute the DFT value $X(N - k)$.
- 8.33** (a) Suppose that $x(n)$ is a finite-duration sequence of $N = 1024$ points. It is desired to evaluate the z -transform $X(z)$ of the sequence at the points

$$z_k = e^{j(2\pi/1024)k}, \quad k = 0, 100, 200, \dots, 1000$$

by using the most efficient method or algorithm possible. Describe an algorithm for performing this computation efficiently. Explain how you arrived at your answer by giving the various options or algorithms that can be used.

- (b) Repeat part (a) if $X(z)$ is to be evaluated at

$$z_k = 2(0.9)^k e^{j[(2\pi/5000)k + \pi/2]}, \quad k = 0, 1, 2, \dots, 999$$

- 8.34** Repeat the analysis for the variance of the quantization error, carried out in Section 8.4.2, for the decimation-in-frequency radix-2 FFT algorithm.
- 8.35** The basic butterfly in the radix-2 decimation-in-time FFT algorithm is

$$\begin{aligned} X_{n+1}(k) &= X_n(k) + W_N^m X_n(l) \\ X_{n+1}(l) &= X_n(k) - W_N^m X_n(l) \end{aligned}$$

- (a) If we require that $|X_n(k)| < \frac{1}{2}$ and $|X_n(l)| < \frac{1}{2}$, show that

$$\begin{aligned} |\operatorname{Re}[X_n X_{n+1}(k)]| &< 1, & |\operatorname{Re}[X_{n+1}(l)]| &< 1 \\ |\operatorname{Im}[X_n X_{n+1}(k)]| &< 1, & |\operatorname{Im}[X_{n+1}(l)]| &< 1 \end{aligned}$$

Thus overflow does not occur.

- (b) Prove that

$$\begin{aligned} \max[|X_{n+1}(k)|, |X_{n+1}(l)|] &\geq \max[|X_n(k)|, |X_n(l)|] \\ \max[|X_{n+1}(k)|, |X_{n+1}(l)|] &\leq 2 \max[|X_n(k)|, |X_n(l)|] \end{aligned}$$

- 8.36 Computation of the DFT** Use an FFT subroutine to compute the following DFTs and plot the magnitudes $|X(k)|$ of the DFTs.

(a) The 64-point DFT of the sequence

$$x(n) = \begin{cases} 1, & n = 0, 1, \dots, 15 \quad (N_1 = 16) \\ 0, & \text{otherwise} \end{cases}$$

(b) The 64-point DFT of the sequence

$$x(n) = \begin{cases} 1, & n = 0, 1, \dots, 7 \quad (N_1 = 8) \\ 0, & \text{otherwise} \end{cases}$$

(c) The 128-point DFT of the sequence in part (a).

(d) The 64-point DFT of the sequence

$$x(n) = \begin{cases} 10e^{j(\pi/8)n}, & n = 0, 1, \dots, 63 \quad (N_1 = 64) \\ 0, & \text{otherwise} \end{cases}$$

Answer the following questions.

1. What is the frequency interval between successive samples for the plots in parts (a), (b), (c), and (d)?
2. What is the value of the spectrum at zero frequency (dc value) obtained from the plots in parts (a), (b), (c), (d)?

From the formula

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j(2\pi/N)nk}$$

compute the theoretical values for the dc value and check these with the computer results.

3. In plots (a), (b), and (c), what is the *frequency interval* between successive nulls in the spectrum? What is the relationship between N_1 of the sequence $x(n)$ and the frequency interval between successive nulls?
4. Explain the difference between the plots obtained from parts (a) and (c).

- 8.37 Identification of pole positions in a system** Consider the system described by the difference equation

$$y(n) = -r^2 y(n-2) + x(n)$$

- (a) Let $r = 0.9$ and $x(n) = \delta(n)$. Generate the output sequence $y(n)$ for $0 \leq n \leq 127$. Compute the $N = 128$ -point DFT $\{Y(k)\}$ and plot $\{|Y(k)|\}$.

- (b) Compute the $N = 128$ -point DFT of the sequence

$$w(n) = (0.92)^{-n} y(n)$$

where $y(n)$ is the sequence generated in part (a). Plot the DFT values $|W(k)|$. What can you conclude from the plots in parts (a) and (b)?

- (c) Let $r = 0.5$ and repeat part (a).
(d) Repeat part (b) for the sequence

$$w(n) = (0.55)^{-n} y(n)$$

where $y(n)$ is the sequence generated in part (c). What can you conclude from the plots in parts (c) and (d)?

- (e) Now let the sequence generated in part (c) be corrupted by a sequence of “measurement” noise which is Gaussian with zero mean and variance $\sigma^2 = 0.1$. Repeat parts (c) and (d) for the noise-corrupted signal.

Implementation of Discrete-Time Systems

The focus of this chapter is on the realization of linear time-invariant discrete-time systems either in software or hardware. As we noted in Chapter 2, there are various configurations or structures for the realization of any FIR and IIR discrete-time system. In Chapter 2 we described the simplest of these structures, namely, the direct-form realizations. However, there are other more practical structures that offer some distinct advantages, especially when quantization effects are taken into consideration.

Of particular importance are the cascade, parallel, and lattice structures, which exhibit robustness in finite-word-length implementations. Also described in this chapter is the frequency-sampling realization for an FIR system, which often has the advantage of being computationally efficient when compared with alternative FIR realizations. Other filter structures are obtained by employing a state-space formulation for linear time-invariant systems. Due to space limitations, state-space structures are not covered.

In addition to describing the various structures for the realization of discrete-time systems, we also treat problems associated with quantization effects in the implementation of digital filters using finite-precision arithmetic. This treatment includes the effects on the filter frequency response characteristics resulting from coefficient quantization and the round-off noise effects inherent in the digital implementation of discrete-time systems.

9.1 Structures for the Realization of Discrete-Time Systems

Let us consider the important class of linear time-invariant discrete-time systems characterized by the general linear constant-coefficient difference equation

$$y(n) = - \sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k) \quad (9.1.1)$$

As we have shown by means of the z -transform, linear time-invariant discrete-time systems of this class are also characterized by the rational system function

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.1.2)$$

which is a ratio of two polynomials in z^{-1} . From the latter characterization, we obtain the zeros and poles of the system function, which depend on the choice of the system parameters $\{b_k\}$ and $\{a_k\}$ and which determine the frequency response characteristics of the system.

Our focus in this chapter is on the various methods of implementing (9.1.1) or (9.1.2) either in hardware, or in software on a programmable digital computer. We shall show that (9.1.1) or (9.1.2) can be implemented in a variety of ways depending on the form in which these two characterizations are arranged.

In general, we can view (9.1.1) as a computational procedure (an algorithm) for determining the output sequence $y(n)$ of the system from the input sequence $x(n)$. However, in various ways, the computations in (9.1.1) can be arranged into equivalent sets of difference equations. Each set of equations defines a computational procedure or an algorithm for implementing the system. From each set of equations we can construct a block diagram consisting of an interconnection of delay elements, multipliers, and adders. In Section 2.5 we referred to such a block diagram as a *realization* of the system or, equivalently, as a *structure* for realizing the system.

If the system is to be implemented in software, the block diagram or, equivalently, the set of equations that are obtained by rearranging (9.1.1), can be converted into a program that runs on a digital computer. Alternatively, the structure in block diagram form implies a hardware configuration for implementing the system.

Perhaps, the one issue that may not be clear to the reader at this point is why we are considering any rearrangements of (9.1.1) or (9.1.2). Why not just implement (9.1.1) or (9.1.2) directly without any rearrangement? If either (9.1.1) or (9.1.2) is rearranged in some manner, what are the benefits gained in the corresponding implementation?

These are the important questions which are answered in this chapter. At this point in our development, we simply state that the major factors that influence our choice of a specific realization are computational complexity, memory requirements, and finite-word-length effects in the computations.

Computational complexity refers to the number of arithmetic operations (multiplications, divisions, and additions) required to compute an output value $y(n)$ for the system. In the past, these were the only items used to measure computational complexity. However, with recent developments in the design and fabrication of rather

sophisticated programmable digital signal processing chips, other factors, such as the number of times a fetch from memory is performed or the number of times a comparison between two numbers is performed per output sample, have become important in assessing the computational complexity of a given realization of a system.

Memory requirements refers to the number of memory locations required to store the system parameters, past inputs, past outputs, and any intermediate computed values.

Finite-word-length effects or finite-precision effects refer to the quantization effects that are inherent in any digital implementation of the system, either in hardware or in software. The parameters of the system must necessarily be represented with finite precision. The computations that are performed in the process of computing an output from the system must be rounded off or truncated to fit within the limited precision constraints of the computer or the hardware used in the implementation. Whether the computations are performed in fixed-point or floating-point arithmetic is another consideration. All these problems are usually called finite-word-length effects and are extremely important in influencing our choice of a system realization. We shall see that different structures of a system, which are equivalent for infinite precision, exhibit different behavior when finite-precision arithmetic is used in the implementation. Therefore, it is very important in practice to select a realization that is not very sensitive to finite-word-length effects.

Although these three factors are the major ones in influencing our choice of the realization of a system of the type described by either (9.1.1) or (9.1.2), other factors, such as whether the structure or the realization lends itself to parallel processing, or whether the computations can be pipelined, may play a role in our selection of the specific implementation. These additional factors are usually important in the realization of more complex digital signal processing algorithms.

In our discussion of alternative realizations, we concentrate on the three major factors just outlined. Occasionally, we will include some additional factors that may be important in some implementations.

9.2 Structures for FIR Systems

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

$$y(n) = \sum_{k=0}^{M-1} b_k x(n-k) \quad (9.2.1)$$

or, equivalently, by the system function

$$H(z) = \sum_{k=0}^{M-1} b_k z^{-k} \quad (9.2.2)$$

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

$$h(n) = \begin{cases} b_n, & 0 \leq n \leq M-1 \\ 0, & \text{otherwise} \end{cases} \quad (9.2.3)$$

The length of the FIR filter is selected as M to conform with the established notation in the technical literature. The reader should note this change in notation in the treatment of FIR filters in this and subsequent chapters.

We shall present several methods for implementing an FIR system, beginning with the simplest structure, called the direct form. A second structure is the cascade-form realization. The third structure that we shall describe is the frequency-sampling realization. Finally, we present a lattice realization of an FIR system. In this discussion we follow the convention often used in the technical literature, which is to use $\{h(n)\}$ for the parameters of an FIR system.

In addition to the four realizations indicated above, an FIR system can be realized by means of the DFT, as described in Section 8.2. From one point of view, the DFT can be considered as a computational procedure rather than a structure for an FIR system. However, when the computational procedure is implemented in hardware, there is a corresponding structure for the FIR system. In practice, hardware implementations of the DFT are based on the use of the fast Fourier transform (FFT) algorithms described in Chapter 8.

9.2.1 Direct-Form Structure

The direct-form realization follows immediately from the nonrecursive difference equation given by (9.2.1) or, equivalently, by the convolution summation

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (9.2.4)$$

The structure is illustrated in Fig. 9.2.1.

We observe that this structure requires $M - 1$ memory locations for storing the $M - 1$ previous inputs, and has a complexity of M multiplications and $M - 1$ additions per output point. Since the output consists of a weighted linear combination of $M - 1$ past values of the input and the weighted current value of the input, the structure in Fig. 9.2.1 resembles a tapped delay line or a transversal system. Consequently, the direct-form realization is often called a transversal or tapped-delay-line filter.

When the FIR system has linear phase, as described in Section 10.2, the unit sample response of the system satisfies either the symmetry or asymmetry condition

$$h(n) = \pm h(M - 1 - n) \quad (9.2.5)$$

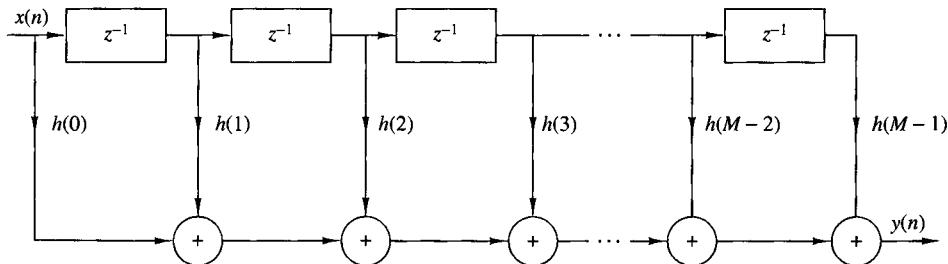


Figure 9.2.1 Direct-form realization of FIR system.

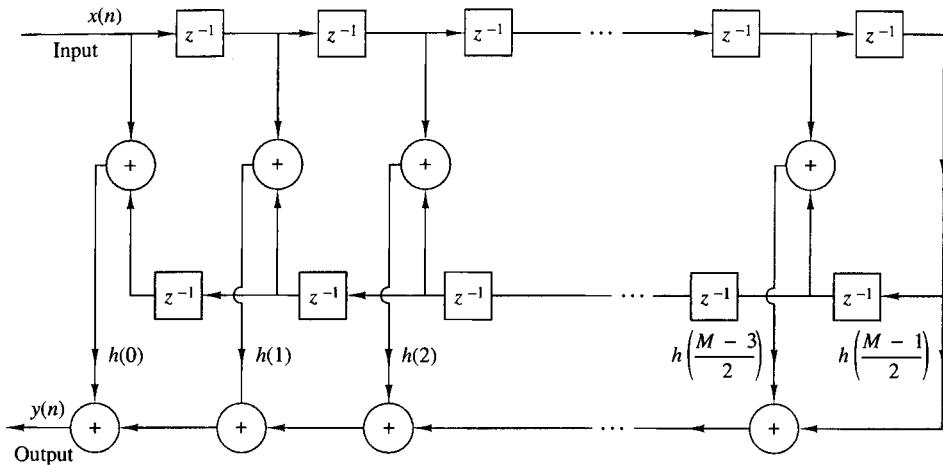


Figure 9.2.2 Direct-form realization of linear-phase FIR system (M odd).

For such a system the number of multiplications is reduced from M to $M/2$ for M even and to $(M - 1)/2$ for M odd. For example, the structure that takes advantage of this symmetry is illustrated in Fig. 9.2.2 for the case in which M is odd.

9.2.2 Cascade-Form Structures

The cascade realization follows naturally from the system function given by (9.2.2). It is a simple matter to factor $H(z)$ into second-order FIR systems so that

$$H(z) = \prod_{k=1}^K H_k(z) \quad (9.2.6)$$

where

$$H_k(z) = b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}, \quad k = 1, 2, \dots, K \quad (9.2.7)$$

and K is the integer part of $(M + 1)/2$. The filter parameter b_0 may be equally distributed among the K filter sections, such that $b_0 = b_{10}b_{20} \cdots b_{K0}$ or it may be assigned to a single filter section. The zeros of $H(z)$ are grouped in pairs to produce the second-order FIR systems of the form (9.2.7). It is always desirable to form pairs of complex-conjugate roots so that the coefficients $\{b_{ki}\}$ in (9.2.7) are real valued. On the other hand, real-valued roots can be paired in any arbitrary manner. The cascade-form realization along with the basic second-order section are shown in Fig. 9.2.3.

In the case of linear-phase FIR filters, the symmetry in $h(n)$ implies that the zeros of $H(z)$ also exhibit a form of symmetry. In particular, if z_k and z_k^* are a pair of complex-conjugate zeros then $1/z_k$ and $1/z_k^*$ are also a pair of complex-conjugate zeros (see Sec. 10.2). Consequently, we gain some simplification by forming fourth-

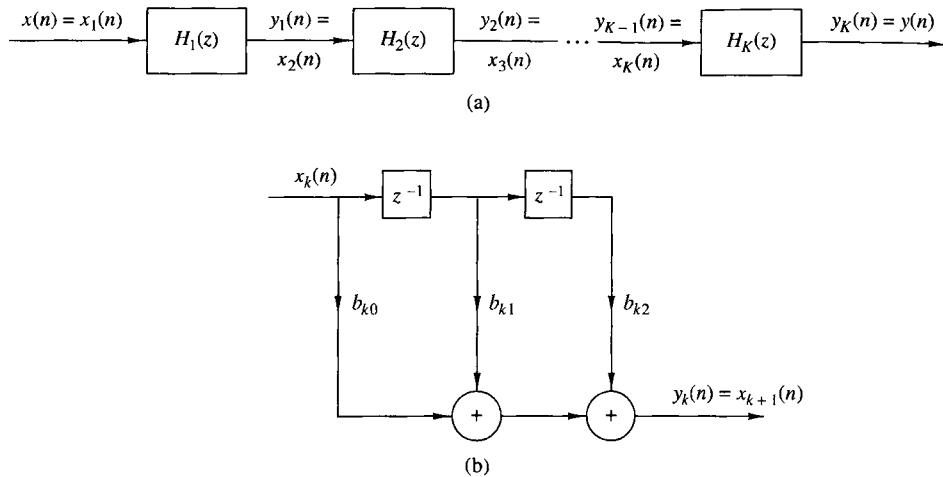


Figure 9.2.3 Cascade realization of an FIR system.

order sections of the FIR system as follows:

$$\begin{aligned} H_k(z) &= c_{k0}(1 - z_k z^{-1})(1 - z_k^* z^{-1})(1 - z^{-1}/z_k)(1 - z^{-1}/z_k^*) \\ &= c_{k0} + c_{k1}z^{-1} + c_{k2}z^{-2} + c_{k1}z^{-3} + c_{k0}z^{-4} \end{aligned} \quad (9.2.8)$$

where the coefficients $\{c_{k1}\}$ and $\{c_{k2}\}$ are functions of z_k . Thus, by combining the two pairs of poles to form a fourth-order filter section, we have reduced the number of multiplications from six to three (i.e., by a factor of 50%). Figure 9.2.4 illustrates the basic fourth-order FIR filter structure.

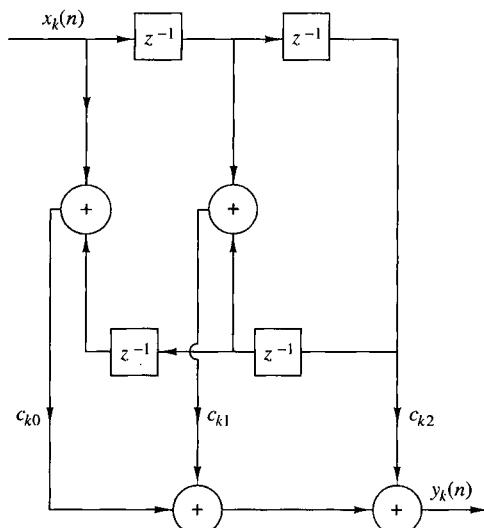


Figure 9.2.4
Fourth-order section in a
cascade realization of an
FIR system.

9.2.3 Frequency-Sampling Structures¹

The frequency-sampling realization is an alternative structure for an FIR filter in which the parameters that characterize the filter are the values of 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 of equally spaced frequencies, namely

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

and solve for the unit sample response $h(n)$ from these equally spaced frequency specifications. Thus we can write the frequency response as

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

and the values of $H(\omega)$ at frequencies $\omega_k = (2\pi/M)(k + \alpha)$ are simply

$$\begin{aligned}H(k + \alpha) &= H\left(\frac{2\pi}{M}(k + \alpha)\right) \\ &= \sum_{n=0}^{M-1} h(n)e^{-j2\pi(k+\alpha)n/M}, \quad k = 0, 1, \dots, M-1\end{aligned}\tag{9.2.9}$$

The set of values $\{H(k + \alpha)\}$ are called the frequency samples of $H(\omega)$. In the case where $\alpha = 0$, $\{H(k)\}$ corresponds to the M -point DFT of $\{h(n)\}$.

It is a simple matter to invert (9.2.9) and express $h(n)$ in terms of the frequency samples. The result is

$$h(n) = \frac{1}{M} \sum_{k=0}^{M-1} H(k + \alpha)e^{j2\pi(k+\alpha)n/M}, \quad n = 0, 1, \dots, M-1\tag{9.2.10}$$

When $\alpha = 0$, (9.2.10) is simply the IDFT of $\{H(k)\}$. Now if we use (9.2.10) to substitute for $h(n)$ in the z -transform $H(z)$, we have

$$\begin{aligned}H(z) &= \sum_{n=0}^{M-1} h(n)z^{-n} \\ &= \sum_{n=0}^{M-1} \left[\frac{1}{M} \sum_{k=0}^{M-1} H(k + \alpha)e^{j2\pi(k+\alpha)n/M} \right] z^{-n}\end{aligned}\tag{9.2.11}$$

¹ The reader may also refer to Section 10.2.3 for additional discussion of frequency-sampling FIR filters.

By interchanging the order of the two summations in (9.2.11) and performing the summation over the index n we obtain

$$\begin{aligned} H(z) &= \sum_{k=0}^{M-1} H(k + \alpha) \left[\frac{1}{M} \sum_{n=0}^{M-1} (e^{j2\pi(k+\alpha)/M} z^{-1})^n \right] \\ &= \frac{1 - z^{-M} e^{j2\pi\alpha}}{M} \sum_{k=0}^{M-1} \frac{H(k + \alpha)}{1 - e^{j2\pi(k+\alpha)/M} z^{-1}} \end{aligned} \quad (9.2.12)$$

Thus the system function $H(z)$ is characterized by the set of frequency samples $\{H(k + \alpha)\}$ instead of $\{h(n)\}$.

We view this FIR filter realization as a cascade of two filters [i.e., $H(z) = H_1(z)H_2(z)$]. One is an all-zero filter, or a comb filter, with system function

$$H_1(z) = \frac{1}{M} (1 - z^{-M} e^{j2\pi\alpha}) \quad (9.2.13)$$

Its zeros are located at equally spaced points on the unit circle at

$$z_k = e^{j2\pi(k+\alpha)/M}, \quad k = 0, 1, \dots, M-1$$

The second filter with system function

$$H_2(z) = \sum_{k=0}^{M-1} \frac{H(k + \alpha)}{1 - e^{j2\pi(k+\alpha)/M} z^{-1}} \quad (9.2.14)$$

consists of a parallel bank of single-pole filters with resonant frequencies

$$p_k = e^{j2\pi(k+\alpha)/M}, \quad k = 0, 1, \dots, M-1$$

Note that the pole locations are identical to the zero locations and that both occur at $\omega_k = 2\pi(k + \alpha)/M$, which are the frequencies at which the desired frequency response is specified. The gains of the parallel bank of resonant filters are simply the complex-valued parameters $\{H(k + \alpha)\}$. This cascade realization is illustrated in Fig. 9.2.5.

When the desired frequency response characteristic of the FIR filter is narrowband, most of the gain parameters $\{H(k + \alpha)\}$ are zero. Consequently, the corresponding resonant filters can be eliminated and only the filters with nonzero gains need be retained. The net result is a filter that requires fewer computations (mul-

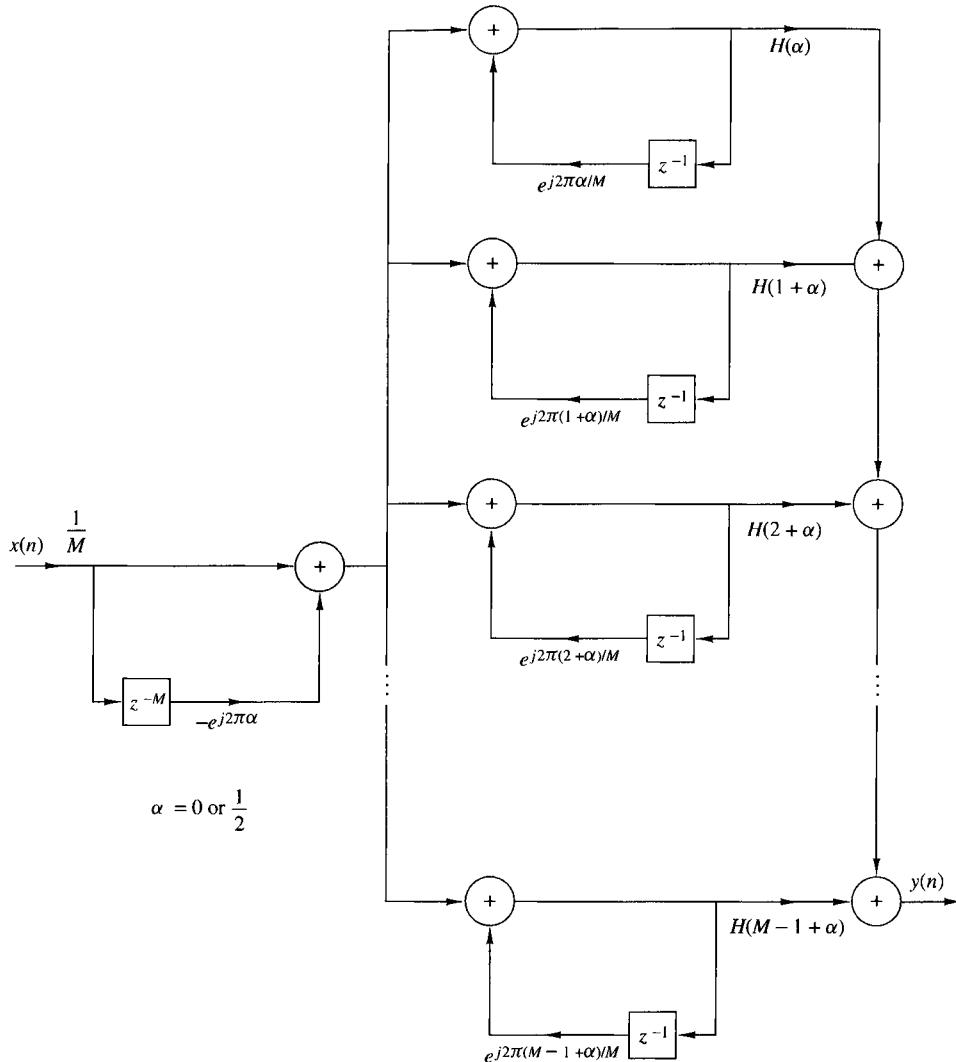


Figure 9.2.5 Frequency-sampling realization of FIR filter.

tiplications and additions) than the corresponding direct-form realization. Thus we obtain a more efficient realization.

The frequency-sampling filter structure can be simplified further by exploiting the symmetry in $H(k + \alpha)$, namely, $H(k) = H^*(M - k)$ for $\alpha = 0$ and

$$H\left(k + \frac{1}{2}\right) = H\left(M - k - \frac{1}{2}\right), \quad \text{for } \alpha = \frac{1}{2}$$

These relations are easily deduced from (9.2.9). As a result of this symmetry, a pair of single-pole filters can be combined to form a single two-pole filter with real-valued

parameters. Thus for $\alpha = 0$ the system function $H_2(z)$ reduces to

$$\begin{aligned} H_2(z) &= \frac{H(0)}{1-z^{-1}} + \sum_{k=1}^{(M-1)/2} \frac{A(k) + B(k)z^{-1}}{1 - 2\cos(2\pi k/M)z^{-1} + z^{-2}}, & M \text{ odd} \\ H_2(z) &= \frac{H(0)}{1-z^{-1}} + \frac{H(M/2)}{1+z^{-1}} + \sum_{k=1}^{(M/2)-1} \frac{A(k) + B(k)z^{-1}}{1 - 2\cos(2\pi k/M)z^{-1} + z^{-2}}, & M \text{ even} \end{aligned} \quad (9.2.15)$$

where, by definition,

$$\begin{aligned} A(k) &= H(k) + H(M-k) \\ B(k) &= H(k)e^{-j2\pi k/M} + H(M-k)e^{j2\pi k/M} \end{aligned} \quad (9.2.16)$$

Similar expressions can be obtained for $\alpha = \frac{1}{2}$.

EXAMPLE 9.2.1

Sketch the block diagram for the direct-form realization and the frequency-sampling realization of the $M = 32$, $\alpha = 0$, linear-phase (symmetric) FIR filter which has frequency samples

$$H\left(\frac{2\pi k}{32}\right) = \begin{cases} 1, & k = 0, 1, 2 \\ \frac{1}{2}, & k = 3 \\ 0, & k = 4, 5, \dots, 15 \end{cases}$$

Compare the computational complexity of these two structures.

Solution. Since the filter is symmetric, we exploit this symmetry and thus reduce the number of multiplications per output point by a factor of 2, from 32 to 16 in the direct-form realization. The number of additions per output point is 31. The block diagram of the direct realization is illustrated in Fig. 9.2.6.

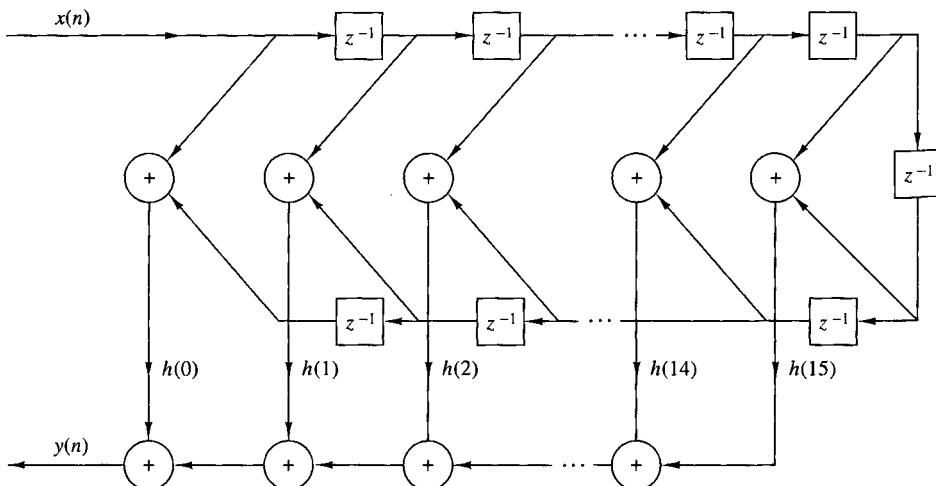


Figure 9.2.6 Direct-form realization of $M = 32$ FIR filter.

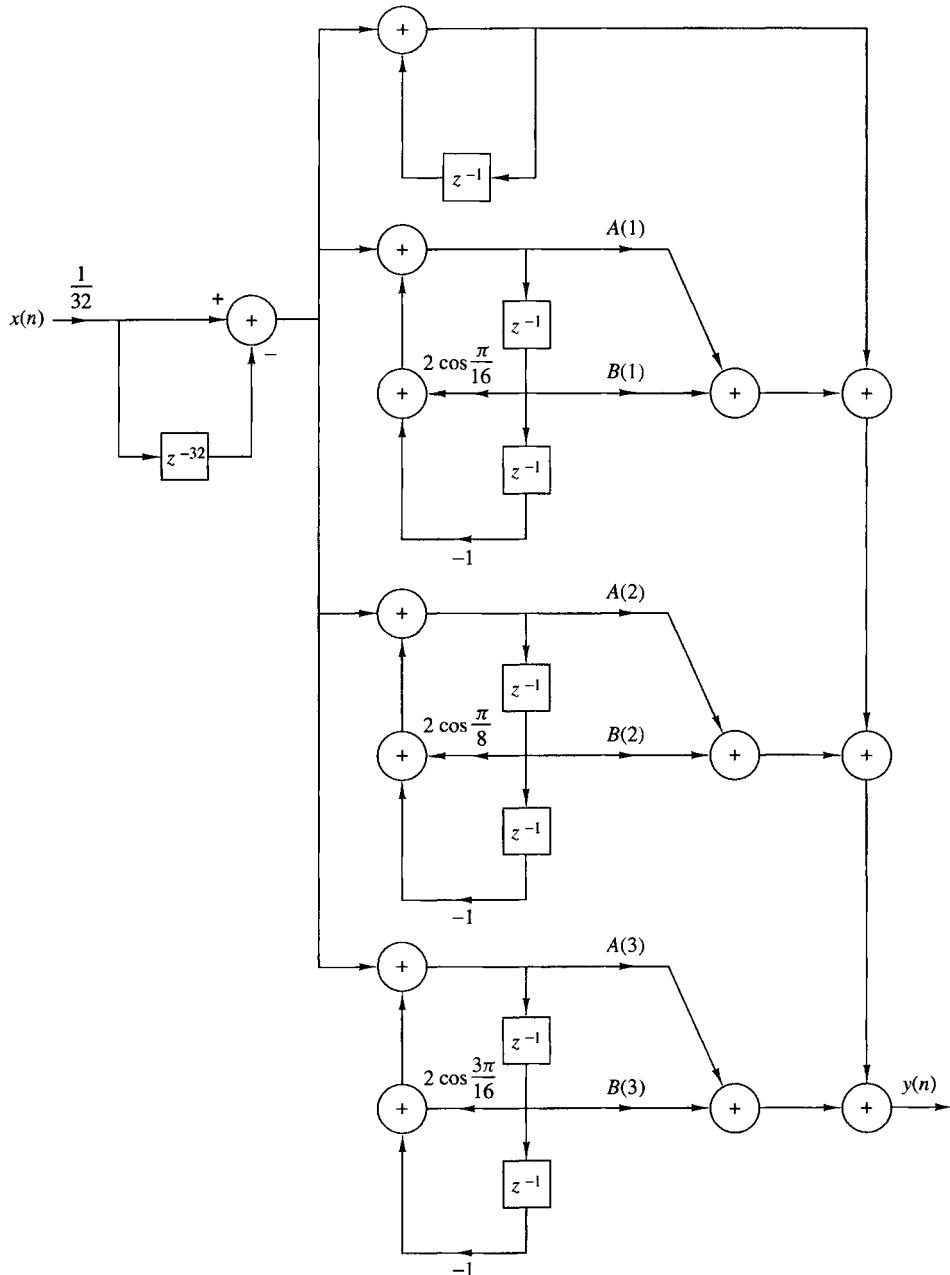


Figure 9.2.7 Frequency-sampling realization for the FIR filter in Example 9.2.1.

We use the form in (9.2.13) and (9.2.15) for the frequency-sampling realization and drop all terms that have zero-gain coefficients $\{H(k)\}$. The nonzero coefficients are $H(k)$ and the corresponding pairs are $H(M - k)$, for $k = 0, 1, 2, 3$. The block diagram of the resulting realization is shown in Fig. 9.2.7. Since $H(0) = 1$, the single-pole filter requires no multipli-

cation. The three double-pole filter sections require three multiplications each for a total of nine multiplications. The total number of additions is 13. Therefore, the frequency-sampling realization of this FIR filter is computationally more efficient than the direct-form realization.

9.2.4 Lattice Structure

In this section we introduce another FIR 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

$$H_m(z) = A_m(z), \quad m = 0, 1, 2, \dots, M - 1 \quad (9.2.17)$$

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

$$A_m(z) = 1 + \sum_{k=1}^m \alpha_m(k)z^{-k}, \quad m \geq 1 \quad (9.2.18)$$

and $A_0(z) = 1$. The unit sample response of the m th 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

$$y(n) = x(n) + \sum_{k=1}^m \alpha_m(k)x(n-k) \quad (9.2.19)$$

Two direct-form structures of the FIR filter are illustrated in Fig. 9.2.8.

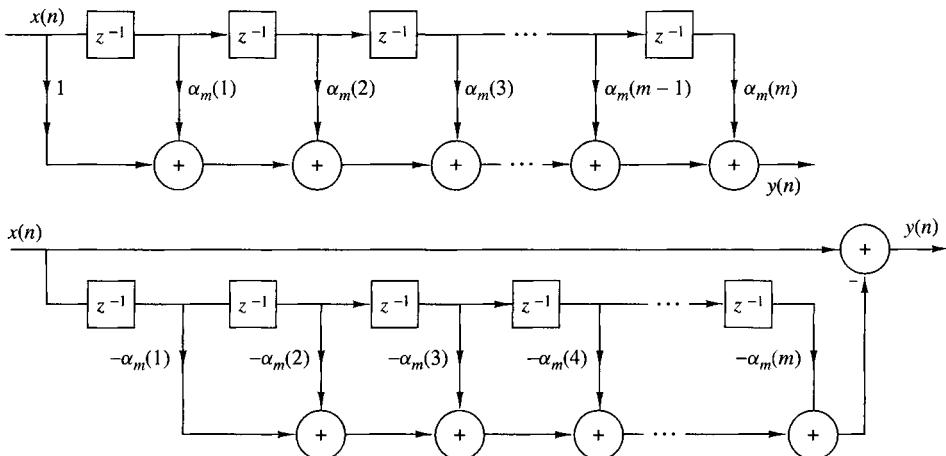


Figure 9.2.8 Direct-form realization of the FIR prediction filter.

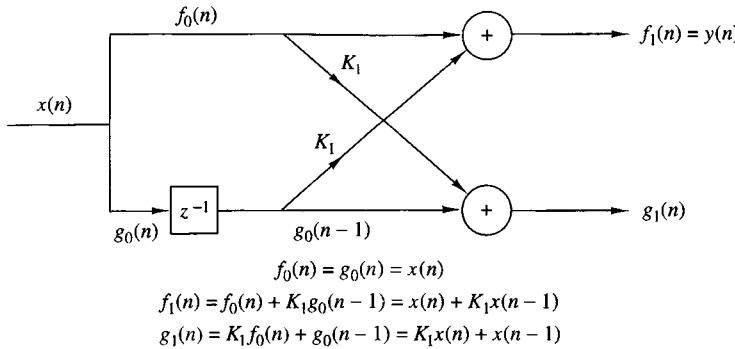


Figure 9.2.9 Single-stage lattice filter.

In Chapter 12, we show that the FIR structures shown in Fig. 9.2.8 are intimately related with the topic of linear prediction, where

$$\hat{x}(n) = - \sum_{k=1}^m \alpha_m(k) x(n-k) \quad (9.2.20)$$

is the one-step forward predicted value of $x(n)$, based on m past inputs, and $y(n) = x(n) - \hat{x}(n)$, given by (9.2.19), represents the prediction error sequence. In this context, the top filter structure in Fig. 9.2.8 is called a *prediction error filter*.

Now suppose that we have a filter of order $m = 1$. The output of such a filter is

$$y(n) = x(n) + \alpha_1(1)x(n-1) \quad (9.2.21)$$

This output can also be obtained from a first-order or single-stage lattice filter, illustrated in Fig. 9.2.9, by exciting both of the inputs by $x(n)$ and selecting the output from the top branch. Thus the output is exactly (9.2.21), if we select $K_1 = \alpha_1(1)$. The parameter K_1 in the lattice is called a reflection coefficient.

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

$$y(n) = x(n) + \alpha_2(1)x(n-1) + \alpha_2(2)x(n-2) \quad (9.2.22)$$

By cascading two lattice stages as shown in Fig. 9.2.10, it is possible to obtain the same output as (9.2.22). Indeed, the output from the first stage is

$$\begin{aligned}
 f_1(n) &= x(n) + K_1 x(n-1) \\
 g_1(n) &= K_1 x(n) + x(n-1)
 \end{aligned} \quad (9.2.23)$$

The output from the second stage is

$$\begin{aligned}
 f_2(n) &= f_1(n) + K_2 g_1(n-1) \\
 g_2(n) &= K_2 f_1(n) + g_1(n-1)
 \end{aligned} \quad (9.2.24)$$

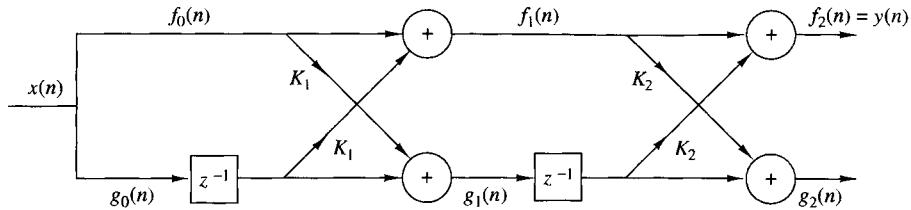


Figure 9.2.10 Two-stage lattice filter.

If we focus our attention on $f_2(n)$ and substitute for $f_1(n)$ and $g_1(n-1)$ from (9.2.23) into (9.2.24), we obtain

$$\begin{aligned} f_2(n) &= x(n) + K_1x(n-1) + K_2[K_1x(n-1) + x(n-2)] \\ &= x(n) + K_1(1 + K_2)x(n-1) + K_2x(n-2) \end{aligned} \quad (9.2.25)$$

Now (9.2.25) is identical to the output of the direct-form FIR filter as given by (9.2.22), if we equate the coefficients, that is,

$$\alpha_2(2) = K_2, \quad \alpha_2(1) = K_1(1 + K_2) \quad (9.2.26)$$

or, equivalently,

$$K_2 = \alpha_2(2), \quad K_1 = \frac{\alpha_2(1)}{1 + \alpha_2(2)} \quad (9.2.27)$$

Thus the reflection coefficients K_1 and K_2 of the lattice filter can be obtained from the coefficients $\{\alpha_m(k)\}$ of the direct-form realization.

By continuing this process, one can easily demonstrate, by induction, the equivalence between an m th-order direct-form FIR filter and an m -order or m -stage lattice filter. The lattice filter is generally described by the following set of order-recursive equations:

$$f_0(n) = g_0(n) = x(n) \quad (9.2.28)$$

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1), \quad m = 1, 2, \dots, M-1 \quad (9.2.29)$$

$$g_m(n) = K_m f_{m-1}(n) + g_{m-1}(n-1), \quad m = 1, 2, \dots, M-1 \quad (9.2.30)$$

Then the output of the $(M-1)$ -stage filter corresponds to the output of an $(M-1)$ -order FIR filter, that is,

$$y(n) = f_{M-1}(n)$$

Figure 9.2.11 illustrates an $(M-1)$ -stage lattice filter in block diagram form along with a typical stage that shows the computations specified by (9.2.29) and (9.2.30).

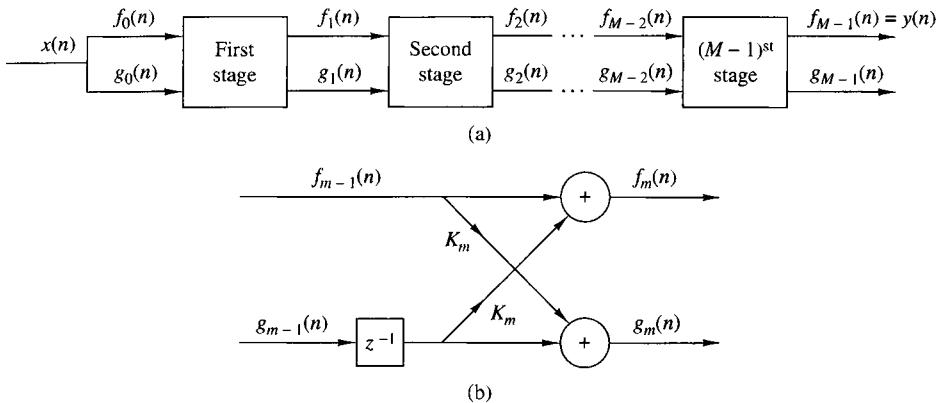


Figure 9.2.11 $(M - 1)$ -stage lattice filter.

As a consequence of the equivalence between an FIR filter and a lattice filter, the output $f_m(n)$ of an m -stage lattice filter can be expressed as

$$f_m(n) = \sum_{k=0}^m \alpha_m(k)x(n-k), \quad \alpha_m(0) = 1 \quad (9.2.31)$$

Since (9.2.31) is a convolution sum, it follows that the z -transform relationship is

$$F_m(z) = A_m(z)X(z)$$

or, equivalently,

$$A_m(z) = \frac{F_m(z)}{X(z)} = \frac{F_m(z)}{F_0(z)} \quad (9.2.32)$$

The other output component from the lattice, namely, $g_m(n)$, can also be expressed in the form of a convolution sum as in (9.2.31), by using another set of coefficients, say $\{\beta_m(k)\}$. That this in fact is the case becomes apparent from observation of (9.2.23) and (9.2.24). From (9.2.23) we note that the filter coefficients for the lattice filter that produces $f_1(n)$ are $\{1, K_1\} = \{1, \alpha_1(1)\}$ while the coefficients for the filter with output $g_1(n)$ are $\{K_1, 1\} = \{\alpha_1(1), 1\}$. We note that these two sets of coefficients are in reverse order. If we consider the two-stage lattice filter, with the output given by (9.2.24), we find that $g_2(n)$ can be expressed in the form

$$\begin{aligned} g_2(n) &= K_2 f_1(n) + g_1(n-1) \\ &= K_2[x(n) + K_1 x(n-1)] + K_1 x(n-1) + x(n-2) \\ &= K_2 x(n) + K_1(1 + K_2)x(n-1) + x(n-2) \\ &= \alpha_2(2)x(n) + \alpha_2(1)x(n-1) + x(n-2) \end{aligned}$$

Consequently, the filter coefficients are $\{\alpha_2(2), \alpha_2(1), 1\}$, whereas the coefficients for the filter that produces the output $f_2(n)$ are $\{1, \alpha_2(1), \alpha_2(2)\}$. Here, again, the two sets of filter coefficients are in reverse order.

From this development it follows that the output $g_m(n)$ from an m -stage lattice filter can be expressed by the convolution sum of the form

$$g_m(n) = \sum_{k=0}^m \beta_m(k)x(n-k) \quad (9.2.33)$$

where the filter coefficients $\{\beta_m(k)\}$ are associated with a filter that produces $f_m(n) = y(n)$ but operates in reverse order. Consequently,

$$\beta_m(k) = \alpha_m(m-k), \quad k = 0, 1, \dots, m \quad (9.2.34)$$

with $\beta_m(m) = 1$.

In the context of linear prediction, suppose that the data $x(n)$, $x(n-1)$, \dots , $x(n-m+1)$ is used to linearly predict the signal value $x(n-m)$ by use of a linear filter with coefficients $\{-\beta_m(k)\}$. Thus the predicted value is

$$\hat{x}(n-m) = - \sum_{k=0}^{m-1} \beta_m(k)x(n-k) \quad (9.2.35)$$

Since the data are run in reverse order through the predictor, the prediction performed in (9.2.35) is called *backward prediction*. In contrast, the FIR filter with system function $A_m(z)$ is called a *forward predictor*.

In the z -transform domain, (9.2.33) becomes

$$G_m(z) = B_m(z)X(z) \quad (9.2.36)$$

or, equivalently,

$$B_m(z) = \frac{G_m(z)}{X(z)} \quad (9.2.37)$$

where $B_m(z)$ represents the system function of the FIR filter with coefficients $\{\beta_m(k)\}$, that is,

$$B_m(z) = \sum_{k=0}^m \beta_m(k)z^{-k} \quad (9.2.38)$$

Since $\beta_m(k) = \alpha_m(m-k)$, (9.2.38) may be expressed as

$$\begin{aligned} B_m(z) &= \sum_{k=0}^m \alpha_m(m-k)z^{-k} \\ &= \sum_{l=0}^m \alpha_m(l)z^{l-m} \\ &= z^{-m} \sum_{l=0}^m \alpha_m(l)z^l \\ &= z^{-m} A_m(z^{-1}) \end{aligned} \quad (9.2.39)$$

The relationship in (9.2.39) implies that the zeros of the FIR filter with system function $B_m(z)$ are simply the reciprocals of the zeros of $A_m(z)$. Hence $B_m(z)$ is called the reciprocal or *reverse polynomial* of $A_m(z)$.

Now that we have established these interesting relationships between the direct-form FIR filter and the lattice structure, let us return to the recursive lattice equations in (9.2.28) through (9.2.30) and transfer them to the z -domain. Thus we have

$$F_0(z) = G_0(z) = X(z) \quad (9.2.40)$$

$$F_m(z) = F_{m-1}(z) + K_m z^{-1} G_{m-1}(z), \quad m = 1, 2, \dots, M-1 \quad (9.2.41)$$

$$G_m(z) = K_m F_{m-1}(z) + z^{-1} G_{m-1}(z), \quad m = 1, 2, \dots, M-1 \quad (9.2.42)$$

If we divide each equation by $X(z)$, we obtain the desired results in the form

$$A_0(z) = B_0(z) = 1 \quad (9.2.43)$$

$$A_m(z) = A_{m-1}(z) + K_m z^{-1} B_{m-1}(z), \quad m = 1, 2, \dots, M-1 \quad (9.2.44)$$

$$B_m(z) = K_m A_{m-1}(z) + z^{-1} B_{m-1}(z), \quad m = 1, 2, \dots, M-1 \quad (9.2.45)$$

Thus a lattice stage is described in the z -domain by the matrix equation

$$\begin{bmatrix} A_m(z) \\ B_m(z) \end{bmatrix} = \begin{bmatrix} 1 & K_m \\ K_m & 1 \end{bmatrix} \begin{bmatrix} A_{m-1}(z) \\ z^{-1} B_{m-1}(z) \end{bmatrix} \quad (9.2.46)$$

Before concluding this discussion, it is desirable to develop the relationships for converting the lattice parameters $\{K_i\}$, that is, the reflection coefficients, to the direct-form filter coefficients $\{\alpha_m(k)\}$, and vice versa.

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:

$$A_0(z) = B_0(z) = 1 \quad (9.2.47)$$

$$A_m(z) = A_{m-1}(z) + K_m z^{-1} B_{m-1}(z), \quad m = 1, 2, \dots, M-1 \quad (9.2.48)$$

$$B_m(z) = z^{-m} A_m(z^{-1}), \quad m = 1, 2, \dots, M-1 \quad (9.2.49)$$

The solution is obtained recursively, beginning with $m = 1$. Thus we obtain a sequence of $(M - 1)$ FIR filters, one for each value of m . The procedure is best illustrated by means of an example.

EXAMPLE 9.2.2

Given a three-stage lattice filter with coefficients $K_1 = \frac{1}{4}$, $K_2 = \frac{1}{4}$, $K_3 = \frac{1}{3}$, determine the FIR filter coefficients for the direct-form structure.

Solution. We solve the problem recursively, beginning with (9.2.48) for $m = 1$. Thus we have

$$\begin{aligned} A_1(z) &= A_0(z) + K_1 z^{-1} B_0(z) \\ &= 1 + K_1 z^{-1} = 1 + \frac{1}{4} z^{-1} \end{aligned}$$

Hence the coefficients of an FIR filter corresponding to the single-stage lattice are $\alpha_1(0) = 1$, $\alpha_1(1) = K_1 = \frac{1}{4}$. Since $B_m(z)$ is the reverse polynomial of $A_m(z)$, we have

$$B_1(z) = \frac{1}{4} + z^{-1}$$

Next we add the second stage to the lattice. For $m = 2$, (9.2.48) yields

$$\begin{aligned} A_2(z) &= A_1(z) + K_2 z^{-1} B_1(z) \\ &= 1 + \frac{3}{8} z^{-1} + \frac{1}{2} z^{-2} \end{aligned}$$

Hence the FIR filter parameters corresponding to the two-stage lattice are $\alpha_2(0) = 1$, $\alpha_2(1) = \frac{3}{8}$, $\alpha_2(2) = \frac{1}{2}$. Also,

$$B_2(z) = \frac{1}{2} + \frac{3}{8} z^{-1} + z^{-2}$$

Finally, the addition of the third stage to the lattice results in the polynomial

$$\begin{aligned} A_3(z) &= A_2(z) + K_3 z^{-1} B_2(z) \\ &= 1 + \frac{13}{24} z^{-1} + \frac{5}{8} z^{-2} + \frac{1}{3} z^{-3} \end{aligned}$$

Consequently, the desired direct-form FIR filter is characterized by the coefficients

$$\alpha_3(0) = 1, \quad \alpha_3(1) = \frac{13}{24}, \quad \alpha_3(2) = \frac{5}{8}, \quad \alpha_3(3) = \frac{1}{3}$$

As this example illustrates, the lattice structure with parameters K_1, K_2, \dots, K_m , corresponds to a class of m direct-form FIR filters with system functions $A_1(z), A_2(z), \dots, A_m(z)$. It is interesting to note that a characterization of this class of m FIR filters in direct form requires $m(m + 1)/2$ filter coefficients. In contrast, the lattice-form characterization requires only the m reflection coefficients $\{K_i\}$. The reason that the lattice provides a more compact representation for the class of m FIR filters is simply that the addition of stages to the lattice does not alter the parameters of the previous stages. On the other hand, the addition of the m th stage to a lattice with $(m - 1)$ stages results in an FIR filter with system function $A_m(z)$ that has coefficients totally different from the coefficients of the lower-order FIR filter with system function $A_{m-1}(z)$.

A formula for determining the filter coefficients $\{\alpha_m(k)\}$ recursively can be easily derived from polynomial relationships in (9.2.47) through (9.2.49). From the relationship in (9.2.48) we have

$$A_m(z) = A_{m-1}(z) + K_m z^{-1} B_{m-1}(z) \quad (9.2.50)$$

$$\sum_{k=0}^m \alpha_m(k) z^{-k} = \sum_{k=0}^{m-1} \alpha_{m-1}(k) z^{-k} + K_m \sum_{k=0}^{m-1} \alpha_{m-1}(m-1-k) z^{-(k+1)}$$

By equating the coefficients of equal powers of z^{-1} and recalling that $\alpha_m(0) = 1$ for $m = 1, 2, \dots, M - 1$, we obtain the desired recursive equation for the FIR filter coefficients in the form

$$\alpha_m(0) = 1 \quad (9.2.51)$$

$$\alpha_m(m) = K_m \quad (9.2.52)$$

$$\begin{aligned} \alpha_m(k) &= \alpha_{m-1}(k) + K_m \alpha_{m-1}(m-k) \\ &= \alpha_{m-1}(k) + \alpha_m(m) \alpha_{m-1}(m-k), \quad \begin{matrix} 1 \leq k \leq m-1 \\ m = 1, 2, \dots, M-1 \end{matrix} \end{aligned} \quad (9.2.53)$$

We note that (9.2.51) through (9.2.53) are simply the Levinson–Durbin recursive equations given in Chapter 12.

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 $m = M-1, M-2, \dots, 1$. Consequently, we need to compute the polynomials $A_m(z)$ starting from $m = M-1$ and “stepping down” successively to $m = 1$.

The desired recursive relation for the polynomials is easily determined from (9.2.44) and (9.2.45). We have

$$\begin{aligned} A_m(z) &= A_{m-1}(z) + K_m z^{-1} B_{m-1}(z) \\ &= A_{m-1}(z) + K_m [B_m(z) - K_m A_{m-1}(z)] \end{aligned}$$

If we solve for $A_{m-1}(z)$, we obtain

$$A_{m-1}(z) = \frac{A_m(z) - K_m B_m(z)}{1 - K_m^2}, \quad m = M-1, M-2, \dots, 1 \quad (9.2.54)$$

Thus we compute all lower-degree polynomials $A_m(z)$ beginning with $A_{M-1}(z)$ and obtain the desired lattice coefficients from the relation $K_m = \alpha_m(m)$. We observe that the procedure works as long as $|K_m| \neq 1$ for $m = 1, 2, \dots, M-1$.

EXAMPLE 9.2.3

Determine the lattice coefficients corresponding to the FIR filter with system function

$$H(z) = A_3(z) = 1 + \frac{13}{24}z^{-1} + \frac{5}{8}z^{-2} + \frac{1}{3}z^{-3}$$

Solution. First we note that $K_3 = \alpha_3(3) = \frac{1}{3}$. Furthermore,

$$B_3(z) = \frac{1}{3} + \frac{5}{8}z^{-1} + \frac{13}{24}z^{-2} + z^{-3}$$

The step-down relationship in (9.2.54) with $m = 3$ yields

$$\begin{aligned} A_2(z) &= \frac{A_3(z) - K_3 B_3(z)}{1 - K_3^2} \\ &= 1 + \frac{3}{8}z^{-1} + \frac{1}{2}z^{-2} \end{aligned}$$

Hence $K_2 = \alpha_2(2) = \frac{1}{2}$ and $B_2(z) = \frac{1}{2} + \frac{3}{8}z^{-1} + z^{-2}$. By repeating the step-down recursion in (9.2.51), we obtain

$$\begin{aligned} A_1(z) &= \frac{A_2(z) - K_2 B_2(z)}{1 - K_2^2} \\ &= 1 + \frac{1}{4}z^{-1} \end{aligned}$$

Hence $K_1 = \alpha_1(1) = \frac{1}{4}$.

From the step-down recursive equation in (9.2.54), it is relatively easy to obtain a formula for recursively computing K_m , beginning with $m = M - 1$ and stepping down to $m = 1$. For $m = M - 1, M - 2, \dots, 1$ we have

$$K_m = \alpha_m(m), \quad \alpha_{m-1}(0) = 1 \quad (9.2.55)$$

$$\begin{aligned} \alpha_{m-1}(k) &= \frac{\alpha_m(k) - K_m \beta_m(k)}{1 - K_m^2} \\ &= \frac{\alpha_m(k) - \alpha_m(m)\alpha_m(m-k)}{1 - \alpha_m^2(m)}, \quad 1 \leq k \leq m-1 \quad (9.2.56) \end{aligned}$$

As indicated above, the recursive equation in (9.2.56) breaks down if any lattice parameters $|K_m| = 1$. If this occurs, it is indicative of the fact that the polynomial $A_{m-1}(z)$ has a root on the unit circle. Such a root can be factored out from $A_{m-1}(z)$ and the iterative process in (9.2.56) is carried out for the reduced-order system.

9.3 Structures for IIR Systems

In this section we consider different IIR systems structures described by the difference equation in (9.1.1) or, equivalently, by the system function in (9.1.2). Just as in the case of FIR systems, there are several types of structures or realizations, including direct-form structures, cascade-form structures, lattice structures, and lattice-ladder structures. In addition, IIR systems lend themselves to a parallel-form realization. We begin by describing two direct-form realizations.

9.3.1 Direct-Form Structures

The rational system function as given by (9.1.2) that characterizes an IIR system can be viewed as two systems in cascade, that is,

$$H(z) = H_1(z)H_2(z) \quad (9.3.1)$$

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

$$H_1(z) = \sum_{k=0}^M b_k z^{-k} \quad (9.3.2)$$

and

$$H_2(z) = \frac{1}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.3.3)$$

In Section 2.5 we described two different direct-form realizations, characterized by whether $H_1(z)$ precedes $H_2(z)$, or vice versa. Since $H_1(z)$ is an FIR system, its direct-form realization was illustrated in Fig. 9.2.1. By attaching the all-pole system in cascade with $H_1(z)$, we obtain the direct form I realization depicted in Fig. 9.3.1. This realization requires $M + N + 1$ multiplications, $M + N$ additions, and $M + N + 1$ memory locations.

If the all-pole filter $H_2(z)$ is placed before the all-zero filter $H_1(z)$, a more compact structure is obtained as illustrated in Section 2.5. Recall that the difference

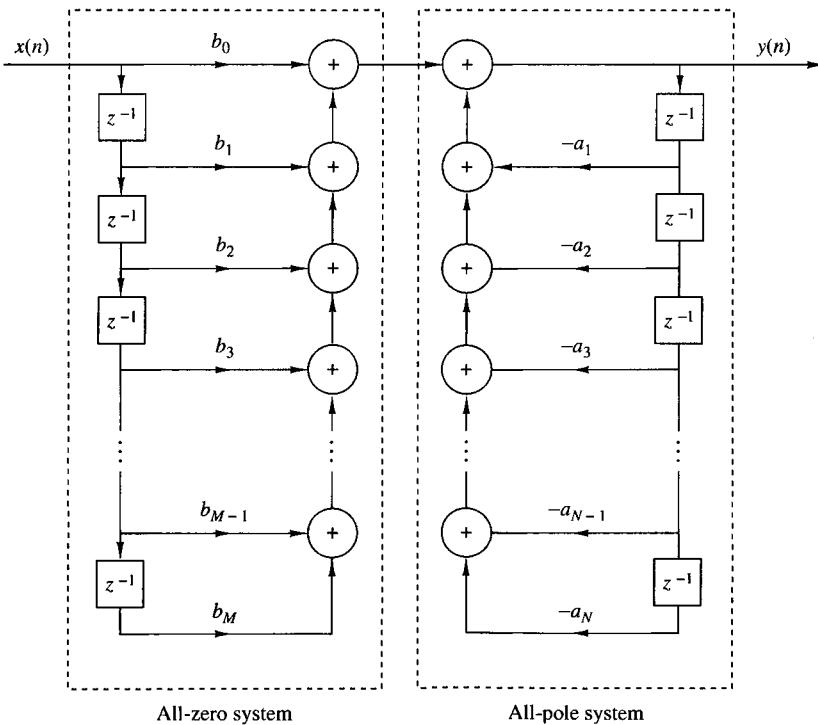


Figure 9.3.1 Direct form I realization.

equation for the all-pole filter is

$$w(n) = - \sum_{k=1}^N a_k w(n-k) + x(n) \quad (9.3.4)$$

Since $w(n)$ is the input to the all-zero system, its output is

$$y(n) = \sum_{k=0}^M b_k w(n-k) \quad (9.3.5)$$

We note that both (9.3.4) and (9.3.5) involve delayed versions of the sequence $\{w(n)\}$. Consequently, only a single delay line or a single set of memory locations is required for storing the past values of $\{w(n)\}$. The resulting structure that implements (9.3.4) and (9.3.5) is called a direct form II realization and is depicted in Fig. 9.3.2. This structure requires $M + N + 1$ multiplications, $M + N$ additions, and the maximum of memory locations. Since the direct form II realization minimizes the number of memory locations, it is said to be *canonic*. However, we should indicate that other IIR structures also possess this property, so that this terminology is perhaps unjustified.

The structures in Figs. 9.3.1 and 9.3.2 are both called “direct-form” realizations because they are obtained directly from the system function $H(z)$ without any rearrangement of $H(z)$. Unfortunately, both structures are extremely sensitive to parameter quantization, in general, and are not recommended in practical applications. This topic is discussed in detail in Section 9.6, where we demonstrate that when N is large, a small change in a filter coefficient due to parameter quantization results in a large change in the location of the poles and zeros of the system.

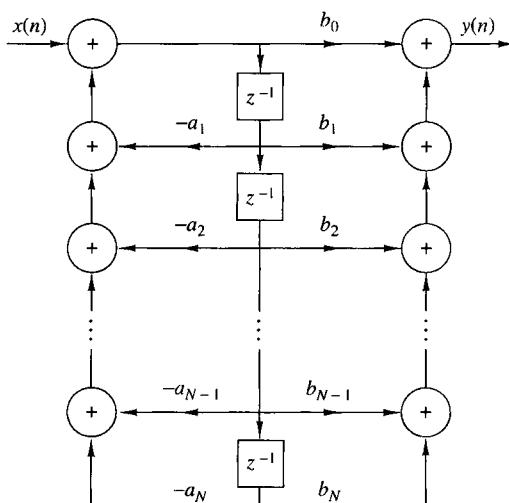


Figure 9.3.2
Direct form II realization
($N = M$).

9.3.2 Signal Flow Graphs and Transposed Structures

A signal flow graph provides an alternative, but equivalent, graphical representation to a block diagram structure that we have been using to illustrate various system realizations. The basic elements of a flow graph are branches and nodes. A signal flow graph is basically a set of 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 a node of a flow graph is equal to the sum of the signals from all branches connecting to the node.

To illustrate these basic notions, let us consider the two-pole and two-zero IIR system depicted in block diagram form in Fig. 9.3.3(a). The system block diagram can be converted to the signal flow graph shown in Fig. 9.3.3(b). We note that the flow graph contains five nodes labeled 1 through 5. Two of the nodes (1, 3) are summing nodes (i.e., they contain adders), while the other three nodes represent branching points. Branch transmittances are indicated for the branches in the flow graph. Note that a delay is indicated by the branch transmittance z^{-1} . When the branch transmittance is unity, it is left unlabeled. The input to the system originates at a *source node* and the output signal is extracted at a *sink node*.

We observe that the signal flow graph contains the same basic information as the block diagram realization of the system. The only apparent difference is that *both*

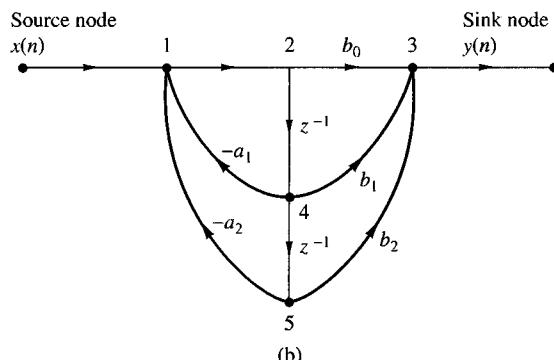
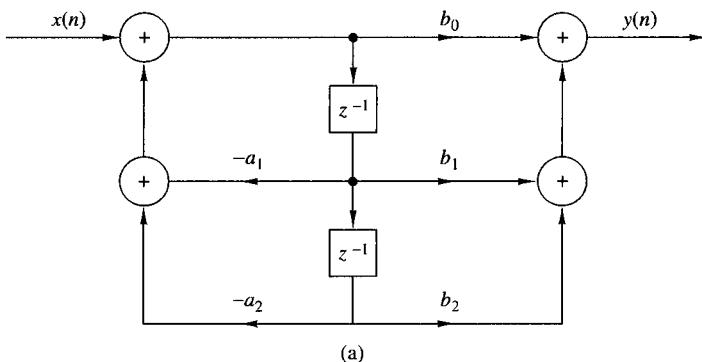


Figure 9.3.3 Second-order filter structure (a) and its signal flow graph (b).

branch points and adders in the block diagram are represented by nodes in the signal flow graph.

The subject of linear signal flow graphs is an important one in the treatment of networks, and many interesting results are available. One basic notion involves the transformation of one flow graph into another without changing the basic input-output relationship. Specifically, one technique that is useful in deriving new system structures for FIR and IIR systems stems from the *transposition* or *flow-graph reversal theorem*. This theorem simply states that if we reverse the directions of all branch transmittances and interchange the input and output in the flow graph, the system function remains unchanged. The resulting structure is called a *transposed structure* or a *transposed form*.

For example, the transposition of the signal flow graph in Fig. 9.3.3(b) is illustrated in Fig. 9.3.4(a). The corresponding block diagram realization of the transposed form is depicted in Fig. 9.3.4(b). It is interesting to note that the transposition of the original flow graph resulted in branching nodes becoming adder nodes, and vice versa.

Let us apply the transposition theorem to the direct form II structure. First, we reverse all the signal flow directions in Fig. 9.3.2. Second, we change nodes into adders and adders into nodes, and finally, we interchange the input and the output.

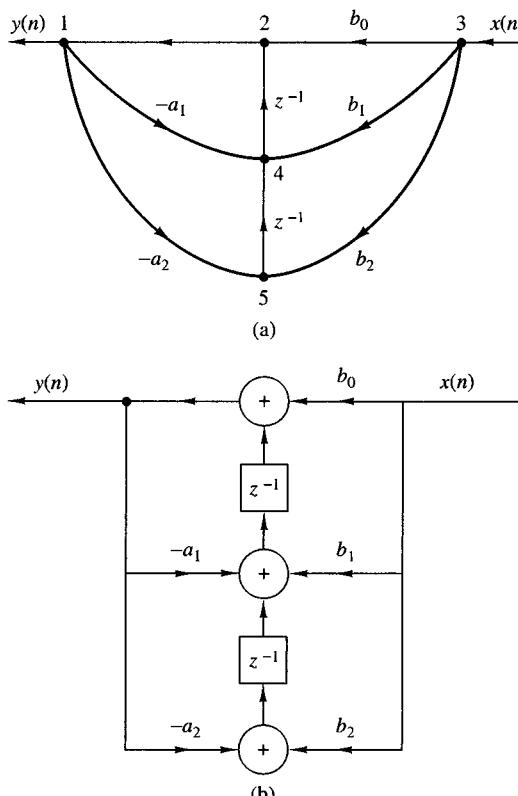


Figure 9.3.4
Signal flow graph of
transposed structure (a) and
its realization (b).

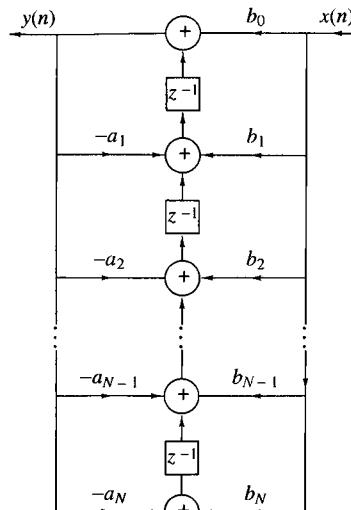


Figure 9.3.5
Transposed direct form II
structure.

These operations result in the transposed direct form II structure shown in Fig. 9.3.5. This structure can be redrawn as in Fig. 9.3.6, which shows the input on the left and the output on the right.

The transposed direct form II realization that we have obtained can be described by the set of difference equations

$$y(n) = w_1(n-1) + b_0 x(n) \quad (9.3.6)$$

$$w_k(n) = w_{k+1}(n-1) - a_k y(n) + b_k x(n), \quad k = 1, 2, \dots, N-1 \quad (9.3.7)$$

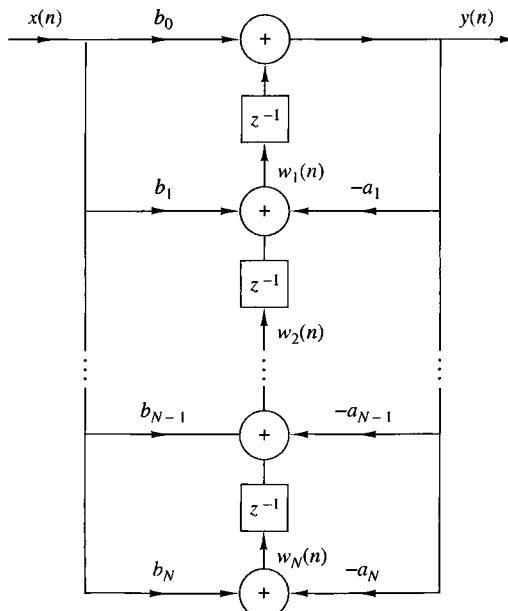


Figure 9.3.6
Transposed direct form II
structure.

$$w_N(n) = b_N x(n) - a_N y(n) \quad (9.3.8)$$

Without loss of generality, we have assumed that $M = N$ in writing equations. It is also clear from observation of Fig. 9.3.6 that this set of difference equations is equivalent to the single difference equation

$$y(n) = -\sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k) \quad (9.3.9)$$

Finally, we observe that the transposed direct form II structure requires the same number of multiplications, additions, and memory locations as the original direct-form II structure.

Although our discussion of transposed structures has been concerned with the general form of an IIR system, it is interesting to note that an FIR system, obtained from (9.3.9) by setting the $a_k = 0$, $k = 1, 2, \dots, N$, also has a transposed direct form as illustrated in Fig. 9.3.7. This structure is simply obtained from Fig. 9.3.6 by setting $a_k = 0$, $k = 1, 2, \dots, N$. This transposed-form realization may be described by the set of difference equations

$$w_M(n) = b_M x(n) \quad (9.3.10)$$

$$w_k(n) = w_{k+1}(n-1) + b_k x(n), \quad k = M-1, M-2, \dots, 1 \quad (9.3.11)$$

$$y(n) = w_1(n-1) + b_0 x(n) \quad (9.3.12)$$

In summary, Table 9.1 illustrates the direct-form structures and the corresponding difference equations for a basic two-pole and two-zero IIR system with system function

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \quad (9.3.13)$$

This is the basic building block in the cascade realization of high-order IIR systems, as described in the following section. Of the three direct-form structures given in Table 9.1, the direct form II structures are preferable due to the smaller number of memory locations required in their implementation.

Finally, we note that in the z -domain, the set of difference equations describing a linear signal flow graph constitute a linear set of equations. Any rearrangement of such a set of equations is equivalent to a rearrangement of the signal flow graph to obtain a new structure, and vice versa.

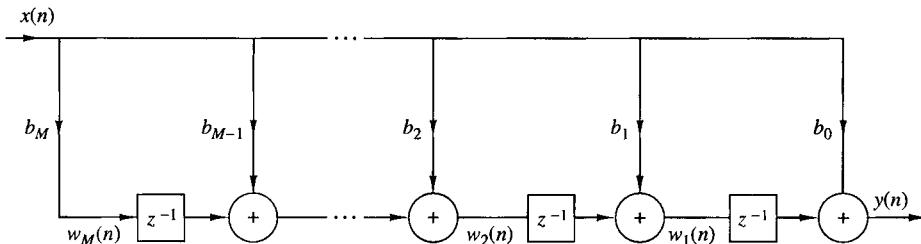


Figure 9.3.7 Transposed FIR structure.

TABLE 9.1 Some Second-Order Modules for Discrete-Time Systems

Structure	Implementation Equations	System Function
Direct Form I	$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) - a_1y(n-1) - a_2y(n-2)$	$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}$
Regular Direct Form II	$w(n) = -a_1w(n-1) - a_2w(n-2) + x(n)$ $y(n) = b_0w(n) + b_1w(n-1) + b_2w(n-2)$	$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}$
Transposed Direct Form II	$y(n) = b_0x(n) + w_1(n-1)$ $w_1(n) = b_1x(n) - a_1y(n) + w_2(n-1)$ $w_2(n) = b_2x(n) - a_2y(n)$	$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}$

9.3.3 Cascade-Form Structures

Let us consider a high-order IIR system with system function given by (9.1.2). Without loss of generality we assume that $N \geq M$. The system can be factored into a cascade of second-order subsystems, such that $H(z)$ can be expressed as

$$H(z) = \prod_{k=1}^K H_k(z) \quad (9.3.14)$$

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

$$H_k(z) = \frac{b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}} \quad (9.3.15)$$

As in the case of FIR systems based on a cascade-form realization, the parameter b_0 can be distributed equally among the K filter sections so that $b_0 = b_{10}b_{20}\dots b_{K0}$.

The coefficients $\{a_{ki}\}$ and $\{b_{ki}\}$ in the second-order subsystems are real. This implies that in forming the second-order subsystems or quadratic factors in (9.3.15),

we should group together a pair of complex-conjugate poles and we should group together a pair of complex-conjugate zeros. However, the pairing of two complex-conjugate poles with a pair of complex-conjugate zeros or real-valued zeros to form a subsystem of the type given by (9.3.15) can be done arbitrarily. Furthermore, any two real-valued zeros can be paired together to form a quadratic factor and, likewise, any two real-valued poles can be paired together to form a quadratic factor. Consequently, the quadratic factor in the numerator of (9.3.15) may consist of either a pair of real roots or a pair of complex-conjugate roots. The same statement applies to the denominator of (9.3.15).

If $N > M$, some of the second-order subsystems have numerator coefficients that are zero, that is, either $b_{k2} = 0$ or $b_{k1} = 0$ or both $b_{k2} = b_{k1} = 0$ for some k . Furthermore, if N is odd, one of the subsystems, say $H_k(z)$, must have $a_{k2} = 0$, so that the subsystem is of first order. To preserve the modularity in the implementation of $H(z)$, it is often preferable to use the basic second-order subsystems in the cascade structure and have some zero-valued coefficients in some of the subsystems.

Each of the second-order subsystems with system function of the form (9.3.15) can be realized in either direct form I, or direct form II, or transposed direct form II. Since there are many ways to pair the poles and zeros of $H(z)$ into a cascade of second-order sections, and several ways to order the resulting subsystems, it is possible to obtain a variety of cascade realizations. Although all cascade realizations are equivalent for infinite-precision arithmetic, the various realizations may differ significantly when implemented with finite-precision arithmetic.

The general form of the cascade structure is illustrated in Fig. 9.3.8. 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.

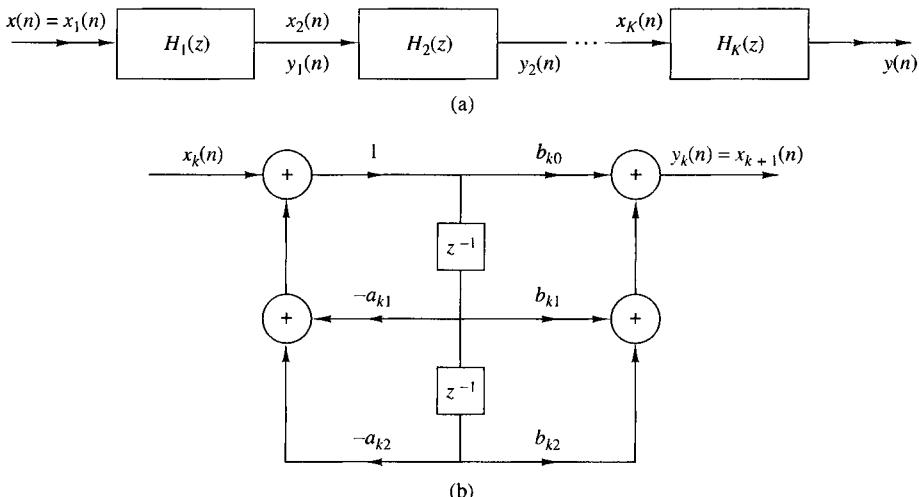


Figure 9.3.8 Cascade structure of second-order systems and a realization of each second-order section.

$$y_0(n) = x(n) \quad (9.3.16)$$

$$w_k(n) = -a_{k1}w_k(n-1) - a_{k2}w_k(n-2) + y_{k-1}(n), \quad k = 1, 2, \dots, K \quad (9.3.17)$$

$$y_k(n) = b_{k0}w_k(n) + b_{k1}w_k(n-1) + b_{k2}w_k(n-2), \quad k = 1, 2, \dots, K \quad (9.3.18)$$

$$y(n) = y_K(n) \quad (9.3.19)$$

Thus this set of equations provides a complete description of the cascade structure based on direct form II sections.

9.3.4 Parallel-Form Structures

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

$$H(z) = C + \sum_{k=1}^N \frac{A_k}{1 - p_k z^{-1}} \quad (9.3.20)$$

where $\{p_k\}$ are the poles, $\{A_k\}$ are the coefficients (residues) in the partial-fraction expansion, and the constant C is defined as $C = b_N/a_N$. The structure implied by (9.3.20) is shown in Fig. 9.3.9. It consists of a parallel bank of single-pole filters.

In general, some of the poles of $H(z)$ may be complex valued. In such a case, the corresponding coefficients A_k are also complex valued. To avoid multiplications by complex numbers, we can combine pairs of complex-conjugate poles to form two-pole subsystems. In addition, we can combine, in an arbitrary manner, pairs of real-valued poles to form two-pole subsystems. Each of these subsystems has the form

$$H_k(z) = \frac{b_{k0} + b_{k1}z^{-1}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}} \quad (9.3.21)$$

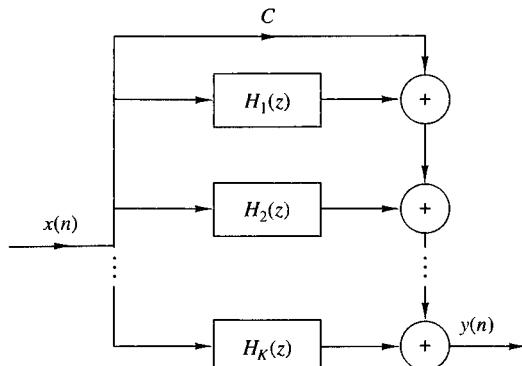


Figure 9.3.9
Parallel structure of IIR system.

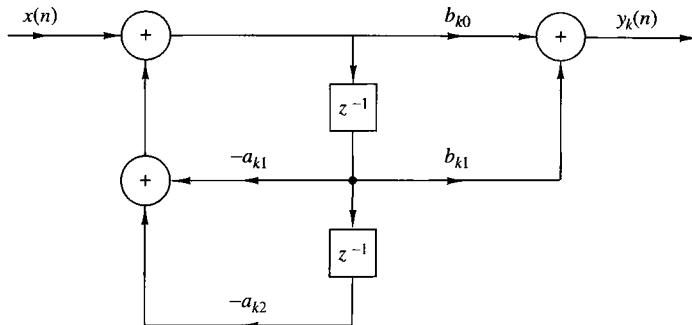


Figure 9.3.10 Structure of second-order section in a parallel IIR system realization.

where the coefficients $\{b_{ki}\}$ and $\{a_{ki}\}$ are real-valued system parameters. The overall function can now be expressed as

$$H(z) = C + \sum_{k=1}^K H_k(z) \quad (9.3.22)$$

where K is the integer part of $(N + 1)/2$. When N is odd, one of the $H_k(z)$ is really a single-pole system (i.e., $b_{k1} = a_{k2} = 0$).

The individual second-order sections which are the basic building blocks for $H(z)$ can be implemented in either of the direct forms or in a transposed direct form. The direct form II structure is illustrated in Fig. 9.3.10. With this structure as a basic building block, the parallel-form realization of the FIR system is described by the following set of equations:

$$w_k(n) = -a_{k1}w_k(n-1) - a_{k2}w_k(n-2) + x(n), \quad k = 1, 2, \dots, K \quad (9.3.23)$$

$$y_k(n) = b_{k0}w_k(n) + b_{k1}w_k(n-1), \quad k = 1, 2, \dots, K \quad (9.3.24)$$

$$y(n) = Cx(n) + \sum_{k=1}^K y_k(n) \quad (9.3.25)$$

EXAMPLE 9.3.1

Determine the cascade and parallel realizations for the system described by the system function

$$H(z) = \frac{10(1 - \frac{1}{2}z^{-1})(1 - \frac{2}{3}z^{-1})(1 + 2z^{-1})}{(1 - \frac{3}{4}z^{-1})(1 - \frac{1}{8}z^{-1})[1 - (\frac{1}{2} + j\frac{1}{2})z^{-1}][1 - (\frac{1}{2} - j\frac{1}{2})z^{-1}]}$$

Solution. The cascade realization is easily obtained from this form. One possible pairing of poles and zeros is

$$H_1(z) = \frac{1 - \frac{2}{3}z^{-1}}{1 - \frac{7}{8}z^{-1} + \frac{3}{32}z^{-2}}$$

$$H_2(z) = \frac{1 + \frac{3}{2}z^{-1} - z^{-2}}{1 - z^{-1} + \frac{1}{2}z^{-2}}$$

and hence

$$H(z) = 10H_1(z)H_2(z)$$

The cascade realization is depicted in Fig. 9.3.11(a).

To obtain the parallel-form realization, $H(z)$ must be expanded in partial fractions. Thus we have

$$H(z) = \frac{A_1}{1 - \frac{3}{4}z^{-1}} + \frac{A_2}{1 - \frac{1}{8}z^{-1}} + \frac{A_3}{1 - (\frac{1}{2} + j\frac{1}{2})z^{-1}} + \frac{A_3^*}{1 - (\frac{1}{2} - j\frac{1}{2})z^{-1}}$$

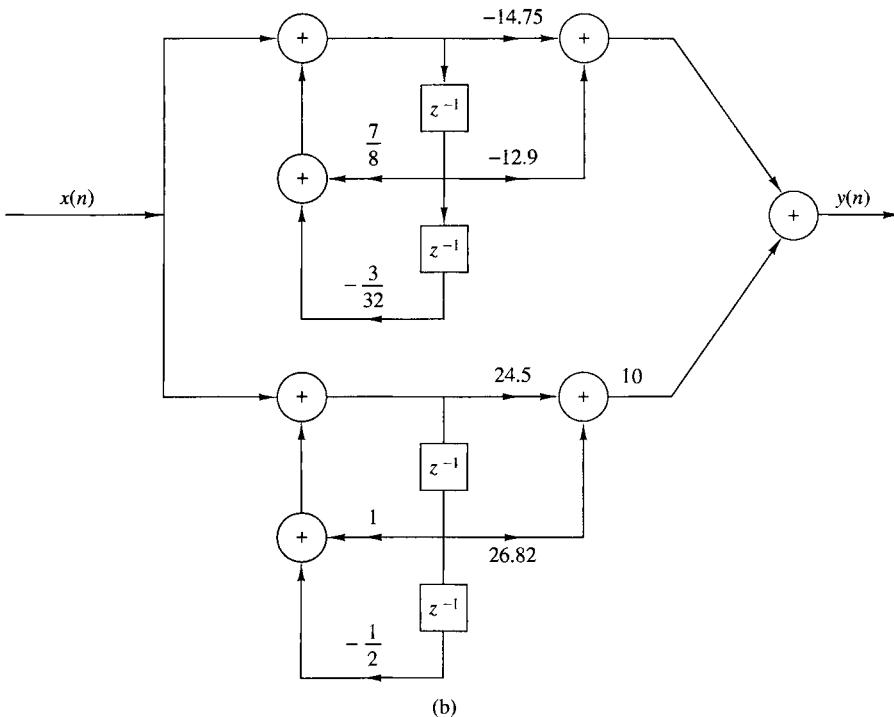
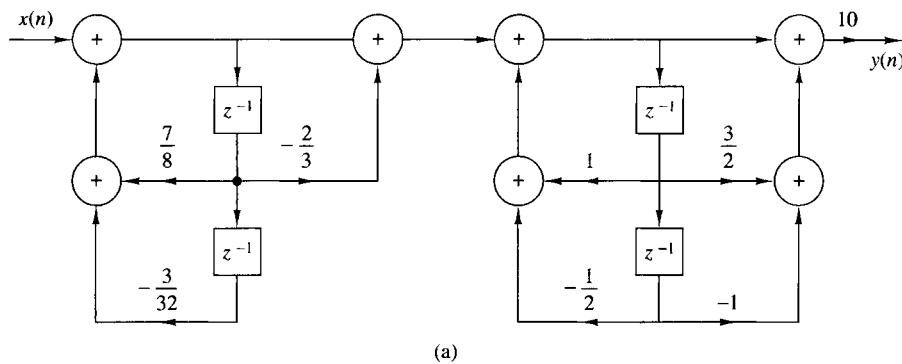


Figure 9.3.11 Cascade and parallel realizations for the system in Example 9.3.1.

where A_1 , A_2 , A_3 , and A_3^* are to be determined. After some arithmetic we find that

$$A_1 = 2.93, \quad A_2 = -17.68, \quad A_3 = 12.25 - j14.57, \quad A_3^* = 12.25 + j14.57$$

Upon recombining pairs of poles, we obtain

$$H(z) = \frac{-14.75 - 12.90z^{-1}}{1 - \frac{7}{8}z^{-1} + \frac{3}{32}z^{-2}} + \frac{24.50 + 26.82z^{-1}}{1 - z^{-1} + \frac{1}{2}z^{-2}}$$

The parallel-form realization is illustrated in Fig. 9.3.11(b).

9.3.5 Lattice and Lattice-Ladder Structures for IIR Systems

In Section 9.2.4 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

$$H(z) = \frac{1}{1 + \sum_{k=1}^N a_N(k)z^{-k}} = \frac{1}{A_N(z)} \quad (9.3.26)$$

The direct-form realization of this system is illustrated in Fig. 9.3.12. The difference equation for this IIR system is

$$y(n) = -\sum_{k=1}^N a_N(k)y(n-k) + x(n) \quad (9.3.27)$$

It is interesting to note that if we interchange the roles of input and output [i.e., interchange $x(n)$ with $y(n)$ in (9.3.27)], we obtain

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

or, equivalently,

$$y(n) = x(n) + \sum_{k=1}^N a_N(k)x(n-k) \quad (9.3.28)$$

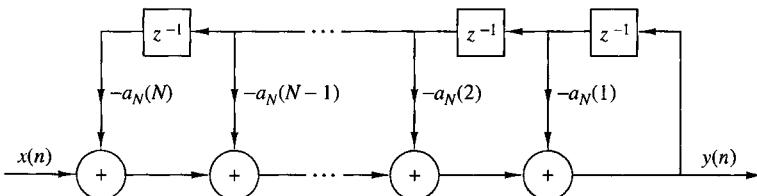


Figure 9.3.12 Direct-form realization of an all-pole system.

We note that the equation in (9.3.28) describes an FIR system having the system function $H(z) = A_N(z)$, while the system described by the difference equation in (9.3.27) represents an IIR system with system function $H(z) = 1/A_N(z)$. One system can be obtained from the other simply by interchanging the roles of the input and output.

Based on this observation, we shall use the all-zero (FIR) lattice described in Section 9.2.4 to obtain a lattice structure for an all-pole IIR system by interchanging the roles of the input and output. First, we take the all-zero lattice filter illustrated in Fig. 9.2.11 and then redefine the input as

$$x(n) = f_N(n) \quad (9.3.29)$$

and the output as

$$y(n) = f_0(n) \quad (9.3.30)$$

These are exactly the opposite of the definitions for the all-zero lattice filter. These definitions dictate that the quantities $\{f_m(n)\}$ be computed in descending order [i.e., $f_N(n)$, $f_{N-1}(n), \dots$]. This computation can be accomplished by rearranging the recursive equation in (9.2.29) and thus solving for $f_{m-1}(n)$ in terms of $f_m(n)$, that is,

$$f_{m-1}(n) = f_m(n) - K_m g_{m-1}(n-1), \quad m = N, N-1, \dots, 1$$

The equation (9.2.30) for $g_m(n)$ remains unchanged.

The result of these changes is the set of equations

$$f_N(n) = x(n) \quad (9.3.31)$$

$$f_{m-1}(n) = f_m(n) - K_m g_{m-1}(n-1), \quad m = N, N-1, \dots, 1 \quad (9.3.32)$$

$$g_m(n) = K_m f_{m-1}(n) + g_{m-1}(n-1), \quad m = N, N-1, \dots, 1 \quad (9.3.33)$$

$$y(n) = f_0(n) = g_0(n) \quad (9.3.34)$$

which correspond to the structure shown in Fig. 9.3.13.

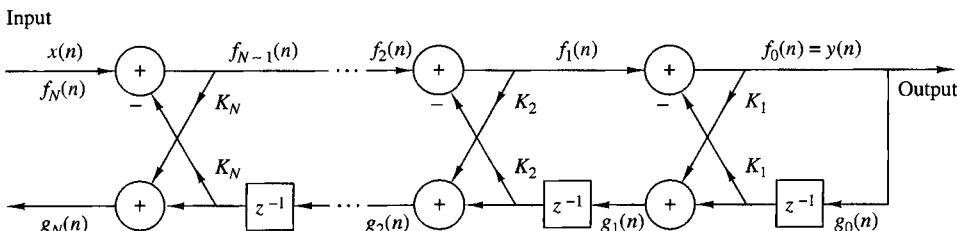


Figure 9.3.13 Lattice structure for an all-pole IIR system.

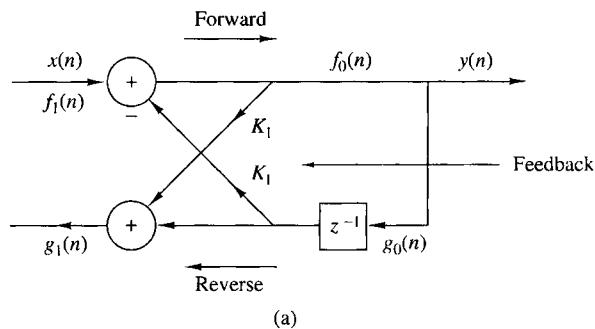
To demonstrate that the set of equations (9.3.31) through (9.3.34) represent an all-pole IIR system, let us consider the case where $N = 1$. The equations reduce to

$$\begin{aligned} x(n) &= f_1(n) \\ f_0(n) &= f_1(n) - K_1 g_0(n-1) \\ g_1(n) &= K_1 f_0(n) + g_0(n-1) \\ y(n) &= f_0(n) \\ &= x(n) - K_1 y(n-1) \end{aligned} \quad (9.3.35)$$

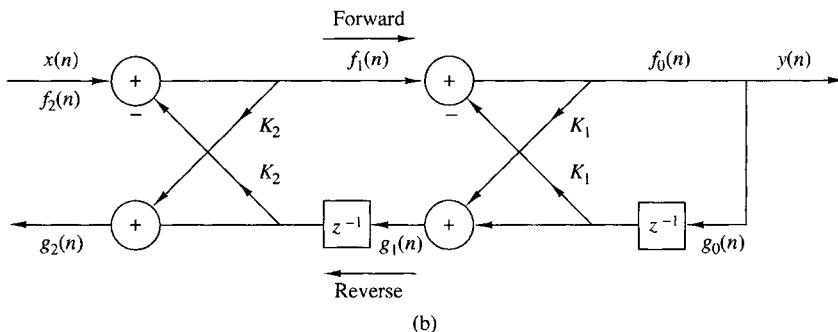
Furthermore, the equation for $g_1(n)$ can be expressed as

$$g_1(n) = K_1 y(n) + y(n-1) \quad (9.3.36)$$

We observe that (9.3.35) represents a first-order all-pole IIR system while (9.3.36) represents a first-order FIR system. The pole is a result of the feedback introduced by the solution of the $\{f_m(n)\}$ in descending order. This feedback is depicted in Fig. 9.3.14(a).



(a)



(b)

Figure 9.3.14 Single-pole and two-pole lattice system.

Next, let us consider the case $N = 2$, which corresponds to the structure in Fig. 9.3.14(b). The equations corresponding to this structure are

$$\begin{aligned} f_2(n) &= x(n) \\ f_1(n) &= f_2(n) - K_2 g_1(n-1) \\ g_2(n) &= K_2 f_1(n) + g_1(n-1) \\ f_0(n) &= f_1(n) - K_1 g_0(n-1) \\ g_1(n) &= K_1 f_0(n) + g_0(n-1) \\ y(n) &= f_0(n) = g_0(n) \end{aligned} \tag{9.3.37}$$

After some simple substitutions and manipulations we obtain

$$y(n) = -K_1(1 + K_2)y(n-1) - K_2y(n-2) + x(n) \tag{9.3.38}$$

$$g_2(n) = K_2y(n) + K_1(1 + K_2)y(n-1) + y(n-2) \tag{9.3.39}$$

Clearly, the difference equation in (9.3.38) represents a two-pole IIR system, and the relation in (9.3.39) is the input-output equation for a two-zero FIR system. Note that the coefficients for the FIR system are identical to those in the IIR system except that they occur in reverse order.

In general, these conclusions hold for any N . Indeed, with the definition of $A_m(z)$ given in (9.2.32), the system function for the all-pole IIR system is

$$H_a(z) = \frac{Y(z)}{X(z)} = \frac{F_0(z)}{F_m(z)} = \frac{1}{A_m(z)} \tag{9.3.40}$$

Similarly, the system function of the all-zero (FIR) system is

$$H_b(z) = \frac{G_m(z)}{Y(z)} = \frac{G_m(z)}{G_0(z)} = B_m(z) = z^{-m} A_m(z^{-1}) \tag{9.3.41}$$

where we used the previously established relationships in (9.2.36) through (9.2.42). Thus the coefficients in the FIR system $H_b(z)$ are identical to the coefficients in $A_m(z)$, except that they occur in reverse order.

It is interesting to note that the all-pole lattice structure has an all-zero path with input $g_0(n)$ and output $g_N(n)$, which is identical to its counterpart all-zero path in the all-zero lattice structure. The polynomial $B_m(z)$, which represents the system function of the all-zero path common to both lattice structures, is usually called the *backward system function*, because it provides a backward path in the all-pole lattice structure.

From this discussion the reader should observe that the all-zero and all-pole lattice structures are characterized by the same set of lattice parameters, namely, K_1, K_2, \dots, K_N . The two lattice structures differ only in the interconnections of their signal flow graphs. Consequently, the algorithms for converting between the

system parameters $\{\alpha_m(k)\}$ in the direct-form realization of an FIR system, and the parameters of its lattice counterpart apply as well to the all-pole structure.

We recall that the roots of the polynomial $A_N(z)$ lie inside the unit circle if and only if the lattice parameters $|K_m| < 1$ for all $m = 1, 2, \dots, N$. Therefore, the all-pole lattice structure is a stable system if and only if its parameters $|K_m| < 1$ for all m .

In practical applications the all-pole lattice structure has been used to model the human vocal tract and a stratified earth. In such cases the lattice parameters, $\{K_m\}$, have the physical significance of being identical to reflection coefficients in the physical medium. This is the reason that the lattice parameters are often called *reflection coefficients*. In such applications, a stable model of the medium requires that the reflection coefficients, obtained by performing measurements on output signals from the medium, be less than unity.

The all-pole lattice provides the basic building block for lattice-type structures that implement IIR systems that contain both poles and zeros. To develop the appropriate structure, let us consider an IIR system with system function

$$H(z) = \frac{\sum_{k=0}^M c_M(k)z^{-k}}{1 + \sum_{k=1}^N a_N(k)z^{-k}} = \frac{C_M(z)}{A_N(z)} \quad (9.3.42)$$

where the notation for the numerator polynomial has been changed to avoid confusion with our previous development. Without loss of generality, we assume that $N \geq M$.

In the direct form II structure, the system in (9.3.42) is described by the difference equations

$$w(n) = -\sum_{k=1}^N a_N(k)w(n-k) + x(n) \quad (9.3.43)$$

$$y(n) = \sum_{k=0}^M c_M(k)w(n-k) \quad (9.3.44)$$

Note that (9.3.43) is the input–output of an all-pole IIR system and that (9.3.44) is the input–output of an all-zero system. Furthermore, we observe that the output of the all-zero system is simply a linear combination of delayed outputs from the all-pole system. This is easily seen by observing the direct form II structure redrawn as in Fig. 9.3.15.

Since zeros result from forming a linear combination of previous outputs, we can carry over this observation to construct a pole–zero IIR system using the all-pole lattice structure as the basic building block. We have already observed that $g_m(n)$ is a linear combination of present and past outputs. In fact, the system

$$H_b(z) = \frac{G_m(z)}{Y(z)} = B_m(z)$$

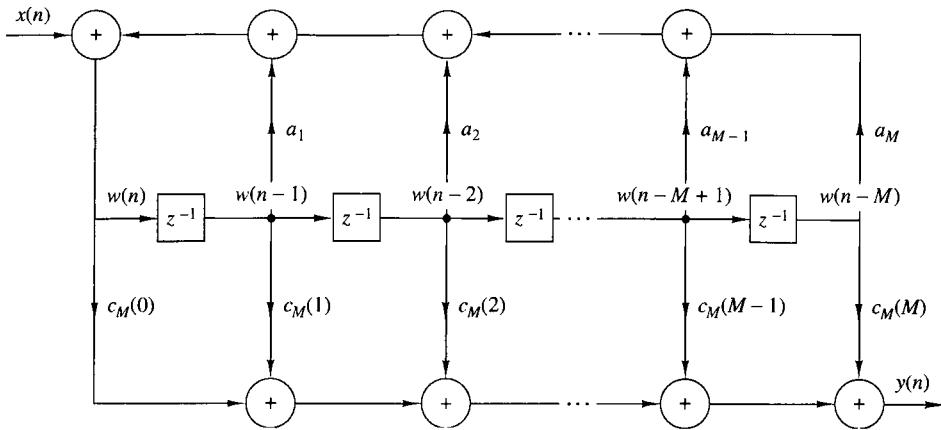


Figure 9.3.15 Direct form II realization of IIR system.

is an all-zero system. Therefore, any linear combination of $\{g_m(n)\}$ is also an all-zero system.

Thus we begin with an all-pole lattice structure with parameters K_m , $1 \leq m \leq N$, and we add a *ladder* part by taking as the output a weighted linear combination of $\{g_m(n)\}$. The result is a pole–zero IIR system which has the *lattice-ladder* structure shown in Fig. 9.3.16 for $M = N$. Its output is

$$y(n) = \sum_{m=0}^M v_m g_m(n) \quad (9.3.45)$$

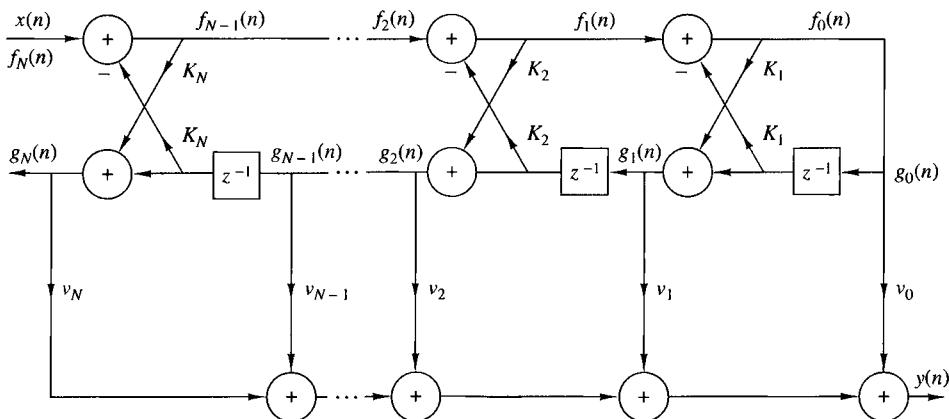


Figure 9.3.16 Lattice-ladder structure for the realization of a pole–zero system.

where $\{v_m\}$ are the parameters that determine the zeros of the system. The system function corresponding to (9.3.45) is

$$\begin{aligned} H(z) &= \frac{Y(z)}{X(z)} \\ &= \sum_{m=0}^M v_m \frac{G_m(z)}{X(z)} \end{aligned} \quad (9.3.46)$$

Since $X(z) = F_N(z)$ and $F_0(z) = G_0(z)$, (9.3.46) can be written as

$$\begin{aligned} H(z) &= \sum_{m=0}^M v_m \frac{G_m(z)}{G_0(z)} \frac{F_0(z)}{F_N(z)} \\ &= \sum_{m=0}^M v_m \frac{B_m(z)}{A_N(z)} \\ &= \frac{\sum_{m=0}^M v_m B_m(z)}{A_N(z)} \end{aligned} \quad (9.3.47)$$

If we compare (9.3.41) with (9.3.47), we conclude that

$$C_M(z) = \sum_{m=0}^M v_m B_m(z) \quad (9.3.48)$$

This is the desired relationship that can be used to determine the weighting coefficients $\{v_m\}$. Thus, we have demonstrated that the coefficients of the numerator polynomial $C_M(z)$ determine the ladder parameters $\{v_m\}$, whereas the coefficients in the denominator polynomial $A_N(z)$ determine the lattice parameters $\{K_m\}$.

Given the polynomials $C_M(z)$ and $A_N(z)$, where $N \geq M$, the parameters of the all-pole lattice are determined first, as described previously, by the conversion algorithm given in Section 9.2.4, which converts the direct-form coefficients into lattice parameters. By means of the step-down recursive relations given by (9.2.54), we obtain the lattice parameters $\{K_m\}$ and the polynomials $B_m(z)$, $m = 1, 2, \dots, N$.

The ladder parameters are determined from (9.3.48), which can be expressed as

$$C_m(z) = \sum_{k=0}^{m-1} v_k B_k(z) + v_m B_m(z) \quad (9.3.49)$$

or, equivalently, as

$$C_m(z) = C_{m-1}(z) + v_m B_m(z) \quad (9.3.50)$$

Thus $C_m(z)$ can be computed recursively from the reverse polynomials $B_m(z)$, $m = 1, 2, \dots, M$. Since $\beta_m(m) = 1$ for all m , the parameters v_m , $m = 0, 1, \dots, M$ can be determined by first noting that

$$v_m = c_m(m), \quad m = 0, 1, \dots, M \quad (9.3.51)$$

Then, by rewriting (9.3.50) as

$$C_{m-1}(z) = C_m(z) - v_m B_m(z) \quad (9.3.52)$$

and running this recursive relation backward in m (i.e., $m = M, M-1, \dots, 2$), we obtain $c_m(m)$ and therefore the ladder parameters according to (9.3.51).

The lattice-ladder filter structures that we have presented require the minimum amount of memory but not the minimum number of multiplications. Although lattice structures with only one multiplier per lattice stage exist, the two multiplier-per-stage lattice that we have described is by far the most widely used in practical applications. In conclusion, the modularity, the built-in stability characteristics embodied in the coefficients $\{K_m\}$, and its robustness to finite-word-length effects make the lattice structure very attractive in many practical applications, including speech processing systems, adaptive filtering, and geophysical signal processing.

9.4 Representation of Numbers

Up to this point we have considered the implementation of discrete-time systems without being concerned about the finite-word-length effects that are inherent in any digital realization, whether it be in hardware or in software. In fact, we have analyzed systems that are modeled as linear when, in fact, digital realizations of such systems are inherently nonlinear.

In this and the following two sections, we consider the various forms of quantization effects that arise in digital signal processing. Although we describe floating-point arithmetic operations briefly, our major concern is with fixed-point realizations of digital filters.

In this section we consider the representation of numbers for digital computations. The main characteristic of digital arithmetic is the limited (usually fixed) number of digits used to represent numbers. This constraint leads to finite numerical precision in computations, which leads to round-off errors and nonlinear effects in the performance of digital filters. We now provide a brief introduction to digital arithmetic.

9.4.1 Fixed-Point Representation of Numbers

The representation of numbers in a fixed-point format is a generalization of the familiar decimal representation of a number as a string of digits with a decimal point. In this notation, the digits to the left of the decimal point represent the integer part

of the number, and the digits to the right of the decimal point represent the fractional part of the number. Thus a real number X can be represented as

$$\begin{aligned} X &= (b_{-A}, \dots, b_{-1}, b_0, b_1, \dots, b_B)_r \\ &= \sum_{i=-A}^B b_i r^{-i}, \quad 0 \leq b_i \leq (r-1) \end{aligned} \tag{9.4.1}$$

where b_i represents the digit, r is the radix or base, A is the number of integer digits, and B is the number of fractional digits. As an example, the decimal number $(123.45)_{10}$ and the binary number $(101.01)_2$ represent the following sums:

$$(123.45)_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2}$$

$$(101.01)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$

Let us focus our attention on the binary representation since it is the most important for digital signal processing. In this case $r = 2$ and the digits $\{b_i\}$ are called binary digits or bits and take the values $\{0, 1\}$. The binary digit b_{-A} is called the most significant bit (MSB) of the number, and the binary digit b_B is called the least significant bit (LSB). The “binary point” between the digits b_0 and b_1 does not exist physically in the computer. Simply, the logic circuits of the computer are designed so that the computations result in numbers that correspond to the assumed location of this point.

By using an n -bit integer format ($A = n - 1$, $B = 0$), we can represent unsigned integers with magnitude in the range 0 to $2^n - 1$. Usually, we use the fraction format ($A = 0$, $B = n - 1$), with a binary point between b_0 and b_1 , that permits numbers in the range from 0 to $1 - 2^{-n}$. Note that any integer or mixed number can be represented in a fraction format by factoring out the term r^A in (9.4.1). In the sequel we focus our attention on the binary fraction format because mixed numbers are difficult to multiply and the number of bits representing an integer cannot be reduced by truncation or rounding.

There are three ways to represent negative numbers. This leads to three formats for the representation of signed binary fractions. The format for positive fractions is the same in all three representations, namely,

$$X = 0.b_1 b_2 \cdots b_B = \sum_{i=1}^B b_i \cdot 2^{-i}, \quad X \geq 0 \tag{9.4.2}$$

Note that the MSB b_0 is set to zero to represent the positive sign. Consider now the negative fraction

$$X = -0.b_1 b_2 \cdots b_B = -\sum_{i=1}^B b_i \cdot 2^{-i} \tag{9.4.3}$$

This number can be represented using one of the following three formats.

Sign-magnitude format. In this format, the MSB is set to 1 to represent the negative sign,

$$X_{SM} = 1.b_1b_2 \cdots b_B, \quad \text{for } X \leq 0 \quad (9.4.4)$$

One's-complement format. In this format the negative numbers are represented as

$$X_{1C} = 1.\bar{b}_1\bar{b}_2 \cdots \bar{b}_B, \quad X \leq 0 \quad (9.4.5)$$

where $\bar{b}_i = 1 - b_i$ is the one's complement of b_i . Thus if X is a positive number, the corresponding negative number is determined by complementing (changing 1's to 0's and 0's to 1's) all the bits. An alternative definition for X_{1C} can be obtained by noting that

$$X_{1C} = 1 \times 2^0 + \sum_{i=1}^B (1 - b_i) \cdot 2^{-i} = 2 - 2^{-B}|X| \quad (9.4.6)$$

Two's-complement format. In this format a negative number is represented by forming the two's complement of the corresponding positive number. In other words, the negative number is obtained by subtracting the positive number from 2.0. More simply, the two's complement is formed by complementing the positive number and adding one LSB. Thus

$$X_{2C} = 1.\bar{b}_1\bar{b}_2 \cdots \bar{b}_B + 00 \cdots 01, \quad X < 0 \quad (9.4.7)$$

where $+$ represents modulo-2 addition that ignores any carry generated from the sign bit. For example, the number $-\frac{3}{8}$ is simply obtained by complementing 0011 ($\frac{3}{8}$) to obtain 1100 and then adding 0001. This yields 1101, which represents $-\frac{3}{8}$ in two's complement.

From (9.4.6) and (9.4.7) it can easily be seen that

$$X_{2C} = X_{1C} + 2^{-B} = 2 - |X| \quad (9.4.8)$$

To demonstrate that (9.4.7) truly represents a negative number, we use the identity

$$1 = \sum_{i=1}^B 2^{-i} + 2^{-B} \quad (9.4.9)$$

The negative number X in (9.4.3) can be expressed as

$$\begin{aligned} X_{2C} &= - \sum_{i=1}^B b_i \cdot 2^{-i} + 1 - 1 \\ &= -1 + \sum_{i=1}^B (1 - b_i) 2^{-i} + 2^{-B} \\ &= -1 + \sum_{i=1}^B \bar{b}_i \cdot 2^{-1} + 2^{-B} \end{aligned}$$

which is exactly the two's-complement representation of (9.4.7).

In summary, the value of a binary string $b_0 b_1 \dots b_B$ depends on the format used. For positive numbers, $b_0 = 0$, and the number is given by (9.4.2). For negative numbers, we use these corresponding formulas for the three formats.

EXAMPLE 9.4.1

Express the fraction $\frac{7}{8}$ and $-\frac{7}{8}$ in sign-magnitude, two's-complement, and one's-complement format.

Solution. $X = \frac{7}{8}$ is represented as $2^{-1} + 2^{-2} + 2^{-3}$, so that $X = 0.111$. In sign-magnitude format, $X = -\frac{7}{8}$ is represented as 1.111. In one's complement, we have

$$X_{1C} = 1.000$$

In two's complement, the result is

$$X_{2C} = 1.000 + 0.001 = 1.001$$

The basic arithmetic operations of addition and multiplication depend on the format used. For one's-complement and two's-complement formats, addition is carried out by adding the numbers bit by bit. The formats differ only in the way in which a carry bit affects the MSB. For example, $\frac{4}{8} - \frac{3}{8} = \frac{1}{8}$. In two's complement, we have

$$0100 \oplus 1101 = 0001$$

where \oplus indicates modulo-2 addition. Note that the carry bit, if present in the MSB, is dropped. On the other hand, in one's-complement arithmetic, the carry in the MSB, if present, is carried around to the LSB. Thus the computation $\frac{4}{8} - \frac{3}{8} = \frac{1}{8}$ becomes

$$0100 \oplus 1100 = 0000 \oplus 0001 = 0001$$

Addition in the sign-magnitude format is more complex and can involve sign checks, complementing, and the generation of a carry. On the other hand, direct multiplication of two sign-magnitude numbers is relatively straightforward, whereas a special algorithm is usually employed for one's-complement and two's-complement multiplication.

Most fixed-point digital signal processors use two's-complement arithmetic. Hence, the range for $(B + 1)$ -bit numbers is from -1 to $1 - 2^{-B}$. These numbers can be viewed in a wheel format as shown in Fig. 9.4.1 for $B = 2$. Two's-complement arithmetic is basically arithmetic modulo 2^{B+1} [i.e., any number that falls outside the range (overflow or underflow) is reduced to this range by subtracting an appropriate multiple of 2^{B+1}]. This type of arithmetic can be viewed as counting using the wheel of Fig. 9.4.1. A very important property of two's-complement addition is that if the final sum of a string of numbers X_1, X_2, \dots, X_N is within the range, it will be computed correctly, even if individual partial sums result in overflows. This and other characteristics of two's-complement arithmetic are considered in Problem 9.29.

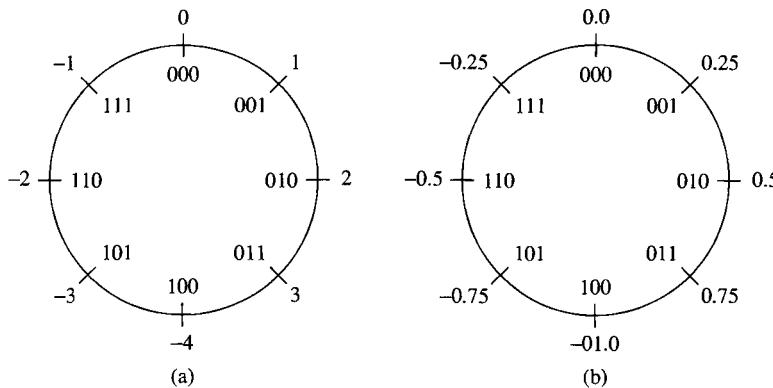


Figure 9.4.1 Counting wheel for 3-bit two's-complement numbers:
(a) integers and (b) fractions.

In general, the multiplication of two fixed-point numbers each b bits in length results in a product of $2b$ bits in length. In fixed-point arithmetic, the product is either truncated or rounded back to b bits. As a result we have a truncation or round-off error in the b least significant bits. The characterization of such errors is treated below.

9.4.2 Binary Floating-Point Representation of Numbers

A fixed-point representation of numbers allows us to cover a range of numbers, say, $x_{\max} - x_{\min}$ with a resolution

$$\Delta = \frac{x_{\max} - x_{\min}}{m - 1}$$

where $m = 2^b$ is the number of levels and b is the number of bits. A basic characteristic of the fixed-point representation is that the resolution is fixed. Furthermore, Δ increases in direct proportion to an increase in the dynamic range.

A floating-point representation can be employed as a means for covering a larger dynamic range. The binary floating-point representation commonly used in practice consists of a mantissa M , which is the fractional part of the number and falls in the range $\frac{1}{2} \leq M < 1$, multiplied by the exponential factor 2^E , where the exponent E is either a positive or negative integer. Hence a number X is represented as

$$X = M \cdot 2^E$$

The mantissa requires a sign bit for representing positive and negative numbers, and the exponent requires an additional sign bit. Since the mantissa is a signed fraction, we can use any of the four fixed-point representations just described.

For example, the number $X_1 = 5$ is represented by the following mantissa and exponent:

$$M_1 = 0.101000$$

$$E_1 = 011$$

while the number $X_2 = \frac{3}{8}$ is represented by the following mantissa and exponent

$$M_2 = 0.110000$$

$$E_2 = 101$$

where the leftmost bit in the exponent represents the sign bit.

If the two numbers are to be multiplied, the mantissas are multiplied and the exponents are added. Thus the product of these two numbers is

$$\begin{aligned} X_1 X_2 &= M_1 M_2 \cdot 2^{E_1+E_2} \\ &= (0.011110) \cdot 2^{010} \\ &= (0.111100) \cdot 2^{001} \end{aligned}$$

On the other hand, the addition of the two floating-point numbers requires that the exponents be equal. This can be accomplished by shifting the mantissa of the smaller number to the right and compensating by increasing the corresponding exponent. Thus the number X_2 can be expressed as

$$M_2 = 0.000011$$

$$E_2 = 011$$

With $E_2 = E_1$, we can add the two numbers X_1 and X_2 . The result is

$$X_1 + X_2 = (0.101011) \cdot 2^{011}$$

It should be observed that the shifting operation required to equalize the exponent of X_2 with that for X_1 results in loss of precision, in general. In this example the six-bit mantissa was sufficiently long to accommodate a shift of four bits to the right for M_2 without dropping any of the ones. However, a shift of five bits would have caused the loss of a single bit and a shift of six bits to the right would have resulted in a mantissa of $M_2 = 0.000000$, unless we round upward after shifting so that $M_2 = 0.000001$.

Overflow occurs in the multiplication of two floating-point numbers when the sum of the exponents exceeds the dynamic range of the fixed-point representation of the exponent.

In comparing a fixed-point representation with a floating-point representation, each with the same number of total bits, it is apparent that the floating-point representation allows us to cover a larger dynamic range by varying the resolution across the range. The resolution decreases with an increase in the size of successive numbers. In other words, the distance between two successive floating-point numbers increases as the numbers increase in size. It is this variable resolution that results in a larger dynamic range. Alternatively, if we wish to cover the same dynamic range with both fixed-point and floating-point representations, the floating-point representation provides finer resolution for small numbers but coarser resolution for the larger

numbers. In contrast, the fixed-point representation provides a uniform resolution throughout the range of numbers.

For example, if we have a computer with a word size of 32 bits, it is possible to represent 2^{32} numbers. If we wish to represent the positive integers beginning with zero, the largest possible integer that can be accommodated is

$$2^{32} - 1 = 4,294,967,295$$

The distance between successive numbers (the resolution) is 1. Alternatively, we can designate the leftmost bit as the sign bit and use the remaining 31 bits for the magnitude. In such a case a fixed-point representation allows us to cover the range

$$-(2^{31} - 1) = -2,147,483,647 \quad \text{to} \quad (2^{31} - 1) = 2,147,483,647$$

again with a resolution of 1.

On the other hand, suppose that we increase the resolution by allocating 10 bits for a fractional part, 21 bits for the integer part, and 1 bit for the sign. Then this representation allows us to cover the dynamic range

$$-(2^{31} - 1) \cdot 2^{-10} = -(2^{21} - 2^{-10}) \quad \text{to} \quad (2^{31} - 1) \cdot 2^{-10} = 2^{21} - 2^{-10}$$

or, equivalently,

$$-2,097,151.999 \quad \text{to} \quad 2,097,151.999$$

In this case, the resolution is 2^{-10} . Thus, the dynamic range has been decreased by a factor of approximately 1000 (actually 2^{10}), while the resolution has been increased by the same factor.

For comparison, suppose that the 32-bit word is used to represent floating-point numbers. In particular, let the mantissa be represented by 23 bits plus a sign bit and let the exponent be represented by 7 bits plus a sign bit. Now, the smallest number in magnitude will have the representation,

$$\begin{array}{ccccccccc} \text{sign} & & 23 \text{ bits} & \text{sign} & & 7 \text{ bits} \\ 0. & 100 \cdots 0 & & 1 & 1111111 & = \frac{1}{2} \times 2^{-127} \approx 0.3 \times 10^{-38} \end{array}$$

At the other extreme, the largest number that can be represented with this floating-point representation is

$$\begin{array}{ccccccccc} \text{sign} & & 23 \text{ bits} & \text{sign} & & 7 \text{ bits} \\ 0 & 111 \cdots 1 & & 0 & 1111111 & = (1 - 2^{-23}) \times 2^{127} \approx 1.7 \times 10^{38} \end{array}$$

Thus, we have achieved a dynamic range of approximately 10^{76} , but with varying resolution. In particular, we have fine resolution for small numbers and coarse resolution for larger numbers.

The representation of zero poses some special problems. In general, only the mantissa has to be zero, but not the exponent. The choice of M and E , the representation of zero, the handling of overflows, and other related issues have resulted in various floating-point representations on different digital computers. In an effort to define a common floating-point format, the Institute of Electrical and Electronic Engineers (IEEE) introduced the IEEE 754 standard, which is widely used in practice. For a 32-bit machine, the IEEE 754 standard single-precision, floating-point number is represented as $X = (-1)^s \cdot 2^{E-127}(M)$, where

0	1	8	9	31
S	E			M

This number has the following interpretations:

- If $E = 255$ and $M \neq 0$, then X is not a number
- If $E = 255$ and $M = 0$, then $X = (-1)^s \cdot \infty$
- If $0 < E < 255$, then $X = (-1)^s \cdot 2^{E-127}(1.M)$
- If $E = 0$ and $M \neq 0$, then $X = (-1)^s \cdot 2^{-126}(0.M)$
- If $E = 0$ and $M = 0$, then $X = (-1)^s \cdot 0$

where $0.M$ is a fraction and $1.M$ is a mixed number with one integer bit and 23 fractional bits. For example, the number

0	1 0 0 0 0 0 1 0	1 0 1 0 \cdots 0 0
S	E	M

has the value $X = -1^0 \times 2^{130-127} \times 1.1010\ldots 0 = 2^3 \times \frac{13}{8} = 13$. The magnitude range of the 32-bit IEEE 754 floating-point numbers is from $2^{-126} \times 2^{-23}$ to $(2 - 2^{-23}) \times 2^{127}$ (i.e., from 1.18×10^{-38} to 3.40×10^{38}). Computations with numbers outside this range result in either underflow or overflow.

9.4.3 Errors Resulting from Rounding and Truncation

In performing computations such as multiplications with either fixed-point or floating-point arithmetic, we are usually faced with the problem of quantizing a number via truncation or rounding, from a given level of precision to a level of lower precision. The effect of rounding and truncation is to introduce an error whose value depends on the number of bits in the original number relative to the number of bits after quantization. The characteristics of the errors introduced through either truncation or rounding depend on the particular form of number representation.

To be specific, let us consider a fixed-point representation in which a number x is quantized from b_u bits to b bits. Thus the number

$$x = \overbrace{0.1011\cdots 01}^{b_u}$$

consisting of b_u bits prior to quantization is represented as

$$x = \overbrace{0.101 \cdots 1}^b$$

after quantization, where $b < b_u$. For example, if x represents the sample of an analog signal, then b_u may be taken as infinite. In any case if the quantizer truncates the value of x , the truncation error is defined as

$$E_t = Q_t(x) - x \quad (9.4.10)$$

First, we consider the range of values of the error for sign-magnitude and two's-complement representation. In both of these representations, the positive numbers have identical representations. For positive numbers, truncation results in a number that is smaller than the unquantized number. Consequently, the truncation error resulting from a reduction of the number of significant bits from b_u to b is

$$-(2^{-b} - 2^{-b_u}) \leq E_t \leq 0 \quad (9.4.11)$$

where the largest error arises from discarding $b_u - b$ bits, all of which are ones.

In the case of negative fixed-point numbers based on the sign-magnitude representation, the truncation error is positive, since truncation basically reduces the magnitude of the numbers. Consequently, for negative numbers, we have

$$0 \leq E_t \leq (2^{-b} - 2^{-b_u}) \quad (9.4.12)$$

In the two's-complement representation, the negative of a number is obtained by subtracting the corresponding positive number from 2. As a consequence, the effect of truncation on a negative number is to increase the magnitude of the negative number. Consequently, $x > Q_t(x)$ and hence

$$-(2^{-b} - 2^{-b_u}) \leq E_t \leq 0 \quad (9.4.13)$$

Hence we conclude that *the truncation error for the sign-magnitude representation is symmetric about zero and falls in the range*

$$-(2^{-b} - 2^{-b_u}) \leq E_t \leq (2^{-b} - 2^{-b_u}) \quad (9.4.14)$$

On the other hand, *for two's-complement representation, the truncation error is always negative and falls in the range*

$$-(2^{-b} - 2^{-b_u}) \leq E_t \leq 0 \quad (9.4.15)$$

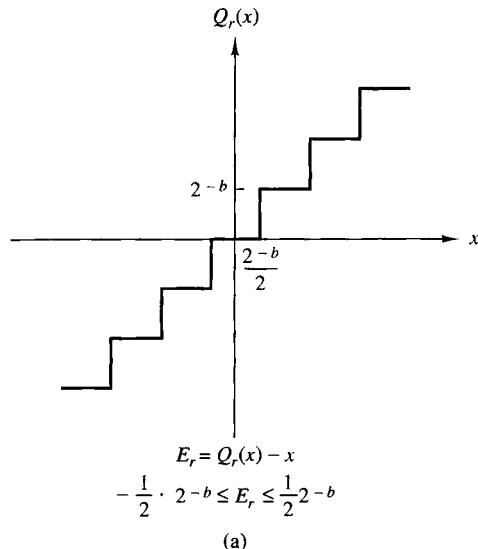
Next, let us consider the quantization errors due to rounding of a number. A number x , represented by b_u bits before quantization and b bits after quantization, incurs a quantization error

$$E_r = Q_r(x) - x \quad (9.4.16)$$

Basically, rounding involves only the magnitude of the number and, consequently, the round-off error is independent of the type of fixed-point representation. The maximum error that can be introduced through rounding is $(2^{-b} - 2^{-b_u})/2$ and this can be either positive or negative, depending on the value of x . Therefore, *the round-off error is symmetric about zero and falls in the range*

$$-\frac{1}{2}(2^{-b} - 2^{-b_u}) \leq E_r \leq \frac{1}{2}(2^{-b} - 2^{-b_u}) \quad (9.4.17)$$

These relationships are summarized in Fig. 9.4.2 when x is a continuous signal amplitude ($b_u = \infty$).



(a)

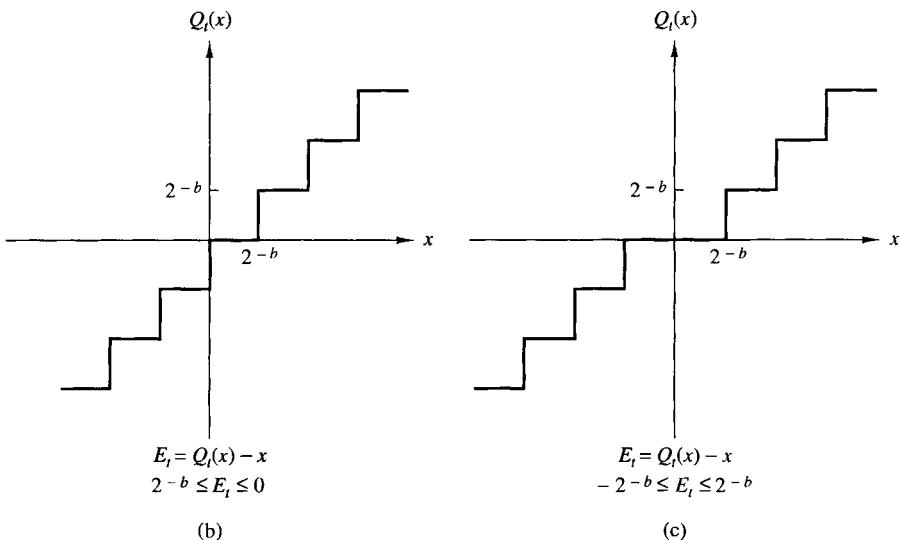


Figure 9.4.2 Quantization errors in rounding and truncation: (a) rounding; (b) truncation in two's complement; (c) truncation in sign-magnitude.

In a floating-point representation, the mantissa is either rounded or truncated. Due to the nonuniform resolution, the corresponding error in a floating-point representation is proportional to the number being quantized. An appropriate representation for the quantized value is

$$Q(x) = x + ex \quad (9.4.18)$$

where e is called the relative error. Now

$$Q(x) - x = ex \quad (9.4.19)$$

In the case of truncation based on two's-complement representation of the mantissa, we have

$$-2^E 2^{-b} < e_t x < 0 \quad (9.4.20)$$

for positive numbers. Since $2^{E-1} \leq x < 2^E$, it follows that

$$-2^{-b+1} < e_t \leq 0, \quad x > 0 \quad (9.4.21)$$

On the other hand, for a negative number in two's-complement representation, the error is

$$0 \leq e_t x < 2^E 2^{-b}$$

and hence

$$0 \leq e_t < 2^{-b+1}, \quad x < 0 \quad (9.4.22)$$

In the case where the mantissa is rounded, the resulting error is symmetric relative to zero and has a maximum value of $\pm 2^{-b}/2$. Consequently, the round-off error becomes

$$-2^E \cdot 2^{-b}/2 < e_r x \leq 2^E \cdot 2^{-b}/2 \quad (9.4.23)$$

Again, since x falls in the range $2^{E-1} \leq x < 2^E$, we divide through by 2^{E-1} so that

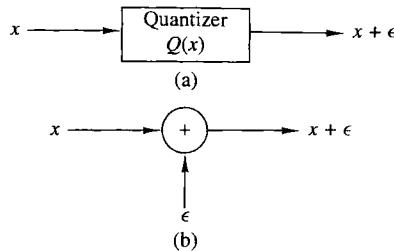
$$-2^{-b} < e_r \leq 2^{-b} \quad (9.4.24)$$

In arithmetic computations involving quantization via truncation and rounding, it is convenient to adopt a statistical approach to the characterization of such errors. The quantizer can be modeled as introducing an additive noise to the unquantized value x . Thus we can write

$$Q(x) = x + \epsilon$$

where $\epsilon = E_r$ for rounding and $\epsilon = E_t$ for truncation. This model is illustrated in Fig. 9.4.3.

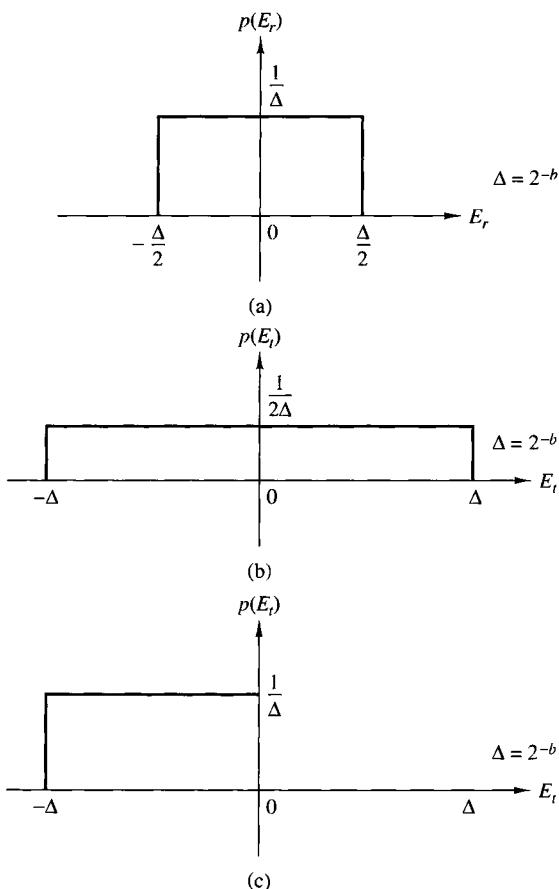
Since x can be any number that falls within any of the levels of the quantizer, the quantization error is usually modeled as a random variable that falls within the limits specified. This random variable is assumed to be uniformly distributed within the ranges specified for the fixed-point representations. Furthermore, in practice, $b_u \gg b$, so that we can neglect the factor of 2^{-b_u} in the formulas given below. Under these conditions, the probability density functions for the round-off and truncation errors in the two fixed-point representations are illustrated in Fig. 9.4.4. We note that

**Figure 9.4.3**

Additive noise model for the nonlinear quantization process: (a) actual system; (b) model for quantization.

in the case of truncation of the two's-complement representation of the number, the average value of the error has a bias of $2^{-b}/2$, whereas in all other cases just illustrated, the error has an average value of zero.

We shall use this statistical characterization of the quantization errors in our treatment of such errors in digital filtering and in the computation of the DFT for fixed-point implementation.

**Figure 9.4.4**

Statistical characterization of quantization errors:
 (a) round-off error;
 (b) truncation error for sign-magnitude;
 (c) truncation error for two's complement.

9.5 Quantization of Filter Coefficients

In the realization of FIR and IIR filters in hardware or in software on a general-purpose computer, the accuracy with which filter coefficients can be specified is limited by the word length of the computer or the length of the register provided to store the coefficients. Since the coefficients used in implementing a given filter are not exact, the poles and zeros of the system function will, in general, be different from the desired poles and zeros. Consequently, we obtain a filter having a frequency response that is different from the frequency response of the filter with unquantized coefficients.

In Section 9.5.1, we demonstrate that the sensitivity of the filter frequency response characteristics to quantization of the filter coefficients is minimized by realizing a filter having a large number of poles and zeros as an interconnection of second-order filter sections. This leads us to the parallel-form and cascade-form realizations in which the basic building blocks are second-order filter sections.

9.5.1 Analysis of Sensitivity to Quantization of Filter Coefficients

To illustrate the effect of quantization of the filter coefficients in a direct-form realization of an IIR filter, let us consider a general IIR filter with system function

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (9.5.1)$$

The direct-form realization of the IIR filter with quantized coefficients has the system function

$$\bar{H}(z) = \frac{\sum_{k=0}^M \bar{b}_k z^{-k}}{1 + \sum_{k=1}^N \bar{a}_k z^{-k}} \quad (9.5.2)$$

where the quantized coefficients $\{\bar{b}_k\}$ and $\{\bar{a}_k\}$ can be related to the unquantized coefficients $\{b_k\}$ and $\{a_k\}$ by the relations

$$\begin{aligned} \bar{a}_k &= a_k + \Delta a_k, & k &= 1, 2, \dots, N \\ \bar{b}_k &= b_k + \Delta b_k, & k &= 0, 1, \dots, M \end{aligned} \quad (9.5.3)$$

and $\{\Delta a_k\}$ and $\{\Delta b_k\}$ represent the quantization errors.

The denominator of $H(z)$ may be expressed in the form

$$D(z) = 1 + \sum_{k=0}^N a_k z^{-k} = \prod_{k=1}^N (1 - p_k z^{-1}) \quad (9.5.4)$$

where $\{p_k\}$ are the poles of $H(z)$. Similarly, we can express the denominator of $\bar{H}(z)$ as

$$\bar{D}(z) = \prod_{k=1}^N (1 - \bar{p}_k z^{-1}) \quad (9.5.5)$$

where $\bar{p}_k = p_k + \Delta p_k$, $k = 1, 2, \dots, N$, and Δp_k is the error or perturbation resulting from the quantization of the filter coefficients.

We shall now relate the perturbation Δp_k to the quantization errors in the $\{a_k\}$.

The perturbation error Δp_i can be expressed as

$$\Delta p_i = \sum_{k=1}^N \frac{\partial p_i}{\partial a_k} \Delta a_k \quad (9.5.6)$$

where $\partial p_i / \partial a_k$, the partial derivative of p_i with respect to a_k , represents the incremental change in the pole p_i due to a change in the coefficient a_k . Thus the total error Δp_i is expressed as a sum of the incremental errors due to changes in each of the coefficients $\{a_k\}$.

The partial derivatives $\partial p_i / \partial a_k$, $k = 1, 2, \dots, N$, can be obtained by differentiating $D(z)$ with respect to each of the $\{a_k\}$. First we have

$$\left(\frac{\partial D(z)}{\partial a_k} \right)_{z=p_i} = \left(\frac{\partial D(z)}{\partial z} \right)_{z=p_i} \left(\frac{\partial p_i}{\partial a_k} \right) \quad (9.5.7)$$

Then

$$\frac{\partial p_i}{\partial a_k} = \frac{(\partial D(z)/\partial a_k)_{z=p_i}}{(\partial D(z)/\partial z)_{z=p_i}} \quad (9.5.8)$$

The numerator of (9.5.8) is

$$\left(\frac{\partial D(z)}{\partial a_k} \right)_{z=p_i} = -z^{-k} \Big|_{z=p_i} = -p_i^{-k} \quad (9.5.9)$$

The denominator of (9.5.8) is

$$\begin{aligned} \left(\frac{\partial D(z)}{\partial z} \right)_{z=p_i} &= \left\{ \frac{\partial}{\partial z} \left[\prod_{l=1}^N (1 - p_l z^{-1}) \right] \right\}_{z=p_i} \\ &= \left\{ \sum_{k=1}^N \frac{p_k}{z^2} \prod_{\substack{l=1 \\ l \neq i}}^N (1 - p_l z^{-1}) \right\}_{z=p_i} \end{aligned} \quad (9.5.10)$$

$$= \frac{1}{p_i^N} \prod_{\substack{l=1 \\ l \neq i}}^N (p_i - p_l)$$

Therefore, (9.5.8) can be expressed as

$$\frac{\partial p_i}{\partial a_k} = \frac{-p_i^{N-k}}{\prod_{\substack{l=1 \\ l \neq i}}^N (p_i - p_l)} \quad (9.5.11)$$

Substitution of the result in (9.5.11) into (9.5.6) yields the total perturbation error Δp_i in the form

$$\Delta p_i = - \sum_{k=1}^N \frac{p_i^{N-k}}{\prod_{\substack{l=1 \\ l \neq i}}^N (p_i - p_l)} \Delta a_k \quad (9.5.12)$$

This expression provides a measure of the sensitivity of the i th pole to changes in the coefficients $\{a_k\}$. An analogous result can be obtained for the sensitivity of the zeros to errors in the parameters $\{b_k\}$.

The terms $(p_i - p_l)$ in the denominator of (9.5.12) represent vectors in the z -plane from the poles $\{p_l\}$ to the pole p_i . If the poles are tightly clustered as they are in a narrowband filter, as illustrated in Fig. 9.5.1, the lengths $|p_i - p_l|$ are small for the poles in the vicinity of p_i . These small lengths will contribute to large errors and hence a large perturbation error Δp_i results.

The error Δp_i can be minimized by maximizing the lengths $|p_i - p_l|$. This can be accomplished by realizing the high-order filter with either single-pole or double-pole filter sections. In general, however, single-pole (and single-zero) filter sections have complex-valued poles and require complex-valued arithmetic operations for their realization. This problem can be avoided by combining complex-valued poles (and zeros) to form second-order filter sections. Since the complex-valued poles are

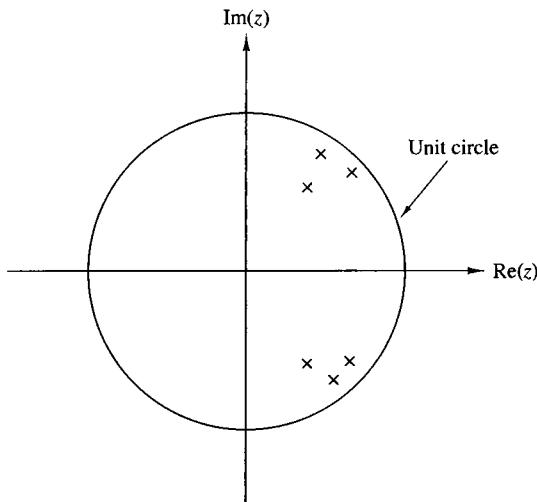


Figure 9.5.1
Pole positions for a bandpass IIR filter.

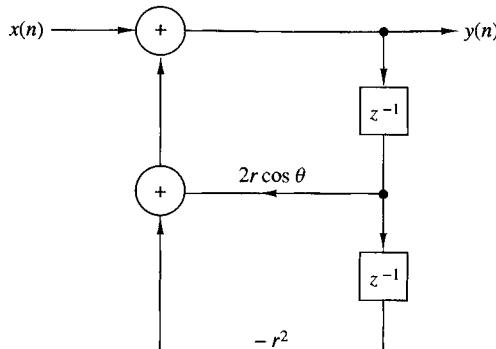


Figure 9.5.2
Realization of a two-pole
IIR filter.

usually sufficiently far apart, the perturbation errors $\{\Delta p_i\}$ are minimized. As a consequence, the resulting filter with quantized coefficients more closely approximates the frequency response characteristics of the filter with unquantized coefficients.

It is interesting to note that even in the case of a two-pole filter section, the structure used to realize the filter section plays an important role in the errors caused by coefficient quantization. To be specific, let us consider a two-pole filter with system function

$$H(z) = \frac{1}{1 - (2r \cos \theta)z^{-1} + r^2 z^{-2}} \quad (9.5.13)$$

This filter has poles at $z = re^{\pm j\theta}$. When realized as shown in Fig. 9.5.2, it has two coefficients, $a_1 = 2r \cos \theta$ and $a_2 = -r^2$. With infinite precision it is possible to achieve an infinite number of pole positions. Clearly, with finite precision (i.e., quantized coefficients a_1 and a_2), the possible pole positions are also finite. In fact, when b bits are used to represent the magnitudes of a_1 and a_2 , there are at most $(2^b - 1)^2$ possible positions for the poles in each quadrant, excluding the case $a_1 = 0$ and $a_2 = 0$.

For example, suppose that $b = 4$. Then there are 15 possible nonzero values for a_1 . There are also 15 possible values for r^2 . We illustrate these possible values in Fig. 9.5.3 for the first quadrant of the z -plane only. There are 169 possible pole positions in this case. The nonuniformity in their positions is due to the fact that we are quantizing r^2 , whereas the pole positions lie on a circular arc of radius r . Of particular significance is the sparse set of poles for values of θ near zero and, due to symmetry, near $\theta = \pi$. This situation would be highly unfavorable for lowpass filters and highpass filters which normally have poles clustered near $\theta = 0$ and $\theta = \pi$.

An alternative realization of the two-pole filter is the coupled-form realization illustrated in Fig. 9.5.4. The two coupled equations are

$$\begin{aligned} y_1(n) &= x(n) + r \cos \theta y_1(n-1) - r \sin \theta y_2(n-1) \\ y(n) &= r \sin \theta y_1(n-1) + r \cos \theta y_2(n-1) \end{aligned} \quad (9.5.14)$$

By transforming these two equations into the z -domain, it is a simple matter to show that

$$\frac{Y(z)}{X(z)} = H(z) = \frac{(r \sin \theta)z^{-1}}{1 - (2r \cos \theta)z^{-1} + r^2 z^{-2}} \quad (9.5.15)$$

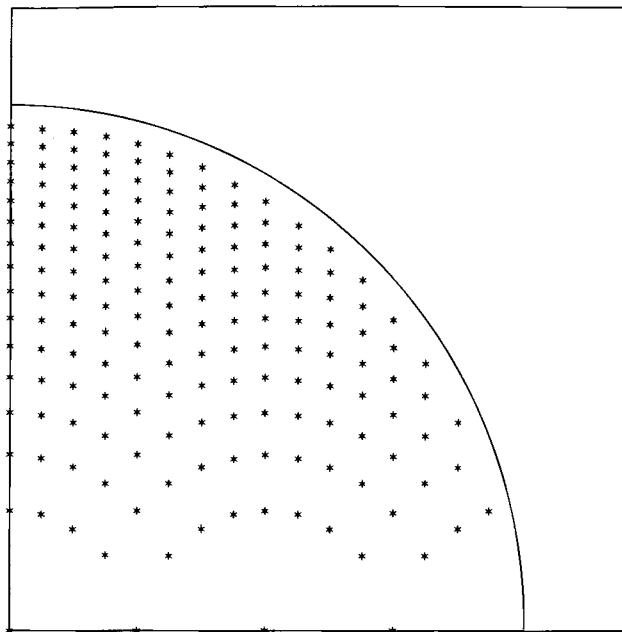


Figure 9.5.3
Possible pole positions
for two-pole IIR filter
realization in Fig. 9.5.2.

In the coupled form we observe that there are also two coefficients, $\alpha_1 = r \sin \theta$ and $\alpha_2 = r \cos \theta$. Since they are both linear in r , the possible pole positions are now equally spaced points on a rectangular grid, as shown in Fig. 9.5.5. As a consequence, the pole positions are now uniformly distributed inside the unit circle, which is a more desirable situation than the previous realization, especially for lowpass filters. (There are 198 possible pole positions in this case.) However, the price that we pay for this uniform distribution of pole positions is an increase in computations. The coupled-form realization requires four multiplications per output point, whereas the realization in Fig. 9.5.2 requires only two multiplications per output point.

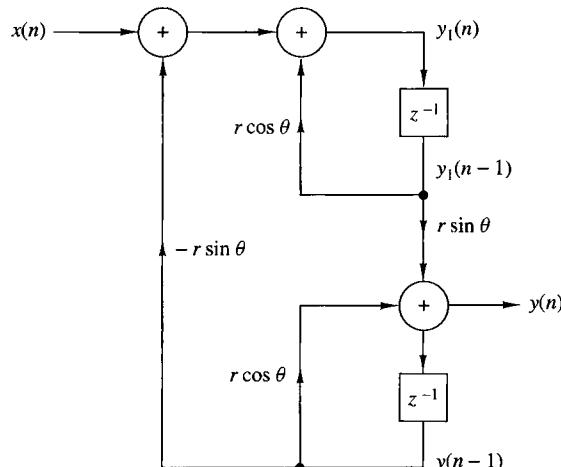


Figure 9.5.4
Coupled-form realization of
a two-pole IIR filter.

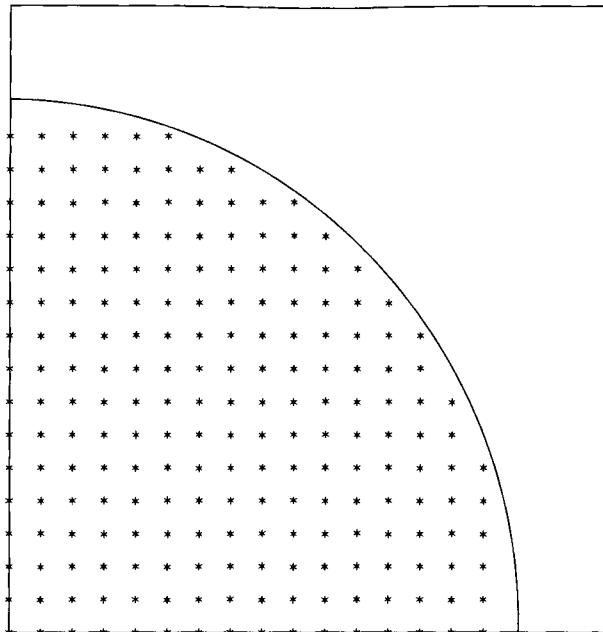


Figure 9.5.5
Possible pole positions for the coupled-form two-pole filter in Fig. 9.5.4.

Since there are various ways in which one can realize a second-order filter section, there are obviously many possibilities for different pole locations with quantized coefficients. Ideally, we should select a structure that provides us with a dense set of points in the regions where the poles lie. Unfortunately, however, there is no simple and systematic method for determining the filter realization that yields this desired result.

Given that a higher-order IIR filter should be implemented as a combination of second-order sections, we still must decide whether to employ a parallel configuration or a cascade configuration. In other words, we must decide between the realization

$$H(z) = \prod_{k=1}^K \frac{b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}} \quad (9.5.16)$$

and the realization

$$H(z) = \sum_{k=1}^K \frac{c_{k0} + c_{k1}z^{-1}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}} \quad (9.5.17)$$

If the IIR filter has zeros on the unit circle, as is generally the case with elliptic and Chebyshev type II filters, each second-order section in the cascade configuration of (9.5.16) contains a pair of complex-conjugate zeros. The coefficients $\{b_k\}$ directly determine the location of these zeros. If the $\{b_k\}$ are quantized, the sensitivity of the system response to the quantization errors is easily and directly controlled by allocating a sufficiently large number of bits to the representation of the $\{b_{ki}\}$. In fact, we can easily evaluate the perturbation effect resulting from quantizing the

coefficients $\{b_{ki}\}$ to some specified precision. Thus we have direct control of both the poles and the zeros that result from the quantization process.

On the other hand, the parallel realization of $H(z)$ provides direct control of the poles of the system only. The numerator coefficients $\{c_{k0}\}$ and $\{c_{k1}\}$ do not specify the location of the zeros directly. In fact, the $\{c_{k0}\}$ and $\{c_{k1}\}$ are obtained by performing a partial-fraction expansion of $H(z)$. Hence they do not directly influence the location of the zeros, but only indirectly through a combination of all the factors of $H(z)$. As a consequence, it is more difficult to determine the effect of quantization errors in the coefficients $\{c_{ki}\}$ on the location of the zeros of the system.

It is apparent that quantization of the parameters $\{c_{ki}\}$ is likely to produce a significant perturbation of the zero positions and usually, it is sufficiently large in fixed-point implementations to move the zeros off the unit circle. This is a highly undesirable situation, which can be easily remedied by use of a floating-point representation. In any case the cascade form is more robust in the presence of coefficient quantization and should be the preferred choice in practical applications, especially where a fixed-point representation is employed.

EXAMPLE 9.5.1

Determine the effect of parameter quantization on the frequency response of the seventh-order elliptic filter given in Table 10.6 when it is realized as a cascade of second-order sections.

Solution. The coefficients for the elliptic filter given in Table 10.6 are specified for the cascade form to six significant digits. We quantized these coefficients to four and then three significant digits (by rounding) and plotted the magnitude (in decibels) and the phase of

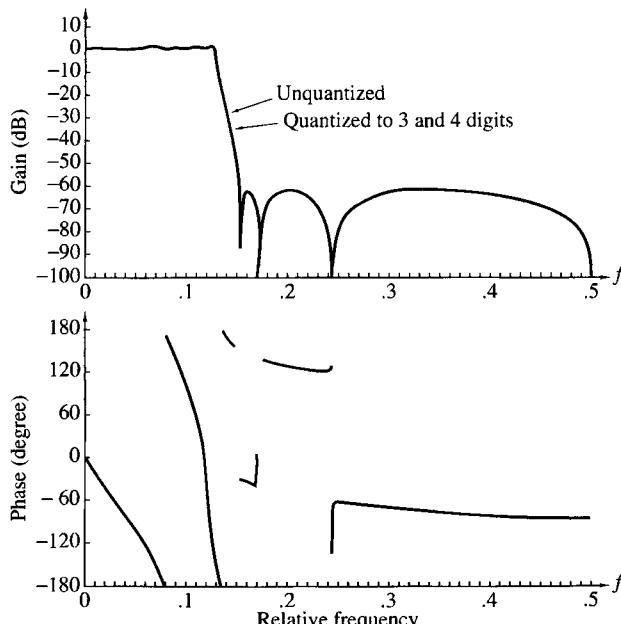


Figure 9.5.6
Effect of coefficient quantization of the magnitude and phase response of an $N = 7$ elliptic filter realized in cascade form.

the frequency response. The results are shown in Fig. 9.5.6 along the frequency response of the filter with unquantized (six significant digits) coefficients. We observe that there is an insignificant degradation due to coefficient quantization for the cascade realization.

EXAMPLE 9.5.2

Repeat the computation of the frequency response for the elliptic filter considered in Example 9.5.1 when it is realized in the parallel form with second-order sections.

Solution. The system function for the 7-order elliptic filter given in Table 10.6 is

$$\begin{aligned} H(z) = & \frac{0.2781304 + 0.0054373108z^{-1}}{1 - 0.790103z^{-1}} \\ & + \frac{-0.3867805 + 0.3322229z^{-1}}{1 - 1.517223z^{-1} + 0.714088z^{-2}} \\ & + \frac{0.1277036 - 0.1558696z^{-1}}{1 - 1.421773z^{-1} + 0.861895z^{-2}} \\ & + \frac{-0.015824186 + 0.38377356z^{-1}}{1 - 1.387447z^{-1} + 0.962242z^{-2}} \end{aligned}$$

The frequency response of this filter with coefficients quantized to four digits is shown in Fig. 9.5.7(a). When this result is compared with the frequency response in Fig. 9.5.6, we observe that the zeros in the parallel realization have been perturbed sufficiently so that the nulls in the magnitude response are now at -80 , -85 , and -92 dB. The phase response has also been perturbed by a small amount.

When the coefficients are quantized to three significant digits, the frequency response characteristic deteriorates significantly, in both magnitude and phase, as illustrated in Fig. 9.5.7(b). It is apparent from the magnitude response that the zeros are no longer on the unit circle as a result of the quantization of the coefficients. This result clearly illustrates the sensitivity of the zeros to quantization of the coefficients in the parallel form.

When compared with the results of Example 9.5.1, it is also apparent that the cascade form is definitely more robust to parameter quantization than the parallel form.

9.5.2 Quantization of Coefficients in FIR Filters

As indicated in the preceding section, the sensitivity analysis performed on the poles of a system also applies directly to the zeros of the IIR filters. Consequently, an expression analogous to (9.5.12) can be obtained for the zeros of an FIR filter. In effect, we should generally realize FIR filters with a large number of zeros as a cascade of second-order and first-order filter sections to minimize the sensitivity to coefficient quantization.

Of particular interest in practice is the realization of linear-phase FIR filters. The direct-form realizations shown in Figs. 9.2.1 and 9.2.2 maintain the linear-phase property even when the coefficients are quantized. This follows easily from the observation that the system function of a linear-phase FIR filter satisfies the property

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

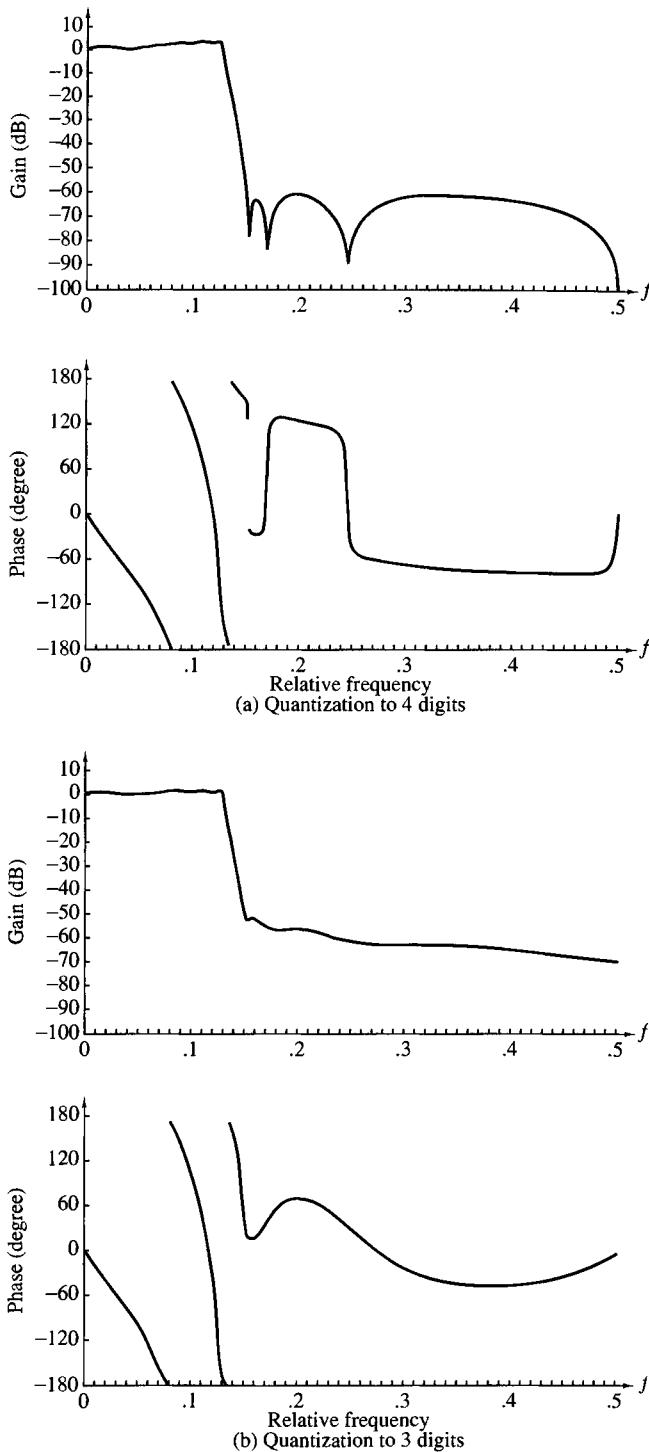


Figure 9.5.7
Effect of coefficient quantization of the magnitude and phase response of an $N = 7$ elliptic filter realized in parallel form:
(a) quantization to four digits; (b) quantization to three digits.

independent of whether the coefficients are quantized or unquantized (see Section 10.2). Consequently, coefficient quantization does not affect the phase characteristic of the FIR filter, but affects only the magnitude. As a result, coefficient quantization effects are not as severe on a linear-phase FIR filter, since the only effect is in the magnitude.

EXAMPLE 9.5.3

Determine the effect of parameter quantization on the frequency response of an $M = 32$ linear-phase FIR bandpass filter. The filter is realized in the direct form.

Solution. The frequency response of a linear-phase FIR bandpass filter with unquantized coefficients is illustrated in Fig. 9.5.8(a). When the coefficients are quantized to four significant digits, the effect on the frequency response is insignificant. However, when the coefficients are quantized to three significant digits, the sidelobes increase by several decibels, as illustrated in Fig. 9.5.8(b). This result indicates that we should use a minimum of 10 bits to represent the coefficients of this FIR filter and, preferably, 12 to 14 bits, if possible.

From this example we learn that a minimum of 10 bits is required to represent the coefficients in a direct-form realization of an FIR filter of moderate length. As the filter length increases, the number of bits per coefficient must be increased to maintain the same error in the frequency response characteristic of the filter.

For example, suppose that each filter coefficient is rounded to $(b + 1)$ bits. Then the maximum error in a coefficient value is bounded as

$$-2^{-(b+1)} < e_h(n) < 2^{-(b+1)}$$

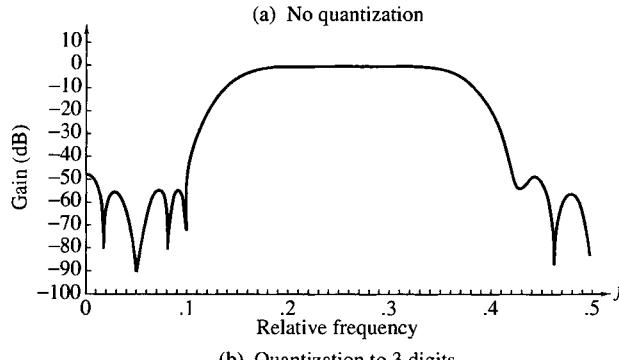
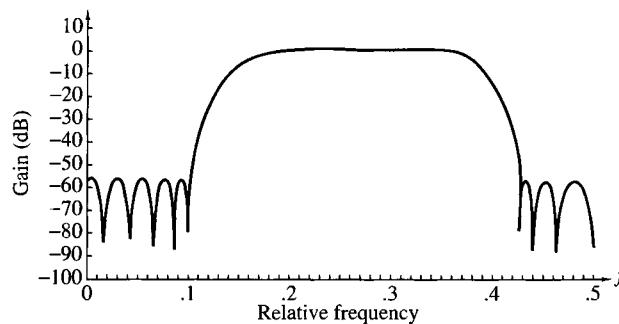


Figure 9.5.8
Effect of coefficient quantization of the magnitude of an $M = 32$ linear-phase FIR filter realized in direct form:
(a) no quantization;
(b) quantization to three digits.

Since the quantized values may be represented as $\bar{h}(n) = h(n) + e_h(n)$, the error in the frequency response is

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

Since $e_h(n)$ is zero mean, it follows that $E_M(\omega)$ is also zero mean. Assuming that the coefficient error sequence $e_h(n)$, $0 \leq n \leq M - 1$, is uncorrelated, the variance of the error $E_M(\omega)$ in the frequency response is just the sum of the variances of the M terms. Thus we have

$$\sigma_E^2 = \frac{2^{-2(b+1)}}{12} M = \frac{2^{-2(b+2)}}{3} M$$

Here we note that the variance of the error in $H(\omega)$ increases linearly with M . Hence the standard deviation of the error in $H(\omega)$ is

$$\sigma_E = \frac{2^{-(b+2)}}{\sqrt{3}} \sqrt{M}$$

Consequently, for every factor-of-4 increase in M , the precision in the filter coefficients must be increased by one additional bit to maintain the standard deviation fixed. This result, taken together with the results of Example 9.5.3, implies that the frequency error remains tolerable for filter lengths up to 256, provided that filter coefficients are represented by 12 to 13 bits. If the word length of the digital signal processor is less than 12 bits or if the filter length exceeds 256, the filter should be implemented as a cascade of smaller length filters to reduce the precision requirements.

In a cascade realization of the form

$$H(z) = G \prod_{k=1}^K H_k(z) \quad (9.5.18)$$

where the second-order sections are given as

$$H_k(z) = 1 + b_{k1}z^{-1} + b_{k2}z^{-2} \quad (9.5.19)$$

the coefficients of complex-valued zeros are expressed as $b_{k1} = -2r_k \cos \theta_k$ and $b_{k2} = r_k^2$. Quantization of b_{k1} and b_{k2} results in zero locations as shown in Fig. 9.5.3, except that the grid extends to points outside the unit circle.

A problem may arise, in this case, in maintaining the linear-phase property, because the quantized pair of zeros at $z = (1/r_k)e^{\pm j\theta_k}$ may not be the mirror image of the quantized zeros at $z = r_k e^{\pm j\theta_k}$. This problem can be avoided by rearranging the factors corresponding to the mirror-image zero. That is, we can write the mirror-image factor as

$$\left(1 - \frac{2}{r_k} \cos \theta_k z^{-1} + \frac{1}{r_k^2} z^{-2}\right) = \frac{1}{r_k^2} (r_k^2 - 2r_k \cos \theta_k z^{-1} + z^{-2}) \quad (9.5.20)$$

The factors $\{1/r_k^2\}$ can be combined with the overall gain factor G , or they can be distributed in each of the second-order filters. The factor in (9.5.20) contains exactly the same parameters as the factor $(1 - 2r_k \cos \theta_k z^{-1} + r_k^2 z^{-2})$, and consequently, the zeros now occur in mirror-image pairs even when the parameters are quantized.

In this brief treatment we have given the reader an introduction to the problems of coefficient quantization in IIR and FIR filters. We have demonstrated that a high-order filter should be reduced to a cascade (for FIR or IIR filters) or a parallel (for IIR filters) realization to minimize the effects of quantization errors in the coefficients. This is especially important in fixed-point realizations in which the coefficients are represented by a relatively small number of bits.

9.6 Round-Off Effects in Digital Filters

In Section 9.4 we characterized the quantization errors that occur in arithmetic operations performed in a digital filter. The presence of one or more quantizers in the realization of a digital filter results in a nonlinear device with characteristics that may be significantly different from the ideal linear filter. For example, a recursive digital filter may exhibit undesirable oscillations in its output, as shown in the following section, even in the absence of an input signal.

As a result of the finite-precision arithmetic operations performed in the digital filter, some registers may overflow if the input signal level becomes large. Overflow represents another form of undesirable nonlinear distortion on the desired signal at the output of the filter. Consequently, special care must be exercised to scale the input signal properly, either to prevent overflow completely or, at least, to minimize its rate of occurrence.

The nonlinear effects due to finite-precision arithmetic make it extremely difficult to precisely analyze the performance of a digital filter. To perform an analysis of quantization effects, we adopt a statistical characterization of quantization errors which, in effect, results in a linear model for the filter. Thus we are able to quantify the effects of quantization errors in the implementation of digital filters. Our treatment is limited to fixed-point realizations where quantization effects are very important.

9.6.1 Limit-Cycle Oscillations in Recursive Systems

In the realization of a digital filter, either in digital hardware or in software on a digital computer, the quantization inherent in the finite-precision arithmetic operations renders the system nonlinear. In recursive systems, the nonlinearities due to the finite-precision arithmetic operations often cause periodic oscillations to occur in the output, even when the input sequence is zero or some nonzero constant value. Such oscillations in recursive systems are called *limit cycles* and are directly attributable to round-off errors in multiplication and overflow errors in addition.

To illustrate the characteristics of a limit-cycle oscillation, let us consider a single-pole system described by the linear difference equation

$$y(n) = ay(n-1) + x(n) \quad (9.6.1)$$

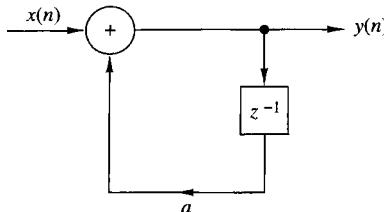


Figure 9.6.1
Ideal single-pole recursive system.

where the pole is at $z = a$. The ideal system is realized as shown in Fig. 9.6.1. On the other hand, the actual system, which is described by the nonlinear difference equation

$$v(n) = Q[av(n - 1)] + x(n) \quad (9.6.2)$$

is realized as shown in Fig. 9.6.2.

Suppose that the actual system in Fig. 9.6.2 is implemented with fixed-point arithmetic based on four bits for the magnitude plus a sign bit. The quantization that takes place after multiplication is assumed to round the resulting product upward.

In Table 9.2 we list the response of the actual system for four different locations of the pole $z = a$, and an input $x(n) = \beta\delta(n)$, where $\beta = 15/16$, which has the binary representation 0.1111. Ideally, the response of the system should decay toward zero exponentially [i.e., $y(n) = a^n \rightarrow 0$ as $n \rightarrow \infty$]. In the actual system, however, the response $v(n)$ reaches a steady-state periodic output sequence with a period that depends on the value of the pole. When the pole is positive, the oscillations occur with a period $N_p = 1$, so that the output reaches a constant value of $\frac{1}{16}$ for $a = \frac{1}{2}$ and $\frac{1}{8}$ for $a = \frac{3}{4}$. On the other hand, when the pole is negative, the output sequence oscillates between positive and negative values ($\pm \frac{1}{16}$ for $a = -\frac{1}{2}$ and $\pm \frac{1}{8}$ for $a = -\frac{3}{4}$). Hence the period is $N_p = 2$.

These limit cycles occur as a result of the quantization effects in multiplications. When the input sequence $x(n)$ to the filter becomes zero, the output of the filter then, after a number of iterations, enters into the limit cycle. The output remains in the limit cycle until another input of sufficient size is applied that drives the system out of the limit cycle. Similarly, zero-input limit cycles occur from nonzero initial conditions with the input $x(n) = 0$. The amplitudes of the output during a limit cycle are confined to a range of values that is called the *dead band* of the filter.

It is interesting to note that when the response of the single-pole filter is in the limit cycle, the actual nonlinear system operates as an equivalent linear system with

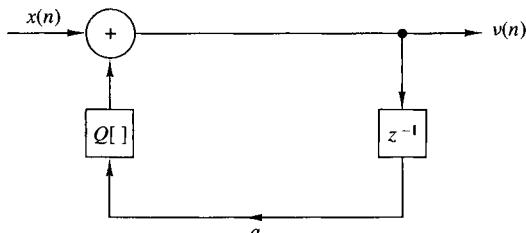


Figure 9.6.2
Actual nonlinear system.

TABLE 9.2 Limit Cycles for Lowpass Single-Pole Filter

n	$a = 0.1000 = \frac{1}{2}$	$a = 1.1000 = -\frac{1}{2}$	$a = 0.1100 = \frac{3}{4}$	$a = 1.1100 = -\frac{3}{4}$
0	0.1111 ($\frac{15}{16}$)	0.1111 ($\frac{15}{16}$)	0.1011 ($\frac{11}{16}$)	0.1011 ($\frac{11}{16}$)
1	0.1000 ($\frac{8}{16}$)	1.1000 ($-\frac{8}{16}$)	0.1000 ($\frac{8}{16}$)	1.1000 ($-\frac{8}{16}$)
2	0.0100 ($\frac{4}{16}$)	0.0100 ($\frac{4}{16}$)	0.0110 ($\frac{6}{16}$)	0.0110 ($\frac{6}{16}$)
3	0.0010 ($\frac{2}{16}$)	1.0010 ($-\frac{2}{16}$)	0.0101 ($\frac{5}{16}$)	1.0101 ($-\frac{5}{16}$)
4	0.0001 ($\frac{1}{16}$)	0.0001 ($\frac{1}{16}$)	0.0100 ($\frac{4}{16}$)	0.0100 ($\frac{4}{16}$)
5	0.0001 ($\frac{1}{16}$)	1.0001 ($-\frac{1}{16}$)	0.0011 ($\frac{3}{16}$)	1.0011 ($-\frac{3}{16}$)
6	0.0001 ($\frac{1}{16}$)	0.0001 ($\frac{1}{16}$)	0.0010 ($\frac{2}{16}$)	0.0010 ($\frac{2}{16}$)
7	0.0001 ($\frac{1}{16}$)	1.0001 ($-\frac{1}{16}$)	0.0010 ($\frac{2}{16}$)	1.0010 ($-\frac{2}{16}$)
8	0.0001 ($\frac{1}{16}$)	0.0001 ($\frac{1}{16}$)	0.0010 ($\frac{2}{16}$)	0.0010 ($\frac{2}{16}$)

a pole at $z = 1$ when the pole is positive and $z = -1$ when the pole is negative. That is,

$$Q_r[av(n-1)] = \begin{cases} v(n-1), & a > 0 \\ -v(n-1), & a < 0 \end{cases} \quad (9.6.3)$$

Since the quantized product $av(n-1)$ is obtained by rounding, it follows that the quantization error is bounded as

$$|Q_r[av(n-1)] - av(n-1)| \leq \frac{1}{2} \cdot 2^{-b} \quad (9.6.4)$$

where b is the number of bits (exclusive of sign) used in the representation of the pole a and $v(n)$. Consequently, (9.6.4) and (9.6.3) lead to

$$|v(n-1)| - |Q_r[av(n-1)]| \leq \frac{1}{2} \cdot 2^{-b}$$

and hence

$$|v(n-1)| \leq \frac{\frac{1}{2} \cdot 2^{-b}}{1 - |a|} \quad (9.6.5)$$

The expression in (9.6.5) defines the dead band for a single-pole filter. For example, when $b = 4$ and $|a| = \frac{1}{2}$, we have a dead band with a range of amplitudes $(-\frac{1}{16}, \frac{1}{16})$. When $b = 4$ and $|a| = \frac{3}{4}$, the dead band increases to $(-\frac{1}{8}, \frac{1}{8})$.

The limit-cycle behavior in a two-pole filter is much more complex and a larger variety of oscillations can occur. In this case the ideal two-pole system is described by the linear difference equation,

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + x(n) \quad (9.6.6)$$

whereas the actual system is described by the nonlinear difference equation

$$v(n) = Q_r[a_1 v(n-1)] + Q_r[a_2 v(n-2)] + x(n) \quad (9.6.7)$$

When the filter coefficients satisfy the condition $a_1^2 < -4a_2$, the poles of the system occur at

$$z = re^{\pm j\theta}$$

where $a_2 = -r^2$ and $a_1 = 2r \cos \theta$. As in the case of the single-pole filter, when the system is in a zero-input or zero-state limit cycle,

$$Q_r[a_2v(n-2)] = -v(n-2) \quad (9.6.8)$$

In other words, the system behaves as an oscillator with complex-conjugate poles on the unit circle (i.e., $a_2 = -r^2 = -1$). Rounding the product $a_2v(n-2)$ implies that

$$|Q_r[a_2v(n-2)] - a_2v(n-2)| \leq \frac{1}{2} \cdot 2^{-b} \quad (9.6.9)$$

Upon substitution of (9.6.8) into (9.6.9), we obtain the result

$$|v(n-2)| - |a_2v(n-2)| \leq \frac{1}{2} \cdot 2^{-b}$$

or equivalently,

$$|v(n-2)| \leq \frac{\frac{1}{2} \cdot 2^{-b}}{1 - |a_2|} \quad (9.6.10)$$

The expression in (9.6.10) defines the dead band of the two-pole filter with complex-conjugate poles. We observe that the dead-band limits depend only on $|a_2|$. The parameter $a_1 = 2r \cos \theta$ determines the frequency of oscillation.

Another possible limit-cycle mode with zero input, which occurs as a result of rounding the multiplications, corresponds to an equivalent second-order system with poles at $z = \pm 1$. In this case it was shown by Jackson (1969) that the two-pole filter exhibits oscillations with an amplitude that falls in the dead band bounded by $2^{-b}/(1 - |a_1| - a_2)$.

It is interesting to note that these limit cycles result from rounding the product of the filter coefficients with the previous outputs, $v(n-1)$ and $v(n-2)$. Instead of rounding, we may choose to truncate the products to b bits. With truncation, we can eliminate many, although not all, of the limit cycles as shown by Claasen et al. (1973). However, recall that truncation results in a biased error unless the sign-magnitude representation is used, in which case the truncation error is symmetric about zero. In general, this bias is undesirable in digital filter implementation.

In a parallel realization of a high-order IIR system, each second-order filter section exhibits its own limit-cycle behavior, with no interaction among the second-order filter sections. Consequently, the output is the sum of the zero-input limit cycles from the individual sections. In the case of a cascade realization for a high-order IIR system, the limit cycles are much more difficult to analyze. In particular, when the first filter section exhibits a zero-input limit cycle, the output limit cycle is filtered by the succeeding sections. If the frequency of the limit cycle falls near a resonance frequency in a succeeding filter section, the amplitude of the sequence is enhanced by the resonance characteristic. In general, we must be careful to avoid such situations.

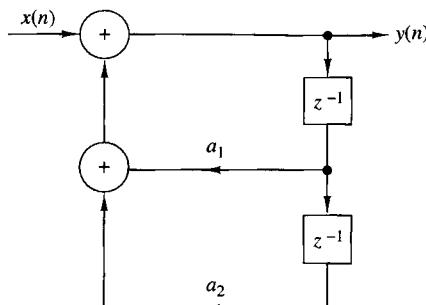


Figure 9.6.3
Two-pole filter realization.

In addition to limit cycles caused by rounding the result of multiplications, there are limit cycles caused by overflows in addition. An overflow in addition of two or more binary numbers occurs when the sum exceeds the word size available in the digital implementation of the system. For example, let us consider the second-order filter section illustrated in Fig. 9.6.3, in which the addition is performed in two's-complement arithmetic. Thus we can write the output $y(n)$ as

$$y(n) = g[a_1y(n-1) + a_2y(n-2) + x(n)] \quad (9.6.11)$$

where the function $g[\cdot]$ represents the two's-complement addition. It is easily verified that the function $g(v)$ versus v is described by the graph in Fig. 9.6.4.

Recall that the range of values of the parameters (a_1, a_2) for a stable filter is given by the stability triangle in Fig. 3.5.1. However, these conditions are no longer sufficient to prevent overflow oscillation with two's-complement arithmetic. In fact, it can easily be shown that a necessary and sufficient condition for ensuring that no zero-input overflow limit cycles occur is

$$|a_1| + |a_2| < 1 \quad (9.6.12)$$

which is extremely restrictive and hence an unreasonable constraint to impose on any second-order section.

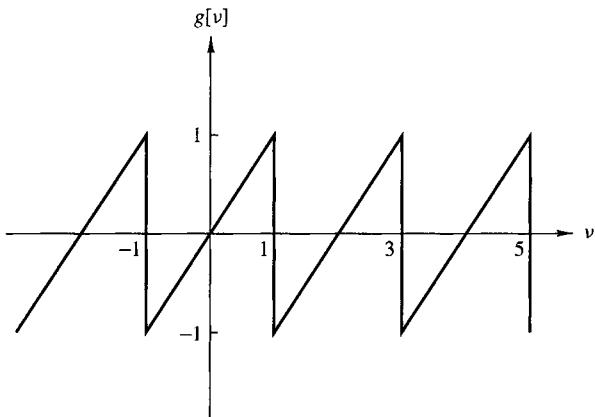


Figure 9.6.4
Characteristic functional relationship for two's-complement addition of two or more numbers.

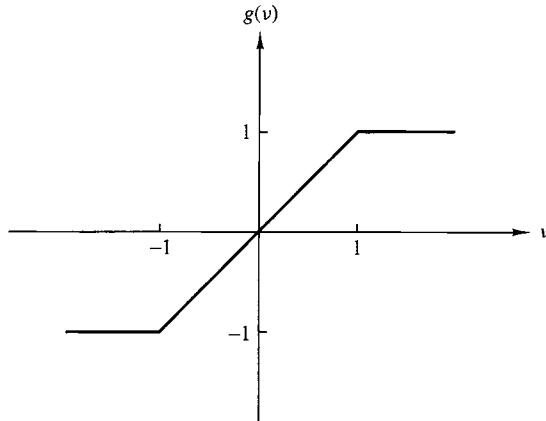


Figure 9.6.5
Characteristic functional relationship for addition with clipping at ± 1 .

An effective remedy for curing the problem of overflow oscillations is to modify the adder characteristic, as illustrated in Fig. 9.6.5, so that it performs saturation arithmetic. Thus when an overflow (or underflow) is sensed, the output of the adder will be the full-scale value of ± 1 . The distortion caused by this nonlinearity in the adder is usually small, provided that saturation occurs infrequently. The use of such a nonlinearity does not preclude the need for scaling of the signals and the system parameters, as described in the following section.

9.6.2 Scaling to Prevent Overflow

Saturation arithmetic as just described eliminates limit cycles due to overflow, on the one hand, but on the other hand, it causes undesirable signal distortion due to the nonlinearity of the clipper. In order to limit the amount of nonlinear distortion, it is important to scale the input signal and the unit sample response, between the input and any internal summing node in the system, such that overflow becomes a rare event.

For fixed-point arithmetic, let us first consider the extreme condition that overflow is not permitted at any node of the system. Let $y_k(n)$ denote the response of the system at the k th node when the input sequence is $x(n)$ and the unit sample response between the node and the input is $h_k(n)$. Then

$$|y_k(n)| = \left| \sum_{m=-\infty}^{\infty} h_k(m)x(n-m) \right| \leq \sum_{m=-\infty}^{\infty} |h_k(m)||x(n-m)|$$

Suppose that $x(n)$ is upper bounded by A_x . Then

$$|y_k(n)| \leq A_x \sum_{m=-\infty}^{\infty} |h_k(m)|, \quad \text{for all } n \quad (9.6.13)$$

Now, if the dynamic range of the computer is limited to $(-1, 1)$, the condition

$$|y_k(n)| < 1$$

can be satisfied by requiring that the input $x(n)$ be scaled such that

$$A_x < \frac{1}{\sum_{m=-\infty}^{\infty} |h_k(m)|} \quad (9.6.14)$$

for all possible nodes in the system. The condition in (9.6.14) is both necessary and sufficient to prevent overflow.

The condition in (9.6.14) is overly conservative, however, to the point where the input signal may be scaled too much. In such a case, much of the precision used to represent $x(n)$ is lost. This is especially true for narrowband sequences, such as sinusoids, where the scaling implied by (9.6.14) is extremely severe. For narrowband signals we can use the frequency response characteristics of the system in determining the appropriate scaling. Since $|H(\omega)|$ represents the gain of the system at frequency ω , a less severe and reasonably adequate scaling is to require that

$$A_x < \frac{1}{\max_{0 \leq \omega \leq \pi} |H_k(\omega)|} \quad (9.6.15)$$

where $H_k(\omega)$ is the Fourier transform of $\{h_k(n)\}$.

In the case of an FIR filter, the condition in (9.6.14) reduces to

$$A_x < \frac{1}{\sum_{m=0}^{M-1} |h_k(m)|} \quad (9.6.16)$$

which is now a sum over the M nonzero terms of the filter unit sample response.

Another approach to scaling is to scale the input so that

$$\sum_{n=-\infty}^{\infty} |y_k(n)|^2 \leq C^2 \sum_{n=-\infty}^{\infty} |x(n)|^2 = C^2 E_x \quad (9.6.17)$$

From Parseval's theorem we have

$$\begin{aligned} \sum_{n=-\infty}^{\infty} |y_k(n)|^2 &= \frac{1}{2\pi} \int_{-\pi}^{\pi} |H(\omega)X(\omega)|^2 d\omega \\ &\leq E_x \frac{1}{2\pi} \int_{-\pi}^{\pi} |H(\omega)|^2 d\omega \end{aligned} \quad (9.6.18)$$

By combining (9.6.17) with (9.6.18), we obtain

$$C^2 \leq \frac{1}{\sum_{n=-\infty}^{\infty} |h_k(n)|^2} = \frac{1}{(1/2\pi) \int_{-\pi}^{\pi} |H(\omega)|^2 d\omega} \quad (9.6.19)$$

If we compare the different scaling factors given above, we find that

$$\left[\sum_{n=-\infty}^{\infty} |h_k(n)|^2 \right]^{1/2} \leq \max_{\omega} |H_k(\omega)| \leq \sum_{n=-\infty}^{\infty} |h_k(n)| \quad (9.6.20)$$

Clearly, (9.6.14) is the most pessimistic constraint.

In the following section we observe the ramifications of this scaling on the output signal-to-noise (power) ratio (SNR) from a first-order and a second-order filter section.

9.6.3 Statistical Characterization of Quantization Effects in Fixed-Point Realizations of Digital Filters

It is apparent from our treatment in the previous section that an analysis of quantization errors in digital filtering, based on deterministic models of quantization effects, is not a very fruitful approach. The basic problem is that the nonlinear effects in quantizing the products of two numbers and in clipping the sum of two numbers to prevent overflow are not easily modeled in large systems that contain many multipliers and many summing nodes.

To obtain more general results on the quantization effects in digital filters, we shall model the quantization errors in multiplication as an additive noise sequence $e(n)$, just as we did in characterizing the quantization errors in A/D conversion of an analog signal. For addition, we consider the effect of scaling the input signal to prevent overflow.

Let us begin our treatment with the characterization of the round-off noise in a single-pole filter which is implemented in fixed-point arithmetic and is described by the nonlinear difference equation

$$v(n) = Q_r[av(n - 1)] + x(n) \quad (9.6.21)$$

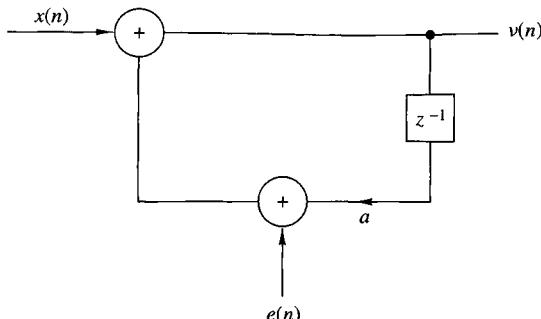
The effect of rounding the product $av(n - 1)$ is modeled as a noise sequence $e(n)$ added to the actual product $av(n - 1)$, that is,

$$Q_r[av(n - 1)] = av(n - 1) + e(n) \quad (9.6.22)$$

With this model for the quantization error, the system under consideration is described by the *linear difference equation*

$$v(n) = av(n - 1) + x(n) + e(n) \quad (9.6.23)$$

The corresponding system is illustrated in block diagram form in Fig. 9.6.6.

**Figure 9.6.6**

Additive noise model for the quantization error in a single-pole filter.

It is apparent from (9.6.23) that the output sequence $v(n)$ of the filter can be separated into two components. One is the response of the system to the input sequence $x(n)$. The second is the response of the system to the additive quantization noise $e(n)$. In fact, we can express the output sequence $v(n)$ as a sum of these two components, that is,

$$v(n) = y(n) + q(n) \quad (9.6.24)$$

where $y(n)$ represents the response of the system to $x(n)$, and $q(n)$ represents the response of the system to the quantization error $e(n)$. Upon substitution from (9.6.24) for $v(n)$ into (9.6.23), we obtain

$$y(n) + q(n) = ay(n - 1) + aq(n - 1) + x(n) + e(n) \quad (9.6.25)$$

To simplify the analysis, we make the following assumptions about the error sequence $e(n)$.

- For any n , the error sequence $\{e(n)\}$ is uniformly distributed over the range $(-\frac{1}{2} \cdot 2^{-b}, \frac{1}{2} \cdot 2^{-b})$. This implies that the mean value of $e(n)$ is zero and its variance is

$$\sigma_e^2 = \frac{2^{-2b}}{12} \quad (9.6.26)$$

- The error $\{e(n)\}$ is a stationary white noise sequence. In other words, the error $e(n)$ and the error $e(m)$ are uncorrelated for $n \neq m$.
- The error sequence $\{e(n)\}$ is uncorrelated with the signal sequence $\{x(n)\}$.

The last assumption allows us to separate the difference equation in (9.6.25) into two uncoupled difference equations, namely,

$$y(n) = ay(n - 1) + x(n) \quad (9.6.27)$$

$$q(n) = aq(n - 1) + e(n) \quad (9.6.28)$$

The difference equation in (9.6.27) represents the input-output relation for the desired system and the difference equation in (9.6.28) represents the relation for the quantization error at the output of the system.

To complete the analysis, we make use of two important relationships developed in Section 12.1. The first is the relationship for the mean value of the output $q(n)$ of a linear shift-invariant filter with impulse response $h(n)$ when excited by a random sequence $e(n)$ having a mean value m_e . The result is

$$m_q = m_e \sum_{n=-\infty}^{\infty} h(n) \quad (9.6.29)$$

or, equivalently,

$$m_q = m_e H(0) \quad (9.6.30)$$

where $H(0)$ is the value of the frequency response $H(\omega)$ of the filter evaluated at $\omega = 0$.

The second important relationship is the expression for the autocorrelation sequence of the output $q(n)$ of the filter with impulse response $h(n)$ when the input random sequence $e(n)$ has an autocorrelation $\gamma_{ee}(n)$. This result is

$$\gamma_{qq}(n) = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} h(k)h(l)\gamma_{ee}(k-l+n) \quad (9.6.31)$$

In the important special case where the random sequence is white (spectrally flat), the autocorrelation $\gamma_{ee}(n)$ is a unit sample sequence scaled by the variance σ_e^2 , that is,

$$\gamma_{ee}(n) = \sigma_e^2 \delta(n) \quad (9.6.32)$$

Upon substituting (9.6.32) into (9.6.31), we obtain the desired result for the autocorrelation sequence at the output of a filter excited by white noise, namely,

$$\gamma_{qq}(n) = \sigma_e^2 \sum_{k=-\infty}^{\infty} h(k)h(k+n) \quad (9.6.33)$$

The variance σ_q^2 of the output noise is simply obtained by evaluating $\gamma_{qq}(n)$ at $n = 0$. Thus

$$\sigma_q^2 = \sigma_e^2 \sum_{k=-\infty}^{\infty} h^2(k) \quad (9.6.34)$$

and with the aid of Parseval's theorem, we have the alternative expression

$$\sigma_q^2 = \frac{\sigma_e^2}{2\pi} \int_{-\pi}^{\pi} |H(\omega)|^2 d\omega \quad (9.6.35)$$

In the case of the single-pole filter under consideration, the unit sample response is

$$h(n) = a^n u(n) \quad (9.6.36)$$

Since the quantization error due to rounding has zero mean, the mean value of the error at the output of the filter is $m_q = 0$. The variance of the error at the output of the filter is

$$\begin{aligned}\sigma_q^2 &= \sigma_e^2 \sum_{k=0}^{\infty} a^{2k} \\ &= \frac{\sigma_e^2}{1 - a^2}\end{aligned}\tag{9.6.37}$$

We observe that the noise power σ_q^2 at the output of the filter is enhanced relative to the input noise power σ_e^2 by the factor $1/(1 - a^2)$. This factor increases as the pole is moved closer to the unit circle.

To obtain a clearer picture of the effect of the quantization error, we should also consider the effect of scaling the input. Let us assume that the input sequence $\{x(n)\}$ is a white noise sequence (wideband signal), whose amplitude has been scaled according to (9.6.14) to prevent overflows in addition. Then

$$A_x < 1 - |a|$$

If we assume that $x(n)$ is uniformly distributed in the range $(-A_x, A_x)$, then, according to (9.6.31) and (9.6.34), the signal power at the output of the filter is

$$\begin{aligned}\sigma_y^2 &= \sigma_x^2 \sum_{k=0}^{\infty} a^{2k} \\ &= \frac{\sigma_x^2}{1 - a^2}\end{aligned}\tag{9.6.38}$$

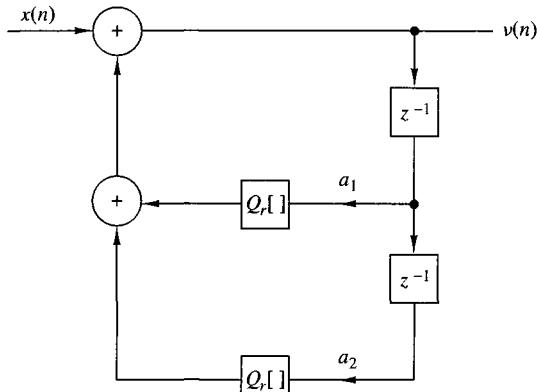
where $\sigma_x^2 = (1 - |a|)^2/3$ is the variance of the input signal. The ratio of the signal power σ_y^2 to the quantization error power σ_q^2 , which is called the signal-to-noise ratio (SNR), is simply

$$\begin{aligned}\frac{\sigma_y^2}{\sigma_q^2} &= \frac{\sigma_x^2}{\sigma_e^2} \\ &= (1 - |a|)^2 \cdot 2^{2(b+1)}\end{aligned}\tag{9.6.39}$$

This expression for the output SNR clearly illustrates the severe penalty paid as a consequence of the scaling of the input, especially when the pole is near the unit circle. By comparison, if the input is not scaled and the adder has a sufficient number of bits to avoid overflow, then the signal amplitude may be confined to the range $(-1, 1)$. In this case, $\sigma_x^2 = \frac{1}{3}$, which is independent of the pole position. Then

$$\frac{\sigma_y^2}{\sigma_q^2} = 2^{2(b+1)}\tag{9.6.40}$$

The difference between the SNRs in (9.6.40) and (9.6.39) clearly demonstrates the need to use more bits in addition than in multiplication. The number of additional

**Figure 9.6.7**

Two-pole digital filter with rounding quantizers.

bits depends on the position of the pole and should be increased as the pole is moved closer to the unit circle.

Next, let us consider a two-pole filter with infinite precision which is described by the linear difference equation

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + x(n) \quad (9.6.41)$$

where $a_1 = 2r \cos \theta$ and $a_2 = -r^2$. When the two products are rounded, we have a system which is described by the nonlinear difference equation

$$v(n) = Q_r[a_1 v(n-1)] + Q_r[a_2 v(n-2)] + x(n) \quad (9.6.42)$$

This system is illustrated in block diagram form in Fig. 9.6.7.

Now there are two multiplications, and hence two quantization errors are produced for each output. Consequently, we should introduce two noise sequences $e_1(n)$ and $e_2(n)$, which correspond to the quantizer outputs

$$\begin{aligned} Q_r[a_1 v(n-1)] &= a_1 v(n-1) + e_1(n) \\ Q_r[a_2 v(n-2)] &= a_2 v(n-2) + e_2(n) \end{aligned} \quad (9.6.43)$$

A block diagram for the corresponding model is shown in Fig. 9.6.8. Note that the error sequences $e_1(n)$ and $e_2(n)$ can be moved directly to the input of the filter.

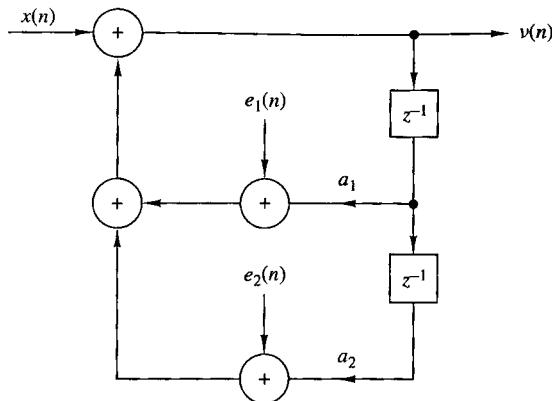
As in the case of the first-order filter, the output of the second-order filter can be separated into two components, the desired signal component and the quantization error component. The former is described by the difference equation

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + x(n) \quad (9.6.44)$$

while the latter satisfies the difference equation

$$q(n) = a_1 q(n-1) + a_2 q(n-2) + e_1(n) + e_2(n) \quad (9.6.45)$$

It is reasonable to assume that the two sequences $e_1(n)$ and $e_2(n)$ are uncorrelated.

**Figure 9.6.8**

Additive noise model for the quantization errors in a two-pole filter realization.

Now the second-order filter has a unit sample response

$$h(n) = \frac{r^n}{\sin \theta} \sin(n+1)\theta u(n) \quad (9.6.46)$$

Hence

$$\sum_{n=0}^{\infty} h^2(n) = \frac{1+r^2}{1-r^2} \frac{1}{r^4 + 1 - 2r^2 \cos 2\theta} \quad (9.6.47)$$

By applying (9.6.34), we obtain the variance of the quantization errors at the output of the filter in the form

$$\sigma_q^2 = \sigma_e^2 \left(\frac{1+r^2}{1-r^2} \frac{1}{r^4 + 1 - 2r^2 \cos 2\theta} \right) \quad (9.6.48)$$

In the case of the signal component, if we scale the input as in (9.6.14) to avoid overflow, the power in the output signal is

$$\sigma_y^2 = \sigma_x^2 \sum_{n=0}^{\infty} h^2(n) \quad (9.6.49)$$

where the power in the input signal $x(n)$ is given by the variance

$$\sigma_x^2 = \frac{1}{3 \left[\sum_{n=0}^{\infty} |h(n)| \right]^2} \quad (9.6.50)$$

Consequently, the SNR at the output of the two-pole filter is

$$\frac{\sigma_y^2}{\sigma_q^2} = \frac{\sigma_x^2}{\sigma_e^2} = \frac{2^{2(b+1)}}{\left[\sum_{n=0}^{\infty} |h(n)| \right]^2} \quad (9.6.51)$$

Although it is difficult to determine the exact value of the denominator term in (9.6.51), it is easy to obtain an upper and a lower bound. In particular, $|h(n)|$ is upper bounded as

$$|h(n)| \leq \frac{1}{\sin \theta} r^n, \quad n \geq 0 \quad (9.6.52)$$

so that

$$\sum_{n=0}^{\infty} |h(n)| \leq \frac{1}{\sin \theta} \sum_{n=0}^{\infty} r^n = \frac{1}{(1-r) \sin \theta} \quad (9.6.53)$$

The lower bound may be obtained by noting that

$$|H(\omega)| = \left| \sum_{n=0}^{\infty} h(n) e^{-j\omega n} \right| \leq \sum_{n=0}^{\infty} |h(n)|$$

But

$$H(\omega) = \frac{1}{(1 - re^{j\theta} e^{-j\omega})(1 - re^{-j\theta} e^{-j\omega})}$$

At $\omega = \theta$, which is the resonant frequency of the filter, we obtain the largest value of $|H(\omega)|$. Hence

$$\sum_{n=0}^{\infty} |h(n)| \geq |H(\theta)| = \frac{1}{(1-r)\sqrt{1+r^2-2r \cos 2\theta}} \quad (9.6.54)$$

Therefore, the SNR is bounded from above and below according to the relation

$$2^{2(b+1)}(1-r)^2 \sin^2 \theta \leq \frac{\sigma_y^2}{\sigma_q^2} \leq 2^{2(b+1)}(1-r)^2(1+r^2-2r \cos 2\theta) \quad (9.6.55)$$

For example, when $\theta = \pi/2$, the expression in (9.6.55) reduces to

$$2^{2(b+1)}(1-r)^2 \leq \frac{\sigma_y^2}{\sigma_q^2} \leq 2^{2(b+1)}(1-r)^2(1+r)^2 \quad (9.6.56)$$

The dominant term in this bound is $(1-r)^2$ which acts to reduce the SNR dramatically as the poles move toward the unit circle. Hence the effect of scaling in the second-order filter is more severe than in the single-pole filter. Note that if $d = 1-r$ is the distance of the pole from the unit circle, the SNR in (9.6.56) is reduced by d^2 , whereas in the single-pole filter the reduction is proportional to d . These results serve to reinforce the earlier statement regarding the use of more bits in addition than in multiplication as a mechanism for avoiding the severe penalty due to scaling.

The analysis of the quantization effects in a second-order filter can be applied directly to higher-order filters based on a parallel realization. In this case each second-order filter section is independent of all the other sections, and therefore the total

quantization noise power at the output of the parallel bank is simply the linear sum of the quantization noise powers of each of the individual sections. On the other hand, the cascade realization is more difficult to analyze. For the cascade interconnection, the noise generated in any second-order filter section is filtered by the succeeding sections. As a consequence, there is the issue of how to pair together real-valued poles to form second-order sections and how to arrange the resulting second-order filters to minimize the total noise power at the output of the high-order filter. This general topic was investigated by Jackson (1970a, b), who showed that poles close to the unit circle should be paired with nearby zeros to reduce the gain of each second-order section. In ordering the second-order sections in cascade, a reasonable strategy is to place the sections in the order of decreasing maximum frequency gain. In this case the noise power generated in the early high-gain section is not boosted significantly by the latter sections.

The following example illustrates the point that proper ordering of sections in a cascade realization is important in controlling the round-off noise at the output of the overall filter.

EXAMPLE 9.6.1

Determine the variance of the round-off noise at the output of the two cascade realizations of the filter with system function

$$H(z) = H_1(z)H_2(z)$$

where

$$H_1(z) = \frac{1}{1 - \frac{1}{2}z^{-1}}$$

$$H_2(z) = \frac{1}{1 - \frac{1}{4}z^{-1}}$$

Solution. Let $h(n)$, $h_1(n)$, and $h_2(n)$ represent the unit sample responses corresponding to the system functions $H(z)$, $H_1(z)$, and $H_2(z)$, respectively. It follows that

$$\begin{aligned} h_1(n) &= (\frac{1}{2})^n u(n), & h_2(n) &= (\frac{1}{4})^n u(n) \\ h(n) &= [2(\frac{1}{2})^n - (\frac{1}{4})^n]u(n) \end{aligned}$$

The two cascade realizations are shown in Fig. 9.6.9.

In the first cascade realization, the variance of the output is

$$\sigma_{q1}^2 = \sigma_e^2 \left[\sum_{n=0}^{\infty} h^2(n) + \sum_{n=0}^{\infty} h_2^2(n) \right]$$

In the second cascade realization, the variance of the output noise is

$$\sigma_{q2}^2 = \sigma_e^2 \left[\sum_{n=0}^{\infty} h^2(n) + \sum_{n=0}^{\infty} h_1^2(n) \right]$$

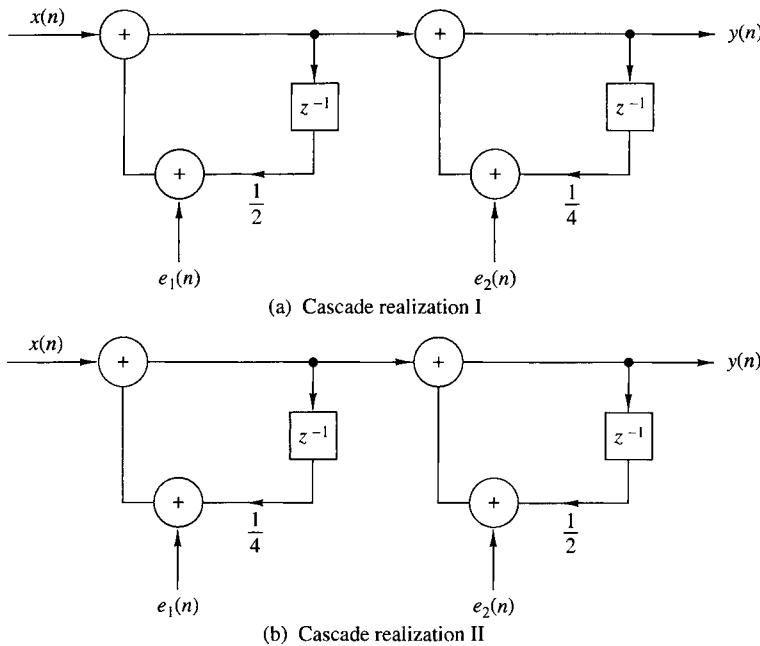


Figure 9.6.9 Two cascade realizations in Example 9.6.1: (a) cascade realization I; (b) cascade realization II.

Now

$$\sum_{n=0}^{\infty} h_1^2(n) = \frac{1}{1 - \frac{1}{4}} = \frac{4}{3}$$

$$\sum_{n=0}^{\infty} h_2^2(n) = \frac{1}{1 - \frac{1}{16}} = \frac{16}{15}$$

$$\sum_{m=0}^{\infty} h^2(n) = \frac{4}{1 - \frac{1}{4}} - \frac{4}{1 - \frac{1}{8}} + \frac{1}{1 - \frac{1}{16}} = 1.83$$

Therefore,

$$\sigma_{q1}^2 = 2.90\sigma_e^2$$

$$\sigma_{q2}^2 = 3.16\sigma_e^2$$

and the ratio of noise variances is

$$\frac{\sigma_{q2}^2}{\sigma_{q1}^2} = 1.09$$

Consequently, the noise power in the second cascade realization is 9% larger than in the first realization.

9.7 Summary and References

From the treatment in this chapter we have seen that there are various realizations of discrete-time systems. FIR systems can be realized in a direct form, a cascade form, a frequency sampling form, and a lattice form. IIR systems can also be realized in a direct form, a cascade form, a lattice or a lattice-ladder form, and a parallel form.

For any given system described by a linear constant-coefficient difference equation, these realizations are equivalent in that they represent the same system and produce the same output for any given input, provided that the internal computations are performed with infinite precision. However, the various structures are not equivalent when they are realized with finite-precision arithmetic.

Additional FIR and IIR filter structures can be obtained by adopting a state-space formulation that provides an internal description of a system. Such state-space realizations were treated in previous editions of this book, but have been dropped from this edition due to space limitations. The use of state-space filter structures in the realization of IIR systems has been proposed by Mullis and Roberts (1976a,b), and further developed by Hwang (1977), Jackson et al. (1979), Jackson (1979), Mills et al. (1981), and Bomar (1985).

Three important factors are presented for choosing among the various FIR and IIR system realizations. These factors are computational complexity, memory requirements, and finite-word-length effects. Depending on either the time-domain or the frequency-domain characteristics of a system, some structures may require less computation and/or less memory than others. Hence our selection must consider these two important factors.

In deriving the transposed structures in Section 9.3, we introduced several concepts and operations on signal flow graphs. Signal flow graphs are treated in depth in the books by Mason and Zimmerman (1960) and Chow and Cassignol (1962).

Another important structure for IIR systems, a *wave digital filter*, has been investigated by Fettweis (1971) and further developed by Sedlmeyer and Fettweis (1973). A treatment of this filter structure can also be found in the book by Antoniou (1979).

Finite-word-length effects are an important factor in the implementation of digital signal processing systems. In this chapter we described the effects of a finite word length in digital filtering. In particular, we considered the following problems dealing with finite-word length effects:

- 1.** Parameter quantization in digital filters
- 2.** Round-off noise in multiplication
- 3.** Overflow in addition
- 4.** Limit cycles

These four effects are internal to the filter and influence the method by which the system will be implemented. In particular, we demonstrated that high-order systems, especially IIR systems, should be realized by using second-order sections as building blocks. We advocated the use of the direct form II realization, either the conventional or the transposed form.

Effects of round-off errors in fixed-point implementations of FIR and IIR filter structures have been investigated by many researchers. We cite the papers by Gold and Rader (1966), Rader and Gold (1967b), Jackson (1970a,b), Liu (1971), Chan and Rabiner (1973a,b,c), and Oppenheim and Weinstein (1972).

Limit-cycle oscillations occur in IIR filters as a result of quantization effects in fixed-point multiplication and rounding. Investigation of limit cycles in digital filtering and their characteristic behavior is treated in the papers by Parker and Hess (1971), Brubaker and Gowdy (1972), Sandberg and Kaiser (1972), and Jackson (1969, 1979). The latter paper deals with limit cycles in state-space structures. Methods have also been devised to eliminate limit cycles caused by round-off errors. For example, the papers by Barnes and Fam (1977), Fam and Barnes (1979), Chang (1981), Butterweck et al. (1984), and Auer (1987) discuss this problem. Overflow oscillations have been treated in the paper by Ebert et al. (1969).

The effects of parameter quantization have been treated in a number of papers. We cite for reference the work of Rader and Gold (1967b), Knowles and Olcayto (1968), Avenhaus and Schuessler (1970), Herrmann and Schuessler (1970b), Chan and Rabiner (1973c), and Jackson (1976).

Finally, we mention that the lattice and lattice-ladder filter structures are known to be robust in fixed-point implementations. For a treatment of these types of filters, the reader is referred to the papers of Gray and Markel (1973), Makhoul (1978), and Morf et al. (1977) and to the book by Markel and Gray (1976).

Problems

- 1** Determine a direct-form realization for the following linear phase filters.

(a) $h(n) = \{1, 2, 3, 4, 3, 2, 1\}$

(b) $h(n) = \{1, 2, 3, 3, 2, 1\}$

- 2** Consider an FIR filter with system function

$$H(z) = 1 + 2.88z^{-1} + 3.4048z^{-2} + 1.74z^{-3} + 0.4z^{-4}$$

Sketch the direct-form and lattice realizations of the filter and determine in detail the corresponding input-output equations. Is the system minimum phase?

- 9.3** Determine the system function and the impulse response of the system shown in Fig. P9.3.

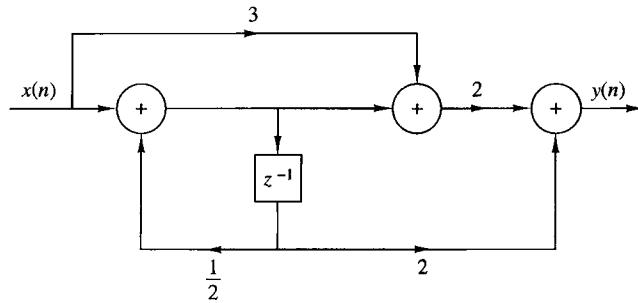


Figure P9.3

- 9.4** Determine the system function and the impulse response of the system shown in Fig. P9.4.

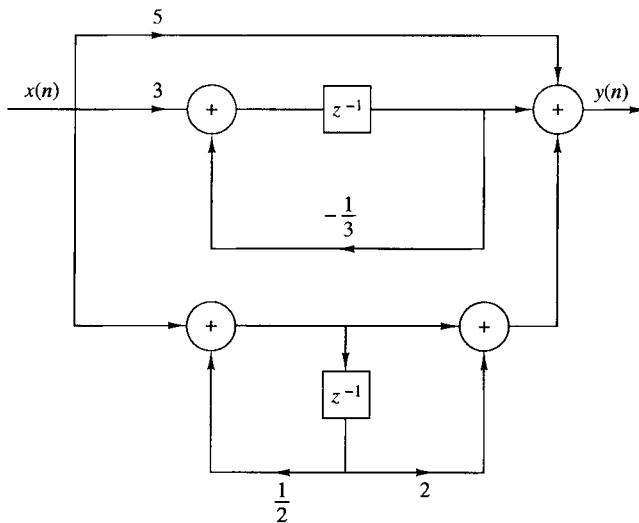


Figure P9.4

- 9.5** Determine the transposed structure of the system in Fig. P9.4 and verify that both the original and the transposed system have the same system function.

- 9.6** Determine a_1 , a_2 and c_1 , and c_0 in terms of b_1 and b_2 so that the two systems in Fig. P9.6 are equivalent.

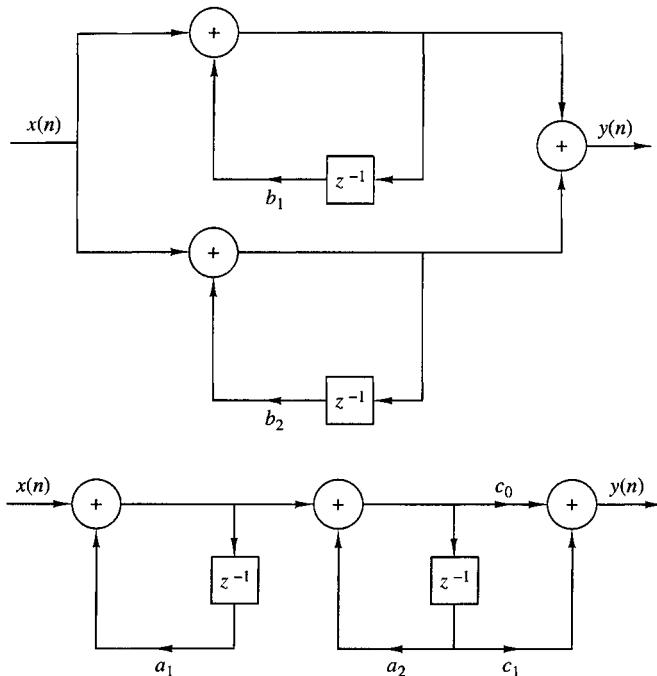


Figure P9.6

- 9.7** Consider the filter shown in Fig. P9.7.

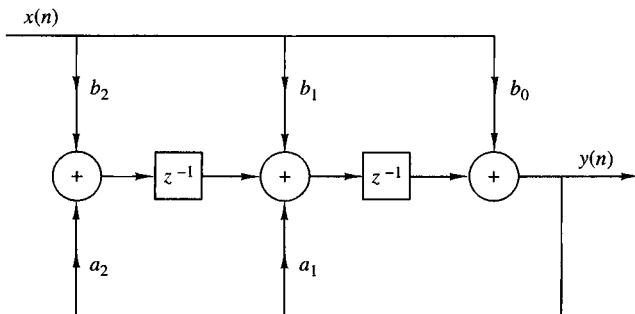


Figure P9.7

- (a) Determine its system function.
- (b) Sketch the pole-zero plot and check for stability if
 1. $b_0 = b_2 = 1, \quad b_1 = 2, \quad a_1 = 1.5, a_2 = -0.9$
 2. $b_0 = b_2 = 1, \quad b_1 = 2, \quad a_1 = 1, a_2 = -2$
- (c) Determine the response to $x(n) = \cos(\pi n/3)$ if $b_0 = 1, b_1 = b_2 = 0, a_1 = 1$, and $a_2 = -0.99$.

9.8 Consider an LTI system, initially at rest, described by the difference equation

$$y(n) = \frac{1}{4}y(n-2) + x(n)$$

(a) Determine the impulse response, $h(n)$, of the system.

(b) What is the response of the system to the input signal

$$x(n) = [(\frac{1}{2})^n + (-\frac{1}{2})^n]u(n)$$

(c) Determine the direct form II, parallel-form, and cascade-form realizations for this system.

(d) Sketch roughly the magnitude response $|H(\omega)|$ of this system.

9.9 Obtain the direct form I, direct form II, cascade, and parallel structures for the following systems.

(a) $y(n) = \frac{3}{4}y(n-1) - \frac{1}{8}y(n-2) + x(n) + \frac{1}{3}x(n-1)$

(b) $y(n) = -0.1y(n-1) + 0.72y(n-2) + 0.7x(n) - 0.252x(n-2)$

(c) $y(n) = -0.1y(n-1) + 0.2y(n-2) + 3x(n) + 3.6x(n-1) + 0.6x(n-2)$

(d) $H(z) = \frac{2(1-z^{-1})(1+\sqrt{2}z^{-1}+z^{-2})}{(1+0.5z^{-1})(1-0.9z^{-1}+0.81z^{-2})}$

(e) $y(n) = \frac{1}{2}y(n-1) + \frac{1}{4}y(n-2) + x(n) + x(n-1)$

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

Which of the systems above are stable?

9.10 Show that the systems in Fig. P9.10 are equivalent.

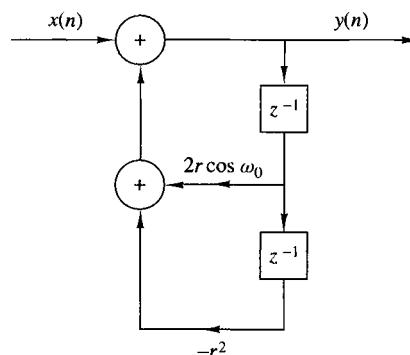
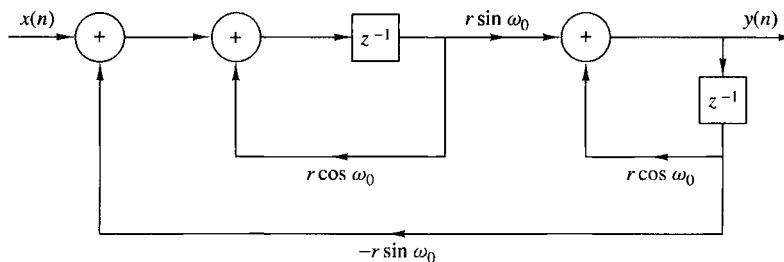


Figure P9.10

- 9.11** Determine all the FIR filters which are specified by the lattice parameters $K_1 = \frac{1}{2}$, $K_2 = 0.6$, $K_3 = -0.7$, and $K_4 = \frac{1}{3}$.
- 9.12** Determine the set of difference equations for describing a realization of an IIR system based on the use of the transposed direct form II structure for the second-order subsystems.
- *9.13** Write a program that implements a parallel-form realization based on transposed direct form II second-order modules.
- *9.14** Write a program that implements a cascade-form realization based on regular direct form II second-order modules.
- 9.15** Determine the parameters $\{K_m\}$ of the lattice filter corresponding to the FIR filter described by the system function

$$H(z) = A_2(z) = 1 + 2z^{-1} + z^{-2}$$

- 9.16 (a)** Determine the zeros and sketch the zero pattern for the FIR lattice filter with parameters

$$K_1 = \frac{1}{2}, \quad K_2 = -\frac{1}{3}, \quad K_3 = 1$$

- (b)** The same as in part (a) but with $K_3 = -1$.
- (c)** You should have found that all the zeros lie exactly on the unit circle. Can this result be generalized? How?
- (d)** Sketch the phase response of the filters in parts (a) and (b). What did you notice? Can this result be generalized? How?
- 9.17** Consider an FIR lattice filter with coefficients $K_1 = 0.65$, $K_2 = -0.34$, and $K_3 = 0.8$.
- (a)** Find its impulse response by tracing a unit impulse input through the lattice structure.
- (b)** Draw the equivalent direct-form structure.
- 9.18** Consider a causal IIR system with system function

$$H(z) = \frac{1 + 2z^{-1} + 3z^{-2} + 2z^{-3}}{1 + 0.9z^{-1} - 0.8z^{-2} + 0.5z^{-3}}$$

- (a)** Determine the equivalent lattice-ladder structure.
- (b)** Check if the system is stable.

- 9.19** Determine the input–output relationship, the system function, and plot the pole–zero pattern for the discrete-time system shown in Fig. P9.19.

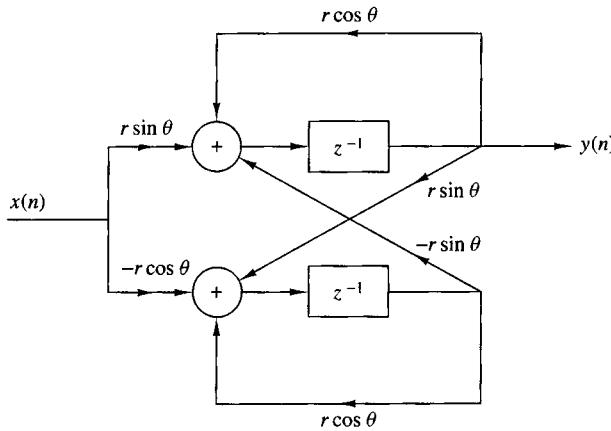


Figure P9.19

- 9.20** Determine the lattice realization for the digital resonator

$$H(z) = \frac{1}{1 - (2r \cos \omega_0)z^{-1} + r^2 z^{-2}}$$

9.21

- (a) Determine the impulse response of an FIR lattice filter with parameters $K_1 = 0.6$, $K_2 = 0.3$, $K_3 = 0.5$, and $K_4 = 0.9$.
- (b) Sketch the direct-form and lattice all-zero and all-pole filters specified by the K -parameters given in part (a).

- 9.22** (a) Determine the lattice-ladder realization for the resonator

$$H(z) = \frac{1 - z^{-1}}{1 - (2r \cos \omega_0)z^{-1} + r^2 z^{-2}}$$

- (b) What happens if $r = 1$?

- 9.23** Sketch the lattice-ladder structure for the system

$$H(z) = \frac{1 - 0.8z^{-1} + 0.15z^{-2}}{1 + 0.1z^{-1} - 0.72z^{-2}}$$

- 9.24** Consider a pole–zero system with system function

$$H(z) = \frac{(1 - 0.5e^{j\pi/4}z^{-1})(1 - 0.5e^{-j\pi/4}z^{-1})}{(1 - 0.8e^{j\pi/3}z^{-1})(1 - 0.8e^{-j\pi/3}z^{-1})}$$

Sketch the regular and transpose direct form II realizations of the system.

- 9.25** Determine a parallel and a cascade realization of the system

$$H(z) = \frac{1 + z^{-1}}{(1 - z^{-1})(1 - 0.8e^{j\pi/4}z^{-1})(1 - 0.8e^{-j\pi/4}z^{-1})}$$

- 9.26** The generic floating-point format for a DSP microprocessor is the following:

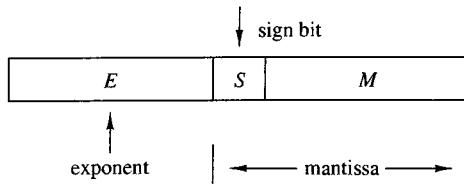
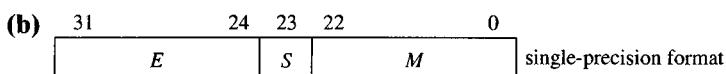
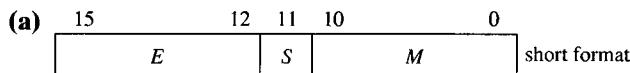


Figure P9.26

The value of the number X is given by

$$X = \begin{cases} 01.M \times 2^E & \text{if } S = 0 \\ 10.M \times 2^E & \text{if } S = 1 \\ 0 & \text{if } E \text{ is the most negative two's-complement value} \end{cases}$$

Determine the range of positive and negative numbers for the following two formats:



- 9.27** Consider the IIR recursive filter shown in Fig. P9.27 and let $h_F(n)$, $h_R(n)$, and $h(n)$ denote the impulse responses of the FIR section, the recursive section, and the overall filter, respectively.

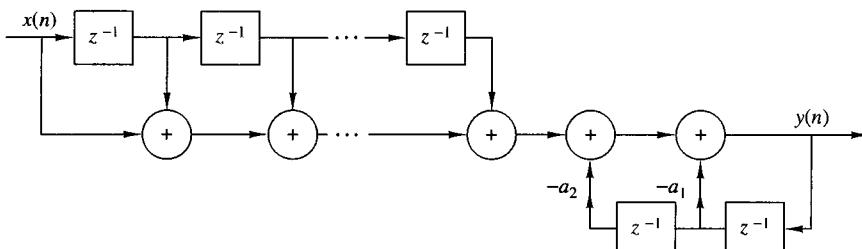


Figure P9.27

- (a) Find all the causal and stable recursive second-order sections with integer coefficients (a_1, a_2) and determine and sketch their impulse responses and frequency responses. These filters do not require complicated multiplications or quantization after multiplications.
- (b) Show that three of the sections obtained in part (a) can be obtained by interconnection of other sections.
- (c) Find a difference equation that describes the impulse response $h(n)$ of the filter and determine the conditions for the overall filter to be FIR.
- (d) Rederive the results in parts (a) to (c) using z -domain considerations.

- 9.28** This problem illustrates the development of digital filter structures using Horner's rule for polynomial evaluation. To this end consider the polynomial

$$p(x) = \alpha_p x^p + a_{p-1} x^{p-1} + \cdots + a_1 x + a_0$$

which computes $p(x)$ with the minimum cost of p multiplications and p additions.

- (a) Draw the structures corresponding to the factorizations

$$H_1(z) = b_0(1 + b_1 z^{-1}(1 + b_2 z^{-1}(1 + b_3 z^{-1})))$$

$$H(z) = b_0(z^{-3} + (b_1 z^{-2} + (b_2 z^{-1} + b_3)))$$

and determine the system function, number of delay elements, and arithmetic operations for each structure

- (b) Draw the Horner structure for the following linear-phase system:

$$H(z) = z^{-1} \left[\alpha_0 + \sum_{k=1}^3 (z^{-k} + z^k) \alpha_k \right]$$

- 9.29** Let x_1 and x_2 be $(b+1)$ -bit binary numbers with magnitude less than 1. To compute the sum of x_1 and x_2 using two's-complement representation we treat them as $(b+1)$ -bit unsigned numbers, perform addition modulo-2 and ignore any carry after the sign bit.

- (a) Show that if the sum of two numbers with the same sign has the opposite sign, this corresponds to overflow.
(b) Show that when we compute the sum of several numbers using two's-complement representation, the result will be correct, even if there are overflows, if the correct sum is less than 1 in magnitude. Illustrate this argument by constructing a simple example with three numbers.

- 9.30** Consider the system described by the difference equation

$$y(n) = ay(n-1) - ax(n) + x(n-1)$$

- (a) Show that it is allpass.
(b) Obtain the direct form II realization of the system
(c) If you quantize the coefficients of the system in part (b), is it still allpass?
(d) Obtain a realization by rewriting the difference equation as

$$y(n) = a[y(n-1) - x(n)] + x(n-1)$$

- (e) If you quantize the coefficients of the system in part (d), is it still allpass?

9.31 Consider the system

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

- (a) Compute its response to the input $x(n) = (\frac{1}{4})^n u(n)$ assuming infinite-precision arithmetic.
- (b) Compute the response of the system $y(n)$, $0 \leq n \leq 5$ to the same input, assuming finite-precision sign-and-magnitude fractional arithmetic with five bits (i.e., the sign bit plus four fractional bits). The quantization is performed by truncation.
- (c) Compare the results obtained in parts (a) and (b).

9.32 The input to the system

$$y(n) = 0.999y(n-1) + x(n)$$

is quantized to $b = 8$ bits. What is the power produced by the quantization noise at the output of the filter?

9.33 Consider the system

$$y(n) = 0.875y(n-1) - 0.125y(n-2) + x(n)$$

- (a) Compute its poles and design the cascade realization of the system.
- (b) Quantize the coefficients of the system using truncation, maintaining a sign bit plus three other bits. Determine the poles of the resulting system.
- (c) Repeat part (b) for the same precision using rounding.
- (d) Compare the poles obtained in parts (b) and (c) with those in part (a). Which realization is better? Sketch the frequency responses of the systems in parts (a), (b), and (c).

9.34 Consider the system

$$H(z) = \frac{1 - \frac{1}{2}z^{-1}}{(1 - \frac{1}{4}z^{-1})(1 + \frac{1}{4}z^{-1})}$$

- (a) Draw all possible realizations of the system.
- (b) Suppose that we implement the filter with fixed-point sign-and-magnitude fractional arithmetic using $(b+1)$ bits (one bit is used for the sign). Each resulting product is rounded into b bits. Determine the variance of the round-off noise created by the multipliers at the output of each one of the realizations in part (a).

- 9.35** The first-order filter shown in Fig. P9.35 is implemented in four-bit (including sign) fixed-point two's-complement fractional arithmetic. Products are rounded to four-bit representation. Using the input $x(n) = 0.10\delta(n)$, determine:

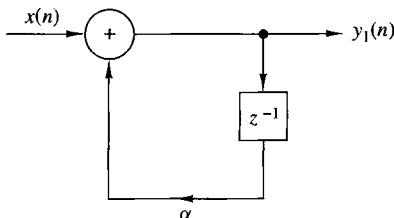


Figure P9.35

- (a) The first five outputs if $\alpha = 0.5$. Does the filter go into a limit cycle?
 (b) The first five outputs if $\alpha = 0.75$. Does the filter go into a limit cycle?
- 9.36** The digital system shown in Fig. P9.36 uses a six-bit (including sign) fixed-point two's-complement A/D converter with rounding, and the filter $H(z)$ is implemented using eight-bit (including sign) fixed-point two's-complement fractional arithmetic with rounding. The input $x(t)$ is a zero-mean uniformly distributed random process having autocorrelation $\gamma_{xx}(\tau) = 3\delta(\tau)$. Assume that the A/D converter can handle input values up to ± 1.0 without overflow.
- (a) What value of attenuation should be applied prior to the A/D converter to assure that it does not overflow?
 (b) With the attenuation above, what is the signal-to-quantization-noise ratio (SQNR) at the A/D converter output?
 (c) The six-bit A/D samples can be left justified, right justified, or centered in the eight-bit word used as the input to the digital filter. What is the correct strategy to use for maximum SNR at the filter output without overflow?
 (d) What is the SNR at the output of the filter due to all quantization noise sources?

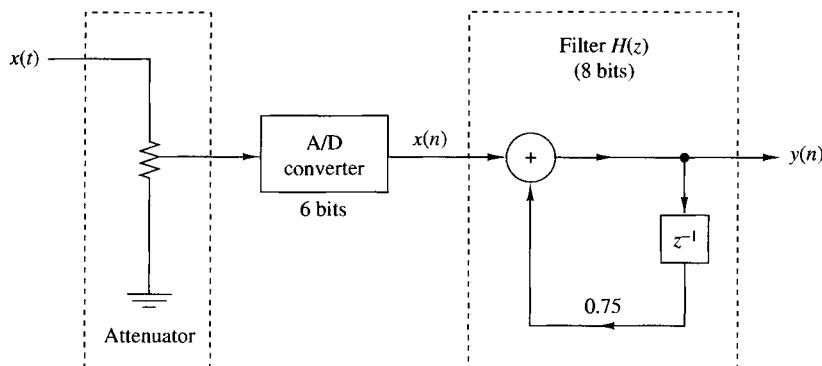


Figure P9.36

- 9.37** Shown in Fig. P9.37 is the coupled-form implementation of a two-pole filter with poles at $x = re^{\pm j\theta}$. There are four real multiplications per output point. Let $e_i(n)$, $i = 1, 2, 3, 4$ represent the round-off noise in a fixed-point implementation of the filter. Assume that the noise sources are zero-mean mutually uncorrelated stationary white noise sequences. For each n the probability density function $p(e)$ is uniform in the range $-\Delta/2 \leq e \leq \Delta/2$, where $\Delta = 2^{-b}$.

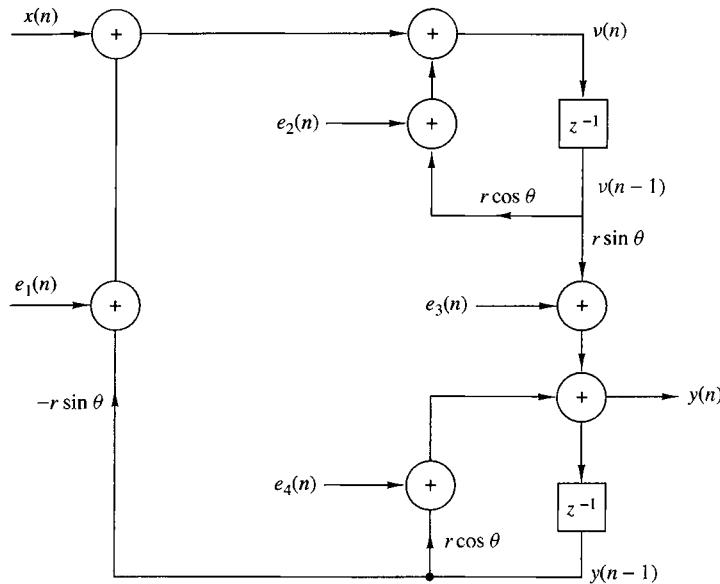


Figure P9.37

- (a) Write the two coupled difference equations for $y(n)$ and $v(n)$, including the noise sources and the input sequence $x(n)$.
- (b) From these two difference equations, show that the filter system functions $H_1(z)$ and $H_2(z)$ between the input noise terms $e_1(n) + e_2(n)$ and $e_3(n) + e_4(n)$ and the output $y(n)$ are:

$$H_1(z) = \frac{r \sin \theta z^{-1}}{1 - 2r \cos \theta z^{-1} + r^2 z^{-2}}$$

$$H_2(z) = \frac{1 - r \cos \theta z^{-1}}{1 - 2r \cos \theta z^{-1} + r^2 z^{-2}}$$

We know that

$$H(z) = \frac{1}{1 - 2r \cos \theta z^{-1} + r^2 z^{-2}} \Rightarrow h(n) = \frac{1}{\sin \theta} r^n \sin(n+1)\theta u(n)$$

Determine $h_1(n)$ and $h_2(n)$.

- (c) Determine a closed-form expression for the variance of the total noise from $e_i(n)$, $i = 1, 2, 3, 4$ at the output of the filter.

- 9.38** Determine the variance of the round-off noise at the output of the two cascade realizations of the filter shown in Fig. P9.38, with system function

$$H(z) = H_1(z)H_2(z)$$

where

$$H_1(z) = \frac{1}{1 - \frac{1}{2}z^{-1}}$$

$$H_2(z) = \frac{1}{1 - \frac{1}{3}z^{-1}}$$

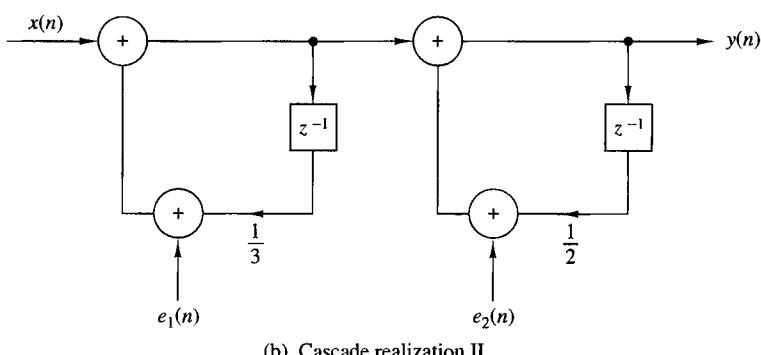
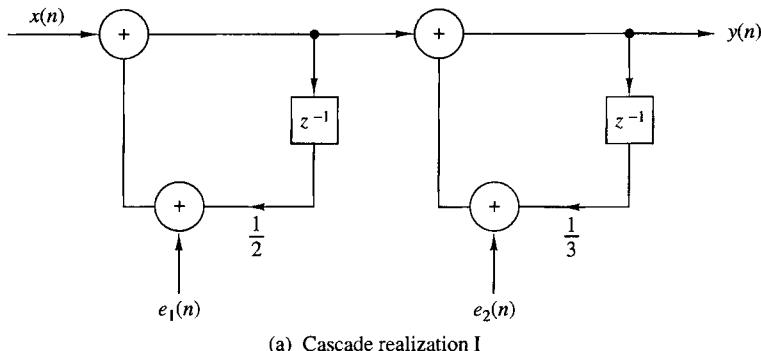


Figure P9.38

- 9.39 Quantization effects in direct-form FIR filters** Consider a direct-form realization of an FIR filter of length M . Suppose that the multiplication of each coefficient with the corresponding signal sample is performed in fixed-point arithmetic with b bits and each product is rounded to b bits. Determine the variance of the quantization noise at the output of the filter by using a statistical characterization of the round-off noise as in Section 9.6.3.

- 9.40** Consider the system specified by the system function

$$H(z) = \frac{B(z)}{A(z)}$$

$$= \left[G_1 \frac{(1 - 0.8e^{j\pi/4}z^{-1})(1 - 0.8e^{-j\pi/4}z^{-1})}{(1 - \frac{1}{2}z^{-1})(1 + \frac{1}{3}z^{-1})} \right] \left[G_2 \frac{(1 + \frac{1}{4}z^{-1})(1 - \frac{5}{8}z^{-1})}{(1 - 0.8e^{j\pi/3}z^{-1})(1 - 0.8e^{-j\pi/3}z^{-1})} \right]$$

- (a) Choose G_1 and G_2 so that the gain of each second-order section at $\omega = 0$ is equal to 1.
 - (b) Sketch the direct form 1, direct form 2, and cascade realizations of the system.
 - (c) Write a program that implements the direct form 1 and direct form 2, and compute the first 100 samples of the impulse response and the step response of the system.
 - (d) Plot the results in part (c) to illustrate the proper functioning of the programs.
- 9.41** Consider the system given in Problem 9.40 with $G_1 = G_2 = 1$.
- (a) Determine a lattice realization for the system

$$H(z) = B(z)$$

- (b) Determine a lattice realization for the system

$$H(z) = \frac{1}{A(z)}$$

- (c) Determine a lattice-ladder realization for the system $H(z) = B(z)/A(z)$.
 - (d) Write a program for the implementation of the lattice-ladder structure in part (c).
 - (e) Determine and sketch the first 100 samples of the impulse responses of the systems in parts (a) through (c) by working with the lattice structures.
 - (f) Compute and sketch the first 100 samples of the convolution of impulse responses in parts (a) and (b). What did you find? Explain your results.
- 9.42** Consider the system given in Problem 9.40. Determine the parallel-form structure and write a program for its implementation.

10

Design of Digital Filters

With the background that we have developed in the preceding chapters, we are now in a position to treat the subject of digital filter design. We shall describe several methods for designing FIR and IIR digital filters.

In the design of frequency-selective filters, the desired filter characteristics are specified in the frequency domain in terms of the desired magnitude and phase response of the filter. In the filter design process, we determine the coefficients of a causal FIR or IIR filter that closely approximates the desired frequency response specifications. The issue of which type of filter to design, FIR or IIR, depends on the nature of the problem and on the specifications of the desired frequency response.

In practice, FIR filters are employed in filtering problems where there is a requirement for a linear-phase characteristic within the passband of the filter. If there is no requirement for a linear-phase characteristic, either an IIR or an FIR filter may be employed. However, as a general rule, an IIR filter has lower sidelobes in the stopband than an FIR filter having the same number of parameters. For this reason, if some phase distortion is either tolerable or unimportant, an IIR filter is preferable, primarily because its implementation involves fewer parameters, requires less memory and has lower computational complexity.

In conjunction with our discussion of digital filter design, we describe frequency transformations in both the analog and digital domains for transforming a lowpass prototype filter into either another lowpass, bandpass, bandstop, or highpass filter.

Today, FIR and IIR digital filter design is greatly facilitated by the availability of numerous computer software programs. In describing the various digital filter design methods in this chapter, our primary objective is to give the reader the background necessary to select the filter that best matches the application and satisfies the design requirements.

10.1 General Considerations

In Section 5.4, we described the characteristics of ideal filters and demonstrated that such filters are not causal and therefore are not physically realizable. In this section, the issue of causality and its implications is considered in more detail. Following this discussion, we present the frequency response characteristics of causal FIR and IIR digital filters.

10.1.1 Causality and Its Implications

Let us consider the issue of causality in more detail by examining the impulse response $h(n)$ of an ideal lowpass filter with frequency response characteristic

$$H(\omega) = \begin{cases} 1, & |\omega| \leq \omega_c \\ 0, & \omega_c < \omega \leq \pi \end{cases} \quad (10.1.1)$$


The impulse response of this filter is

$$h(n) = \begin{cases} \frac{\omega_c}{\pi}, & n = 0 \\ \frac{\pi}{\omega_c} \sin \omega_c n, & n \neq 0 \end{cases} \quad (10.1.2)$$

A plot of $h(n)$ for $\omega_c = \pi/4$ is illustrated in Fig. 10.1.1. It is clear that the ideal lowpass filter is noncausal and hence it cannot be realized in practice.

One possible solution is to introduce a large delay n_0 in $h(n)$ and arbitrarily to set $h(n) = 0$ for $n < n_0$. However, the resulting system no longer has an ideal frequency response characteristic. Indeed, if we set $h(n) = 0$ for $n < n_0$, the Fourier series expansion of $H(\omega)$ results in the Gibbs phenomenon, as will be described in Section 10.2.

Although this discussion is limited to the realization of a lowpass filter, our conclusions hold, in general, for all the other ideal filter characteristics. In brief,

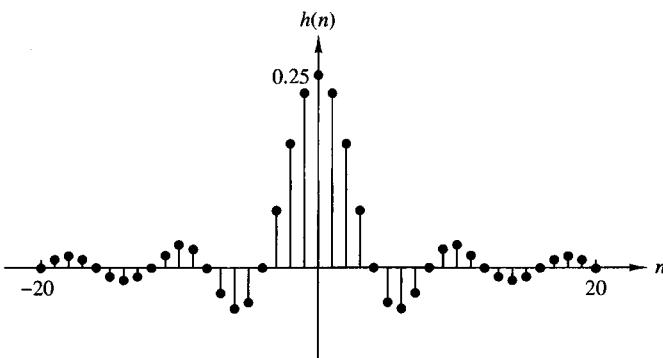


Figure 10.1.1 Unit sample response of an ideal lowpass filter.

none of the ideal filter characteristics previously illustrated in Fig. 5.4.1 are causal, hence all are physically unrealizable.

A question that naturally arises at this point is the following: What are the necessary and sufficient conditions that a frequency response characteristic $H(\omega)$ must satisfy in order for the resulting filter to be causal? The answer to this question is given by the Paley–Wiener theorem, which can be stated as follows:

Paley–Wiener Theorem. If $h(n)$ has finite energy and $h(n) = 0$ for $n < 0$, then [for a reference, see Wiener and Paley (1934)]

$$\int_{-\pi}^{\pi} |\ln|H(\omega)|| d\omega < \infty \quad (10.1.3)$$

Conversely, if $|H(\omega)|$ is square integrable and if the integral in (10.1.3) is finite, then we can associate with $|H(\omega)|$ a phase response $\Theta(\omega)$, so that the resulting filter with frequency response

$$H(\omega) = |H(\omega)|e^{j\Theta(\omega)}$$

is causal.

One important conclusion that we draw from the Paley–Wiener theorem is that the magnitude function $|H(\omega)|$ can be zero at some frequencies, but it cannot be zero over any finite band of frequencies, since the integral then becomes infinite. Consequently, any ideal filter is noncausal.

Apparently, causality imposes some tight constraints on a linear time-invariant system. In addition to the Paley–Wiener condition, causality also implies a strong relationship between $H_R(\omega)$ and $H_I(\omega)$, the real and imaginary components of the frequency response $H(\omega)$. To illustrate this dependence, we decompose $h(n)$ into an even and an odd sequence, that is,

$$h(n) = h_e(n) + h_o(n) \quad (10.1.4)$$

where

$$h_e(n) = \frac{1}{2}[h(n) + h(-n)] \quad (10.1.5)$$

and

$$h_o(n) = \frac{1}{2}[h(n) - h(-n)] \quad (10.1.6)$$

Now, if $h(n)$ is causal, it is possible to recover $h(n)$ from its even part $h_e(n)$ for $0 \leq n \leq \infty$ or from its odd component $h_o(n)$ for $1 \leq n \leq \infty$.

Indeed, it can be easily seen that

$$h(n) = 2h_e(n)u(n) - h_e(0)\delta(n), \quad n \geq 0 \quad (10.1.7)$$

and

$$h(n) = 2h_o(n)u(n) + h(0)\delta(n), \quad n \geq 1 \quad (10.1.8)$$

Since $h_o(n) = 0$ for $n = 0$, we cannot recover $h(0)$ from $h_o(n)$ and hence we also must know $h(0)$. In any case, it is apparent that $h_o(n) = h_e(n)$ for $n \geq 1$, so there is a strong relationship between $h_o(n)$ and $h_e(n)$.

If $h(n)$ is absolutely summable (i.e., BIBO stable), the frequency response $H(\omega)$ exists, and

$$H(\omega) = H_R(\omega) + jH_I(\omega) \quad (10.1.9)$$

In addition, if $h(n)$ is real valued and causal, the symmetry properties of the Fourier transform imply that

$$\begin{aligned} h_e(n) &\xleftrightarrow{F} H_R(\omega) \\ h_o(n) &\xleftrightarrow{F} H_I(\omega) \end{aligned} \quad (10.1.10)$$

Since $h(n)$ is completely specified by $h_e(n)$, it follows that $H(\omega)$ is completely determined if we know $H_R(\omega)$. Alternatively, $H(\omega)$ is completely determined from $H_I(\omega)$ and $h(0)$. In short, $H_R(\omega)$ and $H_I(\omega)$ are interdependent and cannot be specified independently if the system is causal. Equivalently, the magnitude and phase responses of a causal filter are interdependent and hence cannot be specified independently.

Given $H_R(\omega)$ for a corresponding real, even, and absolutely summable sequence $h_e(n)$, we can determine $H(\omega)$. The following example illustrates the procedure.

EXAMPLE 10.1.1

Consider a stable LTI system with real and even impulse response $h(n)$. Determine $H(\omega)$ if

$$H_R(\omega) = \frac{1 - a \cos \omega}{1 - 2a \cos \omega + a^2}, \quad |a| < 1$$

Solution. The first step is to determine $h_e(n)$. This can be done by noting that

$$H_R(\omega) = H_R(z)|_{z=e^{j\omega}}$$

where

$$H_R(z) = \frac{1 - a(z + z^{-1})/2}{1 - a(z + z^{-1}) + a^2} = \frac{z - a(z^2 + 1)/2}{(z - a)(1 - az)}$$

The ROC has to be restricted by the poles at $p_1 = a$ and $p_2 = 1/a$ and should include the unit circle. Hence the ROC is $|a| < |z| < 1/|a|$. Consequently, $h_e(n)$ is a two-sided sequence, with the pole at $z = a$ contributing to the causal part and $p_2 = 1/a$ contributing to the anticausal part. By using a partial-fraction expansion, we obtain

$$h_e(n) = \frac{1}{2}a^{|n|} + \frac{1}{2}\delta(n) \quad (10.1.11)$$

By substituting (10.1.11) into (10.1.7), we obtain $h(n)$ as

$$h(n) = a^n u(n)$$

Finally, we obtain the Fourier transform of $h(n)$ as

$$H(\omega) = \frac{1}{1 - ae^{-j\omega}}$$

The relationship between the real and imaginary components of the Fourier transform of an absolutely summable, causal, and real sequence can be easily established from (10.1.7). The Fourier transform relationship for (10.1.7) is

$$H(\omega) = H_R(\omega) + jH_I(\omega) = \frac{1}{\pi} \int_{-\pi}^{\pi} H_R(\lambda)U(\omega - \lambda)d\lambda - h_e(0) \quad (10.1.12)$$

where $U(\omega)$ is the Fourier transform of the unit step sequence $u(n)$. Although the unit step sequence is not absolutely summable, it has a Fourier transform (see Section 4.2.8).

$$\begin{aligned} U(\omega) &= \pi\delta(\omega) + \frac{1}{1 - e^{-j\omega}} \\ &= \pi\delta(\omega) + \frac{1}{2} - j\frac{1}{2}\cot\frac{\omega}{2}, \quad -\pi \leq \omega \leq \pi \end{aligned} \quad (10.1.13)$$

By substituting (10.1.13) into (10.1.12) and carrying out the integration, we obtain the relation between $H_R(\omega)$ and $H_I(\omega)$ as

$$H_I(\omega) = -\frac{1}{2\pi} \int_{-\pi}^{\pi} H_R(\lambda) \cot\frac{\omega - \lambda}{2} d\lambda \quad (10.1.14)$$

Thus $H_I(\omega)$ is uniquely determined from $H_R(\omega)$ through this integral relationship. The integral is called a *discrete Hilbert transform*. It is left as an exercise to the reader to establish the relationship for $H_R(\omega)$ in terms of the discrete Hilbert transform of $H_I(\omega)$.

To summarize, causality has very important implications in the design of frequency-selective filters. These are: (a) the frequency response $H(\omega)$ cannot be zero, except at a finite set of points in frequency; (b) the magnitude $|H(\omega)|$ cannot be constant in any finite range of frequencies and the transition from passband to stopband cannot be infinitely sharp [this is a consequence of the Gibbs phenomenon, which results from the truncation of $h(n)$ to achieve causality]; and (c) the real and imaginary parts of $H(\omega)$ are interdependent and are related by the discrete Hilbert transform. As a consequence, the magnitude $|H(\omega)|$ and phase $\Theta(\omega)$ of $H(\omega)$ cannot be chosen arbitrarily.

Now that we know the restrictions that causality imposes on the frequency response characteristic and the fact that ideal filters are not achievable in practice, we limit our attention to the class of linear time-invariant systems specified by the difference equation

$$y(n) = -\sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^{M-1} b_k x(n-k)$$

which are causal and physically realizable. As we have demonstrated, such systems have a frequency response

$$H(\omega) = \frac{\sum_{k=0}^{M-1} b_k e^{-j\omega k}}{1 + \sum_{k=1}^N a_k e^{-j\omega k}} \quad (10.1.15)$$

The basic digital filter design problem is to approximate any of the ideal frequency response characteristics with a system that has the frequency response (10.1.15), by properly selecting the coefficients $\{a_k\}$ and $\{b_k\}$. The approximation problem is treated in detail in Sections 10.2 and 10.3, where we discuss techniques for digital filter design.

10.1.2 Characteristics of Practical Frequency-Selective Filters

As we observed from our discussion of the preceding section, ideal filters are non-causal and hence physically unrealizable for real-time signal processing applications. Causality implies that the frequency response characteristic $H(\omega)$ of the filter cannot be zero, except at a finite set of points in the frequency range. In addition, $H(\omega)$ cannot have an infinitely sharp cutoff from passband to stopband, that is, $H(\omega)$ cannot drop from unity to zero abruptly.

Although the frequency response characteristics possessed by ideal filters may be desirable, they are not absolutely necessary in most practical applications. If we relax these conditions, it is possible to realize causal filters that approximate the ideal filters as closely as we desire. In particular, it is not necessary to insist that the magnitude $|H(\omega)|$ be constant in the entire passband of the filter. A small amount of ripple in the passband, as illustrated in Fig. 10.1.2, is usually tolerable. Similarly,

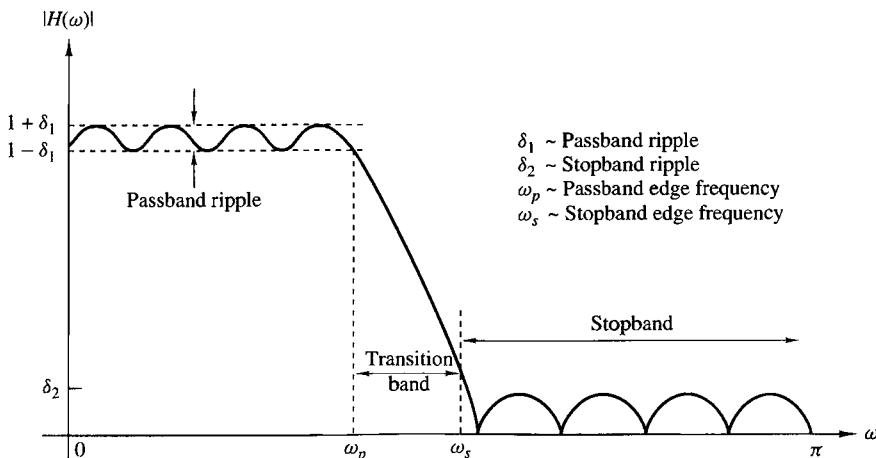


Figure 10.1.2 Magnitude characteristics of physically realizable filters.

it is not necessary for the filter response $|H(\omega)|$ to be zero in the stopband. A small, nonzero value or a small amount of ripple in the stopband is also tolerable.

The transition of the frequency response from passband to stopband defines the *transition band* or *transition region* of the filter, as illustrated in Fig. 10.1.2. The band-edge frequency ω_p defines the edge of the passband, while the frequency ω_s denotes the beginning of the stopband. Thus the width of the transition band is $\omega_s - \omega_p$. The width of the passband is usually called the *bandwidth* of the filter. For example, if the filter is lowpass with a passband edge frequency ω_p , its bandwidth is ω_p .

If there is ripple in the passband of the filter, its value is denoted as δ_1 , and the magnitude $|H(\omega)|$ varies between the limits $1 \pm \delta_1$. The ripple in the stopband of the filter is denoted as δ_2 .

To accommodate a large dynamic range in the graph of the frequency response of any filter, it is common practice to use a logarithmic scale for the magnitude $|H(\omega)|$. Consequently, the ripple in the passband is $20 \log_{10} \delta_1$ decibels, and that in the stopband is $20 \log_{10} \delta_2$.

In any filter design problem we can specify (1) the maximum tolerable passband ripple, (2) the maximum tolerable stopband ripple, (3) the passband edge frequency ω_p , and (4) the stopband edge frequency ω_s . Based on these specifications, we can select the parameters $\{a_k\}$ and $\{b_k\}$ in the frequency response characteristic, given by (10.1.15), which best approximate the desired specification. The degree to which $H(\omega)$ approximates the specifications depends in part on the criterion used in the selection of the filter coefficients $\{a_k\}$ and $\{b_k\}$ as well as on the numbers (M, N) of coefficients.

In the following section we present a method for designing linear-phase FIR filters.

10.2 Design of FIR Filters

In this section we describe several methods for designing FIR filters. Our treatment is focused on the important class of linear-phase FIR filters.

10.2.1 Symmetric and Antisymmetric FIR Filters

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

$$\begin{aligned} y(n) &= b_0x(n) + b_1x(n-1) + \cdots + b_{M-1}x(n-M+1) \\ &= \sum_{k=0}^{M-1} b_kx(n-k) \end{aligned} \quad (10.2.1)$$

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

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (10.2.2)$$

where the lower and upper limits on the convolution sum reflect the causality and finite-duration characteristics of the filter. Clearly, (10.2.1) and (10.2.2) are identical in form and hence it follows that $b_k = h(k)$, $k = 0, 1, \dots, M - 1$.

The filter can also be characterized by its system function

$$H(z) = \sum_{k=0}^{M-1} h(k)z^{-k} \quad (10.2.3)$$

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

$$h(n) = \pm h(M - 1 - n), \quad n = 0, 1, \dots, M - 1 \quad (10.2.4)$$

When the symmetry and antisymmetry conditions in (10.2.4) are incorporated into (10.2.3), we have

$$\left\{ \begin{aligned} H(z) &= h(0) + h(1)z^{-1} + h(2)z^{-2} + \cdots + h(M-2)z^{-(M-2)} + h(M-1)z^{-(M-1)} \\ &= z^{-(M-1)/2} \left\{ h\left(\frac{M-1}{2}\right) + \sum_{n=0}^{(M-3)/2} h(n) [z^{(M-1-2k)/2} \pm z^{-(M-1-2k)/2}] \right\}, \quad M \text{ odd} \\ &= z^{-(M-1)/2} \sum_{n=0}^{(M/2)-1} h(n) [z^{(M-1-2k)/2} \pm z^{-(M-1-2k)/2}], \quad M \text{ even} \end{aligned} \right. \quad (10.2.5)$$

Now, if we substitute z^{-1} for z in (10.2.3) and multiply both sides of the resulting equation by $z^{-(M-1)}$, we obtain

$$z^{-(M-1)} H(z^{-1}) = \pm H(z) \quad (10.2.6)$$

This result implies that the roots of the polynomial $H(z)$ are identical to the roots of the polynomial $H(z^{-1})$. Consequently, the roots of $H(z)$ must occur in reciprocal pairs. In other words, if z_1 is a root or a zero of $H(z)$, then $1/z_1$ is also a root. Furthermore, if the unit sample response $h(n)$ of the filter is real, complex-valued roots must occur in complex-conjugate pairs. Hence, if z_1 is a complex-valued root, z_1^* is also a root. As a consequence of (10.2.6), $H(z)$ also has a zero at $1/z_1^*$. Figure 10.2.1 illustrates the symmetry that exists in the location of the zeros of a linear-phase FIR filter.

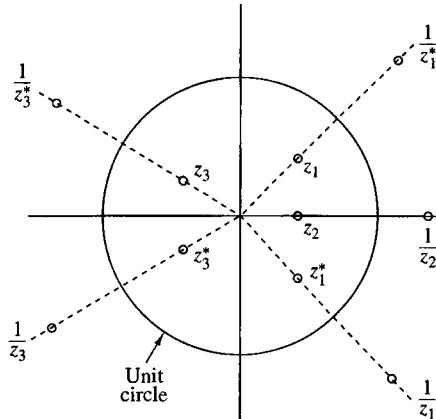


Figure 10.2.1
Symmetry of zero locations
for a linear-phase FIR filter.

The frequency response characteristics of linear-phase FIR filters are obtained by evaluating (10.2.5) on the unit circle. This substitution yields the expression for $H(\omega)$.

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

$$H(\omega) = H_r(\omega)e^{-j\omega(M-1)/2} \quad (10.2.7)$$

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

$$H_r(\omega) = h\left(\frac{M-1}{2}\right) + 2 \sum_{n=0}^{(M-3)/2} h(n) \cos \omega \left(\frac{M-1}{2} - n\right), \quad M \text{ odd} \quad (10.2.8)$$

$$H_r(\omega) = 2 \sum_{n=0}^{(M/2)-1} h(n) \cos \omega \left(\frac{M-1}{2} - n\right), \quad M \text{ even} \quad (10.2.9)$$

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

$$\Theta(\omega) = \begin{cases} -\omega \left(\frac{M-1}{2}\right), & \text{if } H_r(\omega) > 0 \\ -\omega \left(\frac{M-1}{2}\right) + \pi, & \text{if } H_r(\omega) < 0 \end{cases} \quad (10.2.10)$$

When

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

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

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

However, if M is even, each term in $h(n)$ has a matching term of opposite sign.

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

$$H(\omega) = H_r(\omega) e^{j[-\omega(M-1)/2 + \pi/2]} \quad (10.2.11)$$

where

$$H_r(\omega) = 2 \sum_{n=0}^{(M-3)/2} h(n) \sin \omega \left(\frac{M-1}{2} - n \right), \quad M \text{ odd} \quad (10.2.12)$$

$$H_r(\omega) = 2 \sum_{n=0}^{(M/2)-1} h(n) \sin \omega \left(\frac{M-1}{2} - n \right), \quad M \text{ even} \quad (10.2.13)$$

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

$$\Theta(\omega) = \begin{cases} \frac{\pi}{2} - \omega \left(\frac{M-1}{2} \right), & \text{if } H_r(\omega) > 0 \\ \frac{3\pi}{2} - \omega \left(\frac{M-1}{2} \right), & \text{if } H_r(\omega) < 0 \end{cases} \quad (10.2.14)$$

These general frequency response formulas can be used to design linear-phase FIR filters with symmetric and antisymmetric unit sample responses. We note that, for a symmetric $h(n)$, the number of filter coefficients that specify the frequency response is $(M+1)/2$ when M is odd or $M/2$ when M is even. On the other hand, if the unit sample response is antisymmetric,

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

so that there are $(M-1)/2$ filter coefficients when M is odd and $M/2$ coefficients when M is even to be specified.

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, (10.2.12) implies that $H_r(0) = 0$ and $H_r(\pi) = 0$. Consequently, (10.2.12) 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 (10.2.13). 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,

$$H_r(0) = h\left(\frac{M-1}{2}\right) + 2 \sum_{n=0}^{(M-3)/2} h(n), \quad M \text{ odd} \quad (10.2.15)$$

$$H_r(0) = 2 \sum_{n=0}^{(M/2)-1} h(n), \quad M \text{ even} \quad (10.2.16)$$

In summary, the problem of FIR filter design is simply to determine the M coefficients $h(n)$, $n = 0, 1, \dots, M - 1$, from a specification of the desired frequency response $H_d(\omega)$ of the FIR filter. The important parameters in the specification of $H_d(\omega)$ are given in Fig. 10.1.2.

In the following subsections we describe design methods based on specification of $H_d(\omega)$.

10.2.2 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

$$H_d(\omega) = \sum_{n=0}^{\infty} h_d(n)e^{-j\omega n} \quad (10.2.17)$$

where

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(\omega)e^{j\omega n} d\omega \quad (10.2.18)$$

Thus, given $H_d(\omega)$, we can determine the unit sample response $h_d(n)$ by evaluating the integral in (10.2.17).

In general, the unit sample response $h_d(n)$ obtained from (10.2.17) 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

$$w(n) = \begin{cases} 1, & n = 0, 1, \dots, M - 1 \\ 0, & \text{otherwise} \end{cases} \quad (10.2.19)$$

Thus the unit sample response of the FIR filter becomes

$$\begin{aligned} h(n) &= h_d(n)w(n) \\ &= \begin{cases} h_d(n), & n = 0, 1, \dots, M - 1 \\ 0, & \text{otherwise} \end{cases} \end{aligned} \quad (10.2.20)$$

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,

$$W(\omega) = \sum_{n=0}^{M-1} w(n)e^{-j\omega n} \quad (10.2.21)$$

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

$$H(\omega) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(\nu) W(\omega - \nu) d\nu \quad (10.2.22)$$

The Fourier transform of the rectangular window is

$$\begin{aligned} W(\omega) &= \sum_{n=0}^{M-1} e^{-j\omega n} \\ &= \frac{1 - e^{-j\omega M}}{1 - e^{-j\omega}} = e^{-j\omega(M-1)/2} \frac{\sin(\omega M/2)}{\sin(\omega/2)} \end{aligned} \quad (10.2.23)$$

This window function has a magnitude response

$$|W(\omega)| = \frac{|\sin(\omega M/2)|}{|\sin(\omega/2)|}, \quad \pi \leq \omega \leq \pi \quad (10.2.24)$$

and a piecewise linear phase

$$\Theta(\omega) = \begin{cases} -\omega \left(\frac{M-1}{2} \right), & \text{when } \sin(\omega M/2) \geq 0 \\ -\omega \left(\frac{M-1}{2} \right) + \pi, & \text{when } \sin(\omega M/2) < 0 \end{cases} \quad (10.2.25)$$

The magnitude response of the window function is illustrated in Fig. 10.2.2 for $M = 31$ and 61 . The width of the main lobe [width is measured to the first zero of $W(\omega)$] is $4\pi/M$. Hence, as M increases, the main lobe becomes narrower. However, the sidelobes of $|W(\omega)|$ are relatively high and remain unaffected by an increase in M . In fact, even though the width of each sidelobe decreases with an increase in M , the height of each sidelobe increases with an increase in M in such a manner that the area under each sidelobe remains invariant to changes in M . This characteristic behavior is not evident from observation of Fig. 10.2.2 because $W(\omega)$ has been normalized by M such that the normalized peak values of the sidelobes remain invariant to an increase in M .

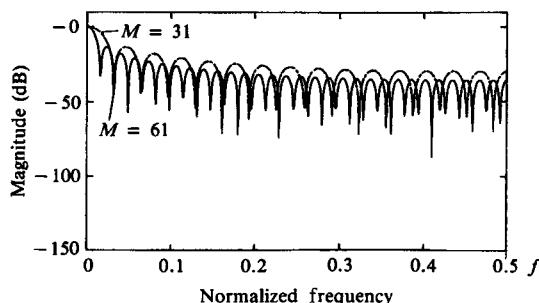


Figure 10.2.2
Frequency response for rectangular window of lengths (a) $M = 31$, (b) $M = 61$.

The characteristics of the rectangular window play a significant role in determining the resulting frequency response of the FIR filter obtained by truncating $h_d(n)$ to length M . Specifically, the convolution of $H_d(\omega)$ with $W(\omega)$ has the effect of smoothing $H_d(\omega)$. As M is increased, $W(\omega)$ becomes narrower, and the smoothing provided by $W(\omega)$ is reduced. On the other hand, the large sidelobes of $W(\omega)$ result in some undesirable ringing effects in the FIR filter frequency response $H(\omega)$, and also in relatively larger sidelobes in $H(\omega)$. These undesirable effects are best alleviated by the use of windows that do not contain abrupt discontinuities in their time-domain characteristics, and have correspondingly low sidelobes in their frequency-domain characteristics.

Table 10.1 lists several window functions that possess desirable frequency response characteristics. Figure 10.2.3 illustrates the time-domain characteristics of the windows. The frequency response characteristics of the Hanning, Hamming, and Blackman windows are illustrated in Figs. 10.2.4 through 10.2.6. All of these window

TABLE 10.1 Window Functions for FIR Filter Design

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

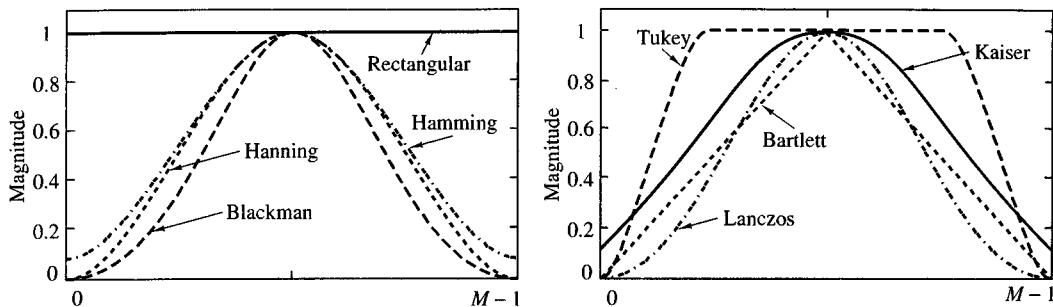


Figure 10.2.3 Shapes of several window functions.

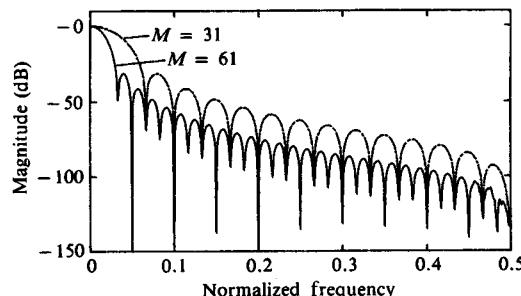


Figure 10.2.4
Frequency responses
of Hanning window
for (a) $M = 31$ and
(b) $M = 61$.

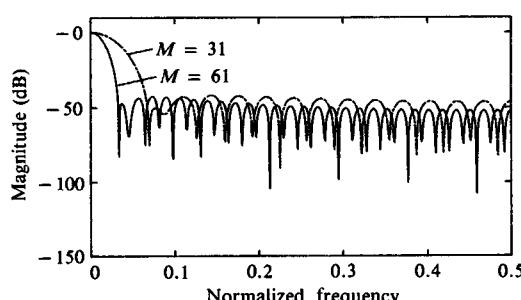


Figure 10.2.5
Frequency responses
for Hamming window
for (a) $M = 31$ and
(b) $M = 61$.

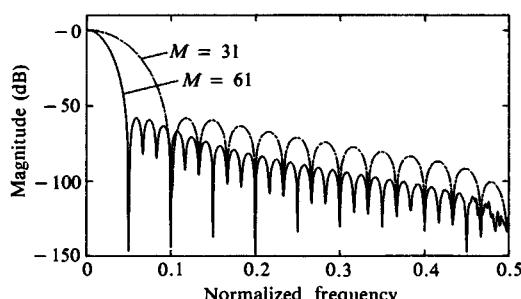


Figure 10.2.6
Frequency responses
for Blackman window
for (a) $M = 31$ and
(b) $M = 61$.

functions have significantly lower sidelobes compared with the rectangular window. However, for the same value of M , the width of the main lobe is also wider for these windows compared to the rectangular window. Consequently, these window functions provide more smoothing through the convolution operation in the frequency domain, and as a result, the transition region in the FIR filter response is wider. To reduce the width of this transition region, we can simply increase the length of the window, which results in a larger filter. Table 10.2 summarizes these important frequency-domain features of the various window functions.

The window technique is best described in terms of a specific example. Suppose that we want to design a symmetric lowpass linear-phase FIR filter having a desired frequency response

$$H_d(\omega) = \begin{cases} 1e^{-j\omega(M-1)/2}, & 0 \leq |\omega| \leq \omega_c \\ 0, & \text{otherwise} \end{cases} \quad (10.2.26)$$

A delay of $(M - 1)/2$ units is incorporated into $H_d(\omega)$ in anticipation of forcing the filter to be of length M . The corresponding unit sample response, obtained by evaluating the integral in (10.2.18), is

$$\begin{aligned} h_d(n) &= \frac{1}{2\pi} \int_{-\omega_c}^{\omega_c} e^{j\omega\left(n - \frac{M-1}{2}\right)} d\omega \\ &= \frac{\sin \omega_c \left(n - \frac{M-1}{2}\right)}{\pi \left(n - \frac{M-1}{2}\right)}, \quad n \neq \frac{M-1}{2} \end{aligned} \quad (10.2.27)$$

Clearly, $h_d(n)$ is noncausal and infinite in duration.

If we multiply $h_d(n)$ by the rectangular window sequence in (10.2.19), we obtain

TABLE 10.2 Important Frequency-Domain Characteristics of Some Window Functions

Type of window	Approximate transition width of main lobe	Peak sidelobe (dB)
Rectangular	$4\pi/M$	-13
Bartlett	$8\pi/M$	-25
Hanning	$8\pi/M$	-31
Hamming	$8\pi/M$	-41
Blackman	$12\pi/M$	-57

an FIR filter of length M having the unit sample response

$$h(n) = \frac{\sin \omega_c \left(n - \frac{M-1}{2} \right)}{\pi \left(n - \frac{M-1}{2} \right)}, \quad 0 \leq n \leq M-1, \quad n \neq \frac{M-1}{2} \quad (10.2.28)$$

If M is selected to be odd, the value of $h(n)$ at $n = (M-1)/2$ is

$$h\left(\frac{M-1}{2}\right) = \frac{\omega_c}{\pi} \quad (10.2.29)$$

The magnitude of the frequency response $H(\omega)$ of this filter is illustrated in Fig. 10.2.7 for $M = 61$ and $M = 101$. We observe that relatively large oscillations or ripples occur near the band edge of the filter. The oscillations increase in frequency as M increases, but they do not diminish in amplitude. As indicated previously, these large oscillations are the direct result of the large sidelobes existing in the frequency characteristic $W(\omega)$ of the rectangular window. As this window function is convolved with the desired frequency response characteristic $H_d(\omega)$, the oscillations occur as the large constant-area sidelobes of $W(\omega)$ move across the discontinuity that exists in $H_d(\omega)$. Since (10.2.17) is basically a Fourier series representation of $H_d(\omega)$, the multiplication of $h_d(n)$ with a rectangular window is identical to truncating the Fourier series representation of the desired filter characteristic $H_d(\omega)$. The truncation of the Fourier series is known to introduce ripples in the frequency response characteristic $H(\omega)$ due to the nonuniform convergence of the Fourier series at a discontinuity. The oscillatory behavior near the band edge of the filter is called the *Gibbs phenomenon*.

To alleviate the presence of large oscillations in both the passband and the stopband, we should use a window function that contains a taper and decays toward zero gradually, instead of abruptly, as it occurs in a rectangular window. Figures 10.2.8 through 10.2.11 illustrate the frequency response of the resulting filter when some of the window functions listed in Table 10.1 are used to taper $h_d(n)$. As illustrated in Figs. 10.2.8 through 10.2.11, the window functions do indeed eliminate the ringing effects at the band edge and do result in lower sidelobes at the expense of an increase in the width of the transition band of the filter.

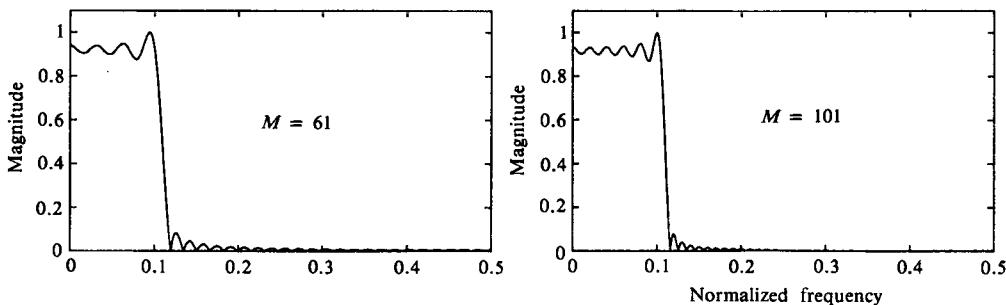


Figure 10.2.7 Lowpass filter designed with a rectangular window: (a) $M = 61$ and (b) $M = 101$.

Figure 10.2.8
Lowpass FIR filter designed
with rectangular window
($M = 61$).

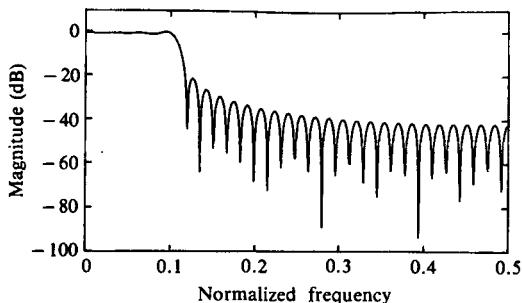


Figure 10.2.9
Lowpass FIR filter designed
with Hamming window
($M = 61$).

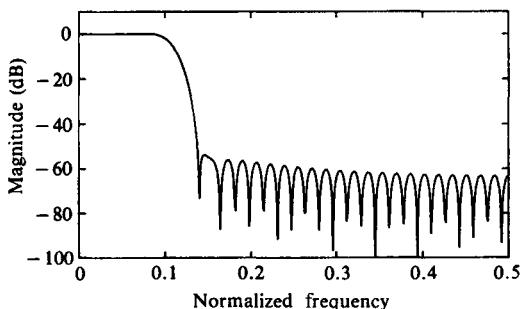


Figure 10.2.10
Lowpass FIR filter designed
with Blackman window
($M = 61$).

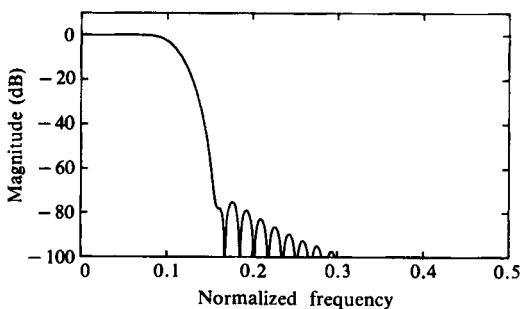
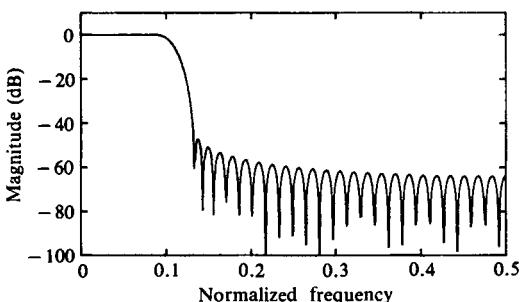


Figure 10.2.11
Lowpass FIR filter designed
with $\alpha = 4$ Kaiser window
($M = 61$).



10.2.3 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

$$\begin{aligned}\omega_k &= \frac{2\pi}{M}(k + \alpha), \quad k = 0, 1, \dots, \frac{M-1}{2}, \quad M \text{ odd} \\ &\quad k = 0, 1, \dots, \frac{M}{2}-1, \quad M \text{ even} \\ \alpha &= 0 \quad \text{or} \quad \frac{1}{2}\end{aligned}\tag{10.2.30}$$

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 $H_d(\omega)$],

$$H(\omega) = \sum_{n=0}^{M-1} h(n)e^{-j\omega n}\tag{10.2.31}$$

Suppose that we specify the frequency response of the filter at the frequencies given by (10.2.30). Then from (10.2.31) we obtain

$$\begin{aligned}H(k + \alpha) &\equiv H\left(\frac{2\pi}{M}(k + \alpha)\right) \\ H(k + \alpha) &\equiv \sum_{n=0}^{M-1} h(n)e^{-j2\pi(k+\alpha)n/M}, \quad k = 0, 1, \dots, M-1\end{aligned}\tag{10.2.32}$$

It is a simple matter to invert (10.2.32) and express $h(n)$ in terms of $H(k + \alpha)$. If we multiply both sides of (10.2.32) by the exponential, $\exp(j2\pi km/M)$, $m = 0, 1, \dots, M-1$, and sum over $k = 0, 1, \dots, M-1$, the right-hand side of (10.2.32) reduces to $Mh(m)\exp(-j2\pi\alpha m/M)$. Thus we obtain

$$h(n) = \frac{1}{M} \sum_{k=0}^{M-1} H(k + \alpha)e^{j2\pi(k+\alpha)n/M}, \quad n = 0, 1, \dots, M-1\tag{10.2.33}$$

The relationship in (10.2.33) allows us to compute the values of the unit sample response $h(n)$ from the specification of the frequency samples $H(k + \alpha)$, $k = 0, 1, \dots, M-1$. Note that when $\alpha = 0$, (10.2.32) reduces to the discrete Fourier

transform (DFT) of the sequence $\{h(n)\}$ and (10.2.33) 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

$$H(k + \alpha) = H^*(M - k - \alpha) \quad (10.2.34)$$

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 (10.2.11) is sampled at the frequencies $\omega_k = 2\pi(k + \alpha)/M$, $k = 0, 1, \dots, M - 1$, we obtain

$$H(k + \alpha) = H_r \left(\frac{2\pi}{M} (k + \alpha) \right) e^{j[\beta\pi/2 - 2\pi(k + \alpha)(M - 1)/2M]} \quad (10.2.35)$$

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

$$G(k + \alpha) = (-1)^k H_r \left(\frac{2\pi}{M} (k + \alpha) \right), \quad k = 0, 1, \dots, M - 1 \quad (10.2.36)$$

We use (10.2.36) in (10.2.35) to eliminate $H_r(\omega_k)$. Thus we obtain

$$H(k + \alpha) = G(k + \alpha) e^{j[\beta\pi/2 - 2\pi(k + \alpha)(M - 1)/2M]} \quad (10.2.37)$$

Now the symmetry condition for $H(k + \alpha)$ given in (10.2.34) translates into a corresponding symmetry condition for $G(k + \alpha)$, which can be exploited by substituting into (10.2.33), 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 10.3. The detailed derivations are left as exercises for the reader.

Although the frequency sampling method provides us with another means for designing linear-phase FIR filters, its major advantage lies in the efficient frequency-sampling structure, which is obtained when most of the frequency samples are zero, as demonstrated in Section 9.2.3.

The following examples illustrate the design of linear-phase FIR filters based on the frequency-sampling method. The optimum values for the samples in the transition band are obtained from the tables in Appendix B, which are taken from the paper by Rabiner et al. (1970).

EXAMPLE 10.2.1

Determine the coefficients of a linear-phase FIR filter of length $M = 15$ which has a symmetric unit sample response and a frequency response that satisfies the conditions

$$H_r \left(\frac{2\pi k}{15} \right) = \begin{cases} 1, & k = 0, 1, 2, 3 \\ 0.4, & k = 4 \\ 0, & k = 5, 6, 7 \end{cases}$$

TABLE 10.3 Unit Sample Response: $h(n) = \pm h(M - 1 - n)$

Symmetric	
	$H(k) = G(k)e^{j\pi k/M}, \quad k = 0, 1, \dots, M - 1$
$\alpha = 0$	$G(k) = (-1)^k H_r\left(\frac{2\pi k}{M}\right), \quad G(k) = -G(M - k)$ $h(n) = \frac{1}{M} \left\{ G(0) + 2 \sum_{k=1}^U G(k) \cos \frac{2\pi k}{M} (n + \frac{1}{2}) \right\}$ $U = \begin{cases} \frac{M-1}{2}, & M \text{ odd} \\ \frac{M}{2} - 1, & M \text{ even} \end{cases}$ $H\left(k + \frac{1}{2}\right) = G\left(k + \frac{1}{2}\right) e^{-j\pi/2} e^{j\pi(2k+1)/2M}$ $G\left(k + \frac{1}{2}\right) = (-1)^k H_r\left[\frac{2\pi}{M}\left(k + \frac{1}{2}\right)\right]$ $G\left(k + \frac{1}{2}\right) = G\left(M - k - \frac{1}{2}\right)$ $h(n) = \frac{2}{M} \sum_{k=0}^U G\left(k + \frac{1}{2}\right) \sin \frac{2\pi}{M} \left(k + \frac{1}{2}\right) \left(n + \frac{1}{2}\right)$
Antisymmetric	
	$H(k) = G(k)e^{j\pi/2}e^{j\pi k/M}, \quad k = 0, 1, \dots, M - 1$
$\alpha = 0$	$G(k) = (-1)^k H_r\left(\frac{2\pi k}{M}\right), \quad G(k) = G(M - k)$ $h(n) = -\frac{2}{M} \sum_{k=1}^{(M-1)/2} G(k) \sin \frac{2\pi k}{M} \left(n + \frac{1}{2}\right), \quad M \text{ odd}$ $h(n) = \frac{1}{M} \left\{ (-1)^{n+1} G(M/2) - 2 \sum_{k=1}^{(M/2)-1} G(k) \sin \frac{2\pi}{M} k \left(n + \frac{1}{2}\right) \right\}, \quad M \text{ even}$ $H\left(k + \frac{1}{2}\right) = G\left(k + \frac{1}{2}\right) e^{j\pi(2k+1)/2M}$ $G\left(k + \frac{1}{2}\right) = (-1)^k H_r\left[\frac{2\pi}{M}\left(k + \frac{1}{2}\right)\right]$ $G\left(k + \frac{1}{2}\right) = -G\left(M - k - \frac{1}{2}\right); \quad G(M/2) = 0 \text{ for } M \text{ odd}$ $h(n) = \frac{2}{M} \sum_{k=0}^V G\left(k + \frac{1}{2}\right) \cos \frac{2\pi}{M} \left(k + \frac{1}{2}\right) \left(n + \frac{1}{2}\right)$ $V = \begin{cases} \frac{M-3}{2}, & M \text{ odd} \\ \frac{M}{2} - 1, & M \text{ even} \end{cases}$

Solution. Since $h(n)$ is symmetric and the frequencies are selected to correspond to the case $\alpha = 0$, we use the corresponding formula in Table 10.3 to evaluate $h(n)$. In this case

$$G(k) = (-1)^k H_r \left(\frac{2\pi k}{15} \right), \quad k = 0, 1, \dots, 7$$

The result of this computation is

$$h(0) = h(14) = -0.014112893$$

$$h(1) = h(13) = -0.001945309$$

$$h(2) = h(12) = 0.04000004$$

$$h(3) = h(11) = 0.01223454$$

$$h(4) = h(10) = -0.09138802$$

$$h(5) = h(9) = -0.01808986$$

$$h(6) = h(8) = 0.3133176$$

$$h(7) = 0.52$$

The frequency response characteristic of this filter is shown in Fig. 10.2.12. We should emphasize that $H_r(\omega)$ is exactly equal to the values given by the specifications above at $\omega_k = 2\pi k/15$.

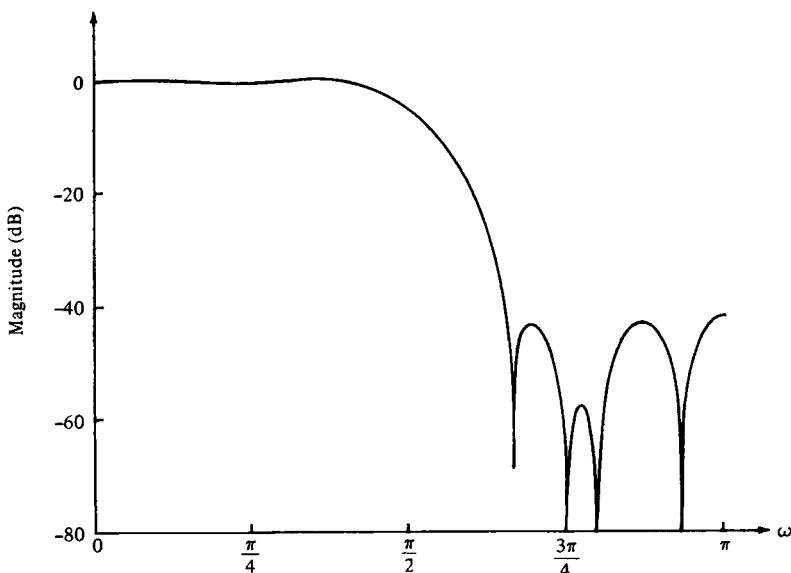


Figure 10.2.12 Frequency response of linear-phase FIR filter in Example 10.2.1.

EXAMPLE 10.2.2

Determine the coefficients of a linear-phase FIR filter of length $M = 32$ which has a symmetric unit sample response and a frequency response that satisfies the condition

$$H_r \left(\frac{2\pi(k + \alpha)}{32} \right) = \begin{cases} 1, & k = 0, 1, 2, 3, 4, 5 \\ T_1, & k = 6 \\ 0, & k = 7, 8, \dots, 15 \end{cases}$$

where $T_1 = 0.3789795$ for $\alpha = 0$, and $T_1 = 0.3570496$ for $\alpha = \frac{1}{2}$. These values of T_1 were obtained from the tables of optimum transition parameters given in Appendix B.

Solution. The appropriate equations for this computation are given in Table 10.3 for $\alpha = 0$ and $\alpha = \frac{1}{2}$. These computations yield the unit sample responses shown in Table 10.4. The corresponding frequency response characteristics are illustrated in Figs. 10.2.13 and 10.2.14, respectively. Note that the bandwidth of the filter for $\alpha = \frac{1}{2}$ is wider than that for $\alpha = 0$.

TABLE 10.4

$M = 32$	$M = 32$
$\text{ALPHA} = 0.$	$\text{ALPHA} = 0.5$
$T1 = 0.3789795E+00$	$T1 = 0.3570496E+00$
$h(0) = -0.7141978E-02$	$h(0) = -0.4089120E-02$
$h(1) = -0.3070801E-02$	$h(1) = -0.9973779E-02$
$h(2) = 0.5891327E-02$	$h(2) = -0.7379891E-02$
$h(3) = 0.1349923E-01$	$h(3) = 0.5949799E-02$
$h(4) = 0.8087033E-02$	$h(4) = 0.1727056E-01$
$h(5) = -0.1107258E-01$	$h(5) = 0.7878412E-02$
$h(6) = -0.2420687E-01$	$h(6) = -0.1798590E-01$
$h(7) = -0.9446550E-02$	$h(7) = -0.2670584E-01$
$h(8) = 0.2544464E-01$	$h(8) = 0.3778549E-02$
$h(9) = 0.3985050E-01$	$h(9) = 0.4191022E-01$
$h(10) = 0.2753036E-02$	$h(10) = 0.2839344E-01$
$h(11) = -0.5913959E-01$	$h(11) = -0.4163144E-01$
$h(12) = -0.6841660E-01$	$h(12) = -0.8254962E-01$
$h(13) = 0.3175741E-01$	$h(13) = 0.2802212E-02$
$h(14) = 0.2080981E+00$	$h(14) = 0.2013655E+00$
$h(15) = 0.3471138E+00$	$h(15) = 0.3717532E+00$

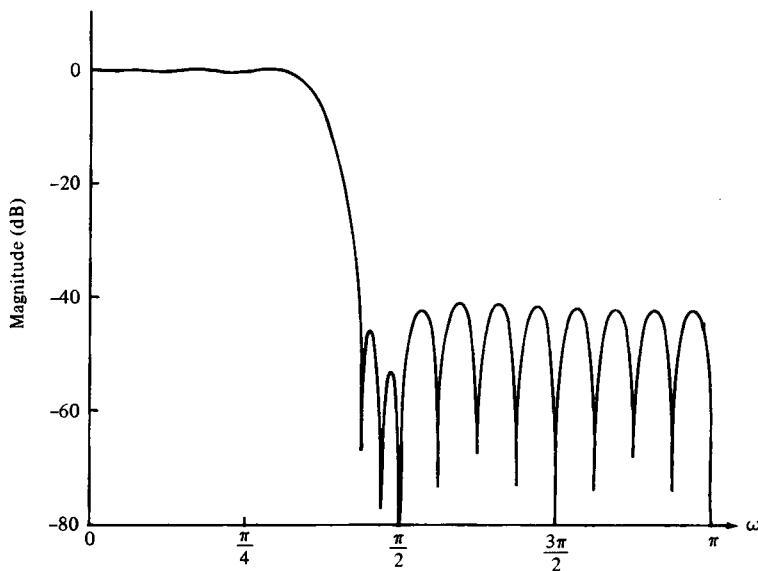


Figure 10.2.13 Frequency response of linear-phase FIR filter in Example 10.2.2 ($M = 32$ and $\alpha = 0$).

The optimization of the frequency samples in the transition region of the frequency response can be explained by evaluating the system function $H(z)$, given by (9.2.12), on the unit circle and using the relationship in (10.2.37) to express $H(\omega)$ in terms of $G(k + \alpha)$. Thus for the symmetric filter we obtain

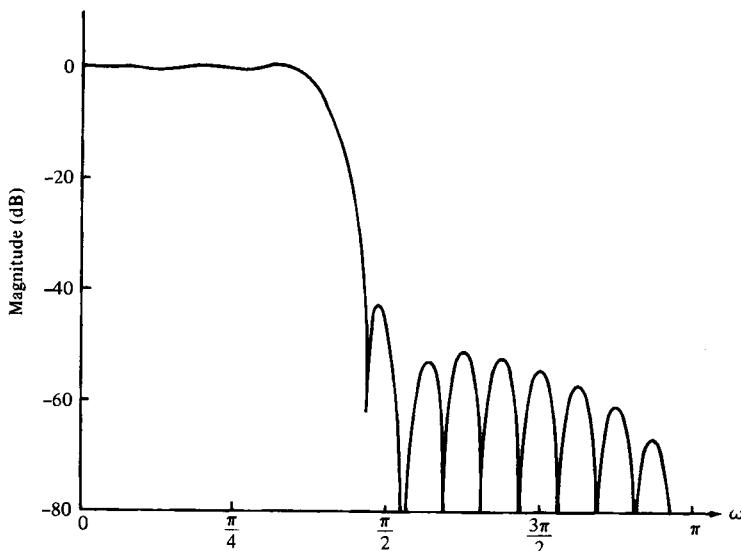


Figure 10.2.14 Frequency response of linear-phase FIR filter in Example 10.2.2 ($M = 32$ and $\alpha = \frac{1}{2}$).

$$H(\omega) = \left\{ \frac{\sin\left(\frac{\omega M}{2} - \pi\alpha\right)}{M} \sum_{k=0}^{M-1} \frac{G(k+\alpha)}{\sin\left[\frac{\omega}{2} - \frac{\pi}{M}(k+\alpha)\right]} \right\} e^{-j\omega(M-1)/2} \quad (10.2.38)$$

where

$$G(k+\alpha) = \begin{cases} -G(M-k), & \alpha = 0 \\ G(M-k-\frac{1}{2}), & \alpha = \frac{1}{2} \end{cases} \quad (10.2.39)$$

Similarly, for the antisymmetric linear-phase FIR filter we obtain

$$H(\omega) = \left\{ \frac{\sin\left(\frac{\omega M}{2} - \pi\alpha\right)}{M} \sum_{k=0}^{M-1} \frac{G(k+\alpha)}{\sin\left[\frac{\omega}{2} - \frac{\pi}{M}(k+\alpha)\right]} \right\} e^{-j\omega(M-1)/2} e^{j\pi/2} \quad (10.2.40)$$

where

$$G(k+\alpha) = \begin{cases} G(M-k), & \alpha = 0 \\ -G\left(M-k-\frac{1}{2}\right), & \alpha = \frac{1}{2} \end{cases} \quad (10.2.41)$$

With these expressions for the frequency response $H(\omega)$ given in terms of the desired frequency samples $\{G(k+\alpha)\}$, we can easily explain the method for selecting the parameters $\{G(k+\alpha)\}$ in the transition band which result in minimizing the peak sidelobe in the stopband. In brief, the values of $G(k+\alpha)$ in the passband are set to $(-1)^k$ and those in the stopband are set to zero. For any choice of $G(k+\alpha)$ in the transition band, the value of $H(\omega)$ is computed at a dense set of frequencies (e.g., at $\omega_n = 2\pi n/K$, $n = 0, 1, \dots, K-1$, where, for example, $K = 10M$). The value of the maximum sidelobe is determined, and the values of the parameters $\{G(k+\alpha)\}$ in the transition band are changed in a direction of steepest descent, which, in effect, reduces the maximum sidelobe. The computation of $H(\omega)$ is now repeated with the new choice of $\{G(k+\alpha)\}$. The maximum sidelobe of $H(\omega)$ is again determined and the values of the parameters $\{G(k+\alpha)\}$ in the transition band are adjusted in a direction of steepest descent, which, in turn, reduces the sidelobe. This iterative process is performed until it converges to the optimum choice of the parameters $\{G(k+\alpha)\}$ in the transition band.

There is a potential problem in the frequency-sampling realization of the FIR linear-phase filter. The frequency-sampling realization of the FIR filter introduces poles and zeros at equally spaced points on the unit circle. In the ideal situation, the zeros cancel the poles and, consequently, the actual zeros of $H(z)$ are determined by the selection of the frequency samples $\{H(k+\alpha)\}$. In a practical implementation of the frequency-sampling realization, however, quantization effects preclude a perfect cancellation of the poles and zeros. In fact, the location of poles on the unit circle provide no damping of the round-off noise that is introduced in the computations. As a result, such noise tends to increase with time and, ultimately, may destroy the normal operation of the filter.

To mitigate this problem, we can move both the poles and zeros from the unit circle to a circle just inside the unit circle, say at radius $r = 1 - \epsilon$, where ϵ is a very small number. Thus the system function of the linear-phase FIR filter becomes

$$H(z) = \frac{1 - r^M z^{-M} e^{j2\pi\alpha}}{M} \sum_{k=0}^{M-1} \frac{H(k + \alpha)}{1 - r e^{j2\omega\pi(k+\alpha)/M} z^{-1}} \quad (10.2.42)$$

The corresponding two-pole filter realization given in Section 9.2.3 can be modified accordingly. The damping provided by selecting $r < 1$ ensures that roundoff noise will be bounded and thus instability is avoided.

10.2.4 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 10.2.6, 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. 10.1.2, in the passband, the filter frequency response satisfies the condition

$$1 - \delta_1 \leq H_r(\omega) \leq 1 + \delta_1, \quad |\omega| \leq \omega_p \quad (10.2.43)$$

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

$$-\delta_2 \leq H_r(\omega) \leq \delta_2, \quad |\omega| > \omega_s \quad (10.2.44)$$

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.

Let us focus on the four different cases that result in a linear-phase FIR filter. These cases were treated in Section 10.2.2 and are summarized below.

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

$$H_r(\omega) = h\left(\frac{M-1}{2}\right) + 2 \sum_{n=0}^{(M-3)/2} h(n) \cos \omega \left(\frac{M-1}{2} - n\right) \quad (10.2.45)$$

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

$$a(k) = \begin{cases} h\left(\frac{M-1}{2}\right), & k=0 \\ 2h\left(\frac{M-1}{2}-k\right), & k=1, 2, \dots, \frac{M-1}{2} \end{cases} \quad (10.2.46)$$

then (10.2.45) reduces to the compact form

$$H_r(\omega) = \sum_{k=0}^{(M-1)/2} a(k) \cos \omega k \quad (10.2.47)$$

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

$$H_r(\omega) = 2 \sum_{n=0}^{(M/2)-1} h(n) \cos \omega \left(\frac{M-1}{2} - n\right) \quad (10.2.48)$$

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

$$b(k) = 2h\left(\frac{M}{2} - k\right), \quad k = 1, 2, \dots, M/2 \quad (10.2.49)$$

With these substitutions (10.2.48) becomes

$$H_r(\omega) = \sum_{k=1}^{M/2} b(k) \cos \omega \left(k - \frac{1}{2}\right) \quad (10.2.50)$$

In carrying out the optimization, it is convenient to rearrange (10.2.50) further into the form

$$H_r(\omega) = \cos \frac{\omega}{2} \sum_{k=0}^{(M/2)-1} \tilde{b}(k) \cos \omega k \quad (10.2.51)$$

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

$$\begin{aligned} \tilde{b}(0) &= \frac{1}{2}b(1), & \tilde{b}(1) &= 2b(1) - 2b(0) \\ \tilde{b}(k) &= 2b(k) - \tilde{b}(k-1), & k &= 1, 2, 3, \dots, \frac{M}{2} - 2 \end{aligned} \quad (10.2.52)$$

$$\tilde{b}\left(\frac{M}{2} - 1\right) = 2b\left(\frac{M}{2}\right)$$

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

$$H_r(\omega) = 2 \sum_{n=0}^{(M-3)/2} h(n) \sin \omega \left(\frac{M-1}{2} - n \right) \quad (10.2.53)$$

If we change the summation in (10.2.53) from n to $k = (M-1)/2 - n$ and define a new set of filter parameters $\{c(k)\}$ as

$$c(k) = 2h\left(\frac{M-1}{2} - k\right), \quad k = 1, 2, \dots, (M-1)/2 \quad (10.2.54)$$

then (10.2.53) becomes

$$H_r(\omega) = \sum_{k=1}^{(M-1)/2} c(k) \sin \omega k \quad (10.2.55)$$

As in the previous case, it is convenient to rearrange (10.2.55) into the form

$$H_r(\omega) = \sin \omega \sum_{k=0}^{(M-3)/2} \tilde{c}(k) \cos \omega k \quad (10.2.56)$$

where the coefficients $\{\tilde{c}(k)\}$ are linearly related to the parameters $\{c(k)\}$. This desired relationship can be derived from (10.2.55) and (10.2.56) and is simply given as

$$\begin{aligned} \tilde{c}\left(\frac{M-3}{2}\right) &= c\left(\frac{M-1}{2}\right) \\ \tilde{c}\left(\frac{M-5}{2}\right) &= 2c\left(\frac{M-3}{2}\right) \\ &\vdots && \vdots \\ \tilde{c}(k-1) - \tilde{c}(k+1) &= 2c(k), & 2 \leq k \leq \frac{M-5}{2} \\ \tilde{c}(0) - \frac{1}{2}\tilde{c}(2) &= c(1) \end{aligned} \quad (10.2.57)$$

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

$$H_r(\omega) = 2 \sum_{n=0}^{(M/2)-1} h(n) \sin \omega \left(\frac{M-1}{2} - n \right) \quad (10.2.58)$$

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

$$d(k) = 2h\left(\frac{M}{2} - k\right), \quad k = 1, 2, \dots, \frac{M}{2} \quad (10.2.59)$$

results in the expression

$$H_r(\omega) = \sum_{k=1}^{M/2} d(k) \sin \omega \left(k - \frac{1}{2} \right) \quad (10.2.60)$$

As in the previous two cases, we find it convenient to rearrange (10.2.60) into the form

$$H_r(\omega) = \sin \frac{\omega}{2} \sum_{k=0}^{(M/2)-1} \tilde{d}(k) \cos \omega k \quad (10.2.61)$$

where the new filter parameters $\{\tilde{d}(k)\}$ are related to $\{d(k)\}$ as follows:

$$\begin{aligned} \tilde{d}\left(\frac{M}{2} - 1\right) &= 2d\left(\frac{M}{2}\right) \\ \tilde{d}(k-1) - \tilde{d}(k) &= 2d(k), \quad 2 \leq k \leq \frac{M}{2} - 1 \\ \tilde{d}(0) - \frac{1}{2}\tilde{d}(1) &= d(1) \end{aligned} \quad (10.2.62)$$

TABLE 10.5 Real-Valued Frequency Response Functions for Linear-Phase FIR Filters

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

The expressions for $H_r(\omega)$ in these four cases are summarized in Table 10.5. We note that the rearrangements that we made in Cases 2, 3, and 4 have allowed us to express $H_r(\omega)$ as

$$H_r(\omega) = Q(\omega)P(\omega) \quad (10.2.63)$$

where

$$Q(\omega) = \begin{cases} 1 & \text{Case 1} \\ \cos \frac{\omega}{2} & \text{Case 2} \\ \sin \omega & \text{Case 3} \\ \sin \frac{\omega}{2} & \text{Case 4} \end{cases} \quad (10.2.64)$$

and $P(\omega)$ has the common form

$$P(\omega) = \sum_{k=0}^L \alpha(k) \cos \omega k \quad (10.2.65)$$

with $\{\alpha(k)\}$ representing the parameters of the filter, which are linearly related to the unit sample response $h(n)$ of the FIR filter. The upper limit L in the sum is $L = (M - 1)/2$ for Case 1, $L = (M - 3)/2$ for Case 3, and $L = M/2 - 1$ for Case 2 and Case 4.

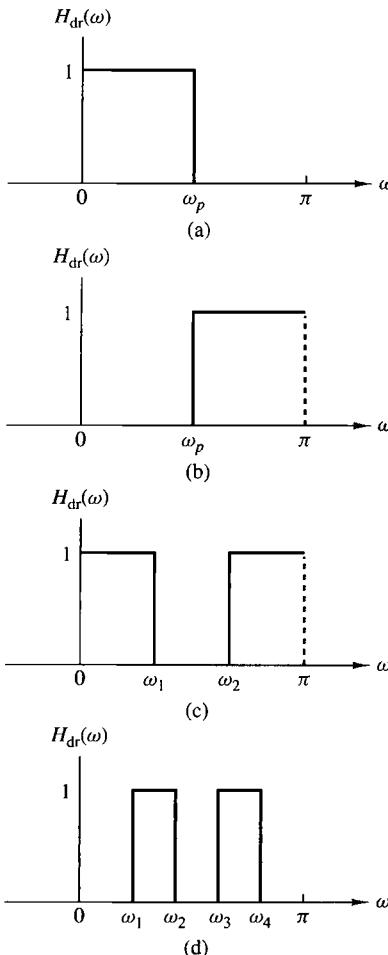
In addition to the common framework given above for the representation of $H_r(\omega)$, we also define the real-valued desired frequency response $H_{dr}(\omega)$ and the weighting function $W(\omega)$ on the approximation error. The real-valued desired frequency response $H_{dr}(\omega)$ is simply defined to be unity in the passband and zero in the stopband. For example, Fig. 10.2.15 illustrates several different types of characteristics for $H_{dr}(\omega)$. The weighting function on the approximation error allows us to choose the relative size of the errors in the different frequency bands (i.e., in the passband and in the stopband). In particular, it is convenient to normalize $W(\omega)$ to unity in the stopband and set $W(\omega) = \delta_2/\delta_1$ in the passband, that is,

$$W(\omega) = \begin{cases} \delta_2/\delta_1, & \omega \text{ in the passband} \\ 1, & \omega \text{ in the stopband} \end{cases} \quad (10.2.66)$$

Then we simply select $W(\omega)$ in the passband to reflect our emphasis on the relative size of the ripple in the stopband to the ripple in the passband.

With the specification of $H_{dr}(\omega)$ and $W(\omega)$, we can now define the weighted approximation error as

$$\begin{aligned} E(\omega) &= W(\omega)[H_{dr}(\omega) - H_r(\omega)] \\ &= W(\omega)[H_{dr}(\omega) - Q(\omega)P(\omega)] \\ &= W(\omega)Q(\omega) \left[\frac{H_{dr}(\omega)}{Q(\omega)} - P(\omega) \right] \end{aligned} \quad (10.2.67)$$

**Figure 10.2.15**

Desired frequency response characteristics for different types of filters.

For mathematical convenience, we define a modified weighting function $\hat{W}(\omega)$ and a modified desired frequency response $\hat{H}_{\text{dr}}(\omega)$ as

$$\begin{aligned}\hat{W}(\omega) &= W(\omega)Q(\omega) \\ \hat{H}_{\text{dr}}(\omega) &= \frac{H_{\text{dr}}(\omega)}{Q(\omega)}\end{aligned}\tag{10.2.68}$$

Then the weighted approximation error may be expressed as

$$E(\omega) = \hat{W}(\omega)[\hat{H}_{\text{dr}}(\omega) - P(\omega)]\tag{10.2.69}$$

for all four different types of linear-phase FIR filters.

Given the error function $E(\omega)$, the Chebyshev approximation problem is basically to determine the filter parameters $\{\alpha(k)\}$ that minimize the maximum absolute

value of $E(\omega)$ over the frequency bands in which the approximation is to be performed. In mathematical terms, we seek the solution to the problem

$$\min_{\text{over } \{\alpha(k)\}} \left[\max_{\omega \in S} |E(\omega)| \right] = \min_{\text{over } \{\alpha(k)\}} \left[\max_{\omega \in S} |\hat{W}(\omega)[\hat{H}_{\text{dr}}(\omega) - \sum_{k=0}^L \alpha(k) \cos \omega k]| \right] \quad (10.2.70)$$

where S represents the set (disjoint union) of frequency bands over which the optimization is to be performed. Basically, the set S consists of the passbands and stopbands of the desired filter.

The solution to this problem is due to Parks and McClellan (1972a), who applied a theorem in the theory of Chebyshev approximation. It is called the *alternation theorem*, which we state without proof.

Alternation Theorem. Let S be a compact subset of the interval $[0, \pi)$. A necessary and sufficient condition for

$$P(\omega) = \sum_{k=0}^L \alpha(k) \cos \omega k$$

to be the unique, best weighted Chebyshev approximation to $\hat{H}_{\text{dr}}(\omega)$ in S , is that the error function $E(\omega)$ exhibit at least $L + 2$ extremal frequencies in S . That is, there must exist at least $L + 2$ frequencies $\{\omega_i\}$ in S such that $\omega_1 < \omega_2 < \dots < \omega_{L+2}$, $E(\omega_i) = -E(\omega_{i+1})$, and

$$|E(\omega_i)| = \max_{\omega \in S} |E(\omega)|, \quad i = 1, 2, \dots, L + 2$$

We note that the error function $E(\omega)$ alternates in sign between two successive extremal frequencies. Hence the theorem is called the alternation theorem.

To elaborate on the alternation theorem, let us consider the design of a lowpass filter with passband $0 \leq \omega \leq \omega_p$ and stopband $\omega_s \leq \omega \leq \pi$. Since the desired frequency response $H_{\text{dr}}(\omega)$ and the weighting function $W(\omega)$ are piecewise constant, we have

$$\begin{aligned} \frac{dE(\omega)}{d\omega} &= \frac{d}{d\omega} \{W(\omega)[H_{\text{dr}}(\omega) - H_r(\omega)]\} \\ &= -\frac{dH_r(\omega)}{d\omega} = 0 \end{aligned}$$

Consequently, the frequencies $\{\omega_i\}$ corresponding to the peaks of $E(\omega)$ also correspond to peaks at which $H_r(\omega)$ meets the error tolerance. Since $H_r(\omega)$ is a trigonometric polynomial of degree L , for Case 1, for example,

$$\begin{aligned}
H_r(\omega) &= \sum_{k=0}^L \alpha(k) \cos \omega k \\
&= \sum_{k=0}^L \alpha(k) \left[\sum_{n=0}^k \beta_{nk} (\cos \omega)^n \right] \\
&= \sum_{k=0}^L \alpha'(k) (\cos \omega)^k
\end{aligned} \tag{10.2.71}$$

it follows that $H_r(\omega)$ can have at most $L - 1$ local maxima and minima in the open interval $0 < \omega < \pi$. In addition, $\omega = 0$ and $\omega = \pi$ are usually extrema of $H_r(\omega)$ and, also, of $E(\omega)$. Therefore, $H_r(\omega)$ has at most $L + 1$ extremal frequencies. Furthermore, the band-edge frequencies ω_p and ω_s are also extrema of $E(\omega)$, since $|E(\omega)|$ is maximum at $\omega = \omega_p$ and $\omega = \omega_s$. As a consequence, there are at most $L + 3$ extremal frequencies in $E(\omega)$ for the unique, best approximation of the ideal lowpass filter. On the other hand, the alternation theorem states that there are at least $L + 2$ extremal frequencies in $E(\omega)$. Thus the error function for the lowpass filter design has either $L + 3$ or $L + 2$ extrema. In general, filter designs that contain more than $L + 2$ alternations or ripples are called *extra ripple filters*. When the filter design contains the maximum number of alternations, it is called a *maximal ripple filter*.

The alternation theorem guarantees a unique solution for the Chebyshev optimization problem in (10.2.70). At the desired extremal frequencies $\{\omega_n\}$, we have the set of equations

$$\hat{W}(\omega_n)[\hat{H}_{dr}(\omega_n) - P(\omega_n)] = (-1)^n \delta, \quad n = 0, 1, \dots, L + 1 \tag{10.2.72}$$

where δ represents the maximum value of the error function $E(\omega)$. In fact, if we select $W(\omega)$ as indicated by (10.2.66), it follows that $\delta = \delta_2$.

The set of linear equations in (10.2.72) can be rearranged as

$$P(\omega_n) + \frac{(-1)^n \delta}{\hat{W}(\omega_n)} = \hat{H}_{dr}(\omega_n), \quad n = 0, 1, \dots, L + 1$$

or, equivalently, in the form

$$\sum_{k=0}^L \alpha(k) \cos \omega_n k + \frac{(-1)^n \delta}{\hat{W}(\omega_n)} = \hat{H}_{dr}(\omega_n), \quad n = 0, 1, \dots, L + 1 \tag{10.2.73}$$

If we treat the $\{a(k)\}$ and δ as the parameters to be determined, (10.2.73) can be expressed in matrix form as

$$\begin{bmatrix} 1 & \cos \omega_0 & \cos 2\omega_0 & \cdots & \cos L\omega_0 & \frac{1}{\hat{W}(\omega_0)} \\ 1 & \cos \omega_1 & \cos 2\omega_1 & \cdots & \cos L\omega_1 & \frac{-1}{\hat{W}(\omega_1)} \\ \vdots & & & & & \\ \vdots & & & & & \\ 1 & \cos \omega_{L+1} & \cos 2\omega_{L+1} & \cdots & \cos L\omega_{L+1} & \frac{(-1)^{L+1}}{\hat{W}(\omega_{L+1})} \end{bmatrix} \begin{bmatrix} \alpha(0) \\ \alpha(1) \\ \vdots \\ \alpha(L) \\ \delta \end{bmatrix} = \begin{bmatrix} \hat{H}_{dr}(\omega_0) \\ \hat{H}_{dr}(\omega_1) \\ \vdots \\ \vdots \\ \hat{H}_{dr}(\omega_{L+1}) \end{bmatrix} \quad (10.2.74)$$

Initially, we know neither the set of extremal frequencies $\{\omega_n\}$ nor the parameters $\{\alpha(k)\}$ and δ . To solve for the parameters, we use an iterative algorithm, called the *Remez exchange algorithm* [see Rabiner et al. (1975)], in which we begin by guessing at the set of extremal frequencies, determine $P(\omega)$ and δ , and then compute the error function $E(\omega)$. From $E(\omega)$ we determine another set of $L + 2$ extremal frequencies and repeat the process iteratively until it converges to the optimal set of extremal frequencies. Although the matrix equation in (10.2.74) can be used in the iterative procedure, matrix inversion is time consuming and inefficient.

A more efficient procedure, suggested in the paper by Rabiner et al. (1975), is to compute δ analytically, according to the formula

$$\delta = \frac{\gamma_0 \hat{H}_{dr}(\omega_0) + \gamma_1 \hat{H}_{dr}(\omega_1) + \cdots + \gamma_{L+1} \hat{H}_{dr}(\omega_{L+1})}{\frac{\gamma_0}{\hat{W}(\omega_0)} - \frac{\gamma_1}{\hat{W}(\omega_1)} + \cdots + \frac{(-1)^{L+1} \gamma_{L+1}}{\hat{W}(\omega_{L+1})}} \quad (10.2.75)$$

where

$$\gamma_k = \prod_{\substack{n=0 \\ n \neq k}}^{L+1} \frac{1}{\cos \omega_k - \cos \omega_n} \quad (10.2.76)$$

The expression for δ in (10.2.75) follows immediately from the matrix equation in (10.2.74). Thus with an initial guess at the $L + 2$ extremal frequencies, we compute δ .

Now since $P(\omega)$ is a trigonometric polynomial of the form

$$P(\omega) = \sum_{k=0}^L \alpha(k) x^k, \quad x = \cos \omega$$

and since we know that the polynomial at the points $x_n \equiv \cos \omega_n$, $n = 0, 1, \dots, L + 1$, has the corresponding values

$$P(\omega_n) = \hat{H}_{dr}(\omega_n) - \frac{(-1)^n \delta}{\hat{W}(\omega_n)}, \quad n = 0, 1, \dots, L + 1 \quad (10.2.77)$$

we can use the Lagrange interpolation formula for $P(\omega)$. Thus $P(\omega)$ can be expressed as [see Hamming (1962)]

$$P(\omega) = \frac{\sum_{k=0}^L P(\omega_k) [\beta_k / (x - x_k)]}{\sum_{k=0}^L [\beta_k / (x - x_k)]} \quad (10.2.78)$$

where $P(\omega_n)$ is given by (10.2.77), $x = \cos \omega$, $x_k = \cos \omega_k$, and

$$\beta_k = \prod_{\substack{n=0 \\ n \neq k}}^L \frac{1}{x_k - x_n} \quad (10.2.79)$$

Having the solution for $P(\omega)$, we can now compute the error function $E(\omega)$ from

$$E(\omega) = \hat{W}(\omega) [\hat{H}_{dr}(\omega) - P(\omega)] \quad (10.2.80)$$

on a dense set of frequency points. Usually, a number of points equal to $16M$, where M is the length of the filter, suffices. If $|E(\omega)| \geq \delta$ for some frequencies on the dense set, then a new set of frequencies corresponding to the $L + 2$ largest peaks of $|E(\omega)|$ are selected and the computational procedure beginning with (10.2.75) is repeated. Since the new set of $L + 2$ extremal frequencies is selected to correspond to the peaks of the error function $|E(\omega)|$, the algorithm forces δ to increase in each iteration until it converges to the upper bound and hence to the optimum solution for the Chebyshev approximation problem. In other words, when $|E(\omega)| \leq \delta$ for all frequencies on the dense set, the optimal solution has been found in terms of the polynomial $H(\omega)$. A flowchart of the algorithm is shown in Fig. 10.2.16 and is due to Remez (1957).

Once the optimal solution has been obtained in terms of $P(\omega)$, the unit sample response $h(n)$ can be computed directly, without having to compute the parameters $\{\alpha(k)\}$. In effect, we have determined

$$H_r(\omega) = Q(\omega)P(\omega)$$

which can be evaluated at $\omega = 2\pi k/M$, $k = 0, 1, \dots, (M-1)/2$, for M odd, or $M/2$ for M even. Then, depending on the type of filter being designed, $h(n)$ can be determined from the formulas given in Table 10.3.

A computer program written by Parks and McClellan (1972b) is available for designing linear-phase FIR filters based on the Chebyshev approximation criterion and implemented with the Remez exchange algorithm. This program can be used to design lowpass, highpass, or bandpass filters, differentiators, and Hilbert transformers. The latter two types of filters are described in the following sections. A number of software packages for designing equiripple linear-phase FIR filters are now available.

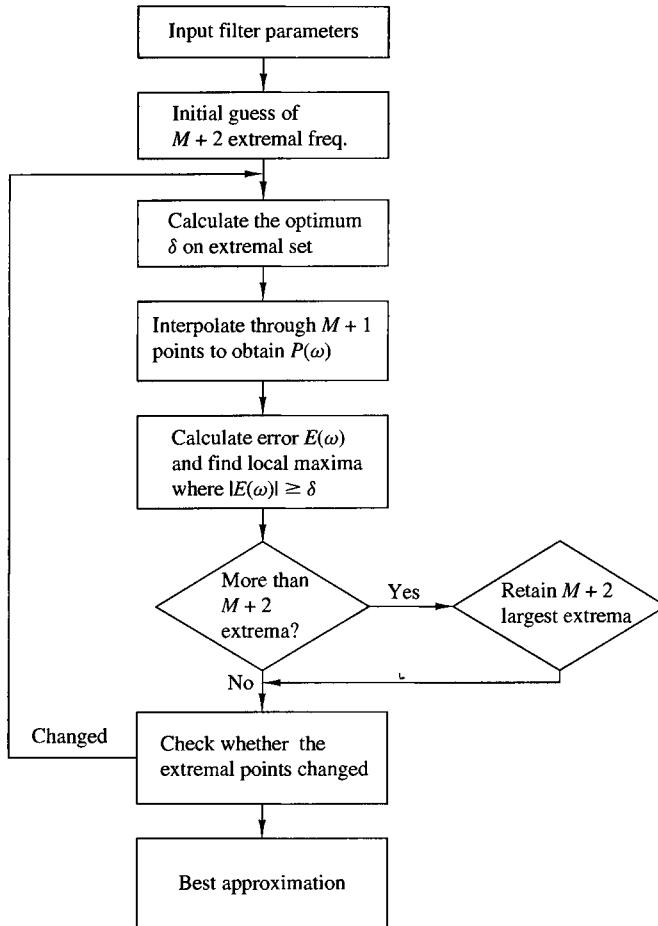


Figure 10.2.16 Flowchart of Remez algorithm.

The Parks–McClellan program requires a number of input parameters which determine the filter characteristics. In particular, the following parameters must be specified:

- NFILT: The filter length, denoted above as M .
- JTYPE: Type of filter:
 JTYPE = 1 results in a multiple passband/stopband filter.
 JTYPE = 2 results in a differentiator.
 JTYPE = 3 results in a Hilbert transformer.
- NBANDS: The number of frequency bands from 2 (for a lowpass filter) to a maximum of 10 (for a multiple-band filter).
- LGRID: The grid density for interpolating the error function $E(\omega)$. The default value is 16 if left unspecified.
- EDGE: The frequency bands specified by lower and upper cutoff frequencies,

up to a maximum of 10 bands (an array of size 20, maximum). The frequencies are given in terms of the variable $f = \omega/2\pi$, where $f = 0.5$ corresponds to the folding frequency.

FX: An array of maximum size 10 that specifies the desired frequency response $H_{dr}(\omega)$ in each band.

WTX: An array of maximum size 10 that specifies the weight function in each band.

The following examples demonstrate the use of this program to design a lowpass and a bandpass filter.

EXAMPLE 10.2.3

Design a lowpass filter of length $M = 61$ with a passband edge frequency $f_p = 0.1$ and a stopband edge frequency $f_s = 0.15$.

Solution. The lowpass filter is a two-band filter with passband edge frequencies $(0, 0.1)$ and stopband edge frequencies $(0.15, 0.5)$. The desired response is $(1, 0)$ and the weight function is arbitrarily selected as $(1, 1)$.

```
61, 1, 2
0.0, 0.1, 0.15, 0.5
1.0, 0.0
1.0, 1.0
```

The impulse response and frequency response are shown in Fig. 10.2.17. The resulting filter has a stopband attenuation of -56 dB and a passband ripple of 0.0135 dB.

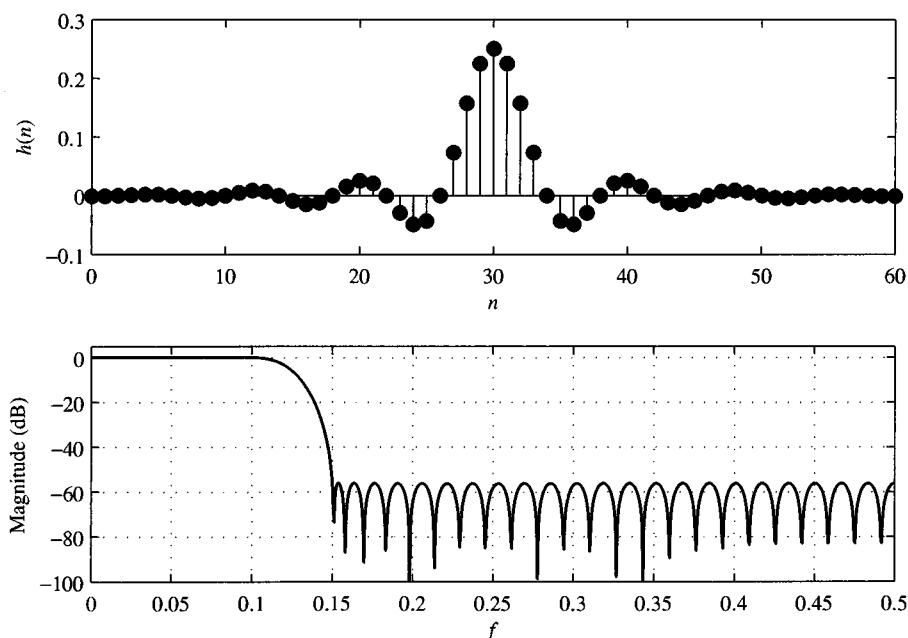


Figure 10.2.17 Impulse response and frequency response of $M = 61$ FIR filter in Example 10.2.3.

If we increase the length of the filter to $M = 101$ while maintaining all the other parameters given above the same, the resulting filter has the impulse response and frequency response characteristics shown in Fig. 10.2.18. Now, the stopband attenuation is -85 dB and the passband ripple is reduced to 0.00046 dB.

We should indicate that it is possible to increase the attenuation in the stopband by keeping the filter length fixed, say at $M = 61$, and decreasing the weighting function $W(\omega) = \delta_2/\delta_1$ in the passband. With $M = 61$ and a weighting function $(0.1, 1)$, we obtain a filter that has a stopband attenuation of -65 dB and a passband ripple of 0.049 dB.

EXAMPLE 10.2.4

Design a bandpass filter of length $M = 32$ with passband edge frequencies $f_{p1} = 0.2$ and $f_{p2} = 0.35$ and stopband edge frequencies of $f_{s1} = 0.1$ and $f_{s2} = 0.425$.

Solution. This passband filter is a three-band filter with a stopband range of $(0, 0.1)$, a passband range of $(0.2, 0.35)$, and a second stopband range of $(0.425, 0.5)$. The weighting function is selected as $(10.0, 1.0, 10.0)$, or as $(1.0, 0.1, 1.0)$, and the desired response in the three bands is $(0.0, 1.0, 0.0)$. Thus the input parameters to the program are

```
32, 1, 3
0.0, 0.1, 0.2, 0.35, 0.425, 0.5
0.0, 1.0, 0.0
10.0, 1.0, 10.0
```

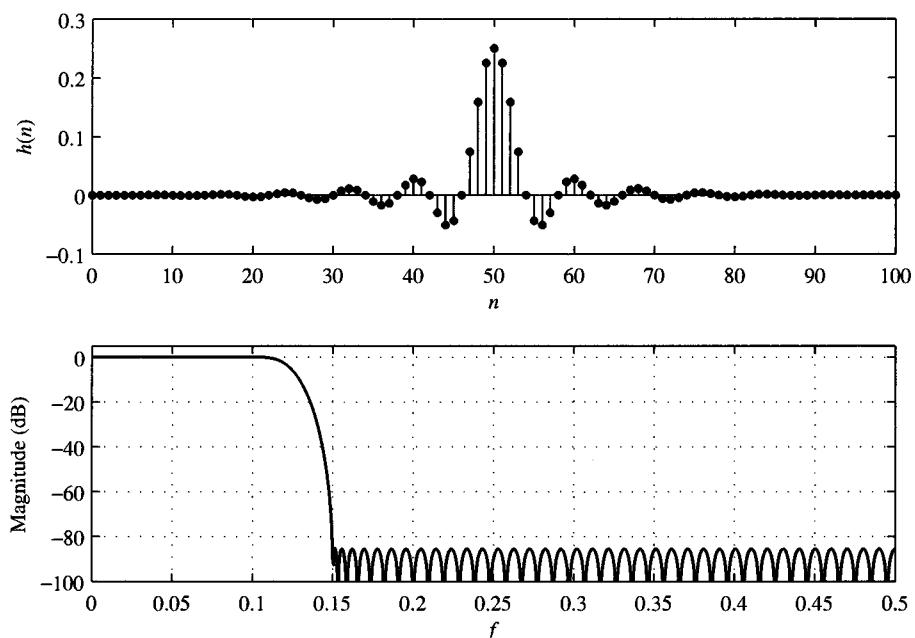


Figure 10.2.18 Impulse response and frequency response of $M = 101$ FIR filter in Example 10.2.3.

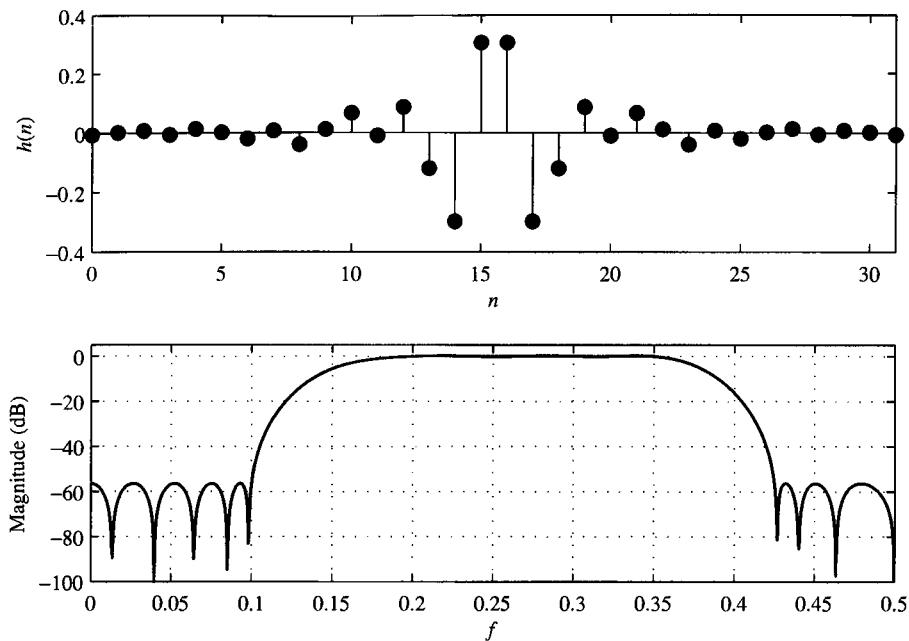


Figure 10.2.19 Impulse response and frequency response of $M = 32$ FIR filter in Example 10.2.4.

We note that the ripple in the stopbands δ_2 is 10 times smaller than the ripple in the passband due to the fact that errors in the stopband were given a weight of 10 compared to the passband weight of unity. The impulse response and frequency response of the bandpass filter are illustrated in Fig. 10.2.19.

These examples serve to illustrate the relative ease with which optimal lowpass, highpass, bandstop, bandpass, and more general multiband linear-phase FIR filters can be designed based on the Chebyshev approximation criterion implemented by means of the Remez exchange algorithm. In the next two sections we consider the design of differentiators and Hilbert transformers.

10.2.5 Design of FIR Differentiators

Differentiators are used in many analog and digital systems to take the derivative of a signal. An ideal differentiator has a frequency response that is linearly proportional to frequency. Similarly, an ideal digital differentiator is defined as one that has the frequency response

$$H_d(\omega) = j\omega, \quad -\pi \leq \omega \leq \pi \quad (10.2.81)$$

The unit sample response corresponding to $H_d(\omega)$ is

$$\begin{aligned} h_d(n) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(\omega) e^{j\omega n} d\omega \\ &= \frac{1}{2\pi} \int_{-\pi}^{\pi} j\omega e^{j\omega n} d\omega \\ &= \frac{\cos \pi n}{n}, \quad -\infty < n < \infty, \quad n \neq 0 \end{aligned} \quad (10.2.82)$$

We observe that the ideal differentiator has an antisymmetric unit sample response [i.e., $h_d(n) = -h_d(-n)$]. Hence, $h_d(0) = 0$.

In this section we consider the design of linear-phase FIR differentiators based on the Chebyshev approximation criterion. In view of the fact that the ideal differentiator has an antisymmetric unit sample response, we shall confine our attention to FIR designs in which $h(n) = -h(M-1-n)$. Hence we consider the filter types classified in the preceding section as Case 3 and Case 4.

We recall that in Case 3, where M is odd, the real-valued frequency response of the FIR filter $H_r(\omega)$ has the characteristic that $H_r(0) = 0$. A zero response at zero frequency is just the condition that the differentiator should satisfy, and we see from Table 10.5 that both filter types satisfy this condition. However, if a full-band differentiator is desired, this is impossible to achieve with an FIR filter having an odd number of coefficients, since $H_r(\pi) = 0$ for M odd. In practice, however, full-band differentiators are rarely required.

In most cases of practical interest, the desired frequency response characteristic need only be linear over the limited frequency range $0 \leq \omega \leq 2\pi f_p$, where f_p is called the bandwidth of the differentiator. In the frequency range $2\pi f_p < \omega \leq \pi$, the desired response may be either left unconstrained or constrained to be zero.

In the design of FIR differentiators based on the Chebyshev approximation criterion, the weighting function $W(\omega)$ is specified in the program as

$$W(\omega) = \frac{1}{\omega}, \quad 0 \leq \omega \leq 2\pi f_p \quad (10.2.83)$$

in order that the relative ripple in the passband be a constant. Thus the absolute error between the desired response ω and the approximation $H_r(\omega)$ increases as ω varies from 0 to $2\pi f_p$. However, the weighting function in (10.2.83) ensures that the relative error

$$\begin{aligned} \delta &= \max_{0 \leq \omega \leq 2\pi f_p} \{W(\omega)[\omega - H_r(\omega)]\} \\ &= \max_{0 \leq \omega \leq 2\pi f_p} \left[1 - \frac{H_r(\omega)}{\omega} \right] \end{aligned} \quad (10.2.84)$$

is fixed within the passband of the differentiator.

EXAMPLE 10.2.5

Use the Remez algorithm to design a linear-phase FIR differentiator of length $M = 60$. The passband edge frequency is 0.1 and the stopband edge frequency is 0.15.

Solution. The input parameters to the program are

$$\begin{array}{lll} 60, & 2, & 2 \\ 0.0, & 0.1, & 0.15, \quad 0.5 \\ 1.0, & 0.0 \\ 1.0, & 1.0 \end{array}$$

The frequency response characteristic is illustrated in Fig. 10.2.20. Also shown in the same figure is the approximation error over the passband $0 \leq f \leq 0.1$ of the filter.

The important parameters in a differentiator are its length M , its bandwidth {band-edge frequency} f_p , and the peak relative error δ of the approximation. The interrelationship among these three parameters can be easily displayed parametrically. In particular, the value of $20 \log_{10} \delta$ versus f_p with M as a parameter is shown in Fig. 10.2.21 for M even and in Fig. 10.2.22 for M odd. These results, due to Rabiner and Schafer (1974a), are useful in the selection of the filter length, given specifications on the inband ripple and the cutoff frequency f_p .

A comparison of the graphs in Figs. 10.2.21 and 10.2.22 reveals that even-length differentiators result in a significantly smaller approximation error δ than comparable odd-length differentiators. Designs based on M odd are particularly poor if the bandwidth exceeds $f_p = 0.45$. The problem is basically the zero in the frequency response at $\omega = \pi(f = 1/2)$. When $f_p < 0.45$, good designs are obtained for M odd, but comparable-length differentiators with M even are always better in the sense that the approximation error is smaller.

In view of the obvious advantage of even-length over odd-length differentiators, a conclusion might be that even-length differentiators are always preferable in practical systems. This is certainly true for many applications. However, we should note that the signal delay introduced by any linear-phase FIR filter is $(M - 1)/2$, which is not an integer when M is even. In many practical applications, this is unimportant. In some applications where it is desirable to have an integer-valued delay in the signal at the output of the differentiator, we must select M to be odd.

These numerical results are based on designs resulting from the Chebyshev approximation criterion. We wish to indicate it is also possible and relatively easy to design linear-phase FIR differentiators based on the frequency-sampling method. For example, Fig. 10.2.23 illustrates the frequency response characteristics of a wide-band ($f_p = 0.5$) differentiator, of length $M = 30$. The graph of the absolute value of the approximation error as a function of frequency is also shown in this figure.

10.2.6 Design of Hilbert Transformers

An ideal Hilbert transformer is an all-pass filter that imparts a 90° phase shift on the signal at its input. Hence the frequency response of the ideal Hilbert transformer is specified as

$$H_d(\omega) = \begin{cases} -j, & 0 < \omega \leq \pi \\ j, & -\pi < \omega < 0 \end{cases} \quad (10.2.85)$$

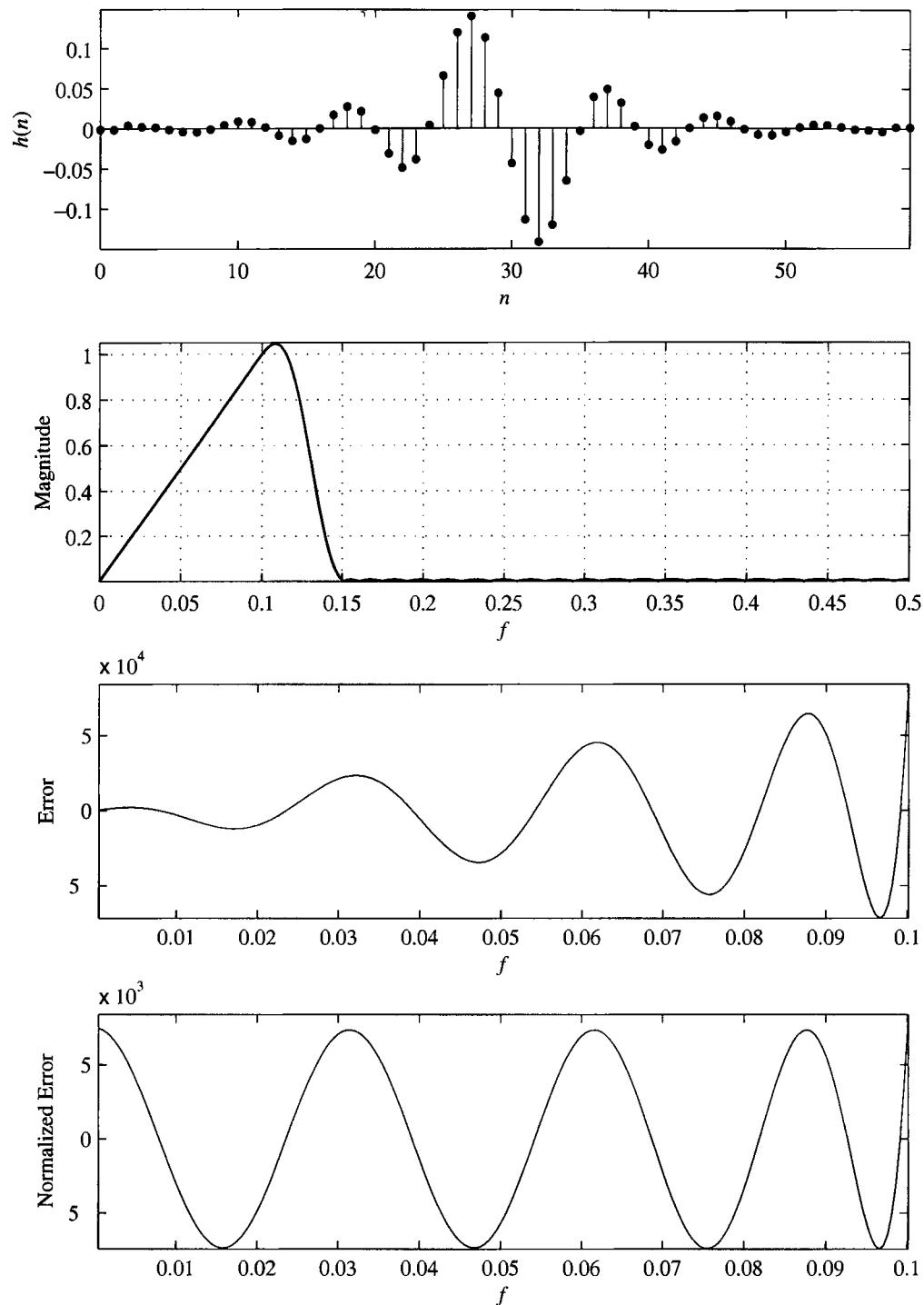


Figure 10.2.20 Frequency response and approximation error for $M = 60$ FIR differentiator of Example 10.2.5.

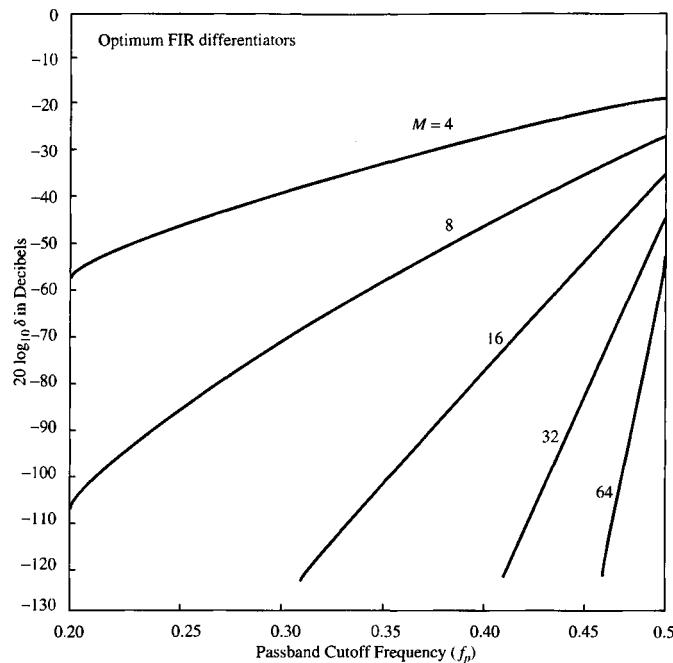


Figure 10.2.21 Curves of $20 \log_{10} \delta$ versus f_p for $M = 4, 8, 16, 32$, and 64 . [From paper by Rabiner and Schafer (1974a). Reprinted with permission of AT&T.]

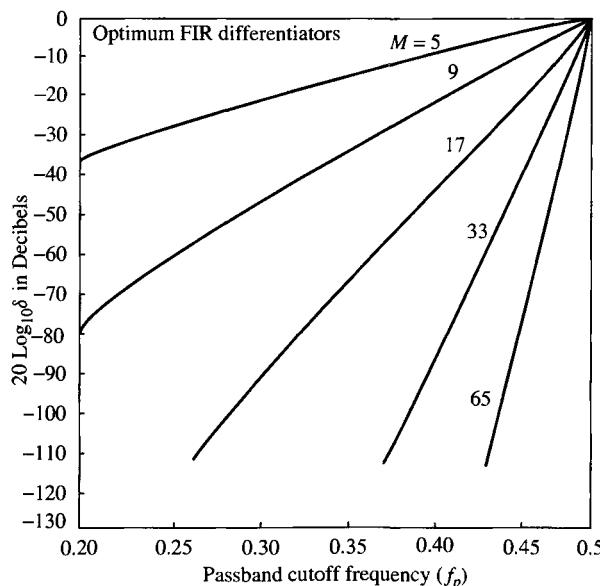


Figure 10.2.22 Curves of $20 \log_{10} \delta$ versus F_p for $M = 5, 9, 17, 33$ and 65 [From paper by Rabiner and Schafer (1974a). Reprinted with permission of AT&T.]

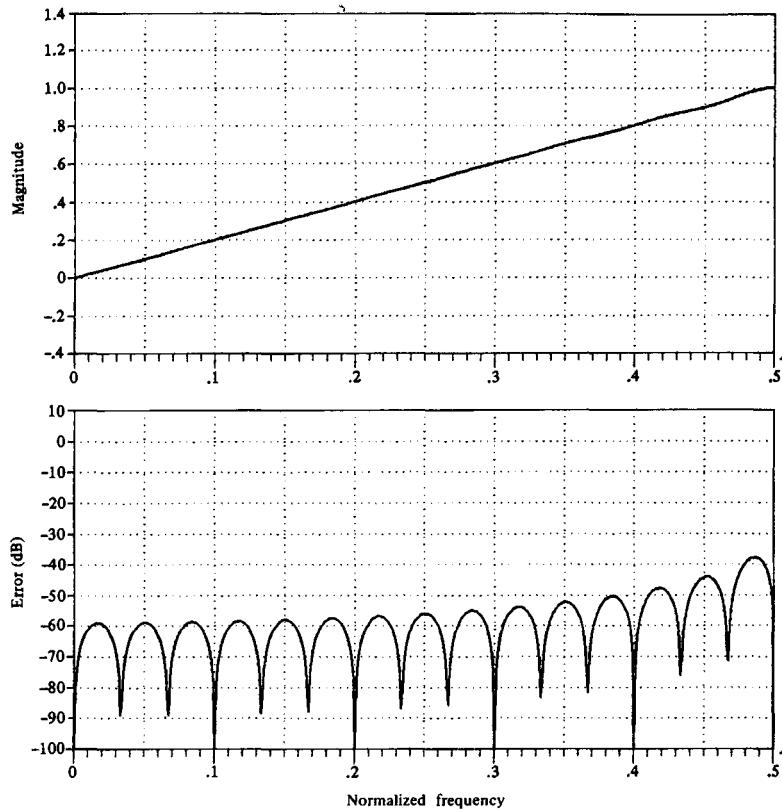


Figure 10.2.23 Frequency response and approximation error for $M = 30$ FIR differentiator designed by frequency-sampling method.

Hilbert transformers are frequently used in communication systems and signal processing, as, for example, in the generation of single-sideband modulated signals, radar signal processing, and speech signal processing.

The unit sample response of an ideal Hilbert transformer is

$$\begin{aligned}
 h_d(n) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(\omega) e^{j\omega n} d\omega \\
 &= \frac{1}{2\pi} \left(\int_{-\pi}^0 j e^{j\omega n} d\omega - \int_0^{\pi} j e^{j\omega n} d\omega \right) \\
 &= \begin{cases} \frac{2 \sin^2(\pi n/2)}{\pi}, & n \neq 0 \\ 0, & n = 0 \end{cases}
 \end{aligned} \tag{10.2.86}$$

As expected, $h_d(n)$ is infinite in duration and noncausal. We note that $h_d(n)$ is antisymmetric [i.e., $h_d(n) = -h_d(-n)$]. In view of this characteristic, we focus our attention on the design of linear-phase FIR Hilbert transformers with antisymmetric

unit sample response [i.e., $h(n) = -h(M-1-n)$]. We also observe that our choice of an antisymmetric unit sample response is consistent with having a purely imaginary frequency response characteristic $H_d(\omega)$.

We recall once again that when $h(n)$ is antisymmetric, the real-valued frequency response characteristic $H_r(\omega)$ is zero at $\omega = 0$ for both M odd and even and at $\omega = \pi$ when M is odd. Clearly, then, it is impossible to design an all-pass digital Hilbert transformer. Fortunately, in practical signal processing applications, an all-pass Hilbert transformer is unnecessary. Its bandwidth need only cover the bandwidth of the signal to be phase shifted. Consequently, we specify the desired real-valued frequency response of a Hilbert transform filter as

$$H_{\text{dr}}(\omega) = 1, \quad 2\pi f_l \leq \omega \leq 2\pi f_u \quad (10.2.87)$$

where f_l and f_u are the lower and upper cutoff frequencies, respectively.

It is interesting to note that the ideal Hilbert transformer with unit sample response $h_d(n)$ as given in (10.2.86) is zero for n even. This property is retained by the FIR Hilbert transformer under some symmetry conditions. In particular, let us consider the Case 3 filter type for which

$$H_r(\omega) = \sum_{k=1}^{(M-1)/2} c(k) \sin \omega k \quad (10.2.88)$$

and suppose that $f_l = 0.5 - f_u$. This ensures a symmetric passband about the midpoint frequency $f = 0.25$. If we have this symmetry in the frequency response, $H_r(\omega) = H_r(\pi - \omega)$ and hence (10.2.88) yields

$$\begin{aligned} \sum_{k=1}^{(M-1)/2} c(k) \sin \omega k &= \sum_{k=1}^{(M-1)/2} c(k) \sin k(\pi - \omega) \\ &= \sum_{k=1}^{(M-1)/2} c(k) \sin \omega k \cos \pi k \\ &= \sum_{k=1}^{(M-1)/2} c(k) (-1)^{k+1} \sin \omega k \end{aligned}$$

or equivalently,

$$\sum_{k=1}^{(M-1)/2} [1 - (-1)^{k+1}] c(k) \sin \omega k = 0 \quad (10.2.89)$$

Clearly, $c(k)$ must be equal to zero for $k = 0, 2, 4, \dots$.

Now, the relationship between $\{c(k)\}$ and the unit sample response $\{h(n)\}$ is, from (10.2.54),

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

or, equivalently,

$$h\left(\frac{M-1}{2} - k\right) = \frac{1}{2}c(k) \quad (10.2.90)$$

If $c(k)$ is zero for $k = 0, 2, 4, \dots$, then (10.2.90) yields

$$h(k) = \begin{cases} 0, & k = 0, 2, 4, \dots, \quad \text{for } \frac{M-1}{2} \text{ even} \\ 0, & k = 1, 3, 5, \dots, \quad \text{for } \frac{M-1}{2} \text{ odd} \end{cases} \quad (10.2.91)$$

Unfortunately, (10.2.91) holds only for M odd. It does not hold for M even. This means that for comparable values of M , the case M odd is preferable since the computational complexity (number of multiplications and additions per output point) is roughly one half of that for M even.

When the design of the Hilbert transformer is performed by the Chebyshev approximation criterion using the Remez algorithm, we select the filter coefficients to minimize the peak approximation error

$$\begin{aligned} \delta &= \max_{2\pi f_l \leq \omega \leq 2\pi f_u} [H_{dr}(\omega) - H_r(\omega)] \\ &= \max_{2\pi f_l \leq \omega \leq 2\pi f_u} [1 - H_r(\omega)] \end{aligned} \quad (10.2.92)$$

Thus the weighting function is set to unity and the optimization is performed over the single frequency band (i.e., the passband of the filter).

EXAMPLE 10.2.6

Design a Hilbert transformer with parameters $M = 31$, $f_l = 0.05$, and $f_u = 0.45$.

Solution. We observe that the frequency response is symmetric, since $f_u = 0.5 - f_l$. The parameters for executing the Remez algorithm are

$$\begin{array}{r} 31, \quad 3, \quad 1 \\ 0.05, \quad 0.45 \\ 1.0 \\ 1.0 \end{array}$$

The result of this design is the unit sample response and frequency response shown in Fig. 10.2.24. We observe that, indeed, every other value of $h(n)$ is essentially zero.

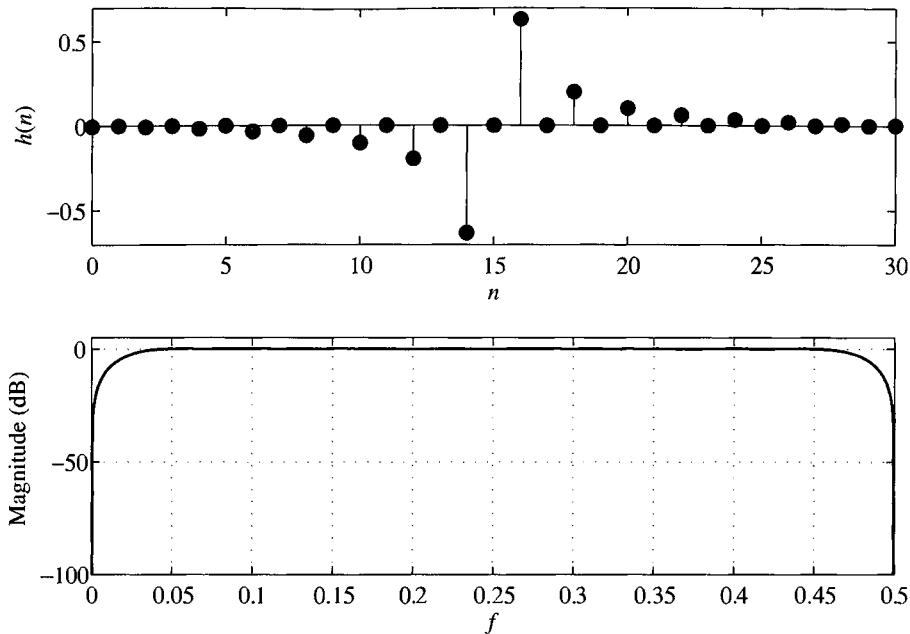


Figure 10.2.24 Frequency of FIR Hilbert transform filter in Example 10.2.6.

Rabiner and Schafer (1974b) have investigated the characteristics of Hilbert transformer designs for both M odd and M even. If the filter design is restricted to a symmetric frequency response, then there are basically three parameters of interest, M , δ , and f_l . Figure 10.2.25 is a plot of $20\log_{10}\delta$ versus f_l (the transition width) with M as a parameter. We observe that for comparable values of M , there is no performance advantage of using M odd over M even, and vice versa. However, the computational complexity in implementing a filter for M odd is less by a factor of 2 over M even as previously indicated. Therefore, M odd is preferable in practice.

For design purposes, the graphs in Fig. 10.2.25 suggest that, as a rule of thumb,

$$Mf_l \approx -0.61\log_{10}\delta \quad (10.2.93)$$

Hence this formula can be used to estimate the size of one of the three basic filter parameters when the other two parameters are specified.

10.2.7 Comparison of Design Methods for Linear-Phase FIR Filters

Historically, the design method based on the use of windows to truncate the impulse response $h_d(n)$ and obtain the desired spectral shaping was the first method proposed for designing linear-phase FIR filters. The frequency-sampling method and the Chebyshev approximation method were developed in the 1970s and have since become very popular in the design of practical linear-phase FIR filters.

The major disadvantage of the window design method is the lack of precise control of the critical frequencies, such as ω_p and ω_s , in the design of a lowpass FIR

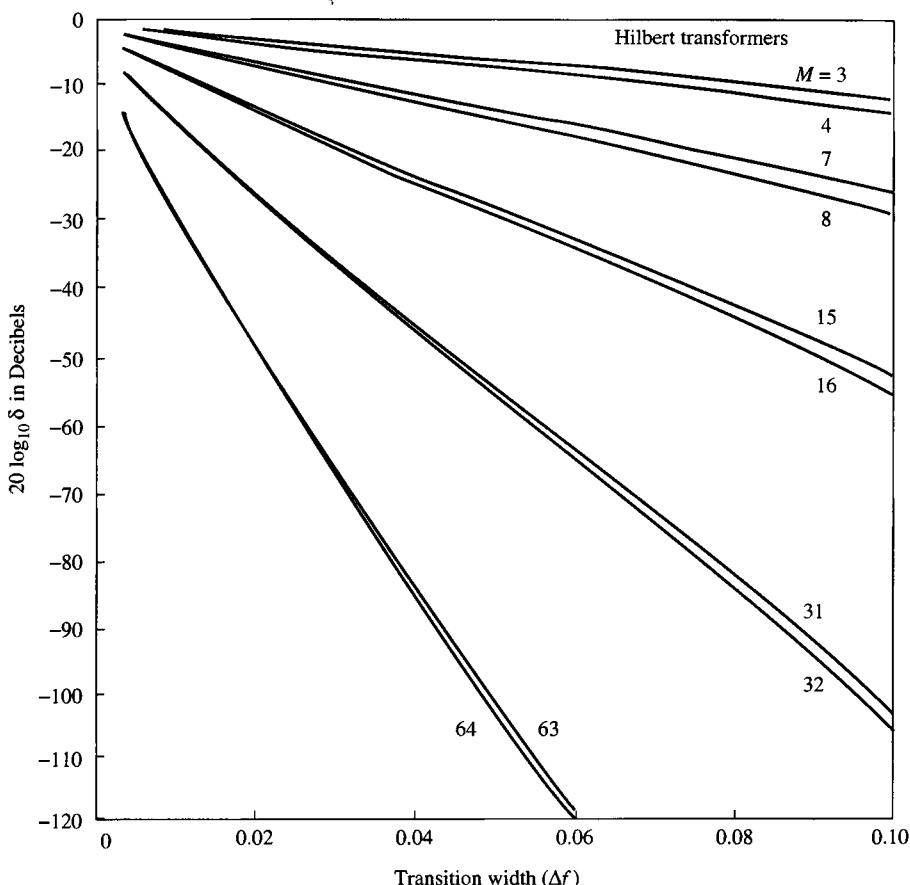


Figure 10.2.25 Curves of $20 \log_{10} \delta$ versus Δf for $M = 3, 4, 7, 8, 15, 16, 31, 32, 63, 64$. [From paper by Rabiner and Schafer (1974b). Reprinted with permission of AT&T.]

filter. The values of ω_p and ω_s , in general, depend on the type of window and the filter length M .

The frequency-sampling method provides an improvement over the window design method, since $H_r(\omega)$ is specified at the frequencies $\omega_k = 2\pi k/M$ or $\omega_k = \pi(2k+1)/M$ and the transition band is a multiple of $2\pi/M$. This filter design method is particularly attractive when the FIR filter is realized either in the frequency domain by means of the DFT or in any of the frequency-sampling realizations. The attractive feature of these realizations is that $H_r(\omega_k)$ is either zero or unity at all frequencies, except in the transition band.

The Chebyshev approximation method provides total control of the filter specifications, and, as a consequence, it is usually preferable over the other two methods. For a lowpass filter, the specifications are given in terms of the parameters ω_p , ω_s , δ_1 , δ_2 , and M . We can specify the parameters ω_p , ω_s , M , and δ , and optimize the filters relative to δ_2 . By spreading the approximation error over the passband and

the stopband of the filter, this method results in an optimal filter design, in the sense that for a given set of specifications just described, the maximum sidelobe level is minimized.

The Chebyshev design procedure based on the Remez exchange algorithm requires that we specify the length of the filter, the critical frequencies ω_p and ω_s , and the ratio δ_2/δ_1 . However, it is more natural in filter design to specify ω_p , ω_s , δ_1 , and δ_2 and to determine the filter length that satisfies the specifications. Although there is no simple formula to determine the filter length from these specifications, a number of approximations have been proposed for estimating M from ω_p , ω_s , δ_1 , and δ_2 . A particularly simple formula attributed to Kaiser for approximating M is

$$\hat{M} = \frac{-20 \log_{10} (\sqrt{\delta_1 \delta_2}) - 13}{14.6 \Delta f} + 1 \quad (10.2.94)$$

where Δf is the transition band, defined as $\Delta f = (\omega_s - \omega_p)/2\pi$. This formula has been given in the paper by Rabiner et al. (1975). A more accurate formula proposed by Herrmann et al. (1973) is

$$\hat{M} = \frac{D_\infty(\delta_1, \delta_2) - f(\delta_1, \delta_2)(\Delta f)^2}{\Delta f} + 1 \quad (10.2.95)$$

where, by definition,

$$\begin{aligned} D_\infty(\delta_1, \delta_2) &= [0.005309(\log_{10} \delta_1)^2 + 0.07114(\log_{10} \delta_1) - 0.4761](\log_{10} \delta_2) \\ &\quad - [0.00266(\log_{10} \delta_1)^2 + 0.5941 \log_{10} \delta_1 + 0.4278] \end{aligned} \quad (10.2.96)$$

$$f(\delta_1, \delta_2) = 11.012 + 0.51244(\log_{10} \delta_1 - \log_{10} \delta_2) \quad (10.2.97)$$

These formulas are extremely useful in obtaining a good estimate of the filter length required to achieve the given specifications Δf , δ_1 , and δ_2 . The estimate is used to carry out the design, and if the resulting δ exceeds the specified δ_2 , the length can be increased until we obtain a sidelobe level that meets the specifications.

10.3 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,

$$H_a(s) = \frac{B(s)}{A(s)} = \frac{\sum_{k=0}^M \beta_k s^k}{\sum_{k=0}^N \alpha_k s^k} \quad (10.3.1)$$

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

$$H_a(s) = \int_{-\infty}^{\infty} h(t)e^{-st} dt \quad (10.3.2)$$

Alternatively, the analog filter having the rational system function $H(s)$ given in (10.3.1) can be described by the linear constant-coefficient differential equation

$$\sum_{k=0}^N \alpha_k \frac{d^k y(t)}{dt^k} = \sum_{k=0}^M \beta_k \frac{d^k x(t)}{dt^k} \quad (10.3.3)$$

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 10.3.1 through 10.3.3. 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.

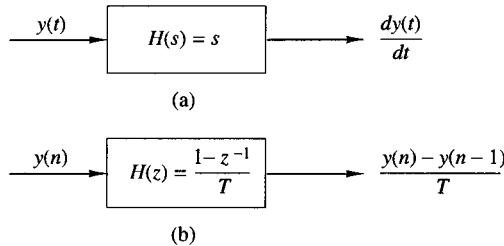
We mentioned in the preceding section that physically realizable and stable IIR filters cannot have linear phase. Recall that a linear-phase filter must have a system function that satisfies the condition

$$H(z) = \pm z^{-N} H(z^{-1}) \quad (10.3.4)$$

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 10.1, we specify the desired magnitude characteristics and accept the phase response that is obtained from the design methodology.

**Figure 10.3.1**

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

10.3.1 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 (10.3.3) 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

$$\begin{aligned} \left. \frac{dy(t)}{dt} \right|_{t=nT} &= \frac{y(nT) - y(nT - T)}{T} \\ &= \frac{y(n) - y(n-1)}{T} \end{aligned} \quad (10.3.5)$$

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. 10.3.1, the frequency-domain equivalent for the relationship in (10.3.5) is

$$s = \frac{1 - z^{-1}}{T} \quad (10.3.6)$$

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

$$\begin{aligned} \left. \frac{d^2y(t)}{dt^2} \right|_{t=nT} &= \frac{d}{dt} \left[\left. \frac{dy(t)}{dt} \right|_{t=nT} \right] \\ &= \frac{[y(nT) - y(nT - T)]/T - [y(nT - T) - y(nT - 2T)]/T}{T} \\ &= \frac{y(n) - 2y(n-1) + y(n-2)}{T^2} \end{aligned} \quad (10.3.7)$$

In the frequency domain, (10.3.7) is equivalent to

$$s^2 = \frac{1 - 2z^{-1} + z^{-2}}{T^2} = \left(\frac{1 - z^{-1}}{T} \right)^2 \quad (10.3.8)$$

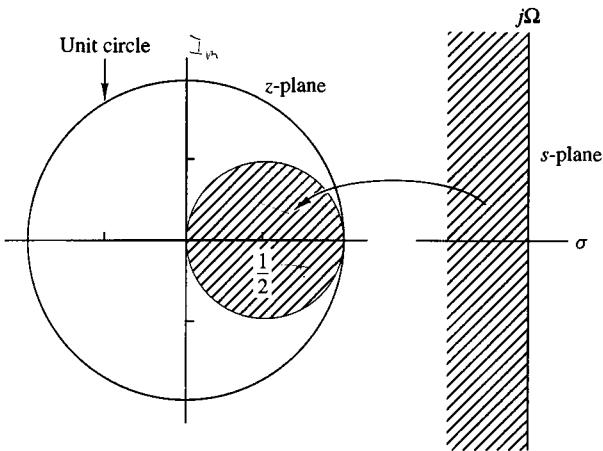


Figure 10.3.2
The mapping
 $s = (1 - z^{-1})/T$ takes LHP
in the s -plane into points
inside the circle of radius
 $\frac{1}{2}$ and center $z = \frac{1}{2}$ in the
 z -plane.

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

$$s^k = \left(\frac{1 - z^{-1}}{T} \right)^k \quad (10.3.9)$$

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

$$H(z) = H_a(s)|_{s=(1-z^{-1})/T} \quad (10.3.10)$$

where $H_a(s)$ is the system function of the analog filter characterized by the differential equation given in (10.3.3).

Let us investigate the implications of the mapping from the s -plane to the z -plane as given by (10.3.6) or, equivalently,

$$z = \frac{1}{1 - sT} \quad (10.3.11)$$

If we substitute $s = j\Omega$ in (10.2.11), we find that

$$\begin{aligned} z &= \frac{1}{1 - j\Omega T} \\ &= \frac{1}{1 + \Omega^2 T^2} + j \frac{\Omega T}{1 + \Omega^2 T^2} \end{aligned} \quad (10.3.12)$$

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. 10.3.2.

It is easily demonstrated that the mapping in (10.3.11) takes points in the LHP of the s -plane into corresponding points inside this circle in the z -plane, and points in the right-hand plane (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 L th-order difference of the form

$$\frac{dy(t)}{dt} \Big|_{t=nT} = \frac{1}{T} \sum_{k=1}^L \alpha_k \frac{y(nT + kT) - y(nT - kT)}{T} \quad (10.3.13)$$

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

$$s = \frac{1}{T} \sum_{k=1}^L \alpha_k (z^k - z^{-k}) \quad (10.3.14)$$

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

$$s = j \frac{2}{T} \sum_{k=1}^L \alpha_k \sin \omega k \quad (10.3.15)$$

which is purely imaginary. Thus

$$\Omega = \frac{2}{T} \sum_{k=1}^L \alpha_k \sin \omega k \quad (10.3.16)$$

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 (10.3.16), 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 L th-order difference as a substitute for the derivative.

EXAMPLE 10.3.1

Convert the analog bandpass filter with system function

$$H_a(s) = \frac{1}{(s + 0.1)^2 + 9}$$

into a digital IIR filter by use of the backward difference for the derivative.

Solution. Substitution for s from (10.3.6) into $H(s)$ yields

$$\begin{aligned} H(z) &= \frac{1}{\left(\frac{1-z^{-1}}{T} + 0.1\right)^2 + 9} \\ &= \frac{T^2/(1+0.2T+9.01T^2)}{1 - \frac{2(1+0.1T)}{1+0.2T+9.01T^2}z^{-1} + \frac{1}{1+0.2T+9.01T^2}z^{-2}} \end{aligned}$$

The system function $H(z)$ has the form of a resonator provided that T is selected small enough (e.g., $T \leq 0.1$), in order for the poles to be near the unit circle. Note that the condition $a_1^2 < 4a_2$ is satisfied, so that the poles are complex valued.

For example, if $T = 0.1$, the poles are located at

$$\begin{aligned} p_{1,2} &= 0.91 \pm j0.27 \\ &= 0.949e^{\pm j16.5^\circ} \end{aligned}$$

We note that the range of resonant frequencies is limited to low frequencies, due to the characteristics of the mapping. The reader is encouraged to plot the frequency response $H(\omega)$ of the digital filter for different values of T and compare the results with the frequency response of the analog filter.

EXAMPLE 10.3.2

Convert the analog bandpass filter in Example 10.3.1 into a digital IIR filter by use of the mapping

$$s = \frac{1}{T}(z - z^{-1})$$

Solution. By substituting for s in $H(s)$, we obtain

$$\begin{aligned} H(z) &= \frac{1}{\left(\frac{z-z^{-1}}{T} + 0.1\right)^2 + 9} \\ &= \frac{z^2T^2}{z^4 + 0.2Tz^3 + (2 + 9.01T^2)z^2 - 0.2Tz + 1} \end{aligned}$$

We observe that this mapping has introduced two additional poles in the conversion from $H_a(s)$ to $H(z)$. As a consequence, the digital filter is significantly more complex than the analog filter. This is a major drawback to the mapping given above.

10.3.2 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,

$$h(n) \equiv h(nT), \quad n = 0, 1, 2, \dots \quad (10.3.17)$$

where T is the sampling interval.

To examine the implications of (10.3.17), we refer back to Section 6.1. 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

$$X(f) = F_s \sum_{k=-\infty}^{\infty} X_a[(f - k)F_s] \quad (10.3.18)$$

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

$$H(f) = F_s \sum_{k=-\infty}^{\infty} H_a[(f - k)F_s] \quad (10.3.19)$$

or, equivalently,

$$H(\omega) = F_s \sum_{k=-\infty}^{\infty} H_a[(\omega - 2\pi k)F_s] \quad (10.3.20)$$

or

$$H(\Omega T) = \frac{1}{T} \sum_{k=-\infty}^{\infty} H_a\left(\Omega - \frac{2\pi k}{T}\right) \quad (10.3.21)$$

Figure 10.3.3 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 to the 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 (10.3.21) which relates the z -transform of $h(n)$ to the Laplace transform of $h_a(t)$. This relationship is

$$H(z)|_{z=e^{sT}} = \frac{1}{T} \sum_{k=-\infty}^{\infty} H_a\left(s - j\frac{2\pi k}{T}\right) \quad (10.3.22)$$

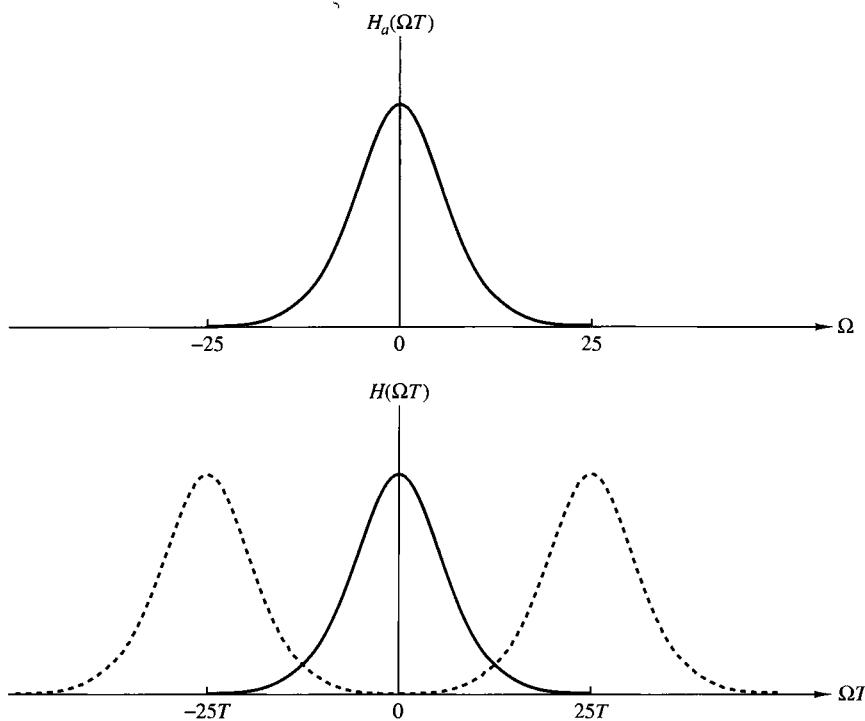


Figure 10.3.3 Frequency response $H_a(\Omega)$ of the analog filter and frequency response of the corresponding digital filter with aliasing.

where

$$H(z) = \sum_{n=0}^{\infty} h(n)z^{-n} \quad (10.3.23)$$

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

Note that when $s = j\Omega$, (10.3.22) reduces to (10.3.21), 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

$$z = e^{sT} \quad (10.3.24)$$

If we substitute $s = \sigma + j\omega$ and express the complex variable z in polar form as $z = re^{j\omega}$, (10.3.24) becomes

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

Clearly, we must have

$$\begin{aligned} r &= e^{\sigma T} \\ \omega &= \Omega T \end{aligned} \quad (10.3.25)$$

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 10.3.4 illustrates the mapping from the s -plane to the z -plane for the relation in (10.3.24).

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

$$H_a(s) = \sum_{k=1}^N \frac{c_k}{s - p_k} \quad (10.3.26)$$

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

$$h_a(t) = \sum_{k=1}^N c_k e^{p_k t}, \quad t \geq 0 \quad (10.3.27)$$

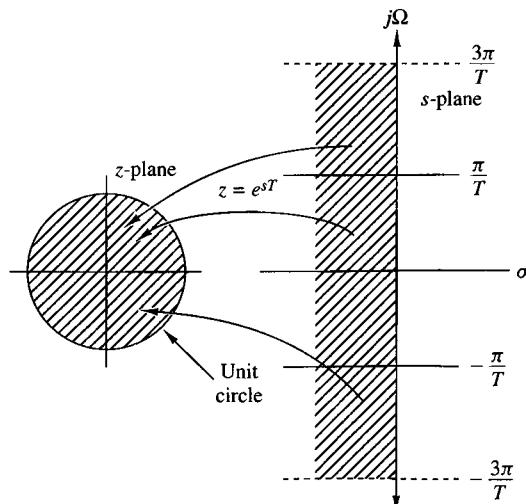


Figure 10.3.4
The mapping of $z = e^{j\Omega T}$ maps strips of width $2\pi/T$ (for $\sigma < 0$) in the s -plane into points in the unit circle in the z -plane.

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

$$\begin{aligned} h(n) &= h_a(nT) \\ &= \sum_{k=1}^N c_k e^{p_k T n} \end{aligned} \quad (10.3.28)$$

Now, with the substitution of (10.3.28), the system function of the resulting digital IIR filter becomes

$$\begin{aligned} H(z) &= \sum_{n=0}^{\infty} h(n) z^{-n} \\ &= \sum_{n=0}^{\infty} \left(\sum_{k=1}^N c_k e^{p_k T n} \right) z^{-n} \\ &= \sum_{k=1}^N c_k \sum_{n=0}^{\infty} (e^{p_k T} z^{-1})^n \end{aligned} \quad (10.3.29)$$

The inner sum in (10.3.29) converges because $p_k < 0$ and yields

$$\sum_{n=0}^{\infty} (e^{p_k T} z^{-1})^n = \frac{1}{1 - e^{p_k T} z^{-1}} \quad (10.3.30)$$

Therefore, the system function of the digital filter is

$$H(z) = \sum_{k=1}^N \frac{c_k}{1 - e^{p_k T} z^{-1}} \quad (10.3.31)$$

We observe that the digital filter has poles at

$$z_k = e^{p_k T}, \quad k = 1, 2, \dots, N \quad (10.3.32)$$

Although the poles are mapped from the s -plane to the z -plane by the relationship in (10.3.32), we should emphasize that the zeros in the two domains do not satisfy the same relationship. Therefore, the impulse invariance method does not correspond to the simple mapping of points given by (10.3.24).

The development that resulted in $H(z)$ given by (10.3.31) was based on a filter having distinct poles. It can be generalized to include multiple-order poles. For brevity, however, we shall not attempt to generalize (10.3.31).

EXAMPLE 10.3.3

Convert the analog filter with system function

$$H_a(s) = \frac{s + 0.1}{(s + 0.1)^2 + 9}$$

into a digital IIR filter by means of the impulse invariance method.

Solution. We note that the analog filter has a zero at $s = -0.1$ and a pair of complex-conjugate poles at

$$p_k = -0.1 \pm j3$$

as illustrated in Fig. 10.3.5.

We do not have to determine the impulse response $h_a(t)$ in order to design the digital IIR filter based on the method of impulse invariance. Instead, we directly determine $H(z)$, as given by (10.3.31), from the partial-fraction expansion of $H_a(s)$. Thus we have

$$H(s) = \frac{\frac{1}{2}}{s + 0.1 - j3} + \frac{\frac{1}{2}}{s + 0.1 + j3}$$

Then

$$H(z) = \frac{\frac{1}{2}}{1 - e^{-0.1T} e^{j3T} z^{-1}} + \frac{\frac{1}{2}}{1 - e^{-0.1T} e^{-j3T} z^{-1}}$$

Since the two poles are complex conjugates, we can combine them to form a single two-pole filter with system function

$$H(z) = \frac{1 - (e^{-0.1T} \cos 3T)z^{-1}}{1 - (2e^{-0.1T} \cos 3T)z^{-1} + e^{-0.2T}z^{-2}}$$

The magnitude of the frequency response characteristic of this filter is plotted in Fig. 10.3.6 for $T = 0.1$ and $T = 0.5$. For purpose of comparison, we have also plotted the magnitude of the frequency response of the analog filter in Fig. 10.3.7. We note that aliasing is significantly more prevalent when $T = 0.5$ than when $T = 0.1$. Also, note the shift of the resonant frequency as T changes.

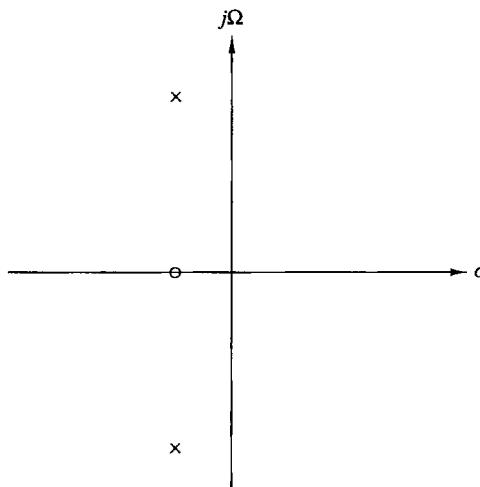


Figure 10.3.5
Pole-zero locations
for analog filter in
Example 10.3.3.

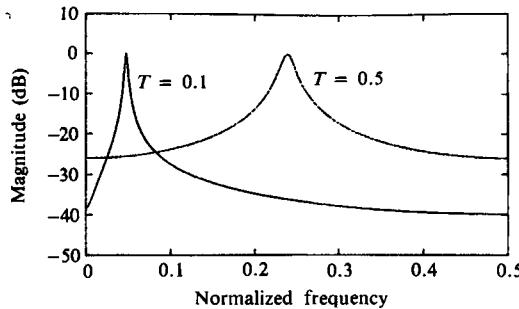


Figure 10.3.6
Frequency response
of digital filter in
Example 10.3.3.

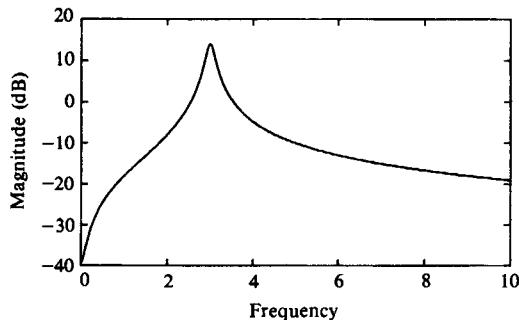


Figure 10.3.7
Frequency response
of analog filter in
Example 10.3.3.

The preceding example illustrates the importance of selecting a small value for T to minimize the effect of aliasing. Due to the presence of aliasing, the impulse invariance method is appropriate for the design of lowpass and bandpass filters only.

10.3.3 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

$$H(s) = \frac{b}{s + a} \quad (10.3.33)$$

This system is also characterized by the differential equation

$$\frac{dy(t)}{dt} + ay(t) = bx(t) \quad (10.3.34)$$

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

$$y(t) = \int_{t_0}^t y'(\tau) d\tau + y(t_0) \quad (10.3.35)$$

where $y'(t)$ denotes the derivative of $y(t)$. The approximation of the integral in (10.3.35) by the trapezoidal formula at $t = nT$ and $t_0 = nT - T$ yields

$$y(nT) = \frac{T}{2} [y'(nT) + y'(nT - T)] + y(nT - T) \quad (10.3.36)$$

Now the differential equation in (10.3.34) evaluated at $t = nT$ yields

$$y'(nT) = -ay(nT) + bx(nT) \quad (10.3.37)$$

We use (10.3.37) to substitute for the derivative in (10.3.36) 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

$$\left(1 + \frac{aT}{2}\right) y(n) - \left(1 - \frac{aT}{2}\right) y(n-1) = \frac{bT}{2} [x(n) + x(n-1)] \quad (10.3.38)$$

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,

$$H(z) = \frac{b}{\frac{2}{T} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right) + a} \quad (10.3.39)$$

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

$$s = \frac{2}{T} \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right) \quad (10.3.40)$$

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 N th-order differential equation.

To investigate the characteristics of the bilinear transformation, let

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

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

Then (10.3.40) 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,

$$\sigma = \frac{2}{T} \frac{r^2 - 1}{1 + r^2 + 2r \cos \omega} \quad (10.3.41)$$

$$\Omega = \frac{2}{T} \frac{2r \sin \omega}{1 + r^2 + 2r \cos \omega} \quad (10.3.42)$$

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

$$\begin{aligned} \Omega &= \frac{2}{T} \frac{\sin \omega}{1 + \cos \omega} \\ &= \frac{2}{T} \tan \frac{\omega}{2} \end{aligned} \quad (10.3.43)$$

or, equivalently,

$$\omega = 2 \tan^{-1} \frac{\Omega T}{2} \quad (10.3.44)$$

The relationship in (10.3.44) between the frequency variables in the two domains is illustrated in Fig. 10.3.8. We observe that the entire range in Ω is mapped only once into the range $-\pi \leq \omega \leq \pi$. 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.

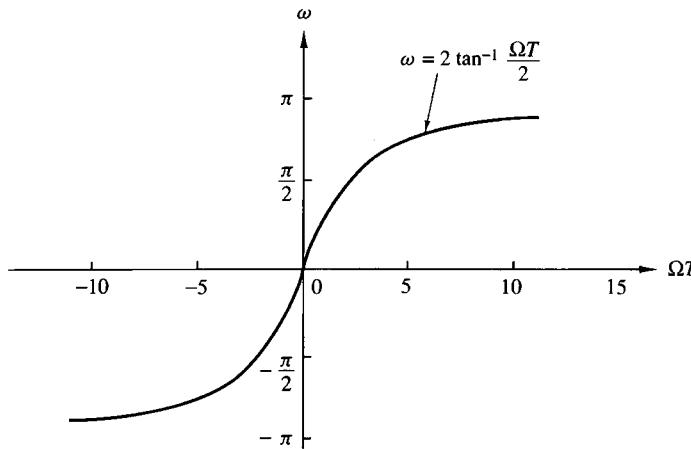


Figure 10.3.8 Mapping between the frequency variables ω and Ω resulting from the bilinear transformation.

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 (10.3.33), which has a zero at $s = \infty$, results in a digital filter that has a zero at $z = -1$.

EXAMPLE 10.3.4

Convert the analog filter with system function

$$H_a(s) = \frac{s + 0.1}{(s + 0.1)^2 + 16}$$

into a digital IIR filter by means of the bilinear transformation. The digital filter is to have a resonant frequency of $\omega_r = \pi/2$.

Solution. First, we note that the analog filter has a resonant frequency $\Omega_r = 4$. This frequency is to be mapped into $\omega_r = \pi/2$ by selecting the value of the parameter T . From the relationship in (10.3.43), we must select $T = \frac{1}{2}$ in order to have $\omega_r = \pi/2$. Thus the desired mapping is

$$s = 4 \left(\frac{1 - z^{-1}}{1 + z^{-1}} \right)$$

The resulting digital filter has the system function

$$H(z) = \frac{0.128 + 0.006z^{-1} - 0.122z^{-2}}{1 + 0.0006z^{-1} + 0.975z^{-2}}$$

We note that the coefficient of the z^{-1} term in the denominator of $H(z)$ is extremely small and can be approximated by zero. Thus we have the system function

$$H(z) = \frac{0.128 + 0.006z^{-1} - 0.122z^{-2}}{1 + 0.975z^{-2}}$$

This filter has poles at

$$p_{1,2} = 0.987e^{\pm j\pi/2}$$

and zeros at

$$z_{1,2} = -1, 0.95$$

Therefore, we have succeeded in designing a two-pole filter that resonates near $\omega = \pi/2$.

In this example the parameter T was selected to map the resonant frequency of the analog filter into the desired resonant frequency of the digital filter. Usually, the design of the digital filter begins with specifications in the digital domain, which involve the frequency variable ω . These specifications in frequency are converted to the analog domain by means of the relation in (10.3.43). The analog filter is then designed that meets these specifications and converted to a digital filter by means of the bilinear transformation in (10.3.40). In this procedure, the parameter T is transparent and may be set to any arbitrary value (e.g., $T = 1$). The following example illustrates this point.

EXAMPLE 10.3.5

Design a single-pole lowpass digital filter with a 3-dB bandwidth of 0.2π , using the bilinear transformation applied to the analog filter

$$H(s) = \frac{\Omega_c}{s + \Omega_c}$$

where Ω_c is the 3-dB bandwidth of the analog filter.

Solution. The digital filter is specified to have its -3 -dB gain at $\omega_c = 0.2\pi$. In the frequency domain of the analog filter $\omega_c = 0.2\pi$ corresponds to

$$\begin{aligned}\Omega_c &= \frac{2}{T} \tan 0.1\pi \\ &= \frac{0.65}{T}\end{aligned}$$

Thus the analog filter has the system function

$$H(s) = \frac{0.65/T}{s + 0.65/T}$$

This represents our filter design in the analog domain.

Now, we apply the bilinear transformation given by (10.3.40) to convert the analog filter into the desired digital filter. Thus we obtain

$$H(z) = \frac{0.245(1 + z^{-1})}{1 - 0.509z^{-1}}$$

where the parameter T has been divided out.

The frequency response of the digital filter is

$$H(\omega) = \frac{0.245(1 + e^{-j\omega})}{1 - 0.509e^{-j\omega}}$$

At $\omega = 0$, $H(0) = 1$, and at $\omega = 0.2\pi$, we have $|H(0.2\pi)| = 0.707$, which is the desired response.

10.3.4 Characteristics of Commonly Used Analog Filters

As we have seen from our discussion above, IIR digital filters can easily be obtained by beginning with an analog filter and then using a mapping to transform the s -plane into the z -plane. Thus the design of a digital filter is reduced to designing an appropriate analog filter and then performing the conversion from $H(s)$ to $H(z)$, in such a way so as to preserve, as much as possible, the desired characteristics of the analog filter.

Analog filter design is a well-developed field and many books have been written on the subject. In this section we briefly describe the important characteristics of commonly used analog filters and introduce the relevant filter parameters. Our discussion is limited to lowpass filters. Subsequently, we describe several frequency transformations that convert a lowpass prototype filter into either a bandpass, high-pass, or band-elimination filter.

Butterworth filters. Lowpass Butterworth filters are all-pole filters characterized by the magnitude-squared frequency response

$$|H(\Omega)|^2 = \frac{1}{1 + (\Omega/\Omega_c)^{2N}} = \frac{1}{1 + \epsilon^2(\Omega/\Omega_p)^{2N}} \quad (10.3.45)$$

where N is the order of the filter, Ω_c is its -3 -dB frequency (usually called the cutoff frequency), Ω_p is the passband edge frequency, and $1/(1+\epsilon^2)$ is the band-edge value of $|H(\Omega)|^2$. Since $H(s)H(-s)$ evaluated at $s = j\Omega$ is simply equal to $|H(\Omega)|^2$, it follows that

$$H(s)H(-s) = \frac{1}{1 + (-s^2/\Omega_c^2)^N} \quad (10.3.46)$$

The poles of $H(s)H(-s)$ occur on a circle of radius Ω_c at equally spaced points. From (10.3.46) we find that

$$\frac{-s^2}{\Omega_c^2} = (-1)^{1/N} = e^{j(2k+1)\pi/N}, \quad k = 0, 1, \dots, N-1$$

and hence

$$s_k = \Omega_c e^{j\pi/2} e^{j(2k+1)\pi/2N}, \quad k = 0, 1, \dots, N-1 \quad (10.3.47)$$

For example, Fig. 10.3.9 illustrates the pole positions for $N = 4$ and $N = 5$ Butterworth filters.

The frequency response characteristics of the class of Butterworth filters are shown in Fig. 10.3.10 for several values of N . We note that $|H(\Omega)|^2$ is monotonic in both the passband and stopband. The order of the filter required to meet an attenuation δ_2 at a specified frequency Ω_s is easily determined from (10.3.45). Thus at $\Omega = \Omega_s$ we have

$$\frac{1}{1 + \epsilon^2(\Omega_s/\Omega_p)^{2N}} = \delta_2^2$$

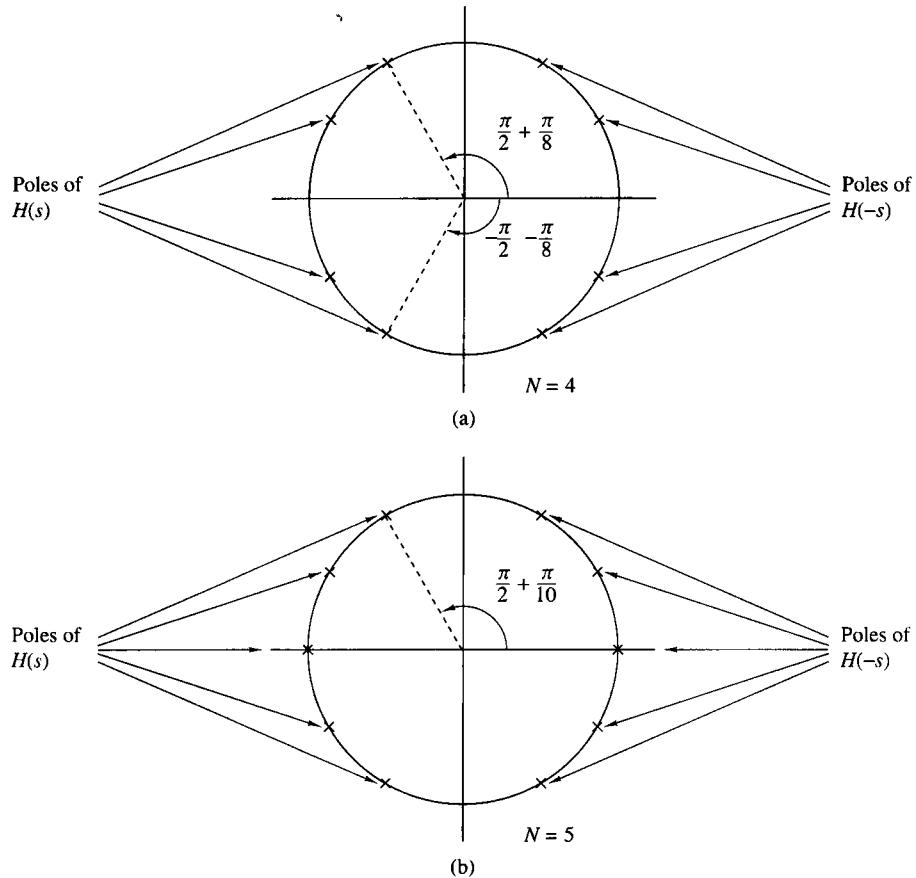


Figure 10.3.9 Pole positions for Butterworth filters.

and hence

$$N = \frac{\log[(1/\delta_2^2) - 1]}{2 \log(\Omega_s/\Omega_c)} = \frac{\log(\delta/\epsilon)}{\log(\Omega_s/\Omega_p)} \quad (10.3.48)$$

where, by definition, $\delta_2 = 1/\sqrt{1+\delta^2}$. Thus the Butterworth filter is completely characterized by the parameters N , δ_2 , ϵ , and the ratio Ω_s/Ω_p .

EXAMPLE 10.3.6

Determine the order and the poles of a lowpass Butterworth filter that has a -3 -dB bandwidth of 500 Hz and an attenuation of 40 dB at 1000 Hz.

Solution. The critical frequencies are the -3 -dB frequency Ω_c and the stopband frequency Ω_s , which are

$$\Omega_c = 1000\pi$$

$$\Omega_s = 2000\pi$$

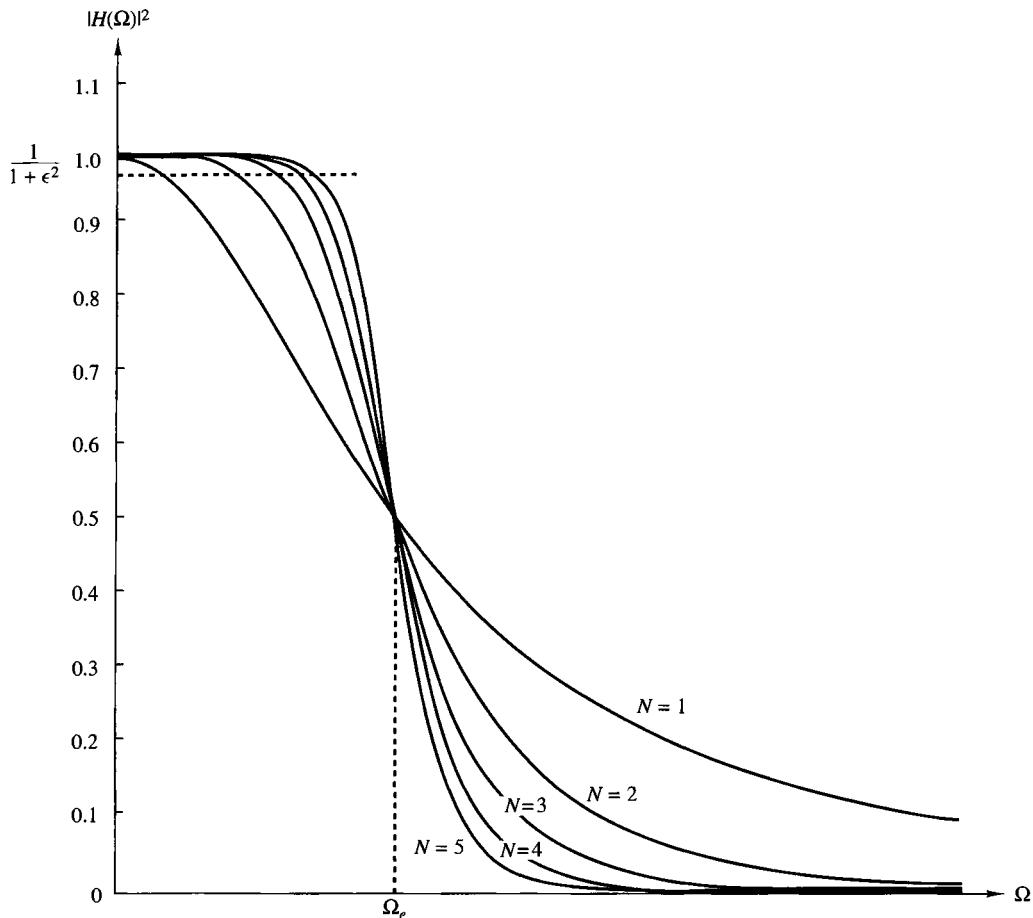


Figure 10.3.10 Frequency response of Butterworth filters.

For an attenuation of 40 dB, $\delta_2 = 0.01$. Hence from (10.3.48) we obtain

$$N = \frac{\log_{10}(10^4 - 1)}{2 \log_{10} 2}$$

$$= 6.64$$

To meet the desired specifications, we select $N = 7$. The pole positions are

$$s_k = 1000\pi e^{j[\pi/2 + (2k+1)\pi/14]}, \quad k = 0, 1, 2, \dots, 6$$

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

$$|H(\Omega)|^2 = \frac{1}{1 + \epsilon^2 T_N^2(\Omega/\Omega_p)} \quad (10.3.49)$$

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

$$T_N(x) = \begin{cases} \cos(N \cos^{-1} x), & |x| \leq 1 \\ \cosh(N \cosh^{-1} x), & |x| > 1 \end{cases} \quad (10.3.50)$$

The Chebyshev polynomials can be generated by the recursive equation

$$T_{N+1}(x) = 2xT_N(x) - T_{N-1}(x), \quad N = 1, 2, \dots \quad (10.3.51)$$

where $T_0(x) = 1$ and $T_1(x) = x$. From (10.3.51) 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. $|T_N(x)| \leq 1$ for all $|x| \leq 1$.
2. $T_N(1) = 1$ for all N .
3. All the roots of the polynomial $T_N(x)$ occur in the interval $-1 \leq x \leq 1$.

The filter parameter ϵ is related to the ripple in the passband, as illustrated in Fig. 10.3.11, 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,

$$\epsilon^2 = \frac{1}{(1 - \delta_1)^2} - 1 \quad (10.3.52)$$

where δ_1 is the value of the passband ripple.

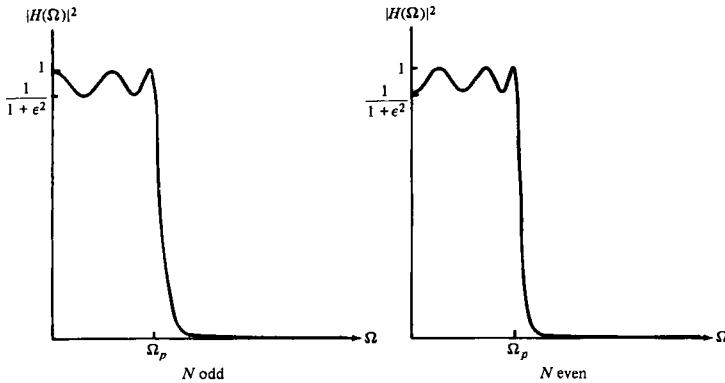


Figure 10.3.11 Type I Chebyshev filter characteristic.

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

$$r_1 = \Omega_p \frac{\beta^2 + 1}{2\beta} \quad (10.3.53)$$

and minor axis

$$r_2 = \Omega_p \frac{\beta^2 - 1}{2\beta} \quad (10.3.54)$$

where β is related to ϵ according to the equation

$$\beta = \left[\frac{\sqrt{1+\epsilon^2} + 1}{\epsilon} \right]^{1/N} \quad (10.3.55)$$

The pole locations are most easily determined for a filter of order N by first locating the poles for an equivalent N th-order Butterworth filter that lie on circles of radius r_1 or radius r_2 , as illustrated in Fig. 10.3.12. If we denote the angular positions of the poles of the Butterworth filter as

$$\phi_k = \frac{\pi}{2} + \frac{(2k+1)\pi}{2N}, \quad k = 0, 1, 2, \dots, N-1 \quad (10.3.56)$$

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

$$\begin{aligned} x_k &= r_2 \cos \phi_k, & k = 0, 1, \dots, N-1 \\ y_k &= r_1 \sin \phi_k, & k = 0, 1, \dots, N-1 \end{aligned} \quad (10.3.57)$$

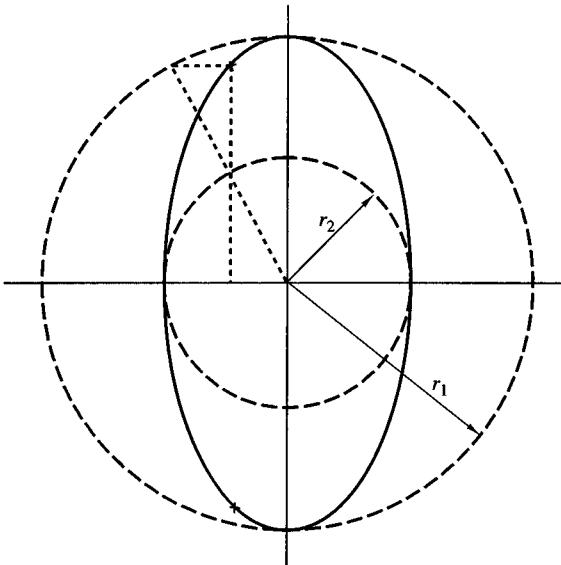


Figure 10.3.12
Determination of the pole locations for a Chebyshev filter.

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

$$|H(\Omega)|^2 = \frac{1}{1 + \epsilon^2 [T_N^2(\Omega_s/\Omega_p)/T_N^2(\Omega_s/\Omega)]} \quad (10.3.58)$$

where $T_N(x)$ is, again, the N th-order Chebyshev polynomial and Ω_s is the stopband frequency as illustrated in Fig. 10.3.13. The zeros are located on the imaginary axis at the points

$$s_k = j \frac{\Omega_s}{\sin \phi_k}, \quad k = 0, 1, \dots, N - 1 \quad (10.3.59)$$

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

$$v_k = \frac{\Omega_s x_k}{\sqrt{x_k^2 + y_k^2}}, \quad k = 0, 1, \dots, N - 1 \quad (10.3.60)$$

$$w_k = \frac{\Omega_s y_k}{\sqrt{x_k^2 + y_k^2}}, \quad k = 0, 1, \dots, N - 1 \quad (10.3.61)$$

where $\{x_k\}$ and $\{y_k\}$ are defined in (10.3.57) with β now related to the ripple in the stopband through the equation

$$\beta = \left[\frac{1 + \sqrt{1 - \delta_2^2}}{\delta_2} \right]^{1/N} \quad (10.3.62)$$

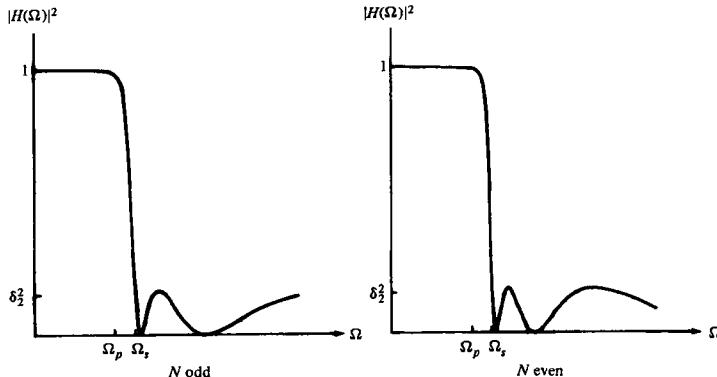


Figure 10.3.13 Type II Chebyshev filters.

From this description, we observe that the Chebyshev filters are characterized by the parameters N , ϵ , δ_2 , and the ratio Ω_s/Ω_p . For a given set of specifications on ϵ , δ_2 , and Ω_s/Ω_p , we can determine the order of the filter from the equation

$$N = \frac{\log \left[\left(\sqrt{1 - \delta_2^2} + \sqrt{1 - \delta_2^2(1 + \epsilon^2)} \right) / \epsilon \delta_2 \right]}{\log \left[(\Omega_s/\Omega_p) + \sqrt{(\Omega_s/\Omega_p)^2 - 1} \right]} \quad (10.3.63)$$

$$= \frac{\cosh^{-1}(\delta/\epsilon)}{\cosh^{-1}(\Omega_s/\Omega_p)}$$

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

EXAMPLE 10.3.7

Determine the order and the poles of a type I lowpass Chebyshev filter that has a 1-dB ripple in the passband, a cutoff frequency $\Omega_p = 1000\pi$, a stopband frequency of 2000π , and an attenuation of 40 dB or more for $\Omega \geq \Omega_s$.

Solution. First, we determine the order of the filter. We have

$$\begin{aligned} 10 \log_{10}(1 + \epsilon^2) &= 1 \\ 1 + \epsilon^2 &= 1.259 \\ \epsilon^2 &= 0.259 \\ \epsilon &= 0.5088 \end{aligned}$$

Also,

$$20 \log_{10} \delta_2 = -40$$

$$\delta_2 = 0.01$$

Hence from (10.3.63) we obtain

$$N = \frac{\log_{10} 196.54}{\log_{10}(2 + \sqrt{3})}$$

$$= 4.0$$

Thus a type I Chebyshev filter having four poles meets the specifications.

The pole positions are determined from the relations in (10.3.53) through (10.3.57). First, we compute β , r_1 , and r_2 . Hence

$$\beta = 1.429$$

$$r_1 = 1.06\Omega_p$$

$$r_2 = 0.365\Omega_p$$

The angles $\{\phi_k\}$ are

$$\phi_k = \frac{\pi}{2} + \frac{(2k+1)\pi}{8}, \quad k = 0, 1, 2, 3$$

Therefore, the poles are located at

$$x_1 + jy_1 = -0.1397\Omega_p \pm j0.979\Omega_p$$

$$x_2 + jy_2 = -0.337\Omega_p \pm j0.4056\Omega_p$$

The filter specifications in Example 10.3.7 are very similar to the specifications given in Example 10.3.6, which involved the design of a Butterworth filter. In that case the number of poles required to meet the specifications was seven. On the other hand, the Chebyshev filter required only four. This result is typical of such comparisons. In general, the Chebyshev filter meets the specifications with fewer poles than the corresponding Butterworth filter. Alternatively, if we compare a Butterworth filter to a Chebyshev filter having the same number of poles and the same passband and stopband specifications, the Chebyshev filter will have a smaller transition bandwidth. For a tabulation of the characteristics of Chebyshev filters and their pole-zero locations, the interested reader is referred to the handbook of Zverev (1967).

Elliptic filters. Elliptic (or Cauer) filters exhibit equiripple behavior in both the passband and the stopband, as illustrated in Fig. 10.3.14 for N odd and N even. This class of filters contains both poles and zeros and is characterized by the magnitude-squared frequency response

$$|H(\Omega)|^2 = \frac{1}{1 + \epsilon^2 U_N(\Omega/\Omega_p)} \quad (10.3.64)$$

where $U_N(x)$ is the Jacobian elliptic function of order N , which has been tabulated by Zverev (1967), and ϵ is a parameter related to the passband ripple. The zeros lie on the $j\Omega$ -axis.

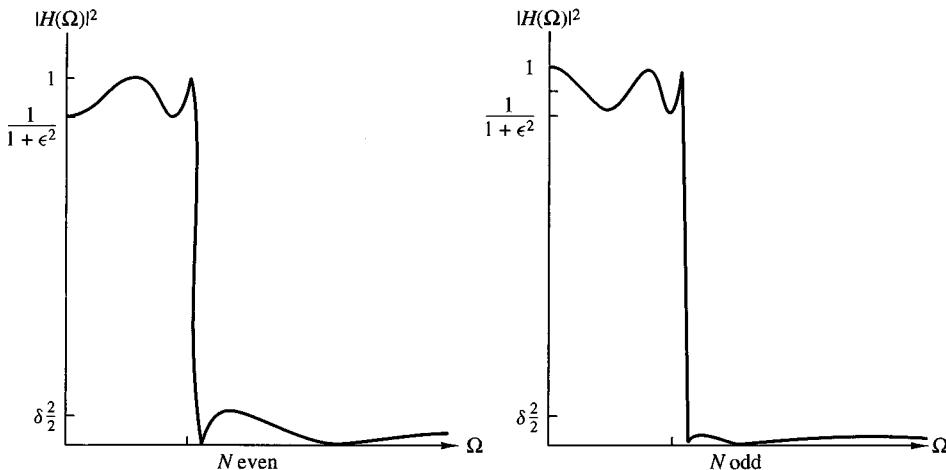


Figure 10.3.14 Magnitude-squared frequency characteristics of elliptic filters.

We recall from our discussion of FIR filters that the most efficient designs occur when we spread the approximation error equally over the passband and the stopband. Elliptic filters accomplish this objective and, as a consequence, are the most efficient from the viewpoint of yielding the smallest-order filter for a given set of specifications. Equivalently, we can say that for a given order and a given set of specifications, an elliptic filter has the smallest transition bandwidth.

The filter order required to achieve a given set of specifications in passband ripple δ_1 , stopband ripple δ_2 , and transition ratio Ω_p/Ω_s is given as

$$N = \frac{K(\Omega_p/\Omega_s)K\left(\sqrt{1-(\epsilon^2/\delta^2)}\right)}{K(\epsilon/\delta)K\left(\sqrt{1-(\Omega_p/\Omega_s)^2}\right)} \quad (10.3.65)$$

where $K(x)$ is the complete elliptic integral of the first kind, defined as

$$K(x) = \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-x^2 \sin^2 \theta}} \quad (10.3.66)$$

and $\delta_2 = 1/\sqrt{1+\delta^2}$. Values of this integral have been tabulated in a number of texts [e.g., the books by Jahnke and Emde (1945) and Dwight (1957)]. The passband ripple is $10 \log_{10}(1+\epsilon^2)$.

We shall not attempt to describe elliptic functions in any detail because such a discussion would take us too far afield. Suffice to say that computer programs are available for designing elliptic filters from the frequency specifications indicated above.

In view of the optimality of elliptic filters, the reader may question the reason for considering the class of Butterworth or the class of Chebyshev filters in practical applications. One important reason that these other types of filters might be preferable in some applications is that they possess better phase response characteristics. The phase response of elliptic filters is more nonlinear in the passband than a comparable Butterworth filter or a Chebyshev filter, especially near the band edge.

Bessel filters. Bessel filters are a class of all-pole filters that are characterized by the system function

$$H(s) = \frac{1}{B_N(s)} \quad (10.3.67)$$

where $B_N(s)$ is the N th-order Bessel polynomial. These polynomials can be expressed in the form

$$B_N(s) = \sum_{k=0}^N a_k s^k \quad (10.3.68)$$

where the coefficients $\{a_k\}$ are given as

$$a_k = \frac{(2N - k)!}{2^{N-k} k! (N - k)!}, \quad k = 0, 1, \dots, N \quad (10.3.69)$$

Alternatively, the Bessel polynomials may be generated recursively from the relation

$$B_N(s) = (2N - 1)B_{N-1}(s) + s^2 B_{N-2}(s) \quad (10.3.70)$$

with $B_0(s) = 1$ and $B_1(s) = s + 1$ as initial conditions.

An important characteristic of Bessel filters is the linear-phase response over the passband of the filter. For example, Fig. 10.3.15 shows a comparison of the magnitude

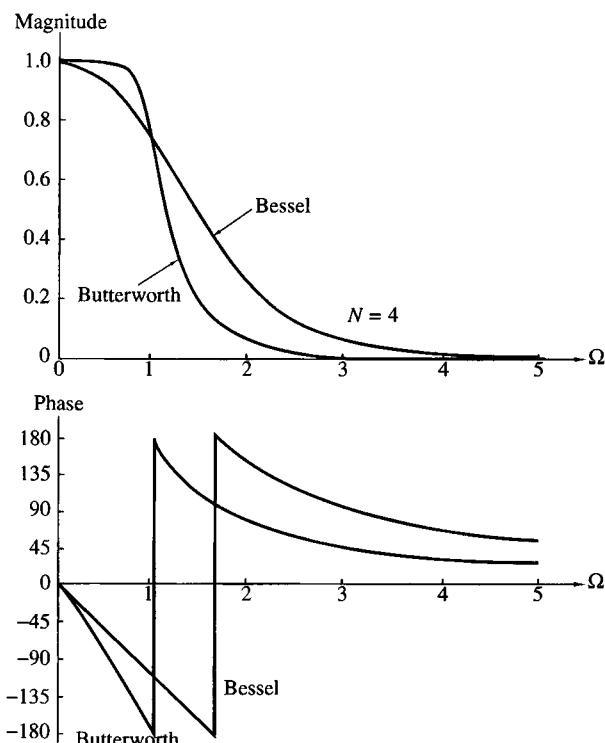


Figure 10.3.15
Magnitude and phase responses of Bessel and Butterworth filters of order $N = 4$.

and phase responses of a Bessel filter and Butterworth filter of order $N = 4$. We note that the Bessel filter has a larger transition bandwidth, but its phase is linear within the passband. However, we should emphasize that the linear-phase characteristics of the analog filter are destroyed in the process of converting the filter into the digital domain by means of the transformations described previously.

10.3.5 Some Examples of Digital Filter Designs Based on the Bilinear Transformation

In this section we present several examples of digital filter designs obtained from analog filters by applying the bilinear transformation to convert $H(s)$ to $H(z)$. These filter designs are performed with the aid of one of several software packages now available for use on a personal computer.

A lowpass filter is designed to meet specifications of a maximum ripple of $\frac{1}{2}$ dB in the passband, 60-dB attenuation in the stopband, a passband edge frequency of $\omega_p = 0.25\pi$, and a stopband edge frequency of $\omega_s = 0.30\pi$.

A Butterworth filter of order $N = 37$ is required to satisfy the specifications. Its frequency response characteristics are illustrated in Fig. 10.3.16. If a Chebyshev filter

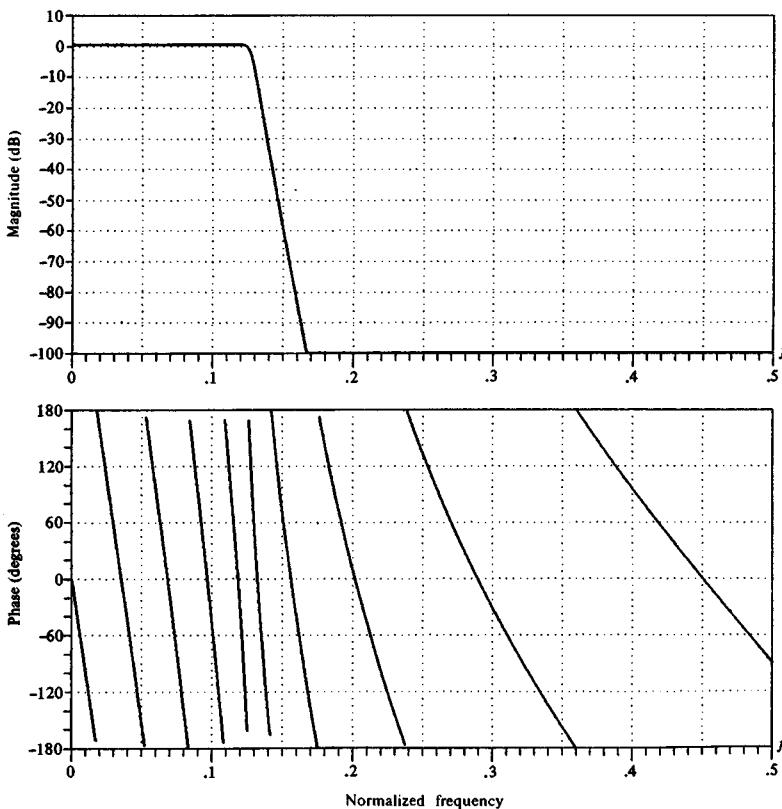


Figure 10.3.16 Frequency response characteristics of a 37-order Butterworth filter.

is used, a filter of order $N = 13$ satisfies the specifications. The frequency response characteristics for a type I Chebyshev filter are shown in Fig. 10.3.17. The filter has a passband ripple of 0.31 dB. Finally, an elliptic filter of order $N = 7$ is designed which also satisfies the specifications. For illustrative purposes, we show in Table 10.6, the numerical values for the filter parameters, and the resulting frequency specifications are shown in Fig. 10.3.18. The following notation is used for the parameters in the function $H(z)$:

$$H(z) = \prod_{i=1}^K \frac{b(i, 0) + b(i, 1)z^{-1} + b(i, 2)z^{-2}}{1 + a(i, 1)z^{-1} + a(i, 2)z^{-2}} \quad (10.3.71)$$

Although we have described only lowpass analog filters in the preceding section, it is a simple matter to convert a lowpass analog filter into a bandpass, bandstop, or highpass analog filter by a frequency transformation, as is described in Section 10.4. The bilinear transformation is then applied to convert the analog filter into an equivalent digital filter. As in the case of the lowpass filters described above, the entire design can be carried out on a computer.

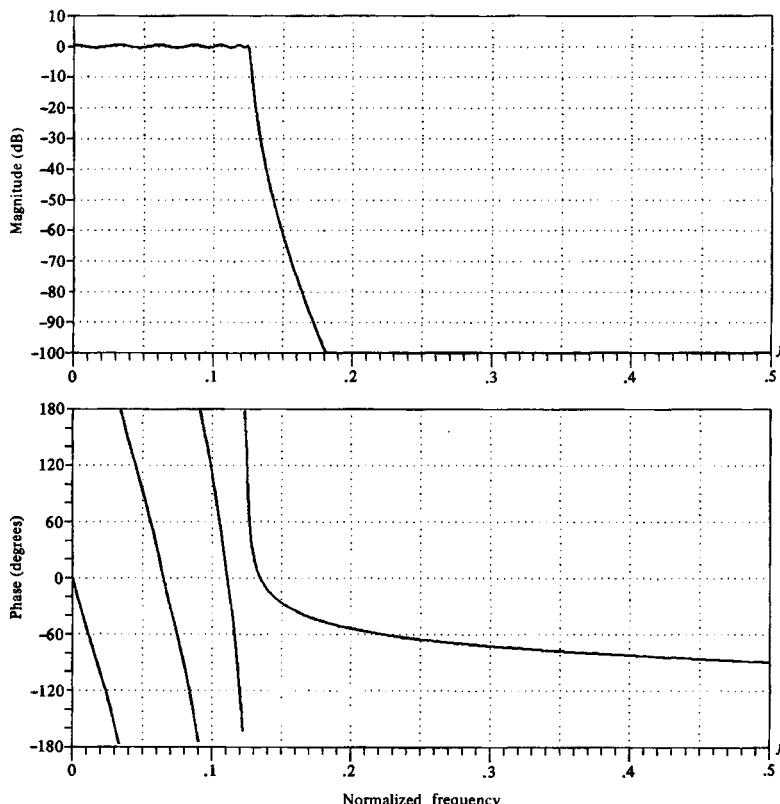


Figure 10.3.17 Frequency response characteristics of a 13-order type I Chebyshev filter.

TABLE 10.6 Filter Coefficients for a 7-Order Elliptic Filter

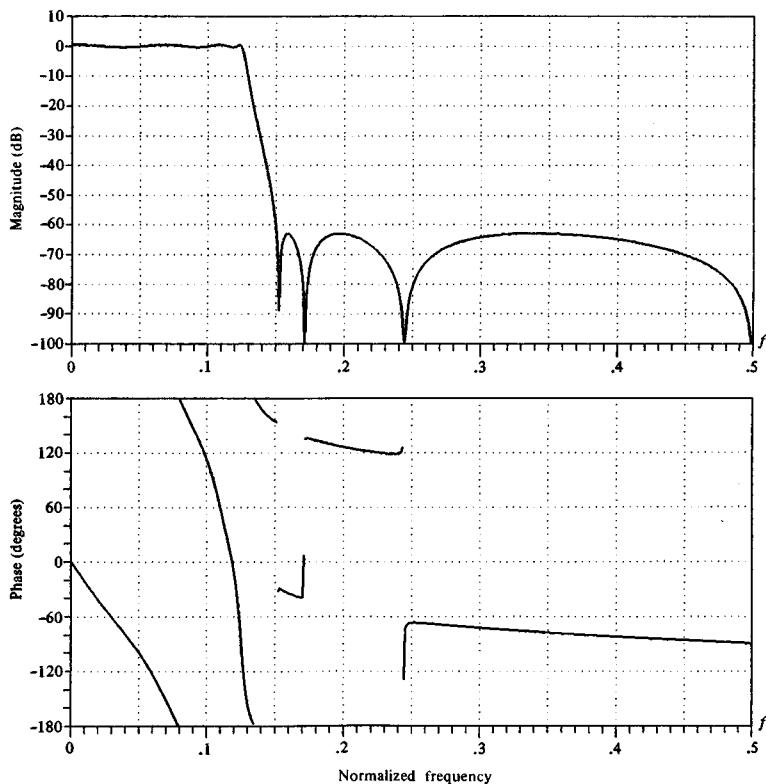
INFINITE IMPULSE RESPONSE (IIR)
ELLIPTIC LOWPASS FILTER
UNQUANTIZED COEFFICIENTS

FILTER ORDER = 7
SAMPLING FREQUENCY = 2.000 KILOHERTZ

I.	A(I, 1)	A(I, 2)	B(I, 0)	B(I, 1)	B(I, 2)
1	-.790103	.000000	.104948	.104948	.000000
2	-1.517223	.714088	.102450	-.007817	.102232
3	-1.421773	.861895	.420100	-.399842	.419864
4	-1.387447	.962252	.714929	-.826743	.714841

*** CHARACTERISTICS OF DESIGNED FILTER ***

	BAND 1	BAND 2
LOWER BAND EDGE	.00000	.30000
UPPER BAND EDGE	.25000	1.00000
NOMINAL GAIN	1.00000	.00000
NOMINAL RIPPLE	.05600	.00100
MAXIMUM RIPPLE	.04910	.00071
RISSLE IN DB	.41634	-63.00399

**Figure 10.3.18** Frequency response characteristics of a 7-order elliptic filter.

10.4 Frequency Transformations

The treatment in the preceding section is focused primarily on the design of lowpass IIR filters. If we wish to design a highpass or a bandpass or a bandstop filter, it is a simple matter to take a lowpass prototype filter (Butterworth, Chebyshev, elliptic, Bessel) and perform a frequency transformation.

One possibility is to perform the frequency transformation in the analog domain and then to convert the analog filter into a corresponding digital filter by a mapping of the s -plane into the z -plane. An alternative approach is first to convert the analog lowpass filter into a lowpass digital filter and then to transform the lowpass digital filter into the desired digital filter by a digital transformation. In general, these two approaches yield different results, except for the bilinear transformation, in which case the resulting filter designs are identical. These two approaches are described below.

10.4.1 Frequency Transformations in the Analog Domain

First, we consider frequency transformations in the analog domain. Suppose that we have a lowpass filter with passband edge frequency Ω_p and we wish to convert it to another lowpass filter with passband edge frequency Ω'_p . The transformation that accomplishes this is

$$s \longrightarrow \frac{\Omega_p}{\Omega'_p} s, \quad (\text{lowpass to lowpass}) \quad (10.4.1)$$

Thus we obtain a lowpass filter with system function $H_l(s) = H_p[(\Omega_p/\Omega'_p)s]$, where $H_p(s)$ is the system function of the prototype filter with passband edge frequency Ω_p .

If we wish to convert a lowpass filter into a highpass filter with passband edge frequency Ω'_p , the desired transformation is

$$s \longrightarrow \frac{\Omega_p \Omega'_p}{s}, \quad (\text{lowpass to highpass}) \quad (10.4.2)$$

The system function of the highpass filter is $H_h(s) = H_p(\Omega_p \Omega'_p/s)$.

The transformation for converting a lowpass analog filter with passband edge frequency Ω_c into a band filter, having a lower band edge frequency Ω_l and an upper band edge frequency Ω_u , can be accomplished by first converting the lowpass filter into another lowpass filter having a band edge frequency $\Omega'_p = 1$ and then performing the transformation

$$s \longrightarrow \frac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)}, \quad (\text{lowpass to bandpass}) \quad (10.4.3)$$

Equivalently, we can accomplish the same result in a single step by means of the transformation

$$s \longrightarrow \Omega_p \frac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)}, \quad (\text{lowpass to bandpass}) \quad (10.4.4)$$

TABLE 10.7 Frequency Transformations for Analog Filters (Prototype Lowpass Filter Has Band Edge Frequency Ω_p)

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

where

$$\Omega_l = \text{lower band edge frequency}$$

$$\Omega_u = \text{upper band edge frequency}$$

Thus we obtain

$$H_b(s) = H_p \left(\Omega_p \frac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)} \right)$$

Finally, if we wish to convert a lowpass analog filter with band-edge frequency Ω_p into a bandstop filter, the transformation is simply the inverse of (10.4.3) with the additional factor Ω_p serving to normalize for the band-edge frequency of the lowpass filter. Thus the transformation is

$$s \rightarrow \Omega_p \frac{s(\Omega_u - \Omega_l)}{s^2 + \Omega_u \Omega_l}, \quad (\text{lowpass to bandstop}) \quad (10.4.5)$$

which leads to

$$H_{bs}(s) = H_p \left(\Omega_p \frac{s(\Omega_u - \Omega_l)}{s^2 + \Omega_u \Omega_l} \right)$$

The mappings in (10.4.1), (10.4.2), (10.4.3), and (10.4.5) are summarized in Table 10.7. The mappings in (10.4.4) and (10.4.5) are nonlinear and may appear to distort the frequency response characteristics of the lowpass filter. However, the effects of the nonlinearity on the frequency response are minor, primarily affecting the frequency scale but preserving the amplitude response characteristics of the filter. Thus an equiripple lowpass filter is transformed into an equiripple bandpass or bandstop or highpass filter.

EXAMPLE 10.4.1

Transform the single-pole lowpass Butterworth filter with system function

$$H(s) = \frac{\Omega_p}{s + \Omega_p}$$

into a bandpass filter with upper and lower band edge frequencies Ω_u and Ω_l , respectively.

Solution. The desired transformation is given by (10.4.4). Thus we have

$$\begin{aligned} H(s) &= \frac{1}{\frac{s^2 + \Omega_l \Omega_u}{s(\Omega_u - \Omega_l)} + 1} \\ &= \frac{(\Omega_u - \Omega_l)s}{s^2 + (\Omega_u - \Omega_l)s + \Omega_l \Omega_u} \end{aligned}$$

The resulting filter has a zero at $s = 0$ and poles at

$$s = \frac{-(\Omega_u - \Omega_l) \pm \sqrt{\Omega_u^2 + \Omega_l^2 - 6\Omega_u \Omega_l}}{2}$$

10.4.2 Frequency Transformations in the Digital Domain

As in the analog domain, frequency transformations can be performed on a digital lowpass filter to convert it to either a bandpass, bandstop, or highpass filter. The transformation involves replacing the variable z^{-1} by a rational function $g(z^{-1})$, which must satisfy the following properties:

1. The mapping $z^{-1} \rightarrow g(z^{-1})$ must map points inside the unit circle in the z -plane into itself.
2. The unit circle must also be mapped into itself.

Condition (2) implies that for $r = 1$,

$$\begin{aligned} e^{-j\omega} &= g(e^{-j\omega}) \equiv g(\omega) \\ &= |g(\omega)|e^{j\arg[g(\omega)]} \end{aligned}$$

It is clear that we must have $|g(\omega)| = 1$ for all ω . That is, the mapping must be all pass. Hence it is of the form

$$g(z^{-1}) = \pm \prod_{k=1}^n \frac{z^{-1} - a_k}{1 - a_k z^{-1}} \quad (10.4.6)$$

where $|a_k| < 1$ to ensure that a stable filter is transformed into another stable filter (i.e., to satisfy condition 1).

From the general form in (10.4.6), we obtain the desired set of digital transformations for converting a prototype digital lowpass filter into either a bandpass, a bandstop, a highpass, or another lowpass digital filter. These transformations are tabulated in Table 10.8.

TABLE 10.8 Frequency Transformation for Digital Filters (Prototype Lowpass Filter Has Band Edge Frequency ω_p)

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

EXAMPLE 10.4.2

Convert the single-pole lowpass Butterworth filter with system function

$$H(z) = \frac{0.245(1 + z^{-1})}{1 - 0.509z^{-1}}$$

into a bandpass filter with upper and lower cutoff frequencies ω_u and ω_l , respectively. The lowpass filter has 3-dB bandwidth, $\omega_p = 0.2\pi$ (see Example 10.3.5).

Solution. The desired transformation is

$$z^{-1} \rightarrow -\frac{z^{-2} - a_1 z^{-1} + a_2}{a_2 z^{-2} - a_1 z^{-1} + 1}$$

where a_1 and a_2 are defined in Table 10.8. Substitution into $H(z)$ yields

$$\begin{aligned} H(z) &= \frac{0.245 \left[1 - \frac{z^{-2} - a_1 z^{-1} + a_2}{a_2 z^{-2} - a_1 z^{-1} + 1} \right]}{1 + 0.509 \left(\frac{z^{-2} - a_1 z^{-1} + a_2}{a_2 z^{-2} - a_1 z^{-1} + 1} \right)} \\ &= \frac{0.245(1 - a_2)(1 - z^{-2})}{(1 + 0.509a_2) - 1.509a_1z^{-1} + (a_2 + 0.509)z^{-2}} \end{aligned}$$

Note that the resulting filter has zeros at $z = \pm 1$ and a pair of poles that depend on the choice of ω_u and ω_l .

For example, suppose that $\omega_u = 3\pi/5$ and $\omega_l = 2\pi/5$. Since $\omega_p = 0.2\pi$, we find that $K = 1$, $a_2 = 0$, and $a_1 = 0$. Then

$$H(z) = \frac{0.245(1 - z^{-2})}{1 + 0.509z^{-2}}$$

This filter has poles at $z = \pm j0.713$ and hence resonates at $\omega = \pi/2$.

Since a frequency transformation can be performed either in the analog domain or in the digital domain, the filter designer has a choice as to which approach to take. However, some caution must be exercised depending on the types of filters being designed. In particular, we know that the impulse invariance method and the mapping of derivatives are inappropriate to use in designing highpass and many bandpass filters, due to the aliasing problem. Consequently, one would not employ an analog frequency transformation followed by conversion of the result into the digital domain by use of these two mappings. Instead, it is much better to perform the mapping from an analog lowpass filter into a digital lowpass filter by either of these mappings, and then to perform the frequency transformation in the digital domain. Thus the problem of aliasing is avoided.

In the case of the bilinear transformation, where aliasing is not a problem, it does not matter whether the frequency transformation is performed in the analog domain or in the digital domain. In fact, in this case only, the two approaches result in identical digital filters.

10.5 Summary and References

We have described in some detail the most important techniques for designing FIR and IIR digital filters based either on frequency-domain specifications expressed in terms of a desired frequency response $H_d(\omega)$ or on the desired impulse response $h_d(n)$.

As a general rule, FIR filters are used in applications where there is a need for a linear-phase filter. This requirement occurs in many applications, especially in telecommunications, where there is a requirement to separate (demultiplex) signals such as data that have been frequency-division multiplexed, without distorting these

signals in the process of demultiplexing. Of the several methods described for designing FIR filters, the frequency-sampling design method and the optimum Chebyshev approximation method yield the best designs.

IIR filters are generally used in applications where some phase distortion is tolerable. Of the class of IIR filters, elliptic filters are the most efficient to implement in the sense that for a given set of specifications, an elliptic filter has a lower order or fewer coefficients than any other IIR filter type. When compared with FIR filters, elliptic filters are also considerably more efficient. In view of this, one might consider the use of an elliptic filter to obtain the desired frequency selectivity, followed then by an all-pass phase equalizer that compensates for the phase distortion in the elliptic filter. However, attempts to accomplish this have resulted in filters with a number of coefficients in the cascade combination that equaled or exceeded the number of coefficients in an equivalent linear-phase FIR filter. Consequently, no reduction in complexity is achievable in using phase-equalized elliptic filters.

Such a rich literature now exists on the design of digital filters that it is not possible to cite all the important references. We shall cite only a few. Some of the early work on digital filter design was done by Kaiser (1963, 1966), Steiglitz (1965), Golden and Kaiser (1964), Rader and Gold (1967a), Shanks (1967), Helms (1968), Gibbs (1969, 1970), and Gold and Rader (1969).

The design of analog filters is treated in the classic books by Storer (1957), Guillemin (1957), Weinberg (1962), and Daniels (1974).

The frequency-sampling method for filter design was first proposed by Gold and Jordan (1968, 1969), and optimized by Rabiner et al. (1970). Additional results were published by Herrmann (1970), Herrmann and Schuessler (1970a), and Hofstetter et al. (1971). The Chebyshev (minimax) approximation method for designing linear-phase FIR filters was proposed by Parks and McClellan (1972a,b) and discussed further by Rabiner et al. (1975). The design of elliptic digital filters is treated in the book by Gold and Rader (1969) and in the paper by Gray and Markel (1976). The latter includes a computer program for designing digital elliptic filters.

The use of frequency transformations in the digital domain was proposed by Constantinides (1967, 1968, 1970). These transformations are appropriate only for IIR filters. The reader should note that when these transformations are applied to a lowpass FIR filter, the resulting filter is IIR.

Direct design techniques for digital filters have been considered in a number of papers, including Shanks (1967), Burrus and Parks (1970), Steiglitz (1970), Deczky (1972), Brophy and Salazar (1973), and Bandler and Bardakjian (1973).

Problems

10.1 Design an FIR linear-phase, digital filter approximating the ideal frequency response

$$H_d(\omega) = \begin{cases} 1, & \text{for } |\omega| \leq \frac{\pi}{6} \\ 0, & \text{for } \frac{\pi}{6} < |\omega| \leq \pi \end{cases}$$

- (a) Determine the coefficients of a 25-tap filter based on the window method with a rectangular window.

- (b) Determine and plot the magnitude and phase response of the filter.
- (c) Repeat parts (a) and (b) using the Hamming window.
- (d) Repeat parts (a) and (b) using a Bartlett window.

10.2 Repeat Problem 10.1 for a bandstop filter having the ideal response

$$H_d(\omega) = \begin{cases} 1, & \text{for } |\omega| \leq \frac{\pi}{6} \\ 0, & \text{for } \frac{\pi}{6} < |\omega| < \frac{\pi}{3} \\ 1, & \text{for } \frac{\pi}{3} \leq |\omega| \leq \pi \end{cases}$$

10.3 Redesign the filter of Problem 10.1 using the Hanning and Blackman windows.

10.4 Redesign the filter of Problem 10.2 using the Hanning and Blackman windows.

10.5 Determine the unit sample response $\{h(n)\}$ of a linear-phase FIR filter of length $M = 4$ for which the frequency response at $\omega = 0$ and $\omega = \pi/2$ is specified as

$$H_r(0) = 1, \quad H_r\left(\frac{\pi}{2}\right) = \frac{1}{2}$$

10.6 Determine the coefficients $\{h(n)\}$ of a linear-phase FIR filter of length $M = 15$ which has a symmetric unit sample response and a frequency response that satisfies the condition

$$H_r\left(\frac{2\pi k}{15}\right) = \begin{cases} 1, & k = 0, 1, 2, 3 \\ 0, & k = 4, 5, 6, 7 \end{cases}$$

10.7 Repeat the filter design problem in Problem 10.6 with the frequency response specifications

$$H_r\left(\frac{2\pi k}{15}\right) = \begin{cases} 1, & k = 0, 1, 2, 3 \\ 0.4, & k = 4 \\ 0, & k = 5, 6, 7 \end{cases}$$

10.8 The ideal analog differentiator is described by

$$y_a(t) = \frac{dx_a(t)}{dt}$$

where $x_a(t)$ is the input and $y_a(t)$ the output signal.

- (a) Determine its frequency response by exciting the system with the input $x_a(t) = e^{j2\pi F t}$.
- (b) Sketch the magnitude and phase response of an ideal analog differentiator band-limited to B hertz.
- (c) The ideal digital differentiator is defined as

$$H(\omega) = j\omega, \quad |\omega| \leq \pi$$

Justify this definition by comparing the frequency response $|H(\omega)|$, $\angle H(\omega)$ with that in part (b).

- (d)** By computing the frequency response $H(\omega)$, show that the discrete-time system

$$y(n) = x(n) - x(n - 1)$$

is a good approximation of a differentiator at low frequencies.

- (e)** Compute the response of the system to the input

$$x(n) = A \cos(\omega_0 n + \theta)$$

- 10.9** Use the window method with a Hamming window to design a 21-tap differentiator as shown in Fig. P10.9. Compute and plot the magnitude and phase response of the resulting filter.

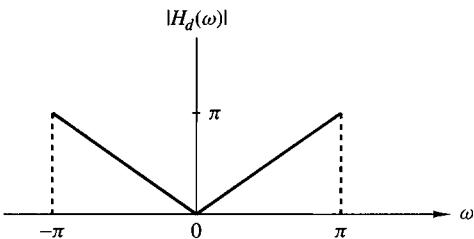


Figure P10.9

- 10.10** Use the bilinear transformation to convert the analog filter with system function

$$H(s) = \frac{s + 0.1}{(s + 0.1)^2 + 9}$$

into a digital IIR filter. Select $T = 0.1$ and compare the location of the zeros in $H(z)$ with the locations of the zeros obtained by applying the impulse invariance method in the conversion of $H(s)$.

- 10.11** Convert the analog bandpass filter designed in Example 10.4.1 into a digital filter by means of the bilinear transformation. Thereby derive the digital filter characteristic obtained in Example 10.4.2 by the alternative approach and verify that the bilinear transformation applied to the analog filter results in the same digital bandpass filter.
- 10.12** An ideal analog integrator is described by the system function $H_a(s) = 1/s$. A digital integrator with system function $H(z)$ can be obtained by use of the bilinear transformation. That is,

$$H(z) = \frac{T}{2} \frac{1 + z^{-1}}{1 - z^{-1}} \equiv H_a(s)|_{s=(2/T)(1-z^{-1})/(1+z^{-1})}$$

- (a)** Write the difference equation for the digital integrator relating the input $x(n)$ to the output $y(n)$.
- (b)** Roughly sketch the magnitude $|H_a(j\Omega)|$ and phase $\Theta(\Omega)$ of the analog integrator.

- (c) It is easily verified that the frequency response of the digital integrator is

$$H(\omega) = -j \frac{T}{2} \frac{\cos(\omega/2)}{\sin(\omega/2)} = -j \frac{T}{2} \cot \frac{\omega}{2}$$

Roughly sketch $|H(\omega)|$ and $\theta(\omega)$.

- (d) Compare the magnitude and phase characteristics obtained in parts (b) and (c). How well does the digital integrator match the magnitude and phase characteristics of the analog integrator?
- (e) The digital integrator has a pole at $z = 1$. If you implement this filter on a digital computer, what restrictions might you place on the input signal sequence $x(n)$ to avoid computational difficulties?

- 10.13** A z -plane pole-zero plot for a certain digital filter is shown in Fig. P10.13. The filter has unity gain at dc.

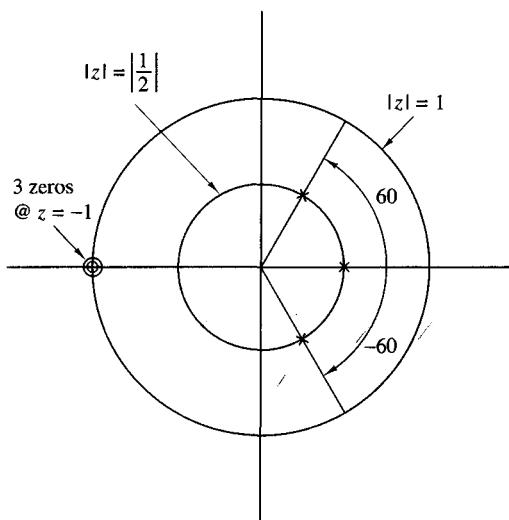


Figure P10.13

- (a) Determine the system function in the form

$$H(z) = A \left[\frac{(1 + a_1 z^{-1})(1 + b_1 z^{-1} + b_2 z^{-2})}{(1 + c_1 z^{-1})(1 + d_1 z^{-1} + d_2 z^{-2})} \right]$$

giving numerical values for the parameters $A, a_1, b_1, b_2, c_1, d_1$, and d_2 .

- (b) Draw block diagrams showing numerical values for path gains in the following forms:
- (a) Direct form II (canonic form)
 - (b) Cascade form (make each section canonic, with real coefficients)

- 10.14** Consider the pole-zero plot shown in Fig. P10.14.

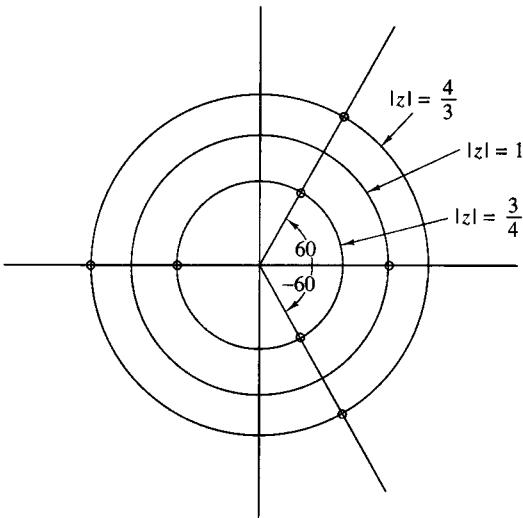


Figure P10.14

- (a) Does it represent an FIR filter?
- (b) Is it a linear-phase system?
- (c) Give a direct-form realization that exploits all symmetries to minimize the number of multiplications. Show all path gains.

- 10.15** A digital low-pass filter is required to meet the following specifications:

Passband ripple: ≤ 1 dB
 Passband edge: 4 kHz
 Stopband attenuation: ≥ 40 dB
 Stopband edge: 6 kHz
 Sample rate: 24 kHz

The filter is to be designed by performing a bilinear transformation on an analog system function. Determine what order Butterworth, Chebyshev, and elliptic analog designs must be used to meet the specifications in the digital implementation.

- 10.16** An IIR digital lowpass filter is required to meet the following specifications:

Passband ripple (or peak-to-peak ripple): ≤ 0.5 dB
 Passband edge: 1.2 kHz
 Stopband attenuation: ≥ 40 dB
 Stopband edge: 2.0 kHz
 Sample rate: 8.0 kHz

Use the design formulas in the book to determine the required filter order for

- (a) A digital Butterworth filter
- (b) A digital Chebyshev filter
- (c) A digital elliptic filter

10.17 Determine the system function $H(z)$ of the lowest-order Chebyshev digital filter that meets the following specifications:

- (a) 1-dB ripple in the passband $0 \leq |\omega| \leq 0.3\pi$.
- (b) At least 60 dB attenuation in the stopband $0.35\pi \leq |\omega| \leq \pi$. Use the bilinear transformation.

10.18 Determine the system function $H(z)$ of the lowest-order Chebyshev digital filter that meets the following specifications:

- (a) $\frac{1}{2}$ -dB ripple in the passband $0 \leq |\omega| \leq 0.24\pi$.
- (b) At least 50-dB attenuation in the stopband $0.35\pi \leq |\omega| \leq \pi$. Use the bilinear transformation.

10.19 An analog signal $x(t)$ consists of the sum of two components $x_1(t)$ and $x_2(t)$. The spectral characteristics of $x(t)$ are shown in the sketch in Fig. P10.19. The signal $x(t)$ is bandlimited to 40 kHz and it is sampled at a rate of 100 kHz to yield the sequence $x(n)$.

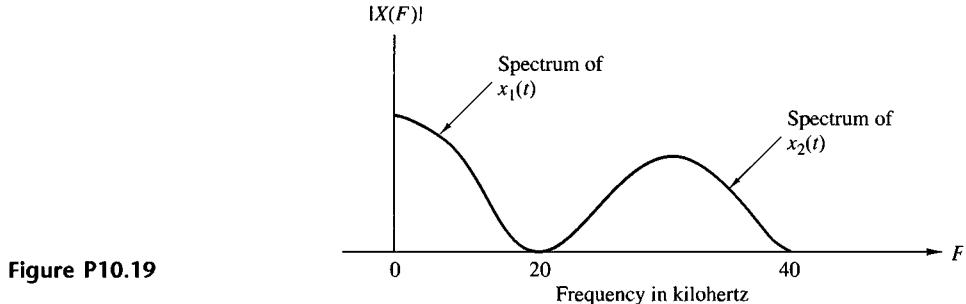


Figure P10.19

It is desired to suppress the signal $x_2(t)$ by passing the sequence $x(n)$ through a digital lowpass filter. The allowable amplitude distortion on $|X_1(f)|$ is $\pm 2\%$ ($\delta_1 = 0.02$) over the range $0 \leq |F| \leq 15$ kHz. Above 20 kHz, the filter must have an attenuation of at least 40 dB ($\delta_2 = 0.01$).

- (a) Use the Remez exchange algorithm to design the *minimum*-order linear-phase FIR filter that meets the specifications above. From the plot of the magnitude characteristic of the filter frequency response, give the actual specifications achieved by the filter.
- (b) Compare the order M obtained in part (a) with the approximate formulas given in equations (10.2.94) and (10.2.95).
- (c) For the order M obtained in part (a), design an FIR digital lowpass filter using the window technique and the Hamming window. Compare the frequency response characteristics of this design with those obtained in part (a).
- (d) Design the *minimum*-order elliptic filter that meets the given amplitude specifications. Compare the frequency response of the elliptic filter with that of the FIR filter in part (a).

- (e)** Compare the complexity of implementing the FIR filter in part (a) versus the elliptic filter obtained in part (d). Assume that the FIR filter is implemented in the direct form and the elliptic filter is implemented as a cascade of two-pole filters. Use storage requirements and the number of multiplications per output point in the comparison of complexity.

10.20 The impulse response of an analog filter is shown in Fig. P10.20.

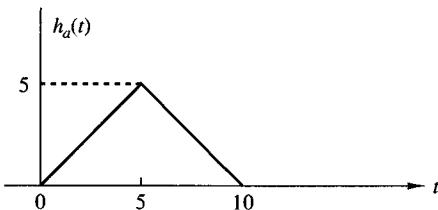


Figure P10.20

- (a)** Let $h(n) = h_a(nT)$, where $T = 1$, be the impulse response of a discrete-time filter. Determine the system function $H(z)$ and the frequency response $H(\omega)$ for this FIR filter.
- (b)** Sketch (roughly) $|H(\omega)|$ and compare this frequency response characteristic with $|H_a(j\Omega)|$.

10.21 In this problem you will be comparing some of the characteristics of analog and digital implementations of the single-pole low-pass analog system

$$H_a(s) = \frac{\alpha}{s + \alpha} \Leftrightarrow h_a(t) = e^{-\alpha t}$$

- (a)** What is the gain at dc? At what radian frequency is the analog frequency response 3 dB down from its dc value? At what frequency is the analog frequency response zero? At what time has the analog impulse response decayed to $1/e$ of its initial value?
- (b)** Give the digital system function $H(z)$ for the impulse-invariant design for this filter. What is the gain at dc? Give an expression for the 3-dB radian frequency. At what (real-valued) frequency is the response zero? How many samples are there in the unit sample time-domain response before it has decayed to $1/e$ of its initial value?
- (c)** “Prewarp” the parameter α and perform the bilinear transformation to obtain the digital system function $H(z)$ from the analog design. What is the gain at dc? At what (real-valued) frequency is the response zero? Give an expression for the 3-dB radian frequency. How many samples are there in the unit sample time-domain response before it has decayed to $1/e$ of its initial value?

- 10.22** We wish to design a FIR bandpass filter having a duration $M = 201$. $H_d(\omega)$ represents the ideal characteristic of the noncausal bandpass filter as shown in Fig. P10.22.

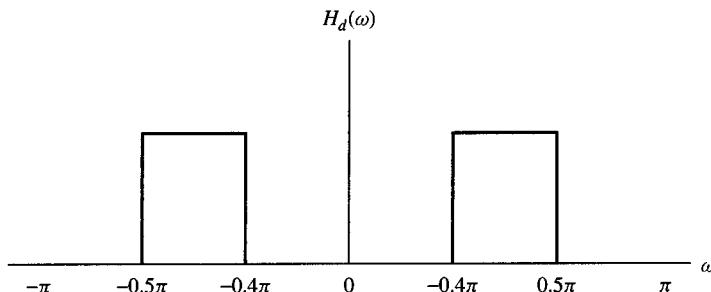


Figure P10.22

- (a) Determine the unit sample (impulse) response $h_d(n)$ corresponding to $H_d(\omega)$.
 (b) Explain how you would use the Hamming window

$$w(n) = 0.54 + 0.46 \cos\left(\frac{2\pi n}{N-1}\right), \quad -\frac{M-1}{2} \leq n \leq \frac{M-1}{2}$$

to design a FIR bandpass filter having an impulse response $h(n)$ for $0 \leq n \leq 200$.

- (c) Suppose that you were to design the FIR filter with $M = 201$ by using the frequency-sampling technique in which the DFT coefficients $H(k)$ are specified instead of $h(n)$. Give the values of $H(k)$ for $0 \leq k \leq 200$ corresponding to $H_d(e^{j\omega})$ and indicate how the frequency response of the actual filter will differ from the ideal. Would the actual filter represent a good design? Explain your answer.

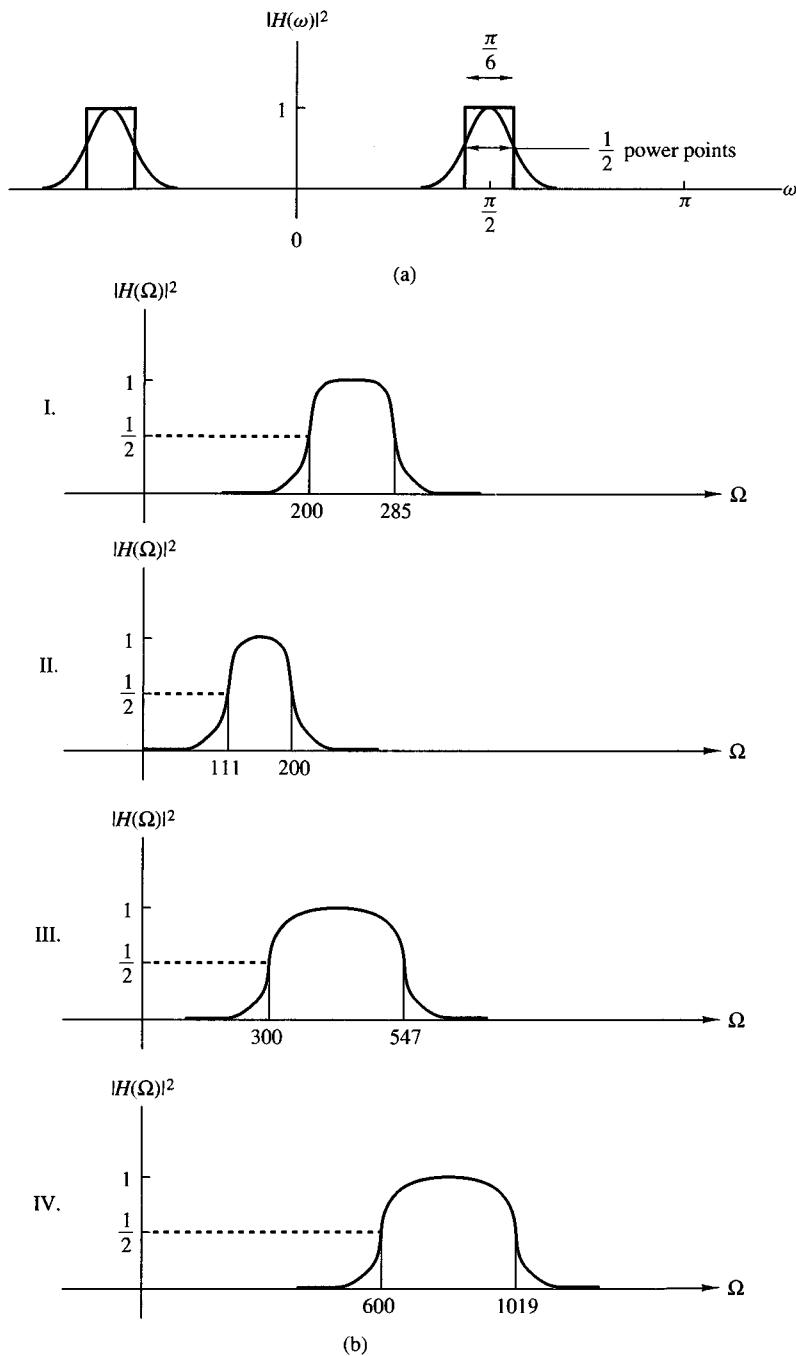
- 10.23** We wish to design a digital bandpass filter from a second-order analog lowpass Butterworth filter prototype using the bilinear transformation. The specifications on the digital filter are shown in Fig. P10.23(a). The cutoff frequencies (measured at the half-power points) for the digital filter should lie at $\omega_l = 5\pi/12$ and $\omega_u = 7\pi/12$.

The analog prototype is given by

$$H_a(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

with the half-power point at $\Omega = 1$.

- (a) Determine the system function for the digital bandpass filter.
 (b) Using the same specs on the digital filter as in part (a), determine which of the analog bandpass prototype filters shown in Fig. P10.23(b) could be transformed directly using the bilinear transformation to give the proper digital filter. Only the plot of the magnitude squared of the frequency is given.

**Figure P10.23**

10.24 Figure P10.24 shows a digital filter designed using the frequency-sampling method.

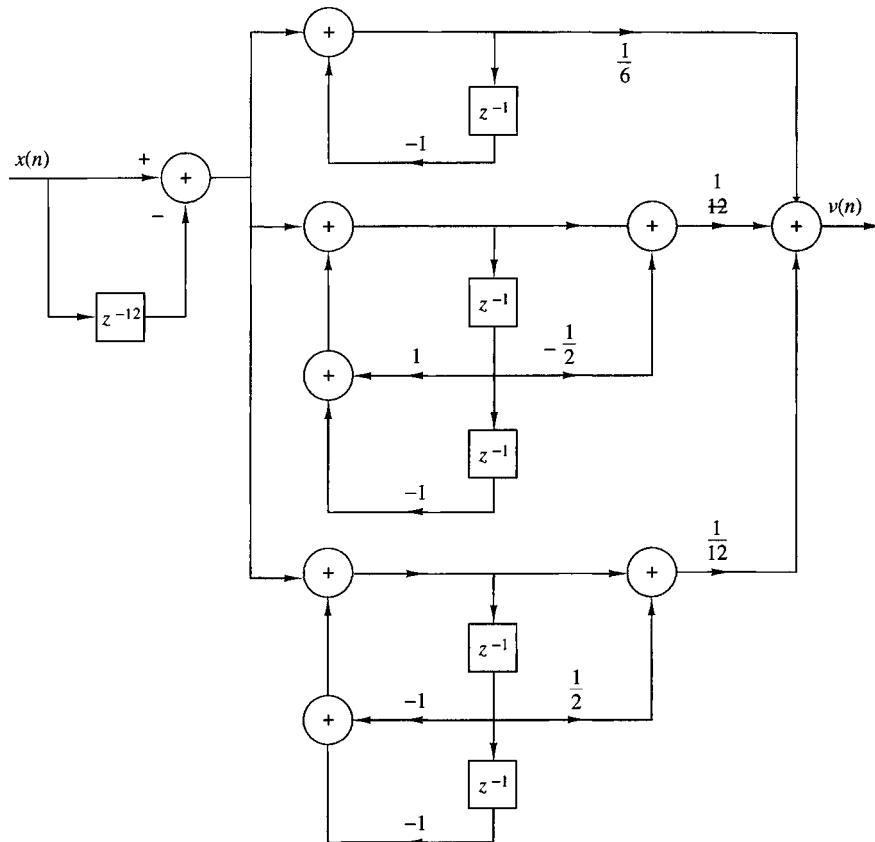


Figure P10.24

- Sketch a z -plane pole-zero plot for this filter.
- Is the filter lowpass, highpass, or bandpass?
- Determine the magnitude response $|H(\omega)|$ at the frequencies $\omega_k = \pi k/6$ for $k = 0, 1, 2, 3, 4, 5, 6$.
- Use the results of part (c) to sketch the magnitude response for $0 \leq \omega \leq \pi$ and confirm your answer to part (b).

10.25 An analog signal of the form $x_a(t) = a(t) \cos 2000\pi t$ is bandlimited to the range $900 \leq F \leq 1100$ Hz. It is used as an input to the system shown in Fig. P10.25.

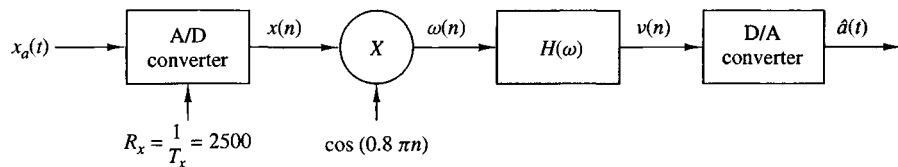


Figure P10.25

- (a) Determine and sketch the spectra for the signals $x(n)$ and $w(n)$.
 (b) Use a Hamming window of length $M = 31$ to design a lowpass linear phase FIR filter $H(\omega)$ that passes $\{a(n)\}$.
 (c) Determine the sampling rate of the A/D converter that would allow us to eliminate the frequency conversion in Fig. P10.25.

10.26 System identification Consider an unknown LTI system and an FIR system model as shown in Fig. P10.26. Both systems are excited by the same input sequence $\{x(n)\}$. The problem is to determine the coefficients $\{h(n), 0 \leq n \leq M - 1\}$ of the FIR model of the system to minimize the average squared error between the outputs of the two systems.

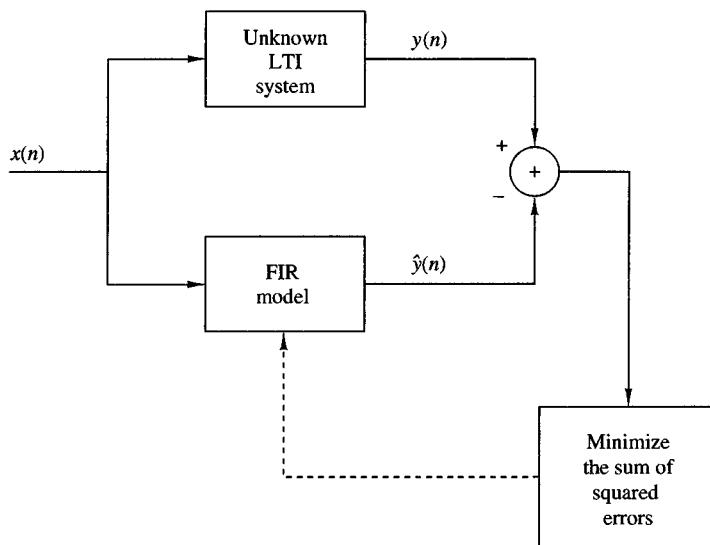


Figure P10.26

- (a) Determine the equation for the FIR filter coefficients $\{h(n), 0 \leq n \leq M - 1\}$ that minimize the least-squares error

$$\mathcal{E} = \sum_{n=0}^N [y(n) - \hat{y}(n)]^2$$

where

$$\hat{y}(n) = \sum_{k=0}^{M-1} h(k)x(n-k), \quad n = K, K+1, \dots, N$$

and $N \gg M$.

- (b) Repeat part (a) if the output of the unknown system is corrupted by an additive white noise $\{w(n)\}$ sequence with variance σ_w^2 .

- 10.27** A linear time-invariant system has an input sequence $x(n)$ and an output sequence $y(n)$. The user has access only to the system output $y(n)$. In addition, the following information is available:

The input signal is periodic with a given fundamental period N and has a flat spectral envelope, that is,

$$x(n) = \sum_{k=0}^{N-1} c_k^x e^{j(2\pi/N)kn}, \quad \text{all } n$$

where $c_k^x = 1$ for all k .

The system $H(z)$ is all pole, that is,

$$H(z) = \frac{1}{1 + \sum_{k=1}^P a_k z^{-k}}$$

but the order p and the coefficients ($a_k, 1 \leq k \leq p$) are unknown. Is it possible to determine the order p and the numerical values of the coefficients $\{a_k, 1 \leq k \leq p\}$ by taking measurements on the output $y(n)$? If yes, explain how. Is this possible for every value of p ?

- 10.28 FIR system modeling** Consider an “unknown” FIR system with impulse response $h(n)$, $0 \leq n \leq 11$, given by

$$\begin{aligned} h(0) &= h(11) = 0.309828 \times 10^{-1} \\ h(1) &= h(10) = 0.416901 \times 10^{-1} \\ h(2) &= h(9) = -0.577081 \times 10^{-1} \\ h(3) &= h(8) = -0.852502 \times 10^{-1} \\ h(4) &= h(7) = 0.147157 \times 10^0 \\ h(5) &= h(6) = 0.449188 \times 10^0 \end{aligned}$$

A potential user has access to the input and output of the system but does not have any information about its impulse response other than that it is FIR. In an effort to determine the impulse response of the system, the user excites it with a zero-mean, random sequence $x(n)$ uniformly distributed in the range $[-0.5, 0.5]$, and records the signal $x(n)$ and the corresponding output $y(n)$ for $0 \leq n \leq 199$.

- (a) By using the available information that the unknown system is FIR, the user employs the method of least squares to obtain an FIR model $h(n)$, $0 \leq n \leq M - 1$. Set up the system of linear equations, specifying the parameters $h(0), h(1), \dots, h(M - 1)$. Specify formulas we should use to determine the necessary autocorrelation and crosscorrelation values.

- (b) Since the order of the system is unknown, the user decides to try models of different orders and check the corresponding total squared error. Clearly, this error will be zero (or very close to it if the order of the model becomes equal to the order of the system). Compute the FIR models $h_M(n)$, $0 \leq n \leq M - 1$ for $M = 8, 9, 10, 11, 12, 13, 14$ as well as the corresponding total squared errors E_M , $M = 8, 9, \dots, 14$. What do you observe?
- (c) Determine and plot the frequency response of the system and the models for $M = 11, 12, 13$. Comment on the results.
- (d) Suppose now that the output of the system is corrupted by additive noise, so instead of the signal $y(n)$, $0 \leq n \leq 199$, we have available the signal

$$v(n) = y(n) + 0.01w(n)$$

where $w(n)$ is a Gaussian random sequence with zero mean and variance $\sigma^2 = 1$.

Repeat part (b) of Problem 10.27 by using $v(n)$ instead of $y(n)$ and comment on the results. The quality of the model can be also determined by the quantity

$$Q = \frac{\sum_{n=0}^{\infty} [h(n) - \hat{h}(n)]^2}{\sum_{n=0}^{\infty} h^2(n)}$$

- 10.29 Filter design by Padé approximation** Let the desired impulse response $h_d(n)$, $n \geq 0$, of an IIR filter be specified. The filter to be designed to approximate $\{h_d(n)\}$ has the system function

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

$H(z)$ has $L = M + N + 1$ parameters, namely, the coefficients $\{a_k\}$ and $\{b_k\}$ to be determined. Suppose the input to the filter is $x(n) = \delta(n)$. Then, the response of the filter is $y(n) = h(n)$, and, hence,

$$\begin{aligned} h(n) &= -a_1 h(n-1) - a_2 h(n-2) - \dots - a_N h(n-N) \\ &\quad + b_0 \delta(n) + b_1 \delta(n-1) + \dots + b_M \delta(n-M) \end{aligned} \tag{1}$$

- (a) Show that equation (1) reduces to

$$h(n) = -a_1 h(n-1) - a_2 h(n-2) - \dots - a_N h(n-N) + b_n, \quad 0 \leq n \leq M \tag{2}$$

- (b) Show that for $n > M$, equation (1) reduces to

$$h(n) = -a_1 h(n-1) - a_2 h(n-2) - \dots - a_N h(n-N), \quad n > M \tag{3}$$

- (c) Explain how equations (2) and (3) can be used to determine $\{a_k\}$ and $\{b_k\}$ by letting $h(n) = h_d(n)$ for $0 \leq n \leq N + M$. (This filter design method in which $h(n)$ exactly matches the desired response $h_d(n)$ for $0 \leq n \leq M + N$ is called the Padé approximation method.)

10.30 Suppose the desired unit sample response is

$$h_d(n) = 2 \left(\frac{1}{2} \right)^n u(n)$$

- (a) Use the Padé approximation described in Problem 10.29 to determine $h(n)$.
- (b) Compare the frequency response of $H(\omega)$ with that of the desired filter response $H_d(\omega)$.

10.31 *Shanks method for least-squares filter design* Suppose that we are given the desired response $h_d(n)$, $n \geq 0$, and we wish to determine the coefficients $\{a_k\}$ and $\{b_k\}$ of an IIR filter with system function

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

such that the sum of squared errors between $h_d(n)$ and $h(n)$ is minimized.

- (a) If the input to $H(z)$ is $x(n) = \delta(n)$, what is the difference equation satisfied by the filter $H(z)$?
- (b) Show that for $n > M$, an estimate of $h_d(n)$ is

$$\hat{h}_d(n) = - \sum_{k=1}^N a_k h_d(n-k)$$

and determine the equations for the coefficients $\{a_k\}$ by minimizing the sum of squared errors

$$\mathcal{E}_1 = \sum_{n=M+1}^{\infty} [h_d(n) - \hat{h}_d(n)]^2$$

Thus, the filter coefficients $\{a_k\}$ that specify the denominator of $H(z)$ are determined.

- (c) To determine the parameters $\{b_k\}$ consider the system shown in Fig. P10.31, where

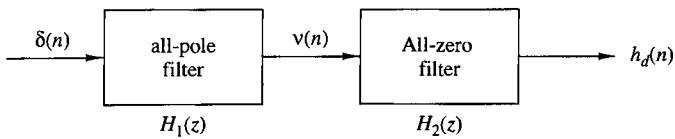
$$H_1(z) = \frac{1}{1 + \sum_{k=1}^N \hat{a}_k z^{-k}}$$

$$H_2(z) = \sum_{k=1}^N b_k z^{-k}$$

and $\{\hat{a}_k\}$ are the coefficients determined in part (b).

If the response of $H_1(z)$ to the input $\delta(n)$ is denoted as

$$v(n) = - \sum_{k=1}^N \hat{a}_k v(n-k) + \delta(n)$$

**Figure P10.31**

and the output of $H_2(z)$ is denoted as $\hat{h}_d(n)$, determine the equation for the parameters $\{b_k\}$ that minimize the sum of squared errors

$$\mathcal{E}_2 = \sum_{n=0}^{\infty} [h_d(n) - \hat{h}_d(n)]^2$$

(The preceding least-squares method for filter design is due to Shanks (1967).)

- 10.32** Use the Shanks method for filter design as described in Problem 10.31 to determine the parameters $\{a_k\}$ and $\{b_k\}$ of

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

when the desired response is the impulse response of the three-pole and three-zero type II lowpass Chebyshev digital filter having a system function

$$H_d(z) = \frac{0.3060(1 + z^{-1})(0.2652 - 0.09z^{-1} + 0.2652z^{-2})}{(1 - 0.3880z^{-1})(1 - 1.1318z^{-1} + 0.5387z^{-2})}$$

- (a) Plot $h_d(n)$ and then observe that $h_d(n) \approx 0$ for $n > 50$.
- (b) Determine the pole and zero positions for the filter $H(z)$ obtained by the Shanks method for $(N, M) = (3, 2)$, $(N, M) = (3, 3)$ and $(N, M) = (4, 3)$, and compare these results with the poles and zeros of $H_d(z)$. Comment on the similarities and differences.

Multirate Digital Signal Processing

In many practical applications of digital signal processing, one is faced with the problem of changing the sampling rate of a signal, either increasing it or decreasing it by some amount. For example, in telecommunication systems that transmit and receive different types of signals (e.g., teletype, facsimile, speech, video, etc.), there is a requirement to process the various signals at different rates commensurate with the corresponding bandwidths of the signals. The process of converting a signal from a given rate to a different rate is called *sampling rate conversion*. In turn, systems that employ multiple sampling rates in the processing of digital signals are called *multirate digital signal processing systems*.

Sampling rate conversion of a digital signal can be accomplished in one of two general methods. One method is to pass the digital signal through a D/A converter, filter it if necessary, and then to resample the resulting analog signal at the desired rate (i.e., to pass the analog signal through an A/D converter). The second method is to perform the sampling rate conversion entirely in the digital domain.

One apparent advantage of the first method is that the new sampling rate can be arbitrarily selected and need not have any special relationship to the old sampling rate. A major disadvantage, however, is the signal distortion, introduced by the D/A converter in the signal reconstruction, and by the quantization effects in the A/D conversion. Sampling rate conversion performed in the digital domain avoids this major disadvantage.

In this chapter we describe sampling rate conversion and multirate signal processing in the digital domain. First we describe sampling rate conversion by a rational factor and present several methods for implementing the rate converter, including single-stage and multistage implementations. Then, we describe a method for sampling rate conversion by an arbitrary factor and discuss its implementation. We

present several applications of sampling rate conversion in multirate signal processing systems, which include the implementation of narrowband filters, digital filter banks, subband coding, transmultiplexers, and quadrature mirror filters.

11.1 Introduction

The process of sampling rate conversion can be developed and understood using the idea of “resampling after reconstruction.” In this theoretical approach, a discrete-time signal is ideally reconstructed and the resulting continuous-time signal is resampled at a different sampling rate. This idea leads to a mathematical formulation that enables the realization of the entire process by means of digital signal processing.

Let $x(t)$ be a continuous-time signal that is sampled at a rate $F_x = 1/T_x$ to generate a discrete-time signal $x(nT_x)$. From the samples $x(nT_x)$ we can generate a continuous-time signal using the interpolation formula

$$y(t) = \sum_{n=-\infty}^{\infty} x(nT_x)g(t - nT_x) \quad (11.1.1)$$

If the bandwidth of $x(t)$ is less than $F_x/2$ and the interpolation function is given by

$$g(t) = \frac{\sin(\pi t/T_x)}{\pi t/T_x} \xleftrightarrow{\mathcal{F}} G(F) = \begin{cases} T_x, & |F| \leq F_x/2 \\ 0, & \text{otherwise} \end{cases} \quad (11.1.2)$$

then $y(t) = x(t)$; otherwise $y(t) \neq x(t)$. In practice, perfect recovery of $x(t)$ is not possible because the infinite summation in (11.1.1) should be replaced by a finite summation.

To perform sampling rate conversion we simply evaluate (11.1.1) at time instants $t = mT_y$, where $F_y = 1/T_y$ is the desired sampling frequency. Therefore, the general formula for sampling rate conversion becomes

$$y(mT_y) = \sum_{n=-\infty}^{\infty} x(nT_x)g(mT_y - nT_x) \quad (11.1.3)$$

which expresses directly the samples of the desired sequence in terms of the samples of the original sequence and sampled values of the reconstruction function at positions $(mT_y - nT_x)$. The computation of $y(nT_y)$ requires (a) the input sequence $x(nT_x)$, (b) the reconstruction function $g(t)$, and (c) the time instants nT_x and mT_y of the input and output samples. The values $y(mT_y)$ calculated by this equation are accurate only if $F_y > F_x$. If $F_y < F_x$, we should filter out the frequency components of $x(t)$ above $F_y/2$ before resampling in order to prevent aliasing. Therefore, the sampling rate conversion formula (11.1.3) yields $y(mT_y) = x(mT_y)$ if we use (11.1.2) and $X(F) = 0$ for $|F| \geq \min\{F_x/2, F_y/2\}$.

If $T_y = T_x$, equation (11.1.3) becomes a convolution summation, which corresponds to an LTI system. To understand the meaning of (11.1.3) for $T_y \neq T_x$, we rearrange the argument of $g(t)$ as follows:

$$y(mT_y) = \sum_{n=-\infty}^{\infty} x(nT_x)g\left(T_x\left(\frac{mT_y}{T_x} - n\right)\right) \quad (11.1.4)$$

The term mT_y/T_x can be decomposed into an integer part k_m and a fractional part Δ_m , $0 \leq \Delta_m < 1$, as

$$\frac{mT_y}{T_x} = k_m + \Delta_m \quad (11.1.5)$$

where

$$k_m = \left\lfloor \frac{mT_y}{T_x} \right\rfloor \quad (11.1.6)$$

and

$$\Delta_m = \frac{mT_y}{T_x} - \left\lfloor \frac{mT_y}{T_x} \right\rfloor \quad (11.1.7)$$

The symbol $\lfloor a \rfloor$ denotes the largest integer contained in a . The quantity Δ_m specifies the position of the current sample within the sample period T_x . Substituting (11.1.5) into (11.1.4) we obtain

$$y(mT_y) = \sum_{n=-\infty}^{\infty} x(nT_x)g((k_m + \Delta_m - n)T_x) \quad (11.1.8)$$

If we change the index of summation in (11.1.8) from n to $k = k_m - n$, we have

$$\begin{aligned} y(mT_y) &= y((k_m + \Delta_m)T_x) \\ &= \sum_{k=-\infty}^{\infty} g(kT_x + \Delta_m T_x)x((k_m - k)T_x) \end{aligned} \quad (11.1.9)$$

Equation (11.1.9) provides the fundamental equation for the discrete-time implementation of sampling rate conversion. This process is illustrated in Figure 11.1.1. We note that (a) given T_x and T_y the input and output sampling times are fixed, (b) the function $g(t)$ is shifted for each m such that the value $g(\Delta_m T_x)$ is positioned at $t = mT_y$, and (c) the required values of $g(t)$ are determined at the input sampling times. For each value of m , the fractional interval Δ_m determines the impulse response coefficients whereas the index k_m specifies the corresponding input samples

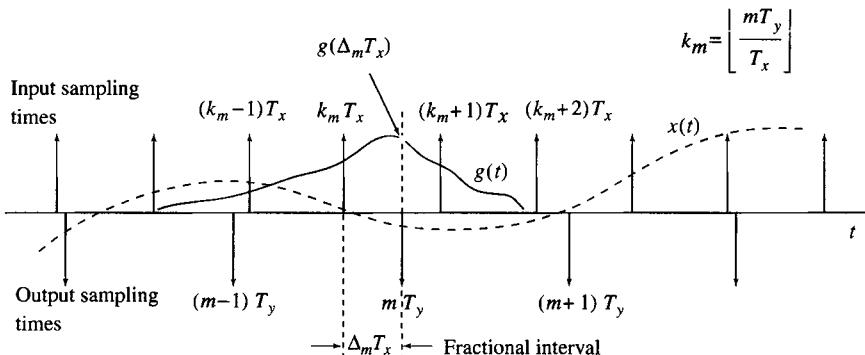
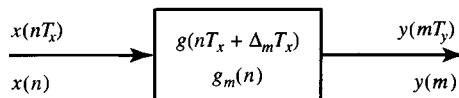


Figure 11.1.1 Illustration of timing relations for sampling rate conversion.

Figure 11.1.2
Discrete-time linear time-varying system for sampling rate conversion.



needed to compute the sample $y(mT_y)$. Since for any given value of m the index k_m is an integer number, $y(mT_y)$ is the convolution between the input sequence $x(nT_x)$ and an impulse response $g((n + \Delta_m)T_x)$. The difference between (11.1.8) and (11.1.9) is that the first shifts a “changing” reconstruction function whereas the second shifts the “fixed” input sequence.

The sampling rate conversion process defined by (11.1.9) is a discrete-time *linear and time-varying* system, because it requires a different impulse response

$$g_m(nT_x) = g((n + \Delta_m)T_x) \quad (11.1.10)$$

for each output sample $y(mT_y)$. Therefore, a new set of coefficients should be computed or retrieved from storage for the computation of *every* output sample (see Figure 11.1.2). This procedure may be inefficient when the function $g(t)$ is complicated and the number of required values is large. This dependence on m prohibits the use of recursive structures because the required past output values must be computed using an impulse response for the present value of Δ_m .

A significant simplification results when the ratio T_y/T_x is constrained to be a rational number, that is,

$$\frac{T_y}{T_x} = \frac{F_x}{F_y} = \frac{D}{I} \quad (11.1.11)$$

where D and I are relatively prime integers. To this end, we express the offset Δ_m as

$$\Delta_m = \frac{mD}{I} - \left\lfloor \frac{mD}{I} \right\rfloor = \frac{1}{I} \left(mD - \left\lfloor \frac{mD}{I} \right\rfloor I \right) = \frac{1}{I} (mD)_I \quad (11.1.12)$$

where $(k)_I$ denotes the value of k modulo I . From (11.1.12) it is clear that Δ_m can take on only I unique values $0, 1/I, \dots, (I-1)/I$, so that there are only I distinct impulse responses possible. Since $g_m(nT_x)$ can take on I distinct sets of values, it is periodic in m ; that is,

$$g_m(nT_x) = g_{m+rI}(nT_x), \quad r = 0, \pm 1, \pm 2, \dots \quad (11.1.13)$$

Thus the system $g_m(nT_x)$ is a linear and *periodically time-varying* discrete-time system. Such systems have been widely studied for a wide range of applications (Meyers and Burrus, 1975). This is a great simplification compared to the *continuously* time-varying discrete-time system in (11.1.10).

To illustrate these concepts we consider two important special cases. We start with the process of reducing the sampling rate by an integer factor D , which is known as *decimation* or *downsampling*. If we set $T_y = DT_x$ in (11.1.3), we have

$$y(mT_y) = y(mDT_x) = \sum_{k=-\infty}^{\infty} x(kT_x)g((mD - k)T_x) \quad (11.1.14)$$

We note that the input signal and the impulse response are sampled with a period T_x . However, the impulse response is shifted at increments of $T_y = DT_x$ because we need to compute only one out of every D samples. Since $I = 1$, we have $\Delta_m = 0$ and therefore there is only one impulse response $g(nT_x)$, for all m . This process is illustrated in Figure 11.1.3 for $D = 2$.

We consider now the process of increasing the sampling rate by an integer factor I , which is called *upsampling* or *interpolation*. If we set $T_y = T_x/I$ in (11.1.3), we have

$$y(mT_y) = \sum_{k=-\infty}^{\infty} x(kT_x)g(m(T_x/I) - kT_x) \quad (11.1.15)$$

We note that both $x(t)$ and $g(t)$ are sampled with a period T_x ; however, the impulse response is shifted at increments of $T_y = T_x/I$ for the computation of each output sample. This is required to “fill-in” an additional number of $(I - 1)$ samples within each period T_x . This is illustrated in Figure 11.1.4(a)–(b) for $I = 2$. Each “fractional shifting” requires that we resample $g(t)$, resulting in a new impulse response $g_m(nT_x) = g(nT_x + mT_x/I)$, for $m = 0, 1, \dots, I-1$ in agreement with (11.1.14). Careful inspection of Figure 11.1.4(a)–(b) shows that if we determine an impulse response sequence $g(nT_y)$ and create a new sequence $v(nT_y)$ by inserting $(I - 1)$ zero samples between successive samples of $x(nT_x)$, we can compute $y(mT_y)$ as the convolution of the sequences $g(nT_y)$ and $x(nT_y)$. This idea is illustrated in Figure 11.1.4(c) for $I = 2$.

In the next few sections we discuss in more detail the properties, design, and structures for the implementation of sampling rate conversion entirely in the discrete-time domain. For convenience, we usually drop the sampling periods T_x and T_y from the argument of discrete-time signals. However, occasionally, it will be beneficial to the reader to reintroduce and think in terms of continuous-time quantities and units.

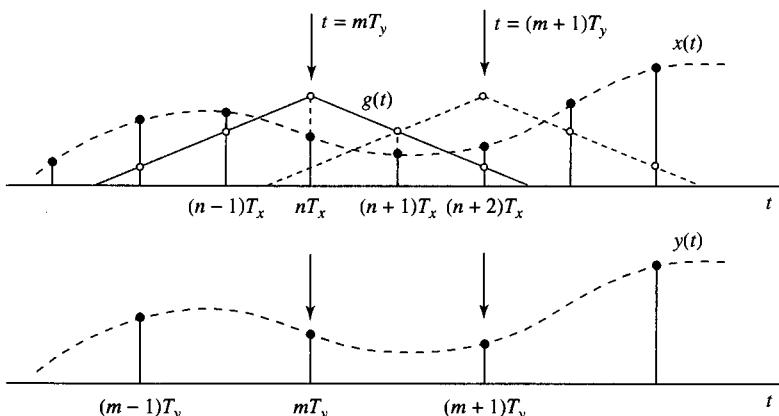


Figure 11.1.3 Illustration of timing relations for sampling rate decrease by an integer factor $D = 2$. A single impulse response, sampled with period T_x , is shifted at steps equal to $T_y = DT_x$ to generate the output samples.

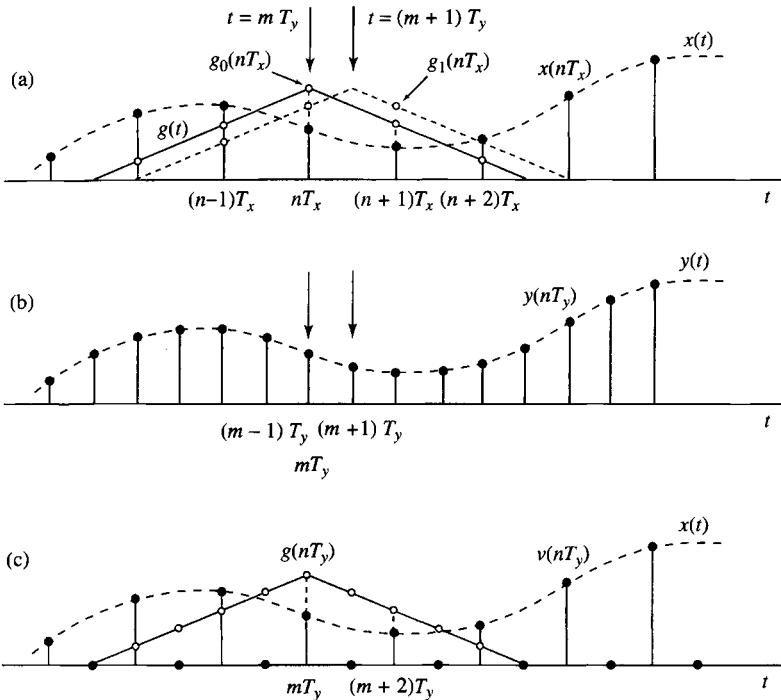


Figure 11.1.4 Illustration of timing relations for sampling rate increase by an integer factor $I = 2$. The approach in (a) requires one impulse response for the even-numbered and one for the odd-numbered output samples. The approach in (c) requires only one impulse response, obtained by interleaving the impulse responses in (a).

11.2 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 $0 \leq |\omega| \leq \pi$ or, equivalently, $|F| \leq F_x/2$. We know that if we reduce the sampling rate simply by selecting every D th 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. 11.2.1. 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

$$H_D(\omega) = \begin{cases} 1, & |\omega| \leq \pi/D \\ 0, & \text{otherwise} \end{cases} \quad (11.2.1)$$

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 $|\omega| \leq \pi/D$ are of interest in further processing of the signal.

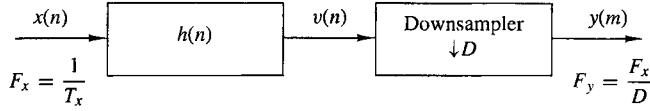


Figure 11.2.1 Decimation by a factor D .

The output of the filter is a sequence $v(n)$ given as

$$v(n) = \sum_{k=0}^{\infty} h(k)x(n-k) \quad (11.2.2)$$

which is then downsampled by the factor D to produce $y(m)$. Thus

$$\begin{aligned} y(m) &= v(mD) \\ &= \sum_{k=0}^{\infty} h(k)x(mD - k) \end{aligned} \quad (11.2.3)$$

Although the filtering operation on $x(n)$ is linear and time invariant, the down-sampling 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 $y(m)$ to the spectrum of the input sequence $x(n)$. First, it is convenient to define a sequence $\tilde{v}(n)$ as

$$\tilde{v}(n) = \begin{cases} v(n), & n = 0, \pm D, \pm 2D, \dots \\ 0, & \text{otherwise} \end{cases} \quad (11.2.4)$$

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. 11.2.2. The discrete Fourier series representation of $p(n)$ is

$$p(n) = \frac{1}{D} \sum_{k=0}^{D-1} e^{j2\pi kn/D} \quad (11.2.5)$$

Hence

$$\tilde{v}(n) = v(n)p(n) \quad (11.2.6)$$

and

$$y(m) = \tilde{v}(mD) = v(mD)p(mD) = v(mD) \quad (11.2.7)$$

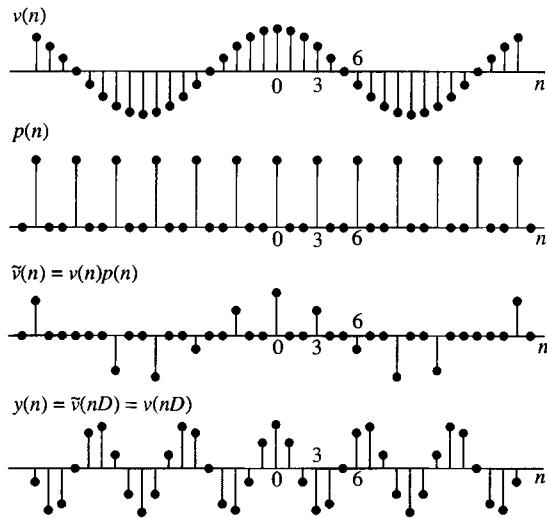


Figure 11.2.2
Steps required to facilitate the mathematical description of downsampling by a factor D , using a sinusoidal sequence for illustration.

Now the z -transform of the output sequence $y(m)$ is

$$\begin{aligned}
 Y(z) &= \sum_{m=-\infty}^{\infty} y(m)z^{-m} \\
 &= \sum_{m=-\infty}^{\infty} \tilde{v}(mD)z^{-m} \\
 Y(z) &= \sum_{m=-\infty}^{\infty} \tilde{v}(m)z^{-m/D}
 \end{aligned} \tag{11.2.8}$$

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 (11.2.5) and (11.2.6) in (11.2.8), we obtain

$$\begin{aligned}
 Y(z) &= \sum_{m=-\infty}^{\infty} v(m) \left[\frac{1}{D} \sum_{k=0}^{D-1} e^{j2\pi mk/D} \right] z^{-m/D} \\
 &= \frac{1}{D} \sum_{k=0}^{D-1} \sum_{m=-\infty}^{\infty} v(m) (e^{-j2\pi k/D} z^{1/D})^{-m} \\
 &= \frac{1}{D} \sum_{k=0}^{D-1} V(e^{-j2\pi k/D} z^{1/D}) \\
 &= \frac{1}{D} \sum_{k=0}^{D-1} H_D(e^{-j2\pi k/D} z^{1/D}) X(e^{-j2\pi k/D} z^{1/D})
 \end{aligned} \tag{11.2.9}$$

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 ,

$$\omega_y = \frac{2\pi F}{F_y} = 2\pi F T_y \quad (11.2.10)$$

Since the sampling rates are related by the expression

$$F_y = \frac{F_x}{D} \quad (11.2.11)$$

it follows that the frequency variables ω_y and

$$\omega_x = \frac{2\pi F}{F_x} = 2\pi F T_x \quad (11.2.12)$$

are related by

$$\omega_y = D\omega_x \quad (11.2.13)$$

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 (11.2.9) on the unit circle, can be expressed as

$$Y(\omega_y) = \frac{1}{D} \sum_{k=0}^{D-1} H_D \left(\frac{\omega_y - 2\pi k}{D} \right) X \left(\frac{\omega_y - 2\pi k}{D} \right) \quad (11.2.14)$$

With a properly designed filter $H_D(\omega)$, the aliasing is eliminated and, consequently, all but the first term in (11.2.14) vanish. Hence

$$\begin{aligned} Y(\omega_y) &= \frac{1}{D} H_D \left(\frac{\omega_y}{D} \right) X \left(\frac{\omega_y}{D} \right) \\ &= \frac{1}{D} X \left(\frac{\omega_y}{D} \right) \end{aligned} \quad (11.2.15)$$

for $0 \leq |\omega_y| \leq \pi$. The spectra for the sequences $x(n)$, $v(n)$, and $y(m)$ are illustrated in Fig. 11.2.3.

EXAMPLE 11.2.1

Design a decimator that downsamples an input signal $x(n)$ by a factor $D = 2$. Use the Remez algorithm to determine the coefficients of the FIR filter that has a 0.1-dB ripple in the passband and is down by at least 30 dB in the stopband.

Solution. A filter length $M = 30$ achieves the design specifications given above. The frequency response is illustrated in Fig. 11.2.4. Note that the cutoff frequency is $\omega_c = \pi/2$.

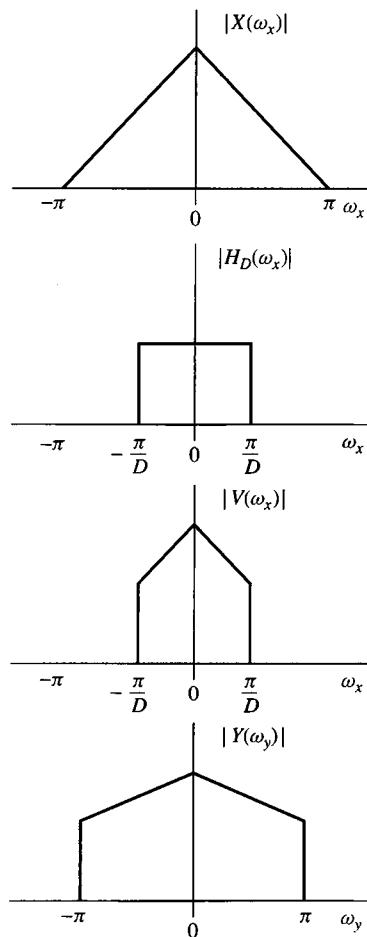


Figure 11.2.3
Spectra of signals in the
decimation of $x(n)$ by a
factor D .

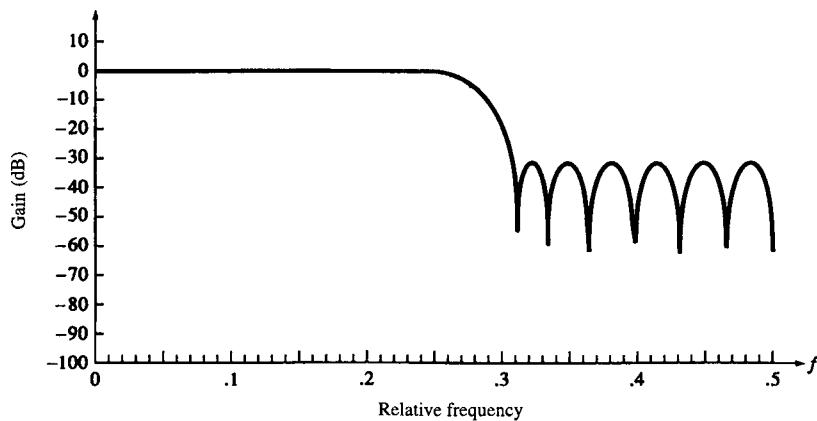


Figure 11.2.4 Magnitude response of linear-phase FIR filter of length $M = 30$ in Example 11.2.1.

11.3 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

$$v(m) = \begin{cases} x(m/I), & m = 0, \pm I, \pm 2I, \dots \\ 0, & \text{otherwise} \end{cases} \quad (11.3.1)$$

and its sampling rate is identical to the rate of $y(m)$. This sequence has a z -transform

$$\begin{aligned} V(z) &= \sum_{m=-\infty}^{\infty} v(m)z^{-m} \\ &= \sum_{m=-\infty}^{\infty} x(m)z^{-mI} \\ &= X(z^I) \end{aligned} \quad (11.3.2)$$

The corresponding spectrum of $v(m)$ is obtained by evaluating (11.3.2) on the unit circle. Thus

$$V(\omega_y) = X(\omega_y I) \quad (11.3.3)$$

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

$$\omega_y = \frac{\omega_x}{I} \quad (11.3.4)$$

The spectra $X(\omega_x)$ and $V(\omega_y)$ are illustrated in Fig. 11.3.1. 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 $0 \leq \omega_y \leq \pi/I$ 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

$$H_I(\omega_y) = \begin{cases} C, & 0 \leq |\omega_y| \leq \pi/I \\ 0, & \text{otherwise} \end{cases} \quad (11.3.5)$$

where C is a scale factor required to properly normalize the output sequence $y(m)$. Consequently, the output spectrum is

$$Y(\omega_y) = \begin{cases} CX(\omega_y I), & 0 \leq |\omega_y| \leq \pi/I \\ 0, & \text{otherwise} \end{cases} \quad (11.3.6)$$

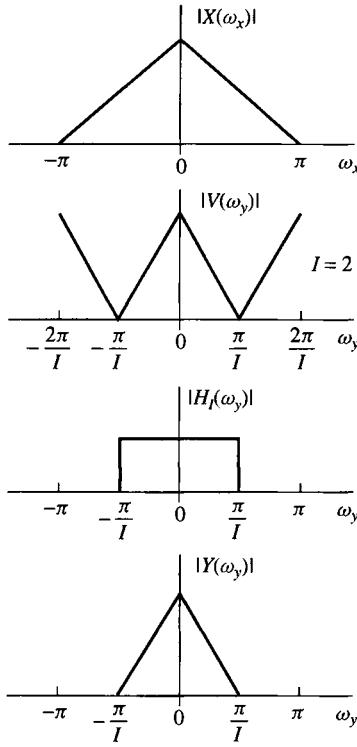


Figure 11.3.1
Spectra of $x(n)$ and $v(n)$
where $V(\omega_y) = X(\omega_y I)$.

The scale factor C is selected so that the output $y(m) = x(m/I)$ for $m = 0, \pm I, \pm 2I, \dots$. For mathematical convenience, we select the point $m = 0$. Thus

$$\begin{aligned} y(0) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} Y(\omega_y) d\omega_y \\ &= \frac{C}{2\pi} \int_{-\pi/I}^{\pi/I} X(\omega_y I) d\omega_y \end{aligned} \quad (11.3.7)$$

Since $\omega_y = \omega_x/I$, (11.3.7) can be expressed as

$$\begin{aligned} y(0) &= \frac{C}{I} \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\omega_x) d\omega_x \\ &= \frac{C}{I} x(0) \end{aligned} \quad (11.3.8)$$

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

$$y(m) = \sum_{k=-\infty}^{\infty} h(m-k)v(k) \quad (11.3.9)$$

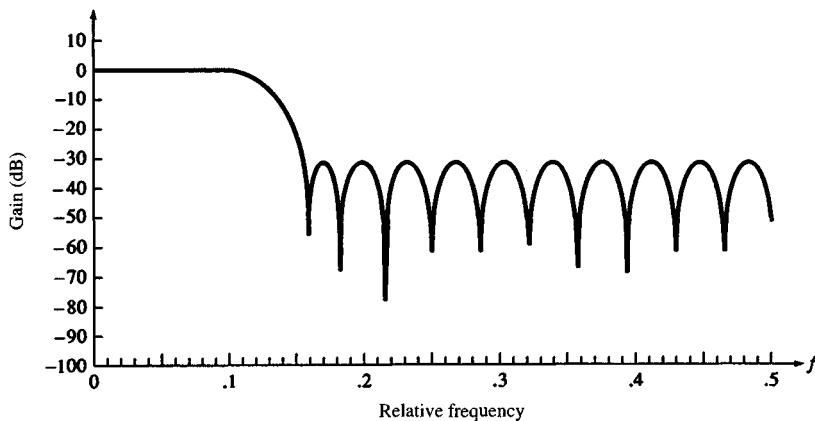


Figure 11.3.2 Magnitude response of linear-phase FIR filter of length $M = 30$ in Example 11.3.1.

Since $v(k) = 0$ except at multiples of I , where $v(kI) = x(k)$, (11.3.9) becomes

$$y(m) = \sum_{k=-\infty}^{\infty} h(m - kI)x(k) \quad (11.3.10)$$

EXAMPLE 11.3.1

Design a interpolator that increases the input sampling rate by a factor of $I = 5$. Use the Remez algorithm to determine the coefficients of the FIR filter that has a 0.1-dB ripple in the passband and is down by at least 30 dB in the stopband.

Solution. A filter length $M = 30$ achieves the design specifications given above. The frequency response of the FIR filter is illustrated in Fig. 11.3.2. Note that the cutoff frequency is $\omega_c = \pi/5$.

11.4 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. 11.4.1.

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. 11.4.1, the two filters with impulse response $\{h_u(k)\}$ and $\{h_d(k)\}$ are operated at the same rate, namely IF_x , and hence can be combined into a single lowpass filter with impulse response $h(k)$ as illustrated in Fig. 11.4.2. The frequency response $H(\omega_v)$ of the combined filter

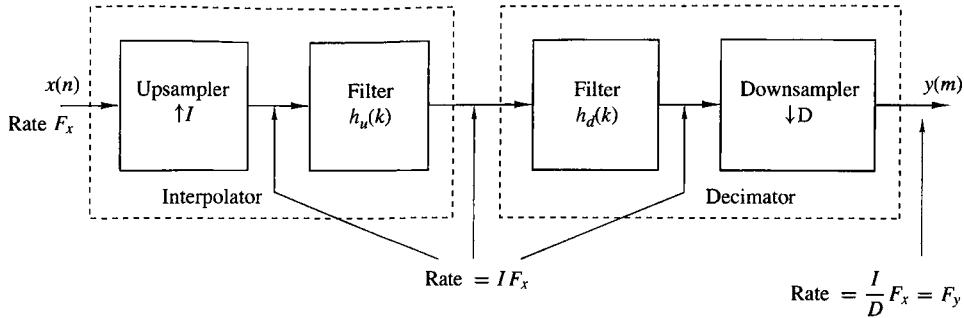


Figure 11.4.1 Method for sampling rate conversion by a factor I/D .

must incorporate the filtering operations for both interpolation and decimation, and hence it should ideally possess the frequency response characteristic

$$H(\omega_v) = \begin{cases} I, & 0 \leq |\omega_v| \leq \min(\pi/D, \pi/I) \\ 0, & \text{otherwise} \end{cases} \quad (11.4.1)$$

where $\omega_v = 2\pi F/F_v = 2\pi F/IF_x = \omega_x/I$.

In the time domain, the output of the upsampler is the sequence

$$v(l) = \begin{cases} x(l/I), & l = 0, \pm I, \pm 2I, \dots \\ 0, & \text{otherwise} \end{cases} \quad (11.4.2)$$

and the output of the linear time-invariant filter is

$$\begin{aligned} w(l) &= \sum_{k=-\infty}^{\infty} h(l-k)v(k) \\ &= \sum_{k=-\infty}^{\infty} h(l-kI)x(k) \end{aligned} \quad (11.4.3)$$

Finally, the output of the sampling rate converter is the sequence $\{y(m)\}$, which is

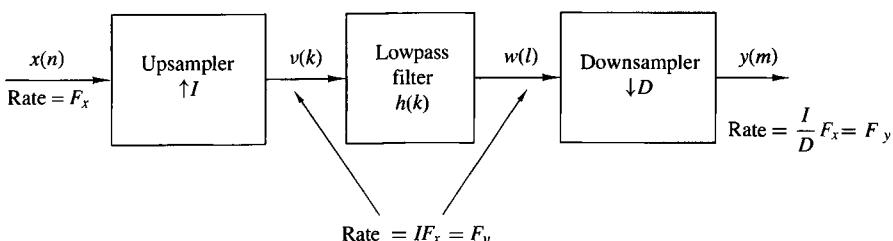


Figure 11.4.2 Method for sampling rate conversion by a factor I/D .

obtained by downsampling the sequence $\{w(l)\}$ by a factor of D . Thus

$$\begin{aligned} y(m) &= w(mD) \\ &= \sum_{k=-\infty}^{\infty} h(mD - kI)x(k) \end{aligned} \quad (11.4.4)$$

It is illuminating to express (11.4.4) in a different form by making a change in variable. Let

$$k = \left\lfloor \frac{mD}{I} \right\rfloor - n \quad (11.4.5)$$

where the notation $\lfloor r \rfloor$ denotes the largest integer contained in r . With this change in variable, (11.4.4) becomes

$$y(m) = \sum_{n=-\infty}^{\infty} h\left(mD - \left\lfloor \frac{mD}{I} \right\rfloor I + nI\right)x\left(\left\lfloor \frac{mD}{I} \right\rfloor - n\right) \quad (11.4.6)$$

We note that

$$\begin{aligned} mD - \left\lfloor \frac{mD}{I} \right\rfloor I &= mD, \quad \text{modulo } I \\ &= (mD)_I \end{aligned}$$

Consequently, (11.4.6) can be expressed as

$$y(m) = \sum_{n=-\infty}^{\infty} h(nI + (mD)_I)x\left(\left\lfloor \frac{mD}{I} \right\rfloor - n\right) \quad (11.4.7)$$

which is the discrete-time version of (11.1.9).

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

$$g(n, m) = h(nI + (mD)_I), \quad -\infty < m, n < \infty \quad (11.4.8)$$

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 ,

$$\begin{aligned} g(n, m + kI) &= h(nI + (mD + kDI)_I) \\ &= h(nI + (mD)_I) \\ &= g(n, m) \end{aligned} \quad (11.4.9)$$

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 $h(I)$ is

$$\begin{aligned} V(\omega_v) &= H(\omega_v)X(\omega_v I) \\ &= \begin{cases} IX(\omega_v I), & 0 \leq |\omega_v| \leq \min(\pi/D, \pi/I) \\ 0, & \text{otherwise} \end{cases} \end{aligned} \quad (11.4.10)$$

The spectrum of the output sequence $y(m)$, obtained by decimating the sequence $v(n)$ by a factor of D , is

$$Y(\omega_y) = \frac{1}{D} \sum_{k=0}^{D-1} V\left(\frac{\omega_y - 2\pi k}{D}\right) \quad (11.4.11)$$

where $\omega_y = D\omega_v$. Since the linear filter prevents aliasing as implied by (11.4.10), the spectrum of the output sequence given by (11.4.11) reduces to

$$Y(\omega_y) = \begin{cases} \frac{I}{D} X\left(\frac{\omega_y}{D}\right), & 0 \leq |\omega_y| \leq \min\left(\pi, \frac{\pi D}{I}\right) \\ 0, & \text{otherwise} \end{cases} \quad (11.4.12)$$

EXAMPLE 11.4.1

Design a sample rate converter that increases the sampling rate by a factor of 2.5. Use the Remez algorithm to determine the coefficients of the FIR filter that has a 0.1-dB ripple in the passband and is down by at least 30 dB in the stopband. Specify the sets of time-varying coefficients $g(n, m)$ used in the realization of the sampling rate converter.

Solution. The FIR filter that meets the specifications of this problem is exactly the same as the filter designed in Example 11.3.1. Its bandwidth is $\pi/5$.

The coefficients of the FIR filter are given by (11.4.8) as

$$\begin{aligned} g(n, m) &= h(nI + (mD)_I) \\ &= h\left(nI + mD - \lfloor \frac{mD}{I} \rfloor I\right) \end{aligned}$$

By substituting $I = 5$ and $D = 2$ we obtain

$$g(n, m) = h\left(5n + 2m - 5\lfloor \frac{2m}{5} \rfloor\right)$$

By evaluating $g(n, m)$ for $n = 0, 1, \dots, 5$ and $m = 0, 1, \dots, 4$ we obtain the following coefficients for the time-variant filter:

$$\begin{aligned} g(0, m) &= \{h(0) \quad h(2) \quad h(4) \quad h(1) \quad h(3)\} \\ g(1, m) &= \{h(5) \quad h(7) \quad h(9) \quad h(6) \quad h(8)\} \\ g(2, m) &= \{h(10) \quad h(12) \quad h(14) \quad h(11) \quad h(13)\} \\ g(3, m) &= \{h(15) \quad h(17) \quad h(19) \quad h(16) \quad h(18)\} \\ g(4, m) &= \{h(20) \quad h(22) \quad h(24) \quad h(21) \quad h(23)\} \\ g(0, m) &= \{h(25) \quad h(27) \quad h(29) \quad h(26) \quad h(28)\} \end{aligned}$$

In summary, sampling rate conversion by the factor I/D can be achieved by first increasing the sampling rate by I , accomplished by inserting $I - 1$ zeros between successive values of the input signal $x(n)$, followed by linear filtering of the resulting sequence to eliminate unwanted images of $X(\omega)$ and, finally, by downsampling the filtered signal by the factor D . When $F_y > F_x$, the lowpass filter acts as an anti-imaging postfilter that removes the spectral replicas at multiples of F_x , but not at multiples of IF_x . When $F_y < F_x$, the lowpass filter acts as an anti-aliasing prefilter that removes the down-shifted spectral replicas at multiples of F_y to avoid overlapping. The design of the lowpass filter can be accomplished by the filter design techniques described in Chapter 10.

11.5 Implementation of Sampling Rate Conversion

In this section we consider the efficient implementation of sampling rate conversion systems using polyphase filter structures. Additional computational simplifications can be obtained using the multistage approach described in Section 11.6.

11.5.1 Polyphase Filter Structures

Polyphase structures for FIR filters were developed for the efficient implementation of sampling rate converters; however, they can be used in other applications. The polyphase structure is based on the fact that any system function can be split as

$$\begin{aligned} H(z) = & \cdots + h(0) & + h(M)z^{-M} + \cdots \\ & \cdots + h(1)z^{-1} & + h(M+1)z^{-(M+1)} + \cdots \\ & & \vdots \\ & \cdots + h(M-1)z^{-(M-1)} & + h(2M-1)z^{-(2M-1)} + \cdots \end{aligned}$$

If we next factor out the term $z^{-(i-1)}$ at the i th row, we obtain

$$\begin{aligned} H(z) = & [\cdots + h(0) & + h(M)z^{-M} + \cdots] \\ & + z^{-1}[\cdots + h(1) & + h(M+1)z^{-M} + \cdots] \\ & & \vdots \\ & + z^{-(M-1)}[\cdots + h(M-1) & + h(2M-1)z^{-M} + \cdots] \end{aligned}$$

The last equation can be compactly expressed as

$$H(z) = \sum_{i=0}^{M-1} z^{-i} P_i(z^M) \quad (11.5.1)$$

where

$$P_i(z) = \sum_{n=-\infty}^{\infty} h(nM+i)z^{-n} \quad (11.5.2)$$

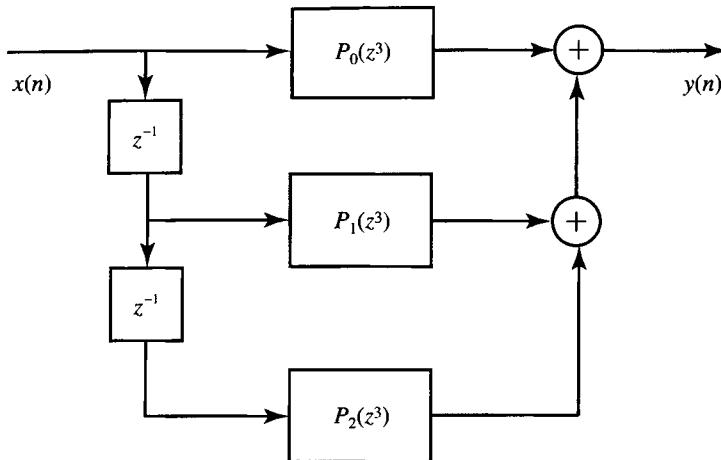


Figure 11.5.1 Block diagram of polyphase filter structure for $M = 3$.

Relation (11.5.1) is called the *M-component polyphase decomposition* and $P_i(z)$ the polyphase components of $H(z)$. Each subsequence

$$p_i(n) = h(nM + i), \quad i = 0, 1, \dots, M - 1 \quad (11.5.3)$$

is obtained by downsampling a delayed (“phase-shifted”) version of the original impulse response.

To develop an M -component polyphase filter structure, we use (11.5.1) for $M = 3$ to express the z -transform of the output sequence as

$$\begin{aligned} Y(z) &= H(z)X(z) \\ &= P_0(z^3)X(z) + z^{-1}P_1(z^3)X(z) + z^{-2}P_2(z^3)X(z) \end{aligned} \quad (11.5.4)$$

$$= P_0(z^3)X(z) + z^{-1}\{P_1(z^3)X(z) + z^{-1}[P_2(z^3)X(z)]\} \quad (11.5.5)$$

Equation (11.5.4) leads to the polyphase structure of Figure 11.5.1. Similarly, (11.5.5) yields the polyphase structure shown in Figure 11.5.2. This is known as transpose polyphase structure because it is similar to the transpose FIR filter realization. The obtained polyphase structures are valid for any filter, FIR or IIR, and any finite value of M and are sufficient for our needs. Additional structures and details can be found in Vaidyanathan (1993).

11.5.2 Interchange of Filters and Downsamplers/Upsamplers

In general, the order of a sampling rate converter (which is a linear time-varying system) and a linear time-invariant system *cannot* be interchanged. We next derive two identities, known as *noble identities*, that help to swap the position of a filter with a downampler or an upampler by properly modifying the filter.

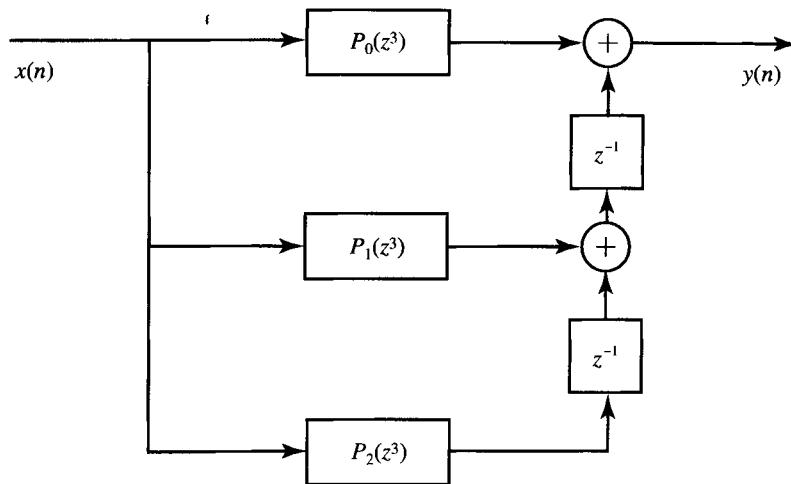


Figure 11.5.2 Illustration of transpose polyphase filter structure for $M = 3$.

To prove the first identity (see Figure 11.5.3), we recall that the input/output description of a downampler is

$$y(n) = x(nD) \xrightarrow{z} Y(z) = \frac{1}{D} \sum_{i=0}^{D-1} X(z^{1/D} W_D^i) \quad (11.5.6)$$

where $W_D = e^{-j2\pi/D}$. The output of the system in Figure 11.5.3(a) can be written as

$$Y(z) = \frac{1}{D} \sum_{i=0}^{D-1} V_1(z^{1/D} W_D^i) = \frac{1}{D} \sum_{i=0}^{D-1} H(z W_D^{iD}) X(z^{1/D} W_D^i) \quad (11.5.7)$$

because $V_1(z) = H(z^D)X(z)$. Taking into consideration $W_D^{iD} = 1$ and Figure 11.5.3(b), relation (11.5.7) results in

$$Y(z) = \frac{1}{D} H(z) \sum_{i=0}^{D-1} X(z^{1/D} W_D^i) = H(z) V_2(z) \quad (11.5.8)$$

which shows the equivalence of the two structures in Figure 11.5.3.

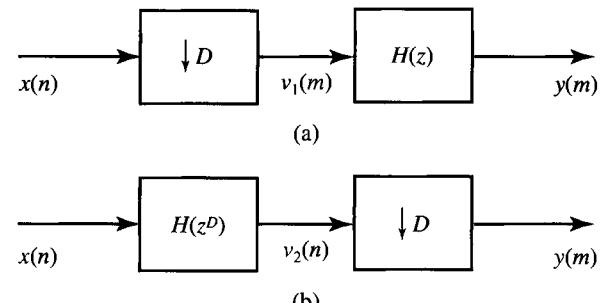


Figure 11.5.3
Two equivalent
downsampling systems (first
noble identity).

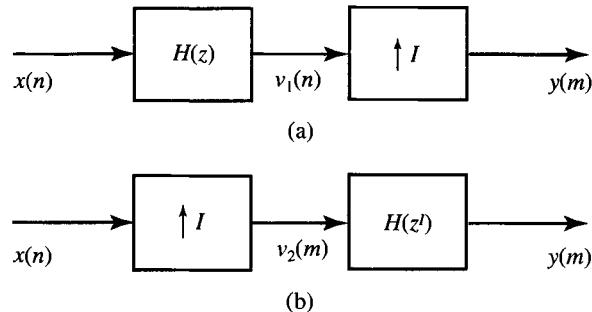


Figure 11.5.4
Two equivalent upsampling systems (second noble identity).

A similar identity can be shown to hold for upsampling. To start, we recall that the input/output description of an upsampler is

$$y(n) = \begin{cases} x\left(\frac{n}{I}\right), & n = 0, \pm I, \pm 2I, \dots \\ 0, & \text{otherwise} \end{cases} \xleftrightarrow{z} Y(z) = X(z^I) \quad (11.5.9)$$

The output of the system in Figure 11.5.4(a) can be written as

$$Y(z) = H(z^I)V_1(z) = H(z^I)X(z^I) \quad (11.5.10)$$

because $V_1(z) = X(z^I)$. The output of the system in Figure 11.5.4(b) is given by

$$Y(z) = V_2(z^I) = H(z^I)X(z^I) \quad (11.5.11)$$

which is equal to (11.5.10). This shows that the two systems in Figure 11.5.4 are identical.

In conclusion, we have shown that it is possible to interchange the operation of linear filtering and downsampling or upsampling if we properly modify the system function of the filter.

11.5.3 Sampling Rate Conversion with Cascaded Integrator Comb Filters

The hardware implementation of the lowpass filter required for sampling rate conversion can be significantly simplified if we choose a comb filter with system function (see Section 5.4.5)

$$H(z) = \sum_{k=0}^{M-1} z^{-k} = \frac{1 - z^{-M}}{1 - z^{-1}} \quad (11.5.12)$$

This system can be implemented by cascading either the “integrator” $1/(1-z^{-1})$ with the comb filter $(1-z^{-M})$ or vice versa. This leads to the name *cascaded integrator comb* (CIC) filter structure. The CIC structure does not require any multiplications or storage for the filter coefficients.

To obtain an efficient decimation structure, we start with an integrator-comb CIC filter followed by the downsampler and then we apply the first noble identity, as shown in Figure 11.5.5. For the interpolation case, we use an upsampler followed

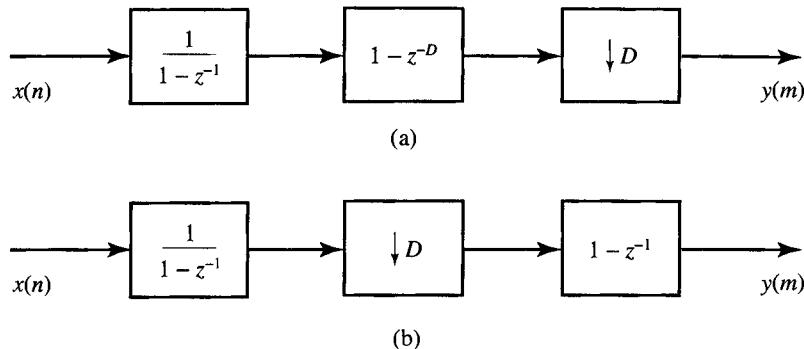


Figure 11.5.5 Using the first noble identity to obtain an efficient CIC filter structure for decimation.

by a comb-integrator CIC filter and then we use the second noble identity, as shown in Figure 11.5.6. To improve the lowpass frequency response required for sampling rate conversion, we can cascade K CIC filters. In this case, we order all integrators on one side of the filter and the comb filters on the other side, and then we apply the noble identities as in the single-stage case. The integrator $1/(1-z^{-1})$ is an unstable system. Therefore, its output may grow without limits, resulting in overflow when the integrator section comes first, as in the decimation structure shown in Figure 11.5.5(b). However, this overflow can be tolerated if the entire filter is implemented using two's-complement fixed-point arithmetic. If $D \neq M$ or $I \neq M$, the comb filter $1-z^{-1}$ in Figures 11.5.5(a) and 11.5.6(b) should be replaced by $1-z^{-M/D}$ or $1-z^{-M/I}$, respectively. A detailed treatment of CIC filters for decimation and interpolation can be found in Hogenauer (1981). Finally, we note that CIC filters are special cases of the frequency sampling structure discussed in Section 10.2.3.

If the CIC filter order is a power of 2, that is, $M = 2^K$, we can decompose the system function (11.5.12) as follows:

$$H(z) = (1+z^{-1})(1+z^{-2})(1+z^{-4}) \dots (1+z^{-2^{K-1}}) \quad (11.5.13)$$

Using this decomposition we can develop decimator structures using nonrecursive

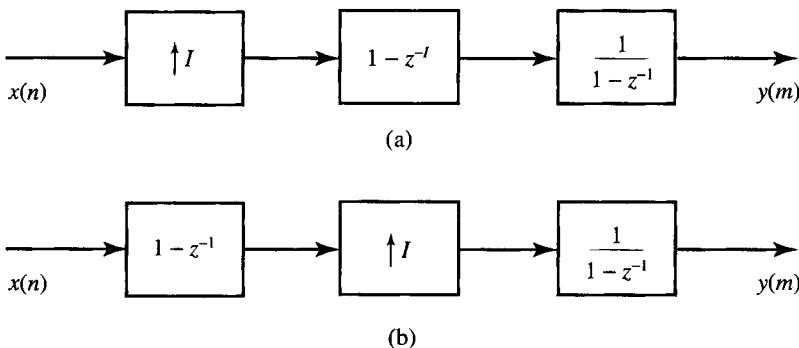


Figure 11.5.6 Using the second noble identity to obtain an efficient CIC filter structure for interpolation.

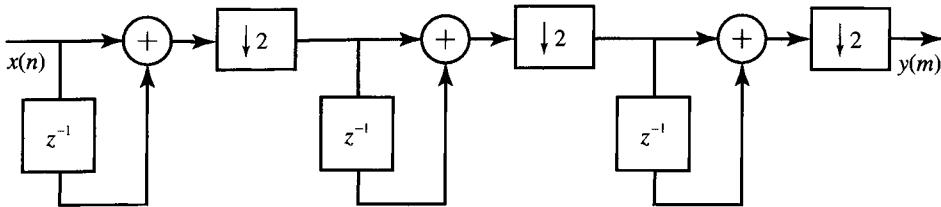


Figure 11.5.7 Efficient filter structure for decimation by $D = 8$ using comb filters.

CIC filters. Figure 11.5.7 shows an example for a decimator with $D = M = 8$. Cascading of N CIC filters can be obtained by providing M first-order sections ($1 - z^{-1}$) between each decimation stage. The constraint $M = 2^K$ can be relaxed by factoring M into a product of prime numbers, as shown in Jang and Yang (2001).

11.5.4 Polyphase Structures for Decimation and Interpolation Filters

To develop a polyphase structure for decimation, we start with the straightforward implementation of the decimation process shown in Figure 11.5.8. The decimated sequence is obtained by passing the input sequence $x(n)$ through a linear filter and then downsampling the filter output by a factor D . In this configuration, the filter is operating at the high sampling rate F_x , while only one out of every D output samples is actually needed. A logical solution would be to find a structure where only the needed samples are computed. We will develop such an efficient implementation by exploiting the polyphase structure in Figure 11.5.1. Since downsampling commutes with addition, combining the structures in Figures 11.5.8 and 11.5.1 yields the structure in Figure 11.5.9(a). If we next apply the identity in Figure 11.5.3, we obtain the desired implementation structure shown in Figure 11.5.9(b). In this filtering structure, only the needed samples are computed and all multiplications and additions are performed at the lower sampling rate F_x/D . Thus, we have achieved the desired efficiency. Additional reduction in computation can be achieved by using an FIR filter with linear phase and exploiting the symmetry of its impulse response.

In practice it is more convenient to implement the polyphase decimator using a *commutator model* as shown in Figure 11.5.10. The commutator rotates *counter-clockwise* starting at time $n = 0$ and distributes a block of D input samples to the polyphase filters starting at filter $i = D - 1$ and continuing in reverse order until $i = 0$. For every block of D input samples, the polyphase filters receive a new input and their outputs are computed and summed to produce one sample of the output signal $y(m)$. The operation of this realization can be also understood by a careful inspection of Figure 11.1.3.

Next, let us consider the efficient implementation of an interpolator, which is realized by first inserting $I - 1$ zeros between successive samples of $x(n)$ and then

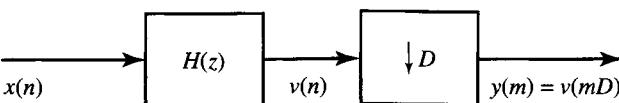


Figure 11.5.8 Decimation system.

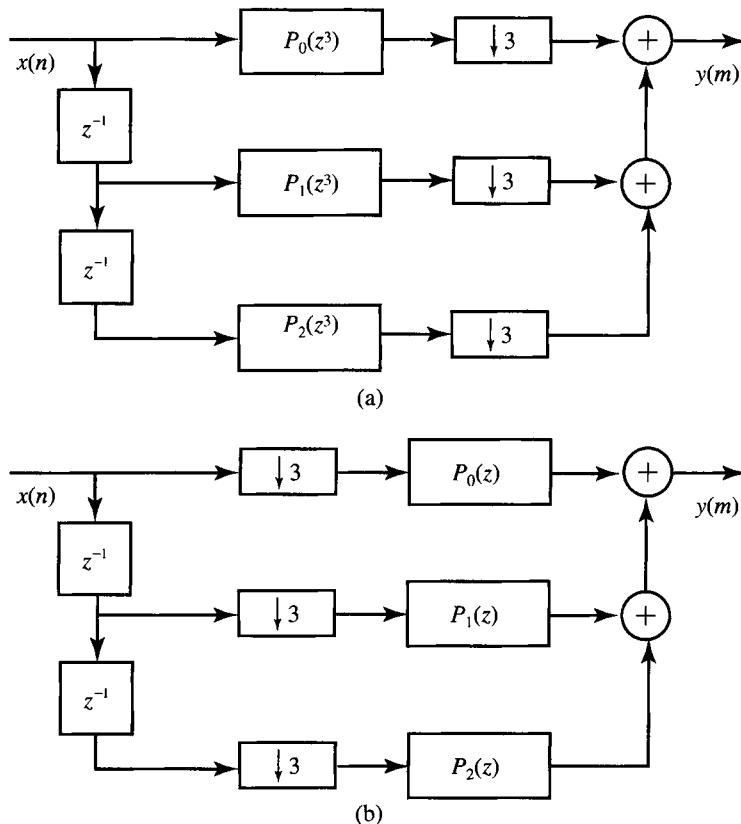


Figure 11.5.9 Implementation of a decimation system using a polyphase structure before (a) and after (b) the use of the first noble identity.

filtering the resulting sequence (see Figure 11.5.11). The major problem with this structure is that the filter computations are performed at the high sampling rate IF_x . The desired simplification is achieved by first replacing the filter in Figure 11.5.11 with the transpose polyphase structure in Figure 11.5.2, as illustrated in Figure 11.5.12(a).

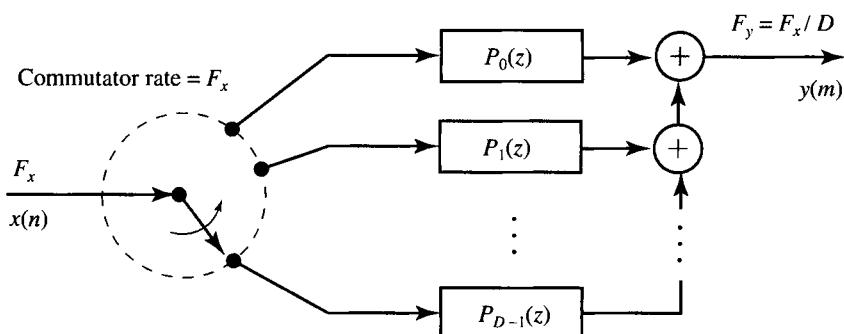


Figure 11.5.10 Decimation using a polyphase filter and a commutator.

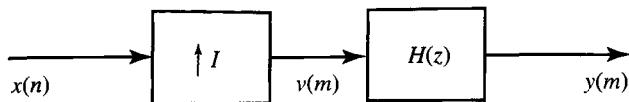
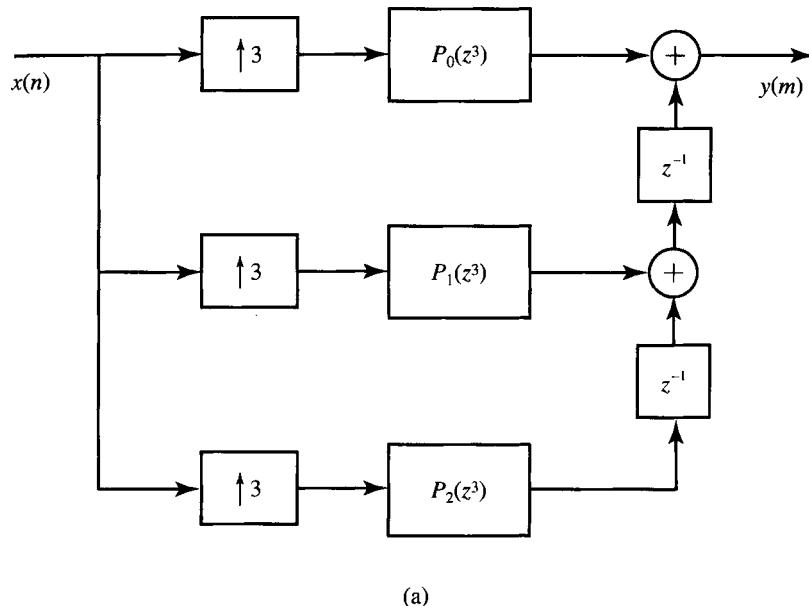
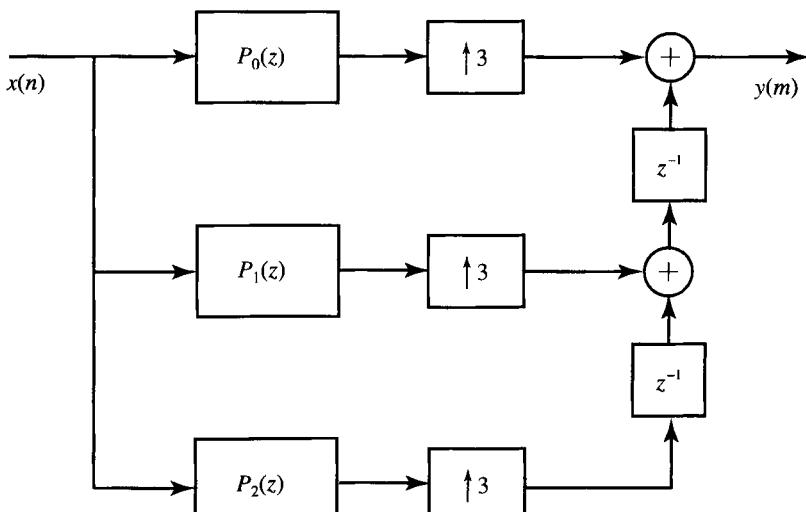


Figure 11.5.11 Interpolation system.



(a)



(b)

Figure 11.5.12 Implementation of an interpolation system using a polyphase structure before (a) and after (b) the use of the second noble identity.

Then, we use the second noble identity (see Figure 11.5.4) to obtain the structure in Figure 11.5.12(b). Thus, all the filtering multiplications are performed at the low rate F_x . It is interesting to note that the structure for an interpolator can be obtained by transposing the structure of a decimator, and vice versa (Cochiere and Rabiner, 1981).

For every input sample, the polyphase filters produce I output samples $y_0(n)$, $y_1(n)$, ..., $y_{I-1}(n)$. Because the output $y_i(n)$ of the i th filter is followed by $(I - 1)$ zeros and it is delayed by i th samples, the polyphase filters contribute nonzero samples at different time slots. In practice, we can implement the part of the structure including the 1-to- I expanders, delays, and adders using the commutator model shown in Figure 11.5.13. The commutator rotates *counterclockwise* starting at time $n = 0$ at branch $i = 0$. For each input sample $x(n)$ the commutator reads the output of every polyphase filter to obtain I samples of the output (interpolated) signal $y(m)$. The operation of this realization can be also understood by a careful inspection of Figure 11.1.4. Each polyphase filter in Figure 11.5.13 operates on the same input data using its unique set of coefficients. Therefore, we can obtain the same results using a single filter by sequentially loading a different set of coefficients.

11.5.5 Structures for Rational Sampling Rate Conversion

A sampling rate converter with a ratio I/D can be efficiently implemented using a polyphase interpolator followed by a downampler. However, since the downampler keeps only every D th polyphase subfilter output, it is not necessary to compute all I interpolated values between successive input samples. To determine which polyphase subfilter outputs to compute, we consider an example with $I = 5$ and $D = 3$. The interpolator polyphase structure has $I = 5$ subfilters which provide interpolated samples at an effective sampling period $T = T_x/I$. The downampler picks every D th of these samples, resulting in a discrete-time signal with sampling

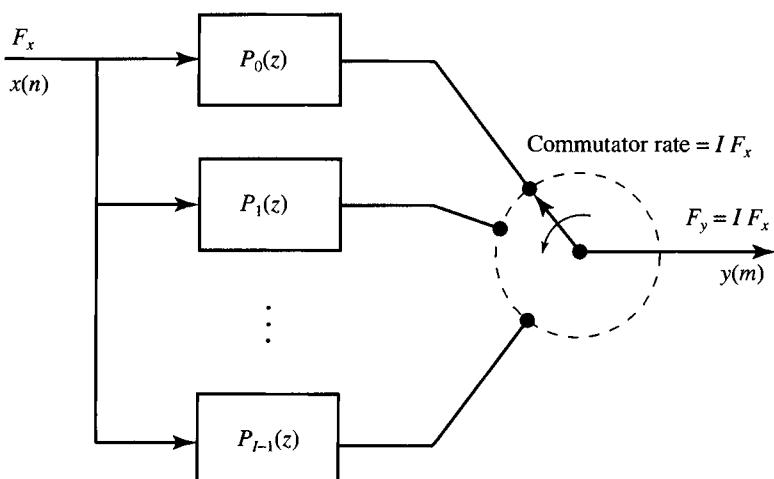


Figure 11.5.13 Interpolation using a polyphase filter and a commutator.

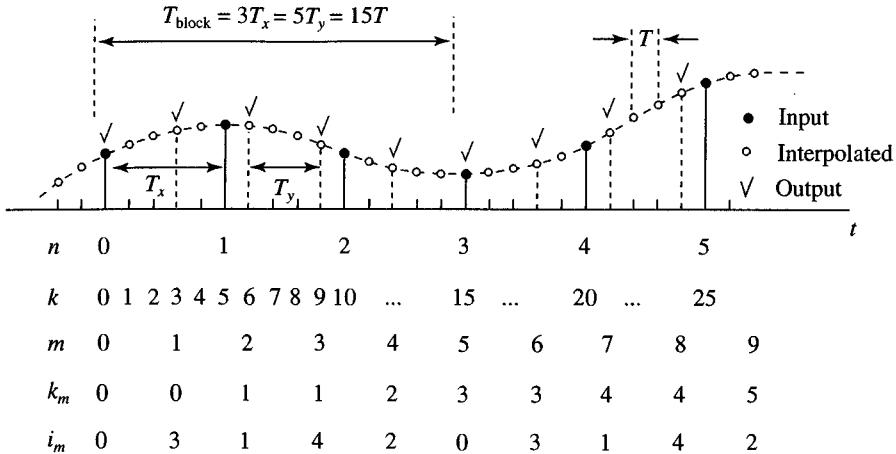


Figure 11.5.14 Illustration of index computation for polyphase implementation of sampling rate conversion for a rational ratio $I/D = 5/3$.

period $T_y = DT = DT_x/I$. It is convenient to think in terms of blocks of duration

$$T_{\text{block}} = IT_y = DT_x = IDT \quad (11.5.14)$$

which contain L output samples or I input samples. The relative time positions of the various sequences and a block of data are illustrated in Figure 11.5.14. The input sequence $x(nT_x)$ is interpolated to produce a sequence $v(kT)$, which is then decimated to yield $y(mT_y)$. If we use an FIR filter with $M = KI$ coefficients, the polyphase subfilters are given by $p_i(n) = h(nI + i)$, $i = 0, 1, \dots, I - 1$, where $n = 0, 1, \dots, K - 1$. To compute the output sample $y(m)$, we use the polyphase subfilter with index i_m which requires the input samples $x(k_m), x(k_m - 1), \dots, x(k_m - K + 1)$. From relation (11.1.9) and Figure 11.5.14, we can easily deduce that

$$k_m = \left\lfloor \frac{mD}{I} \right\rfloor \quad \text{and} \quad i_m = (Dm)_I \quad (11.5.15)$$

For $D = 3$ and $I = 5$ the first data block includes $D = 3$ input samples and $I = 5$ output samples. To compute the samples $\{y(0), y(1), y(2), y(3), y(4)\}$, we use the polyphase subfilter specified by the index $i_m = \{0, 3, 1, 4, 2\}$, respectively. The samples in the filter memory are updated only when k_m changes value. This discussion provides the basic ideas for the efficient software implementation of rational sampling rate conversion using FIR filters.

11.6 Multistage Implementation of Sampling Rate Conversion

In practical applications of sampling-rate conversion we often encounter decimation factors and interpolation factors that are much larger than unity. For example, suppose that we are given the task of altering the sampling rate by the factor

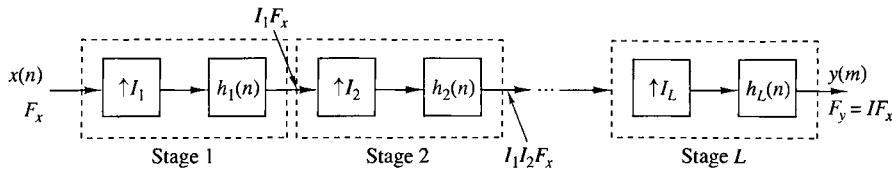


Figure 11.6.1 Multistage implementation of interpolation by a factor I .

$I/D = 130/63$. Although, in theory, this rate alteration can be achieved exactly, the implementation would require a bank of 130 polyphase filters and may be computationally inefficient. In this section we consider methods for performing sampling rate conversion for either $D \gg 1$ and/or $I \gg 1$ in multiple stages.

First, let us consider interpolation by a factor $I \gg 1$ and let us assume that I can be factored into a product of positive integers as

$$I = \prod_{i=1}^L I_i \quad (11.6.1)$$

Then, interpolation by a factor I can be accomplished by cascading L stages of interpolation and filtering, as shown in Fig. 11.6.1. Note that the filter in each of the interpolators eliminates the images introduced by the upsampling process in the corresponding interpolator.

In a similar manner, decimation by a factor D , where D may be factored into a product of positive integers as

$$D = \prod_{i=1}^J D_i \quad (11.6.2)$$

can be implemented as a cascade of J stages of filtering and decimation as illustrated in Fig. 11.6.2. Thus the sampling rate at the output of the i th stage is

$$F_i = \frac{F_{i-1}}{D_i}, \quad i = 1, 2, \dots, J \quad (11.6.3)$$

where the input rate for the sequence $\{x(n)\}$ is $F_0 = F_x$.

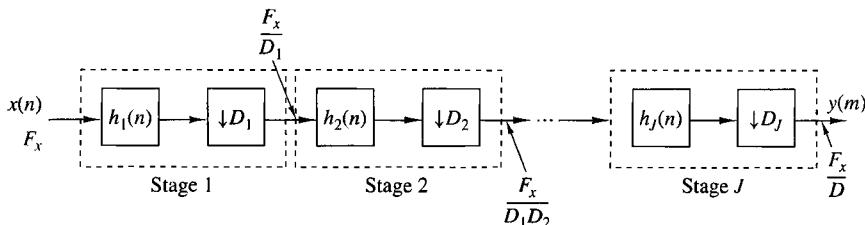


Figure 11.6.2 Multistage implementation of decimation by a factor D .

To ensure that no aliasing occurs in the overall decimation process, we can design each filter stage to avoid aliasing within the frequency band of interest. To elaborate, let us define the desired passband and the transition band in the overall decimator as

$$\begin{aligned} \text{Passband: } 0 &\leq F \leq F_{pc} \\ \text{Transition band: } F_{pc} &\leq F \leq F_{sc} \end{aligned} \quad (11.6.4)$$

where $F_{sc} \leq F_x/2D$. Then, aliasing in the band $0 \leq F \leq F_{sc}$ is avoided by selecting the frequency bands of each filter stage as follows:

$$\begin{aligned} \text{Passband: } 0 &\leq F \leq F_{pc} \\ \text{Transition band: } F_{pc} &\leq F \leq F_i - F_{sc} \\ \text{Stopband: } F_i - F_{sc} &\leq F \leq \frac{F_{i-1}}{2} \end{aligned} \quad (11.6.5)$$

For example, in the first filter stage we have $F_1 = F_x/D_1$, and the filter is designed to have the following frequency bands:

$$\begin{aligned} \text{Passband: } 0 &\leq F \leq F_{pc} \\ \text{Transition band: } F_{pc} &\leq F \leq F_1 - F_{sc} \\ \text{Stopband: } F_1 - F_{sc} &\leq F \leq \frac{F_0}{2} \end{aligned} \quad (11.6.6)$$

After decimation by D_1 , there is aliasing from the signal components that fall in the filter transition band, but the aliasing occurs at frequencies above F_{sc} . Thus there is no aliasing in the frequency band $0 \leq F \leq F_{sc}$. By designing the filters in the subsequent stages to satisfy the specifications given in (11.6.5), we ensure that no aliasing occurs in the primary frequency band $0 \leq F \leq F_{sc}$.

EXAMPLE 11.6.1

Consider an audio-band signal with a nominal bandwidth of 4 kHz that has been sampled at a rate of 8 kHz. Suppose that we wish to isolate the frequency components below 80 Hz with a filter that has a passband $0 \leq F \leq 75$ and a transition band $75 \leq F \leq 80$. Hence $F_{pc} = 75$ Hz and $F_{sc} = 80$. The signal in the band $0 \leq F \leq 80$ may be decimated by the factor $D = F_x/2F_{sc} = 50$. We also specify that the filter have a passband ripple $\delta_1 = 10^{-2}$ and a stopband ripple of $\delta_2 = 10^{-4}$.

The length of the linear phase FIR filter required to satisfy these specifications can be estimated from one of the well-known formulas given in the Section 10.2.7. Recall that a particularly simple formula for approximating the length M , attributed to Kaiser, is

$$\hat{M} = \frac{-10 \log_{10} \delta_1 \delta_2 - 13}{14.6 \Delta f} + 1 \quad (11.6.7)$$

where Δf is the normalized (by the sampling rate) width of the transition region [i.e., $\Delta f = (F_{sc} - F_{pc})/F_s$]. A more accurate formula proposed by Herrmann et al. (1973) is

$$\hat{M} = \frac{D_\infty(\delta_1, \delta_2) - f(\delta_1, \delta_2)(\Delta f)^2}{\Delta f} + 1 \quad (11.6.8)$$

where $D_\infty(\delta_1, \delta_2)$ and $f(\delta_1, \delta_2)$ are defined as

$$\begin{aligned} D_\infty(\delta_1, \delta_2) &= [0.005309(\log_{10} \delta_1)^2 + 0.07114(\log_{10} \delta_1) \\ &\quad - 0.4761]\log_{10} \delta_2 \end{aligned} \quad (11.6.9)$$

$$- [0.00266(\log_{10} \delta_1)^2 + 0.5941 \log_{10} \delta_1 + 0.4278] \quad (11.6.9)$$

$$f(\delta_1, \delta_2) = 11.012 + 0.51244[\log_{10} \delta_1 - \log_{10} \delta_2] \quad (11.6.10)$$

Now a single FIR filter followed by a decimator would require (using the Kaiser formula) a filter of (approximate) length

$$\hat{M} = \frac{-10 \log_{10} 10^{-6} - 13}{14.6(5/8000)} + 1 \approx 5152$$

As an alternative, let us consider a two-stage decimation process with $D_1 = 25$ and $D_2 = 2$. In the first stage we have the specifications $F_1 = 320$ Hz and

$$\text{Passband: } 0 \leq F \leq 75$$

$$\text{Transition band: } 75 < F \leq 240$$

$$\Delta f = \frac{165}{8000}$$

$$\delta_{11} = \frac{\delta_1}{2}, \quad \delta_{21} = \delta_2$$

Note that we have reduced the passband ripple δ_1 by a factor of 2, so that the total passband ripple in the cascade of the two filters does not exceed δ_1 . On the other hand, the stopband ripple is maintained at δ_2 in both stages. Now the Kaiser formula yields an estimate of M_1 as

$$\hat{M}_1 = \frac{-10 \log_{10} \delta_{11} \delta_{21} - 13}{14.6 \Delta f} + 1 \approx 167$$

For the second stage, we have $F_2 = F_1/2 = 160$ and the specifications

$$\text{Passband: } 0 \leq F \leq 75$$

$$\text{Transition band: } 75 < F \leq 80$$

$$\Delta f = \frac{5}{320}$$

$$\delta_{12} = \frac{\delta_1}{2}, \quad \delta_{22} = \delta_2$$

Hence the estimate of the length M_2 of the second filter is

$$\hat{M}_2 \approx 220$$

Therefore, the total length of the two FIR filters is approximately $\hat{M}_1 + \hat{M}_2 = 387$. This represents a reduction in the filter length by a factor of more than 13.

The reader is encouraged to repeat the computation above with $D_1 = 10$ and $D_2 = 5$.

It is apparent from the computations in Example 11.6.1 that the reduction in the filter length results from increasing the factor Δf , which appears in the denominator in (11.6.7) and (11.6.8). By decimating in multiple stages, we are able to increase the width of the transition region through a reduction in the sampling rate.

In the case of a multistage interpolator, the sampling rate at the output of the i th stage is

$$F_{i-1} = I_i F_i, \quad i = J, J-1, \dots, 1$$

and the output rate is $F_0 = I F_J$ when the input sampling rate is F_J . The corresponding frequency band specifications are

$$\text{Passband: } 0 \leq F \leq F_p$$

$$\text{Transition band: } F_p < F \leq F_i - F_{\text{sc}}$$

The following example illustrates the advantages of multistage interpolation.

EXAMPLE 11.6.2

Let us reverse the filtering problem described in Example 11.6.1 by beginning with a signal having a passband $0 \leq F \leq 75$ and a transition band of $75 \leq F \leq 80$. We wish to interpolate by a factor of 50. By selecting $I_1 = 2$ and $I_2 = 25$, we have basically a transposed form of the decimation problem considered in Example 11.6.1. Thus we can simply transpose the two-stage decimator to achieve the two-stage interpolator with $I_1 = 2$, $I_2 = 25$, $\hat{M}_1 \approx 220$, and $\hat{M}_2 \approx 167$.

11.7 Sampling Rate Conversion of Bandpass Signals

In this section we consider the decimation and interpolation of bandpass signals. We begin by noting that any bandpass signal can be converted into an equivalent lowpass signal (see Section 6.5.2) whose sampling rate can be changed using the already developed techniques. However, a simpler and more widely used approach concerns bandpass discrete-time signals with *integer-band positioning*. The concept is similar to the one discussed for continuous-time bandpass signals in Section 6.4.

To be specific, suppose that we wish to decimate by a factor D an integer-positioned bandpass signal with spectrum confined to the bands

$$(k-1)\frac{\pi}{D} < |\omega| < k\frac{\pi}{D} \quad (11.7.1)$$

where k is a positive integer. A bandpass filter defined by

$$H_{\text{BP}}(\omega) = \begin{cases} 1, & (k-1)\frac{\pi}{D} < |\omega| < k\frac{\pi}{D} \\ 0, & \text{otherwise} \end{cases} \quad (11.7.2)$$

would normally be used to eliminate signal frequency components outside the desired frequency range. Then direct decimation of the filtered signal $v(n)$ by the factor D

results in a periodic replication of the bandpass spectrum $V(\omega)$ every $2\pi/D$ radians according to (11.2.14). The spectrum of the decimated signal $y(m)$ is obtained by scaling the frequency axis by $\omega_y = D\omega_x$. This process is illustrated in Figure 11.7.1 for bandpass signal with odd band positioning ($k = 3$) and in Figure 11.7.2 for signals with even band positioning ($k = 4$). In the case where k is odd, there is an inversion of the spectrum of the signal as in the continuous-time case (see Figure 6.4.1(b)). The inversion can be undone by the simple process $y'(m) = (-1)^m y(m)$. Note that violation of the bandwidth constraint given by (11.7.1) results in signal aliasing.

The process of bandpass interpolation by an integer factor I is the inverse of that of bandpass decimation and can be accomplished in a similar manner. The process of upsampling by inserting zeros between the samples of $x(n)$ produces I images in the band $0 \leq \omega \leq \pi$. The desired image can be selected by bandpass filtering. This can be seen by “reversing” the process shown in Figure 11.7.1. Note that the process of interpolation also provides us with the opportunity to achieve frequency translation of the spectrum.

Finally, sampling rate conversion for a bandpass signal by a rational factor I/D can be accomplished by cascading a decimator with an interpolator in a manner that depends on the choice of the parameters D and I . A bandpass filter preceding the sampling converter is usually required to isolate the signal frequency band of

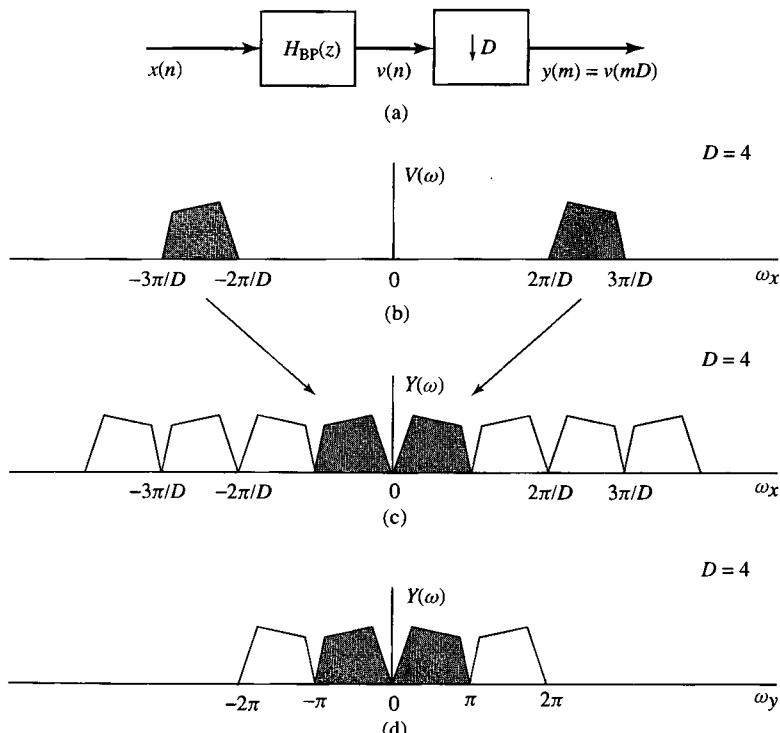


Figure 11.7.1 Spectral interpretation of bandpass decimation for integer-band positioning (odd integer positioning).

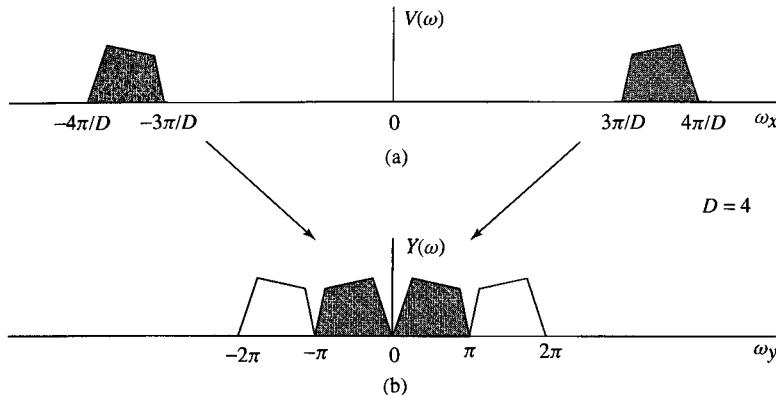


Figure 11.7.2 Spectral interpretation of bandpass decimation for integer-band positioning (even integer positioning).

interest. Note that this approach provides us with a modulation-free method for achieving frequency translation of a signal by selecting $D = I$.

11.8 Sampling Rate Conversion by an Arbitrary Factor

Efficient implementation of sampling rate conversion by a polyphase structure requires that the rates F_x and F_y are fixed and related by a rational factor I/D . In some applications, it is either inefficient or sometimes impossible to use such an exact rate conversion scheme.

For example, suppose 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 implementation is obviously inefficient in memory usage because we need to store a large number of filter coefficients.

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 converter in this case.

In principle, we can convert from any rate F_x to any rate F_y (fixed or variable) using formula (11.1.9), which we repeat here for convenience:

$$y(mT_y) = \sum_{k=K_1}^{K_2} g(kT_x + \Delta_m T_x) x((k_m - k)T_x) \quad (11.8.1)$$

This requires the computation of a new impulse response $p_m(k) = g(kT_x + \Delta_m T_x)$ for each output sample. However, if Δ_m is measured with finite accuracy, there is only a finite set of impulse responses, which may be precomputed and loaded from memory

as needed. We next discuss two practical approaches for sampling rate conversion by an arbitrary factor.

11.8.1 Arbitrary Resampling with Polyphase Interpolators

If we use a polyphase interpolator with I subfilters we can generate samples with spacing T_x/I . Therefore, the number I of stages determines the granularity of the interpolating process. If T_x/I is sufficiently small so that successive values of the signal do not change significantly or the change is less than the quantization step, we can determine the value at any location $t = nT_x + \Delta T_x$, $0 \leq \Delta \leq 1$, using the value of the nearest neighbor (zero-order hold interpolation).

Additional improvement can be obtained using two-point linear interpolation

$$y(nT_x + \Delta T_x) = (1 - \Delta)x(n) + \Delta x(n + 1) \quad (11.8.2)$$

The performance of these interpolation techniques has been discussed in Section 6.5 by analyzing their frequency-domain characteristics. Additional practical details can be found in Ramstad (1984).

11.8.2 Arbitrary Resampling with Farrow Filter Structures

In practice, we typically implement polyphase sampling rate converters by a rational factor using causal FIR lowpass filters. If we use an FIR filter with $M = KI$ coefficients, the coefficients of the polyphase filters are obtained by the mapping

$$p_i(n) = h(nI + i), \quad i = 0, 1, \dots, I - 1 \quad (11.8.3)$$

This mapping can be easily visualized as mapping the one-dimensional sequence $h(n)$ into a two-dimensional array with I rows and K columns by filling successive columns in natural order as follows:

$$\begin{array}{lllll} p_0(k) & \mapsto h(0) & h(I) & \dots & h((K-1)I) \\ p_1(k) & \mapsto h(1) & h(I+1) & \dots & h((K-1)I+1) \\ & \vdots & & & \\ p_i(k) & \mapsto h(i) & h(I+i) & \dots & h((K-1)I+i) \\ p_{i+1}(k) & \mapsto h(i+1) & h(I+i+1) & \dots & \\ & \vdots & & & \\ p_{I-1}(k) & \mapsto h(I-1) & h(2I-1) & \dots & h(KI-1) \end{array} \quad (11.8.4)$$

The polyphase filters $p_i(n)$ are used to compute samples at I equidistant locations $t = nT_x + i(T_x/I)$, $i = 0, 1, \dots, I - 1$ covering each input sampling interval. Suppose now that we wish to compute a sample at $t = nT_x + \Delta T_x$, where $\Delta \neq i/I$ and $0 \leq \Delta \leq 1$. This requires a nonexistent polyphase subfilter, denoted as $p_\Delta(k)$, which would “fall” between two existing subfilters, say, $p_i(k)$ and $p_{i+1}(k)$. This set of coefficients would create a row between the rows with indexes i and $i + 1$. We note that each column of (11.8.4) consists of a segment of I consecutive samples of

the impulse response $h(n)$ and covers one sampling interval T_x . Suppose next that we can approximate the coefficient set in each column by an L -degree polynomial

$$B_k(\Delta) = \sum_{\ell=0}^L b_\ell^{(k)} \Delta^\ell, \quad k = 0, 1, \dots, K-1 \quad (11.8.5)$$

We note that evaluating (11.8.5) at $\Delta = i/I$ will provide the coefficients of $p_i(k)$ polyphase subfilter. The type of the polynomial (Lagrange, Chebyshev, etc.) and the order L can be chosen to avoid any performance degradation compared to the original filter $h(n)$. The sample at location $t = nT_x + \Delta T_x$ is determined by

$$y((n + \Delta)T_x) = \sum_{k=0}^{K-1} B_k(\Delta) x((n - k)T_x), \quad 0 \leq \Delta \leq 1 \quad (11.8.6)$$

where the required filter coefficients are computed using (11.8.5). If we substitute the polynomials (11.8.5) into the filtering formula (11.8.6) and we change the order of summations, we obtain

$$\begin{aligned} y((n + \Delta)T_x) &= \sum_{k=0}^{K-1} \sum_{\ell=0}^L b_\ell^{(k)} \Delta^\ell x((n - k)T_x) \\ &= \sum_{\ell=0}^L \Delta^\ell \sum_{k=0}^{K-1} b_\ell^{(k)} x((n - k)T_x) \end{aligned}$$

The last equation can be written as

$$y((n + \Delta)T_x) = \sum_{\ell=0}^L v(\ell) \Delta^\ell \quad (11.8.7)$$

where

$$v(\ell) = \sum_{k=0}^{K-1} b_\ell^{(k)} x((n - k)T_x), \quad \ell = 0, 1, \dots, L \quad (11.8.8)$$

Equation (11.8.7) can be interpreted as a Taylor series representation of the output sequence, where the terms $v(\ell)$ are successive local derivatives determined from the input sequence. Relation (11.8.8) can be implemented using FIR filtering structures with system functions

$$H_\ell(z) = \sum_{k=0}^{K-1} b_\ell^{(k)} z^{-k} \quad (11.8.9)$$

The most efficient computation of polynomial (11.8.7) can be done using the nested Horner's rule, which is illustrated next for $L = 4$:

$$\begin{aligned} y(\Delta) &= c_0 + c_1 \Delta + c_2 \Delta^2 + c_3 \Delta^3 + c_4 \Delta^4 \\ &= c_0 + \Delta(c_1 + \Delta(c_2 + \Delta(c_3 + \Delta c_4))) \end{aligned} \quad (11.8.10)$$

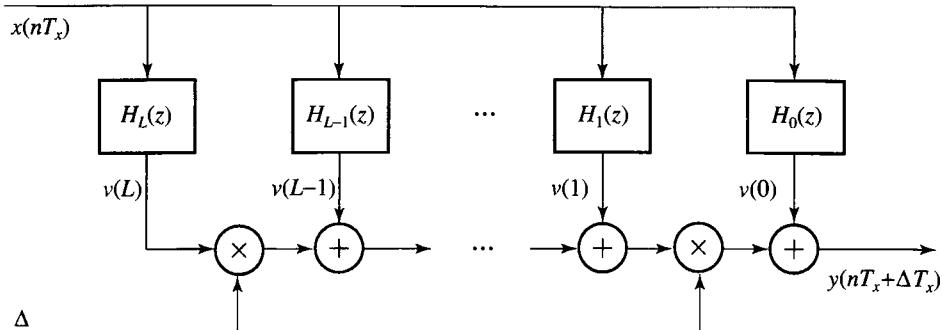


Figure 11.8.1 Block diagram of the Farrow structure for sampling rate change by an arbitrary factor.

This approach leads to the block diagram realization shown in Figure 11.8.1, which is known as Farrow structure (Farrow 1988). Basically, the Farrow structure performs interpolation between signal values by interpolating between filter coefficients. More details can be found in Gardner (1993), Erup et al. (1993), Ramstad (1984), Harris (1997), and Laakso et al. (1996).

11.9 Applications of Multirate Signal Processing

There are numerous practical applications of multirate signal processing. In this section we describe a few of these applications.

11.9.1 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

$$\Theta(\omega) = -\frac{k\omega}{I} \quad (11.9.1)$$

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. 11.9.1. 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 $(k/I)T_x$.

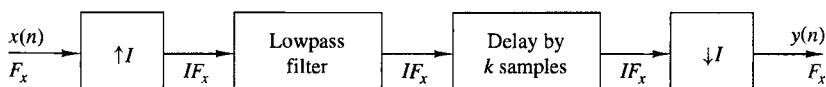


Figure 11.9.1 Method for generating a delay in a discrete-time signal.

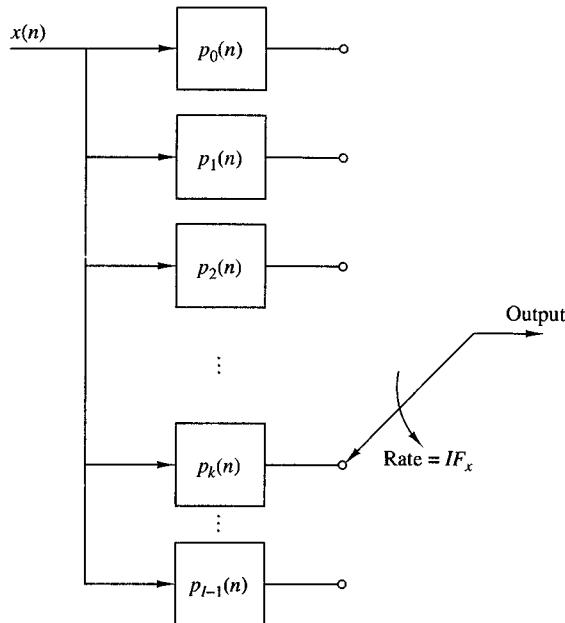


Figure 11.9.2
Polyphase filter structure
for implementing the
system shown in Fig. 11.9.1.

An efficient implementation of the interpolator is the polyphase filter illustrated in Fig. 11.9.2. The delay of k samples is achieved by placing the initial position of the commutator at the output of the k th subfilter. Since decimation by $D = I$ means that we take one out of every I samples from the polyphase filter, the commutator position can be fixed to the output of the k th subfilter. Thus a delay in k/I can be achieved by using only the k th subfilter of the polyphase filter. We note that the polyphase filter introduces an additional delay of $(M - 1)/2$ samples, where M is the length of its impulse response.

Finally, we mention that if the desired delay is a nonrational factor of the sample interval T_x , the methods described in Section 11.8 can be used to obtain the delay.

11.9.2 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. 11.9.3. 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.

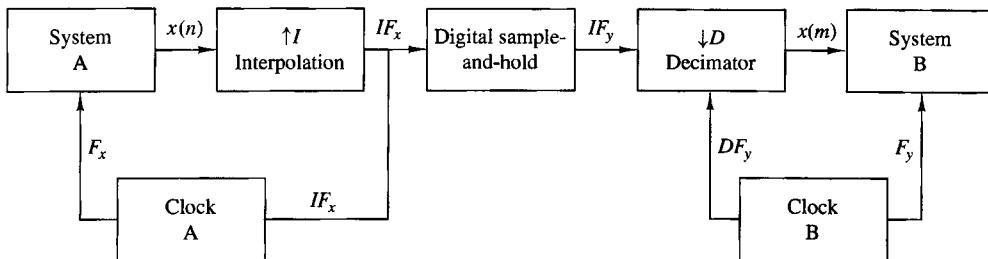


Figure 11.9.3 Interfacing of two digital systems with different sampling rates.

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 we can further reduce the distortion and thus reduce the size of the interpolator.

11.9.3 Implementation of Narrowband Lowpass Filters

In Section 11.6 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 $I = D$.

We demonstrate the procedure by means of an example for the design of a low-pass filter which has the same specifications as the filter that is given in Example 11.6.1.

EXAMPLE 11.9.1

Design a linear-phase FIR filter that satisfies the following specifications:

Sampling frequency:	8000 Hz
Passband:	$0 \leq F \leq 75$ Hz
Transition band:	$75 \text{ Hz} \leq F \leq 80 \text{ Hz}$
Stopband:	$80 \text{ Hz} \leq F \leq 4000 \text{ Hz}$
Passband ripple:	$\delta_1 = 10^{-2}$
Stopband ripple:	$\delta_2 = 10^{-4}$

Solution. If this filter were designed as a single-rate linear-phase FIR filter, the length of the filter required to meet the specifications is (from Kaiser's formula)

$$\hat{M} \approx 5152$$

Now, suppose that we employ a multirate implementation of the lowpass filter based on a decimation and interpolation factor of $D = I = 100$. A single-stage implementation of the decimator-interpolator requires an FIR filter of length

$$\hat{M}_1 = \frac{-10 \log_{10}(\delta_1 \delta_2 / 2) - 13}{14.6 \Delta f} + 1 \approx 5480$$

However, there is a significant savings in computational complexity by implementing the decimator and interpolator filters using their corresponding polyphase filters. If we employ linear-phase (symmetric) decimation and interpolation filters, the use of polyphase filters reduces the multiplication rate by a factor of 100.

A significantly more efficient implementation is obtained by using two stages of decimation followed by two stages of interpolation. For example, suppose that we select $D_1 = 50$, $D_2 = 2$, $I_1 = 2$, and $I_2 = 50$. Then the required filter lengths are

$$\hat{M}_1 = \frac{-10 \log_{10}(\delta_1 \delta_2 / 4) - 13}{14.6 \Delta f} + 1 \approx 177$$

$$\hat{M}_2 = \frac{-10 \log_{10}(\delta_1 \delta_2 / 4) - 13}{14.6 \Delta f} + 1 \approx 233$$

Thus we obtain a reduction in the overall filter length of $2(5480)/2(177 + 233) \approx 13.36$. In addition, we obtain further reduction in the multiplication rate by using polyphase filters. For the first stage of decimation, the reduction in multiplication rate is 50, while for the second stage the reduction in multiplication rate is 100. Further reductions can be obtained by increasing the number of stages of decimation and interpolation.

11.9.4 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. 11.9.4(a). 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. 11.9.4(b).

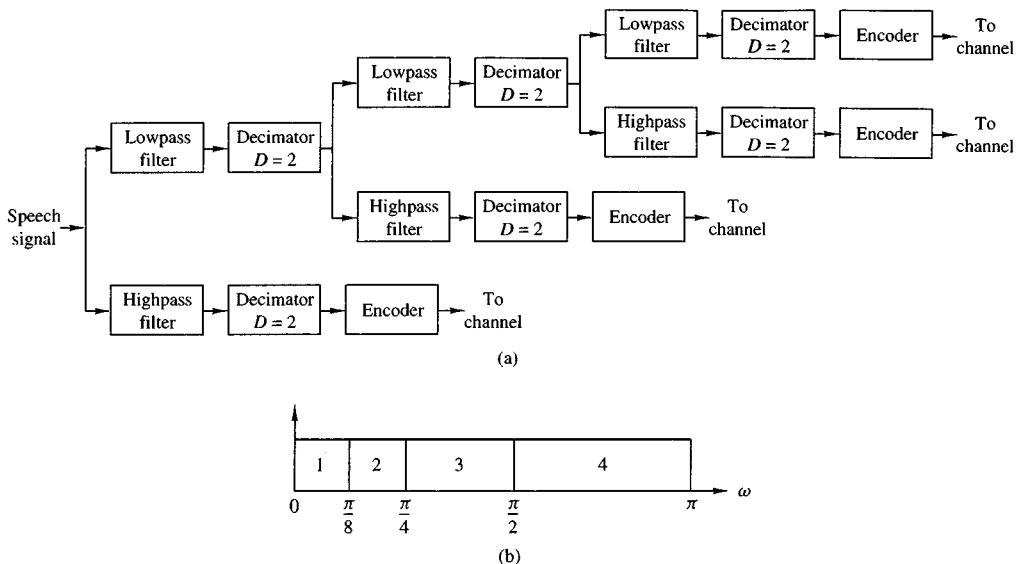


Figure 11.9.4 Block diagram of a subband speech coder.

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.

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. 11.9.5(a), 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. 11.9.5(b). These filters are described in Section 11.11.

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. 11.9.6. 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.

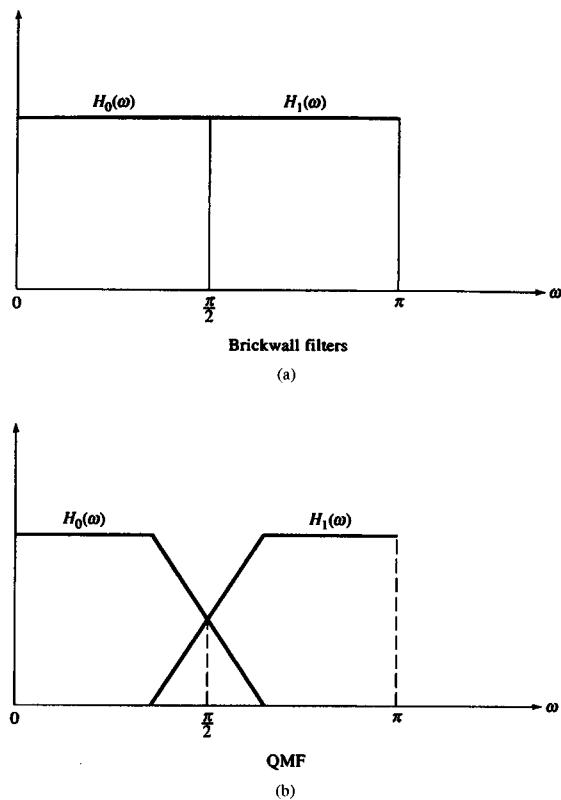


Figure 11.9.5
Filter characteristics for
subband coding.

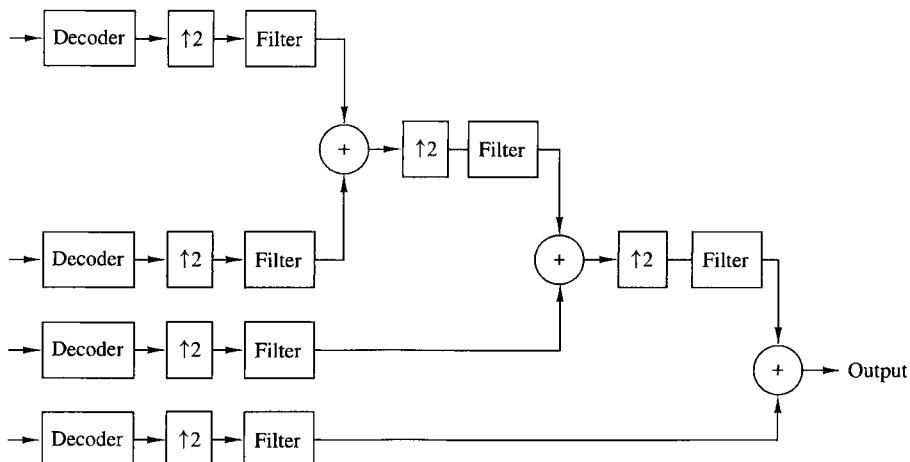


Figure 11.9.6 Synthesis of subband-encoded signals.

11.10 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 $\{H_k(z)\}$, arranged in a parallel bank as illustrated in Fig. 11.10.1(a). The frequency response characteristics of this filter bank split 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. 11.10.1(b), 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 transform (DFT) of a sequence $\{x(n)\}$, the filter bank is called a DFT filter bank. An analysis filter bank consisting of N filters $\{H_k(z), k = 0, 1, \dots, N - 1\}$ is called a *uniform DFT filter bank* if $H_k(z)$, $k = 1, 2, \dots, N - 1$, are derived from a prototype filter

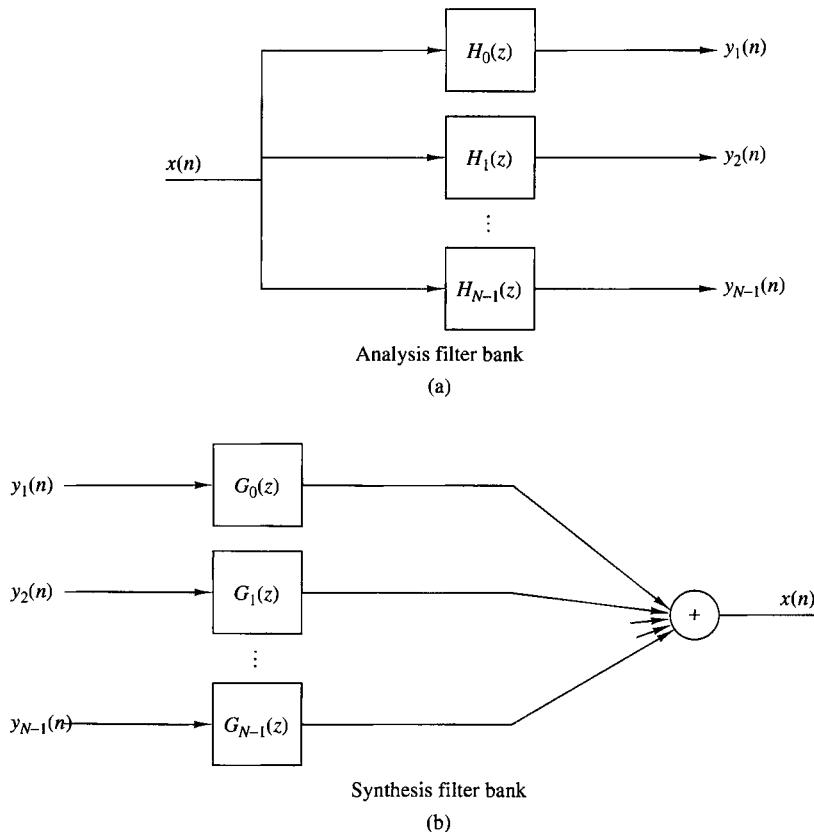


Figure 11.10.1 A digital filter bank.

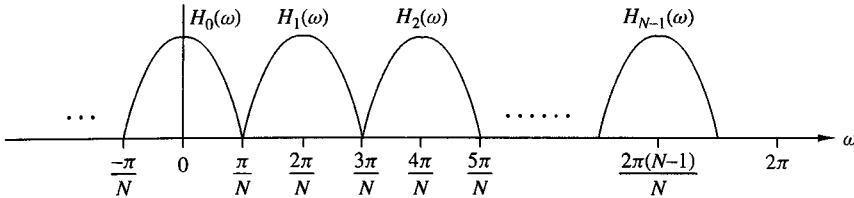


Figure 11.10.2 Illustration of frequency response characteristics of the N filters.

$H_0(z)$, where

$$H_k(\omega) = H_0\left(\omega - \frac{2\pi k}{N}\right), \quad k = 1, 2, \dots, N-1 \quad (11.10.1)$$

Hence the frequency response characteristics of the filters $\{H_k(z), k = 0, 1, \dots, N-1\}$ 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

$$h_k(n) = h_0(n)e^{j2\pi nk/N}, \quad k = 0, 1, \dots, N-1 \quad (11.10.2)$$

where $h_0(n)$ is the impulse response of the prototype filter, which in general may be either an FIR or an IIR filter. If $H_0(z)$ denotes the transfer function of the prototype filter, the transfer function of the k th filter is

$$H_k(z) = H_0(z e^{-j2\pi k/N}), \quad 1 \leq k \leq N-1 \quad (11.10.3)$$

Figure 11.10.2 provides a conceptual illustration of the frequency response characteristics of the N filters.

The uniform DFT analysis filter bank can be realized as shown in Fig. 11.10.3(a), where the frequency components in the sequence $\{x(n)\}$ are translated in frequency to lowpass by multiplying $x(n)$ with the complex exponentials $\exp(-j2\pi nk/N)$, $k = 1, \dots, N-1$, 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

$$X_k(m) = \sum_n h_0(mD - n)x(n)e^{-j2\pi nk/N}, \quad \begin{matrix} k = 0, 1, \dots, N-1 \\ m = 0, 1, \dots \end{matrix} \quad (11.10.4)$$

where $\{X_k(m)\}$ are samples of the DFT at frequencies $\omega_k = 2\pi k/N$.

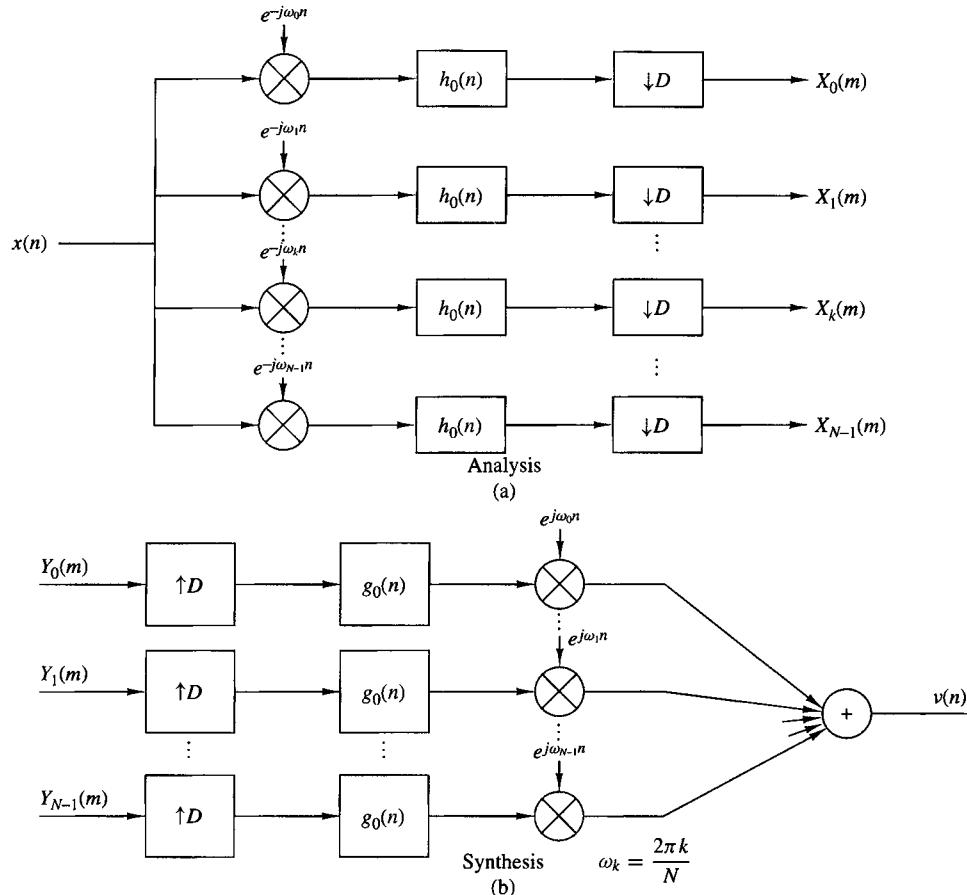


Figure 11.10.3 A uniform DFT filter bank.

The corresponding synthesis filter for each element in the filter bank can be viewed as shown in Fig. 11.10.3(b), where the input signal sequences $\{Y_k(m), k = 0, 1, \dots, N - 1\}$ are upsampled by a factor of $I = D$, filtered to remove the images, and translated in frequency by multiplication by the complex exponentials $\{\exp(j2\pi nk/N), k = 0, 1, \dots, N - 1\}$. The resulting frequency-translated signals from the N filters are then summed. Thus we obtain the sequence

$$\begin{aligned}
 v(n) &= \frac{1}{N} \sum_{k=0}^{N-1} e^{j2\pi nk/N} \left[\sum_m Y_k(m) g_0(n - mI) \right] \\
 &= \sum_m g_0(n - mI) \left[\frac{1}{N} \sum_{k=0}^{N-1} Y_k(m) e^{j2\pi nk/N} \right] \\
 &= \sum_m g_0(n - mI) y_n(m)
 \end{aligned} \tag{11.10.5}$$

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 $I = D$.

The relationship between the output $\{X_k(n)\}$ of the analysis filter bank and the input $\{Y_k(m)\}$ to the synthesis filter bank depends on the application. Usually, $\{Y_k(m)\}$ is a modified version of $\{X_k(m)\}$, where the specific modification is determined by the application.

An alternative realization of the analysis and synthesis filter banks is illustrated in Fig. 11.10.4. The filters are realized as bandpass filters with impulse responses

$$h_k(n) = h_0(n)e^{j2\pi nk/N}, \quad k = 0, 1, \dots, N-1 \quad (11.10.6)$$

The output of each bandpass filter is decimated by a factor D and multiplied by $e^{-j2\pi mk/N}$ to produce the DFT sequence $\{X_k(m)\}$. The modulation by the complex exponential allows us to shift the spectrum of the signal from $\omega_k = 2\pi k/N$ to $\omega_0 = 0$. Hence this realization is equivalent to the realization given in Fig. 11.10.3.

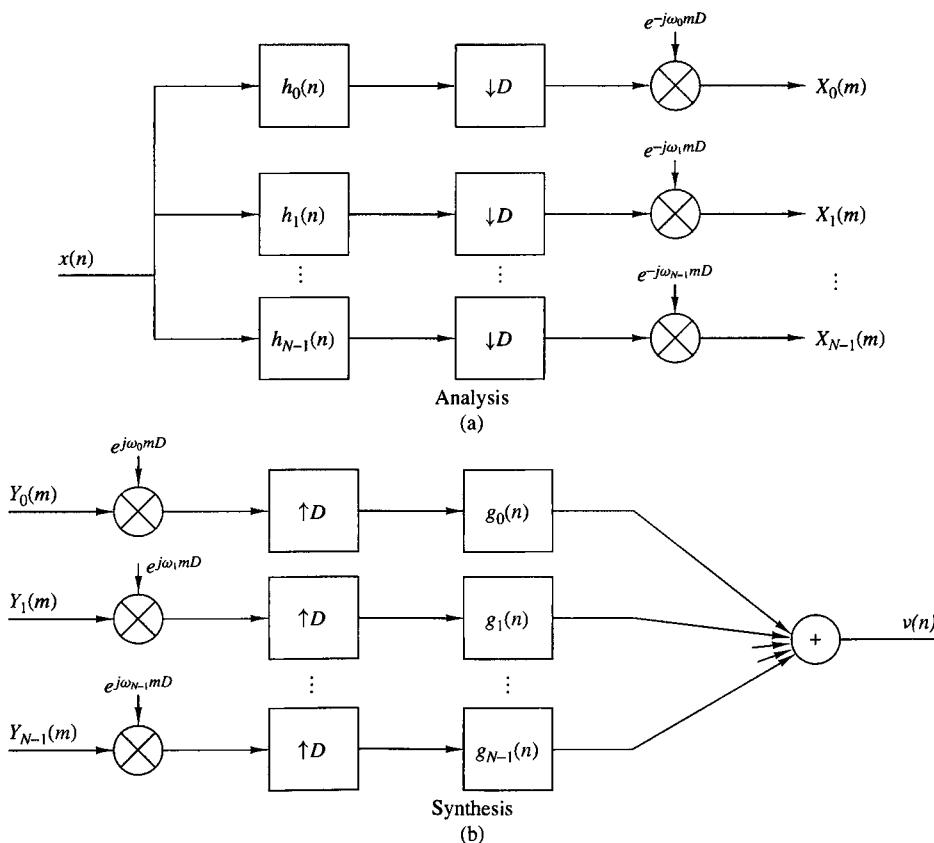


Figure 11.10.4 Alternative realization of a uniform DFT filter bank.

The analysis filter bank output can be written as

$$X_k(m) = \left[\sum_n x(n) h_0(mD - n) e^{j2\pi k(mD-n)/N} \right] e^{-j2\pi mkD/N} \quad (11.10.7)$$

The corresponding filter bank synthesizer can be realized as shown in Fig. 11.10.4-(b), where the input sequences are first multiplied by the exponential factors [$\exp(j2\pi kmD/N)$], upsampled by the factor $I = D$, and the resulting sequences are filtered by the bandpass interpolation filters with impulse responses

$$g_k(n) = g_0(n) e^{j2\pi nk/N} \quad (11.10.8)$$

where $\{g_0(n)\}$ is the impulse response of the prototype filter. The outputs of these filters are then summed to yield

$$v(n) = \frac{1}{N} \sum_{k=0}^{N-1} \left\{ \sum_m [Y_k(m) e^{j2\pi kmI/N}] g_k(n - mI) \right\} \quad (11.10.9)$$

where $I = D$.

In the implementation of digital filter banks, computational efficiency can be achieved by use of polyphase filters for decimation and interpolation. Of particular interest is the case where the decimation factor D is selected to be equal to the number N of frequency bands. When $D = N$, we say that the filter bank is *critically sampled*.

11.10.1 Polyphase Structures of Uniform Filter Banks

For the analysis filter bank, let us define a set of $N = D$ polyphase filters with impulse responses

$$p_k(n) = h_0(nN - k), \quad k = 0, 1, \dots, N - 1 \quad (11.10.10)$$

and the corresponding set of decimated input sequences

$$x_k(n) = x(nN + k), \quad k = 0, 1, \dots, N - 1 \quad (11.10.11)$$

Note that this definition of $\{p_k(n)\}$ implies that the commutator for the decimator rotates clockwise.

The structure of the analysis filter bank based on the use of polyphase filters can be obtained by substituting (11.10.10) and (11.10.11) into (11.10.7) and rearranging the summation into the form

$$X_k(m) = \sum_{n=0}^{N-1} \left[\sum_l p_n(l) x_n(m - l) \right] e^{-j2\pi nk/N}, \quad k = 0, 1, \dots, D - 1 \quad (11.10.12)$$

where $N = D$. Note that the inner summation represents the convolution of $\{p_n(l)\}$ with $\{x_n(l)\}$. The outer summation represents the N -point DFT of the filter outputs. The filter structure corresponding to this computation is illustrated in Fig. 11.10.5. Each sweep of the commutator results in N outputs, denoted as $\{r_n(m)\}$, $n = 0, 1, \dots, N - 1$ from the N polyphase filters. The N -point DFT of this sequence yields the spectral samples $\{X_k(m)\}$. For large values of N , the FFT algorithm provides an efficient means for computing the DFT.

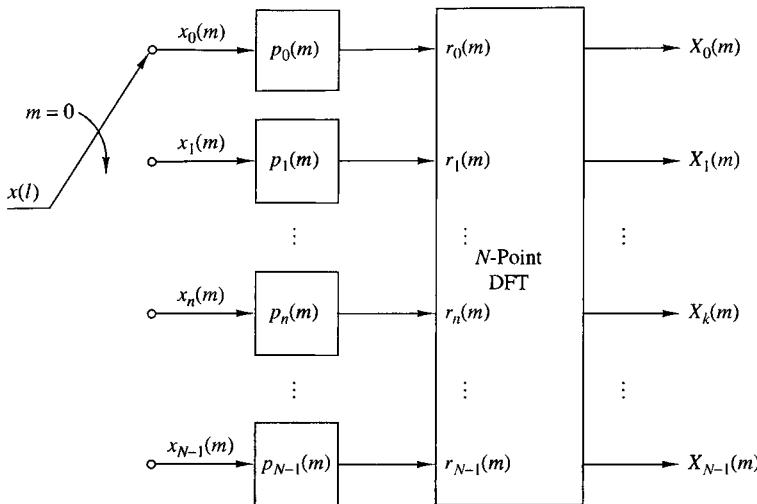


Figure 11.10.5 Digital filter bank structure for the computation of (11.10.12).

Now suppose that the spectral samples $\{X_k(m)\}$ are modified in some manner, prescribed by the application, to produce $\{Y_k(m)\}$. A filter bank synthesis filter based on a polyphase filter structure can be realized in a similar manner. First, we define the impulse response of the N ($D = I = N$) polyphase filters for the interpolation filter as

$$q_k(n) = g_0(nN + k), \quad k = 0, 1, \dots, N - 1 \quad (11.10.13)$$

and the corresponding set of output signals as

$$v_k(n) = v(nN + k), \quad k = 0, 1, \dots, N - 1 \quad (11.10.14)$$

Note that this definition of $\{q_k(n)\}$ implies that the commutator for the interpolator rotates counterclockwise.

By substituting (11.10.13) into (11.10.5), we can express the output $v_l(n)$ of the l th polyphase filter as

$$v_l(n) = \sum_m q_l(n - m) \left[\frac{1}{N} \sum_{k=0}^{N-1} Y_k(m) e^{j2\pi kl/N} \right], \quad l = 0, 1, \dots, N - 1 \quad (11.10.15)$$

The term in brackets is the N -point inverse DFT of $\{Y_k(m)\}$, which we denote as $\{y_l(m)\}$. Hence

$$v_l(n) = \sum_m q_l(n - m) y_l(m), \quad l = 0, 1, \dots, N - 1 \quad (11.10.16)$$

The synthesis structure corresponding to (11.10.16) is shown in Fig. 11.10.6. It is interesting to note that by defining the polyphase interpolation filter as in (11.10.13), the structure in Fig. 11.10.6 is the transpose of the polyphase analysis filter shown in Fig. 11.10.5.

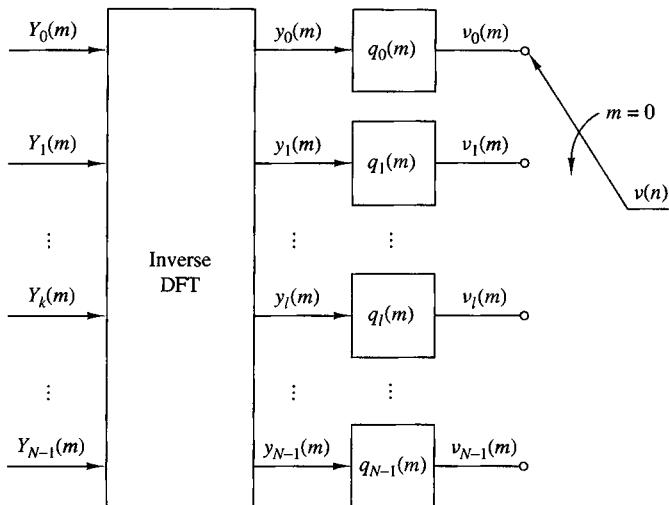


Figure 11.10.6 Digital filter bank structure for the computation of (11.10.16).

In our treatment of digital filter banks we considered the important case of critically sampled DFT filter banks, where $D = N$. Other choices of D and N can be employed in practice, but the implementation of the filters becomes more complex. Of particular importance is the oversampled DFT filter bank, where $N = KD$, D denotes the decimation factor and K is an integer that specifies the oversampling factor. In this case it can be shown that the polyphase filter bank structures for the analysis and synthesis filters can be implemented by use of N subfilters and N -point DFTs and inverse DFTs.

11.10.2 Transmultiplexers

An application of digital filter banks 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 is then modulated on a different carrier frequency 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.

Let us first consider FDM-to-TDM conversion. The analog FDM signal is passed through an A/D converter as shown in Fig. 11.10.7(a). The digital signal is then

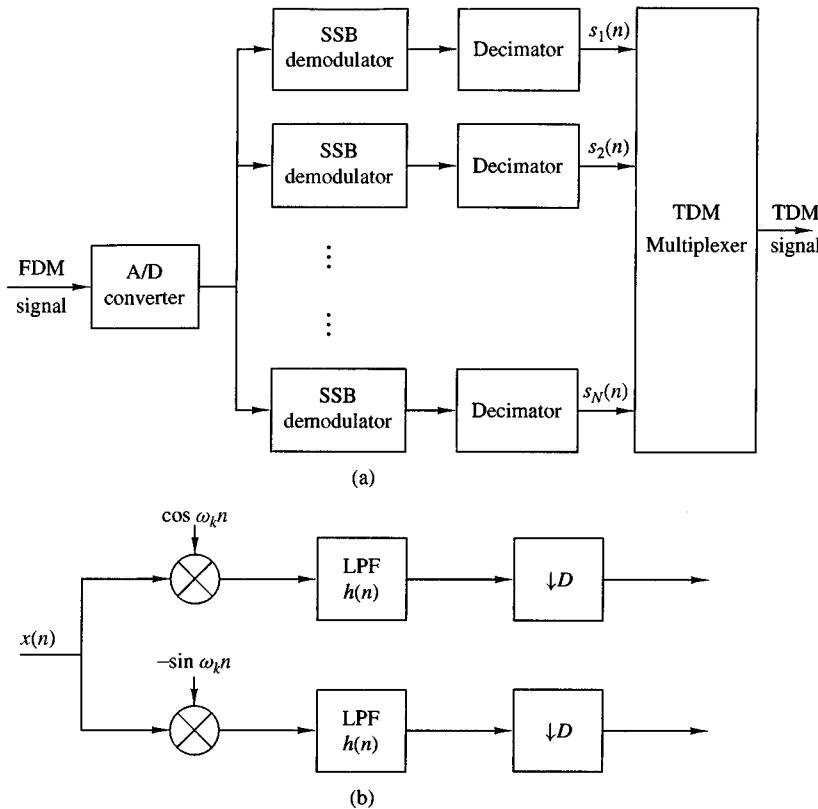


Figure 11.10.7 Block diagram of FDM-to-TDM transmultiplexer.

demodulated to baseband by means of single-sideband demodulators. The output of each demodulator is decimated and fed to the commutator of the TDM system.

To be specific, let us assume that the 12-channel FDM signal is sampled at the Nyquist rate of 96 kHz and passed through a filter-bank demodulator. The basic building block in the FDM demodulator consists of a frequency converter, a lowpass filter, and a decimator, as illustrated in Fig. 11.10.7(b). Frequency conversion can be efficiently implemented by the DFT filter bank described previously. The lowpass filter and decimator are efficiently implemented by use of the polyphase filter structure. Thus the basic structure for the FDM-to-TDM converter has the form of a DFT filter bank analyzer. Since the signal in each channel occupies a 4-kHz bandwidth, its Nyquist rate is 8 kHz, and hence the polyphase filter output can be decimated by a factor of 12. Consequently, the TDM commutator is operating at a rate of 12×8 kHz or 96 kHz.

In TDM-to-FDM conversion, the 12-channel TDM signal is demultiplexed into the 12 individual signals, where each signal has a rate of 8 kHz. The signal in each channel is interpolated by a factor of 12 and frequency converted by a single-sideband modulator, as shown in Fig. 11.10.8. The signal outputs from the 12 single-sideband modulators are summed and fed to the D/A converter. Thus we obtain the analog

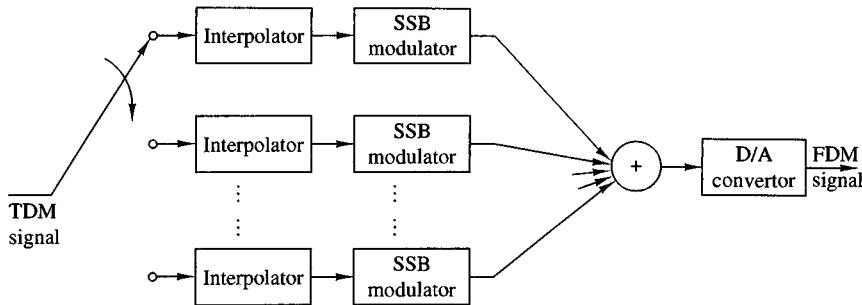


Figure 11.10.8 Block diagram of TDM-to-FDM transmultiplexer.

FDM signal for transmission. As in the case of FDM-to-TDM conversion, the interpolator and the modulator filter are combined and efficiently implemented by use of a polyphase filter. The frequency translation can be accomplished by the DFT. Consequently, the TDM-to-FDM converter encompasses the basic principles introduced previously in our discussion of DFT filter bank synthesis.

11.11 Two-Channel Quadrature Mirror Filter Bank

The basic building block in applications of quadrature mirror filters (QMF) is the two-channel QMF bank shown in Fig. 11.11.1. 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

$$\begin{aligned} X_{a0}(\omega) &= \frac{1}{2} \left[X\left(\frac{\omega}{2}\right) H_0\left(\frac{\omega}{2}\right) + X\left(\frac{\omega - 2\pi}{2}\right) H_0\left(\frac{\omega - 2\pi}{2}\right) \right] \\ X_{a1}(\omega) &= \frac{1}{2} \left[X\left(\frac{\omega}{2}\right) H_1\left(\frac{\omega}{2}\right) + X\left(\frac{\omega - 2\pi}{2}\right) H_1\left(\frac{\omega - 2\pi}{2}\right) \right] \end{aligned} \quad (11.11.1)$$

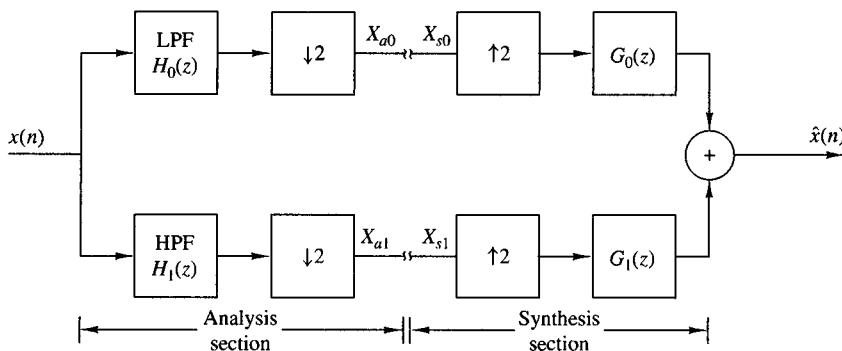


Figure 11.11.1 Two-channel QMF bank.

If $X_{s0}(\omega)$ and $X_{s1}(\omega)$ represent the two inputs to the synthesis section, the output is simply

$$\hat{X}(\omega) = X_{s0}(2\omega)G_0(\omega) + X_{s1}(2\omega)G_1(\omega) \quad (11.11.2)$$

Now, suppose that we connect the analysis filter to the corresponding synthesis filter, so that $X_{a0}(\omega) = X_{s0}(\omega)$ and $X_{a1}(\omega) = X_{s1}(\omega)$. Then, by substituting from (11.11.1) into (11.11.2), we obtain

$$\begin{aligned} \hat{X}(\omega) &= \frac{1}{2} [H_0(\omega)G_0(\omega) + H_1(\omega)G_1(\omega)] X(\omega) \\ &\quad + \frac{1}{2} [H_0(\omega - \pi)G_0(\omega) + H_1(\omega - \pi)G_1(\omega)] X(\omega - \pi) \end{aligned} \quad (11.11.3)$$

The first term in (11.11.3) is the desired signal output from the QMF bank. The second term represents the effect of aliasing, which we would like to eliminate.

In the z -transform domain (11.11.3) is expressed as

$$\begin{aligned} \hat{X}(z) &= \frac{1}{2} [H_0(z)G_0(z) + H_1(z)G_1(z)] X(z) \\ &\quad + \frac{1}{2} [H_0(-z)G_0(z) + H_1(-z)G_1(z)] X(-z) \\ &= Q(z)X(z) + A(z)X(-z) \end{aligned} \quad (11.11.4)$$

where, by definition,

$$\begin{aligned} Q(z) &= \frac{1}{2} [H_0(z)G_0(z) + H_1(z)G_1(z)] \\ A(z) &= \frac{1}{2} [H_0(-z)G_0(z) + H_1(-z)G_1(z)] \end{aligned} \quad (11.11.5)$$

11.11.1 Elimination of Aliasing

To eliminate aliasing, we require that $A(z) = 0$, i.e.,

$$H_0(-z)G_0(z) + H_1(-z)G_1(z) = 0 \quad (11.11.6)$$

In the frequency domain, this condition becomes

$$H_0(\omega - \pi)G_0(\omega) + H_1(\omega - \pi)G_1(\omega) = 0 \quad (11.11.7)$$

This condition can be simply satisfied by selecting $G_0(\omega)$ and $G_1(\omega)$ as

$$G_0(\omega) = H_1(\omega - \pi), \quad G_1(\omega) = -H_0(\omega - \pi) \quad (11.11.8)$$

Thus, the second term in (11.11.3) vanishes and the filter bank is alias-free.

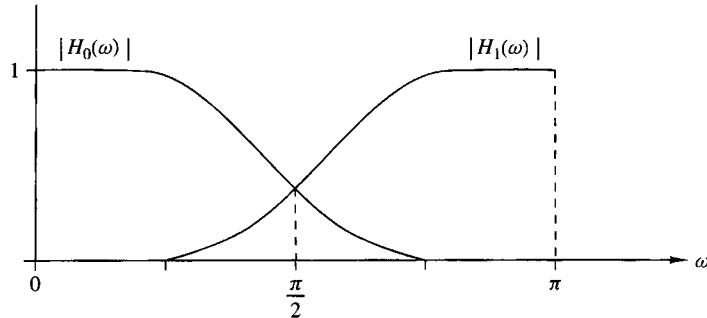


Figure 11.11.2 Mirror image characteristics of the analysis filters $H_0(\omega)$ and $H_1(\omega)$.

To elaborate, let us assume that $H_0(\omega)$ is a lowpass filter and $H_1(\omega)$ is a mirror-image highpass filter as shown in Fig. 11.11.2. Then we can express $H_0(\omega)$ and $H_1(\omega)$ as

$$\begin{aligned} H_0(\omega) &= H(\omega) \\ H_1(\omega) &= H(\omega - \pi) \end{aligned} \quad (11.11.9)$$

where $H(\omega)$ is the frequency response of a lowpass filter. In the time domain, the corresponding relations are

$$\begin{aligned} h_0(n) &= h(n) \\ h_1(n) &= (-1)^n h(n) \end{aligned} \quad (11.11.10)$$

As a consequence, $H_0(\omega)$ and $H_1(\omega)$ have mirror-image symmetry about the frequency $\omega = \pi/2$, as shown in Fig. 11.11.2. To be consistent with the constraint in (11.11.8), we select the lowpass filter $G_0(\omega)$ as

$$G_0(\omega) = H(\omega) \quad (11.11.11)$$

and the highpass filter $G_1(\omega)$ as

$$G_1(\omega) = -H(\omega - \pi) \quad (11.11.12)$$

In the time domain, these relations become

$$\begin{aligned} g_0(n) &= h(n) \\ g_1(n) &= (-1)^n h(n) \end{aligned} \quad (11.11.13)$$

In the z -transform domain, the relations for the elimination of aliasing are:

$$\begin{aligned} H_0(z) &= H(z) \\ H_1(z) &= H(-z) \\ G_0(z) &= H(z) \\ G_1(z) &= -H(-z) \end{aligned} \quad (11.11.14)$$

11.11.2 Condition for Perfect Reconstruction

With $A(z) = 0$, we now consider the condition for which the output $x(n)$ of the QMF bank is identical to the input $x(n)$, except for an arbitrary delay, for all possible inputs. When this condition is satisfied, the filter bank is called a perfect reconstruction QMF bank. Thus, we require that

$$Q(z) = \frac{1}{2} [H_0(z)G_0(z) + H_1(z)G_1(z)] = z^{-k} \quad (11.11.15)$$

By making use of the relations in (11.11.14), the condition for perfect reconstruction may be expressed as

$$H^2(z) - H^2(-z) = 2z^{-k} \quad (11.11.16)$$

or, equivalently,

$$H^2(\omega) - H^2(\omega - \pi) = 2e^{-j\omega k} \quad (11.11.17)$$

Therefore, for perfect reconstruction, the frequency response $H(\omega)$ of the lowpass filter in the two-channel QMF bank must satisfy the magnitude condition

$$\left| H^2(\omega) - H^2(\omega - \pi) \right| = C \quad (11.11.18)$$

where C is a positive constant, e.g., $C = 2$. We note that if $H(\omega)$ satisfies the magnitude condition in (11.11.18) and is designed to have linear phase, then the QMF output $x(n)$ is simply a delayed version of the input sequence $x(n)$. However, linear phase is not a necessary condition for perfect reconstruction.

11.11.3 Polyphase Form of the QMF Bank

The two-channel alias-free QMF bank can be realized efficiently by employing polyphase filters. Toward this goal, $H_0(z)$ and $H_1(z)$ can be expressed as

$$\begin{aligned} H_0(z) &= P_0(z^2) + z^{-1}P_1(z^2) \\ H_1(z) &= P_0(z^2) - z^{-1}P_1(z^2) \end{aligned} \quad (11.11.19)$$

where we have used the relationship given in (11.11.14).

Similarly, using the relations given in (11.11.14), we obtain the polyphase representation of the filters $G_0(z)$ and $G_1(z)$ as

$$\begin{aligned} G_0(z) &= P_0(z^2) + z^{-1}P_1(z^2) \\ G_1(z) &= -[P_0(z^2) - z^{-1}P_1(z^2)] \end{aligned} \quad (11.11.20)$$

Thus, we obtain the polyphase realization of the QMF bank as shown in Figure 11.11.3(a). The corresponding computationally efficient polyphase realization is shown in Figure 11.11.3(b).

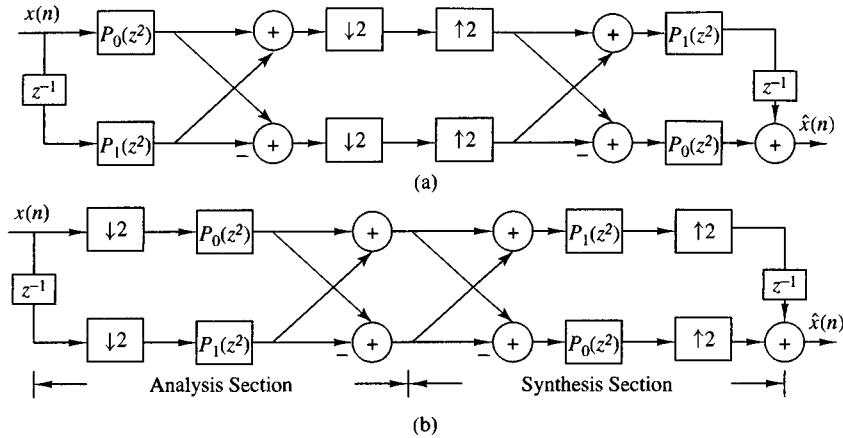


Figure 11.11.3 Polyphase realization of the two-channel QMF bank.

11.11.4 Linear Phase FIR QMF Bank

Now, let us consider the use of a linear phase filter $H(\omega)$. Hence $H(\omega)$ may be expressed in the form

$$H(\omega) = H_r(\omega)e^{-j\omega(N-1)/2} \quad (11.11.21)$$

where N is the filter length. Then

$$\begin{aligned} H^2(\omega) &= H_r^2(\omega)e^{-j\omega(N-1)} \\ &= |H(\omega)|^2e^{-j\omega(N-1)} \end{aligned} \quad (11.11.22)$$

and

$$\begin{aligned} H^2(\omega - \pi) &= H_r^2(\omega - \pi)e^{-j(\omega-\pi)(N-1)} \\ &= (-1)^{N-1}|H(\omega - \pi)|^2e^{-j\omega(N-1)} \end{aligned} \quad (11.11.23)$$

Therefore, the overall transfer function of the two-channel QMF which employs linear-phase FIR filters is

$$\frac{\hat{X}(\omega)}{X(\omega)} = \left[|H(\omega)|^2 - (-1)^{N-1}|H(\omega - \pi)|^2 \right] e^{-j\omega(N-1)} \quad (11.11.24)$$

Note that the overall filter has a delay of $N - 1$ samples and a magnitude characteristic

$$M(\omega) = |H(\omega)|^2 - (-1)^{N-1}|H(\omega - \pi)|^2 \quad (11.11.25)$$

We also note that when N is odd, $M(\pi/2) = 0$, because $|H(\pi/2)| = |H(3\pi/2)|$. This is an undesirable property for a QMF design. On the other hand, when N is even,

$$M(\omega) = |H(\omega)|^2 + |H(\omega - \pi)|^2 \quad (11.11.26)$$

which avoids the problem of a zero at $\omega = \pi/2$. For N even, the ideal two-channel QMF should satisfy the condition

$$M(\omega) = |H(\omega)|^2 + |H(\omega - \pi)|^2 = 1 \quad \text{for all } \omega \quad (11.11.27)$$

which follows from (11.11.25). Unfortunately, the only filter frequency response function that satisfies (11.11.27) is the trivial function $|H(\omega)|^2 = \cos^2 a\omega$. Consequently, any nontrivial linear-phase FIR filter $H(\omega)$ introduces some amplitude distortion.

The amount of amplitude distortion introduced by a nontrivial linear phase FIR filter in the QMF can be minimized by optimizing the FIR filter coefficients. A particularly effective method is to select the filter coefficients of $H(\omega)$ such that $M(\omega)$ is made as flat as possible while simultaneously minimizing (or constraining) the stopband energy of $H(\omega)$. This approach leads to the minimization of the integral squared error

$$J = w \int_{\omega_s}^{\pi} |H(\omega)|^2 d\omega + (1-w) \int_0^{\pi} [M(\omega) - 1]^2 d\omega \quad (11.11.28)$$

where w is a weighting factor in the range $0 < w < 1$. In performing the optimization, the filter impulse response is constrained to be symmetric (linear phase). This optimization is easily done numerically on a digital computer. This approach has been used by Johnston (1980) and Jain and Crochiere (1984) to design two-channel QMFs. Tables of optimum filter coefficients have been tabulated by Johnston (1980).

11.11.5 IIR QMF Bank

As an alternative to linear-phase FIR filters, we can design an IIR filter that satisfies the all-pass constraint given by (11.11.18). For this purpose, elliptic filters provide especially efficient designs. Since the QMF would introduce some phase distortion, the signal at the output of the QMF can be passed through an all-pass phase equalizer designed to minimize phase distortion.

11.11.6 Perfect Reconstruction Two-Channel FIR QMF Bank

We have observed that neither linear-phase FIR filters nor IIR filters described above yield perfect reconstruction in a two-channel QMF bank. However, as shown by Smith and Barnwell (1984), perfect reconstruction can be achieved by designing $H(\omega)$ as a linear-phase FIR half-band filter of length $2N - 1$.

A half-band filter is defined as a zero-phase FIR filter whose impulse response $\{b(n)\}$ satisfies the condition

$$b(2n) = \begin{cases} \text{constant}, & n = 0 \\ 0, & n \neq 0 \end{cases} \quad (11.11.29)$$

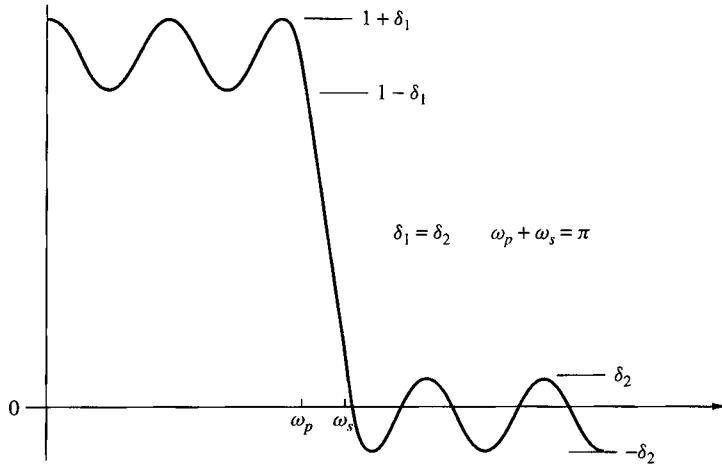


Figure 11.11.4 Frequency response characteristic of FIR half-band filter.

Hence all the even-numbered samples are zero except at $n = 0$. The zero-phase requirement implies that $b(n) = b(-n)$. The frequency response of such a filter is

$$B(\omega) = \sum_{n=-K}^K b(n)e^{-j\omega n} \quad (11.11.30)$$

where K is odd. Furthermore, $B(\omega)$ satisfies the condition that $B(\omega) + B(\pi - \omega)$ is equal to a constant for all frequencies. The typical frequency response characteristic of a half-band filter is shown in Fig. 11.11.4. We note that the filter response is symmetric with respect to $\pi/2$, the band edges frequencies ω_p and ω_s are symmetric about $\omega = \pi/2$, and the peak passband and stopband errors are equal. We also note that the filter can be made causal by introducing a delay of K samples.

Now, suppose that we design an FIR half-band filter of length $2N - 1$, where N is even, with frequency response as shown in Fig. 11.11.5(a). From $B(\omega)$ we construct another half-band filter with frequency response

$$B_+(\omega) = B(\omega) + \delta e^{-j\omega(N-1)} \quad (11.11.31)$$

as shown in Fig. 11.11.5(b). Note that $B_+(\omega)$ is nonnegative and hence it has the spectral factorization

$$B_+(z) = H(z)H(z^{-1})z^{-(N-1)} \quad (11.11.32)$$

or, equivalently,

$$B_+(\omega) = |H(\omega)|^2 e^{-j\omega(N-1)} \quad (11.11.33)$$

where $H(\omega)$ is the frequency response of an FIR filter of length N with real coefficients. Due to the symmetry of $B_+(\omega)$ with respect to $\omega = \pi/2$, we also have

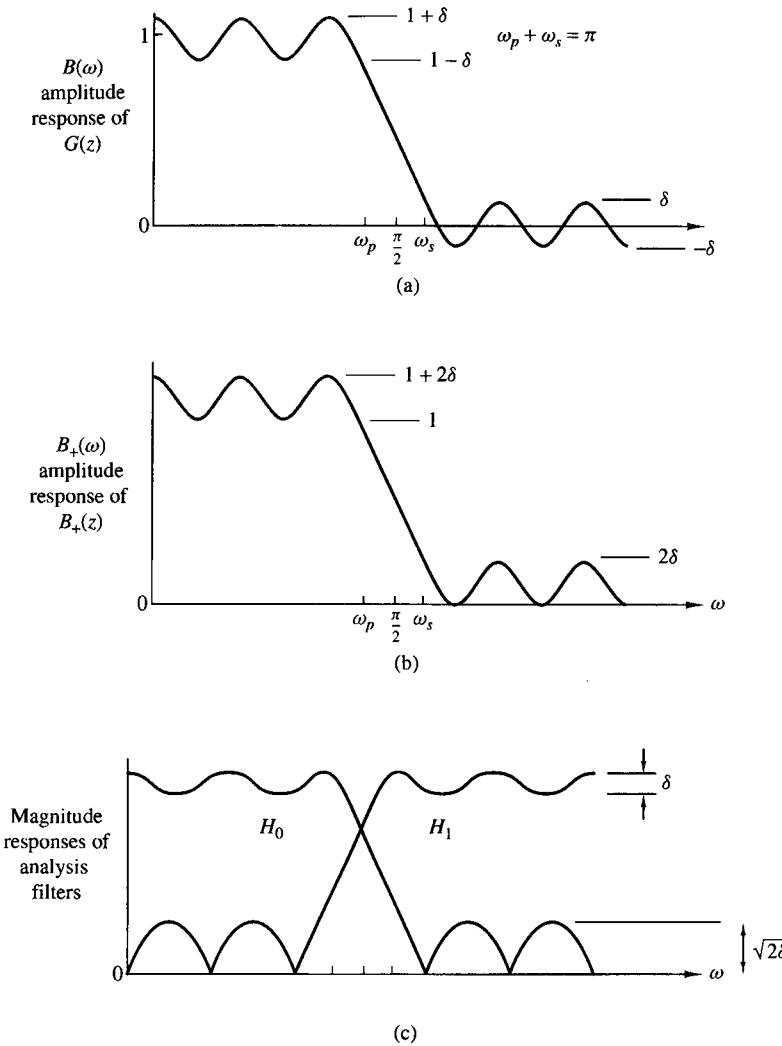


Figure 11.11.5 Frequency response characteristic of FIR half-band filters $B(\omega)$ and $B_+(\omega)$. (From Vaidyanathan (1987))

$$B_+(z) + (-1)^{N-1} B_+(-z) = \alpha z^{-(N-1)} \quad (11.11.34)$$

or, equivalently,

$$B_+(\omega) + (-1)^{N-1} B_+(\omega - \pi) = \alpha e^{-j\omega(N-1)} \quad (11.11.35)$$

where α is a constant. Thus, by substituting (11.11.32) into (11.11.34), we obtain

$$H(z)H(z^{-1}) + H(-z)H(-z^{-1}) = \alpha \quad (11.11.36)$$

Since $H(z)$ satisfies (11.11.36) and since aliasing is eliminated when we have $G_0(z) = H_1(-z)$ and $G_1(z) = -H_0(-z)$, it follows that these conditions are satisfied by choosing $H_1(z)$, $G_0(z)$, and $G_1(z)$ as

$$\begin{aligned} H_0(z) &= H(z) \\ H_1(z) &= -z^{-(N-1)}H_0(-z^{-1}) \\ G_0(z) &= z^{-(N-1)}H_0(z^{-1}) \\ G_1(z) &= z^{-(N-1)}H_1(z^{-1}) = -H_0(-z) \end{aligned} \quad (11.11.37)$$

Thus aliasing distortion is eliminated and, since $\hat{X}(\omega)/X(\omega)$ is a constant, the QMF performs perfect reconstruction so that $x(n) = \alpha x(n - N + 1)$. However, we note that $H(z)$ is not a linear-phase filter. The FIR filters $H_0(z)$, $H_1(z)$, $G_0(z)$, and $G_1(z)$ in the two-channel QMF bank are efficiently realized as polyphase filters as shown previously.

11.11.7 Two-Channel QMF Banks in Subband Coding

In Section 11.9.4 we described a method for efficient encoding of a speech signal based on subdividing the signal into several subbands and encoding each subband separately. For example, in Figure 11.9.4 we illustrated the separation of a signal into four subbands, namely, $0 \leq F \leq F_s/16$, $F_s/16 < F \leq F_s/8$, $F_s/8 < F \leq F_s/4$, and $F_s/4 < F \leq F_s/2$, where F_s is the sampling frequency. The subdivision into four subbands can be accomplished by use of three two-channel QMF analysis sections. After encoding and transmission through the channel, each subband signal is decoded and reconstructed by passing the subbands through three two-channel QMF synthesis filters. The system configuration for subband coding employing four subbands is illustrated in Figure 11.11.6.

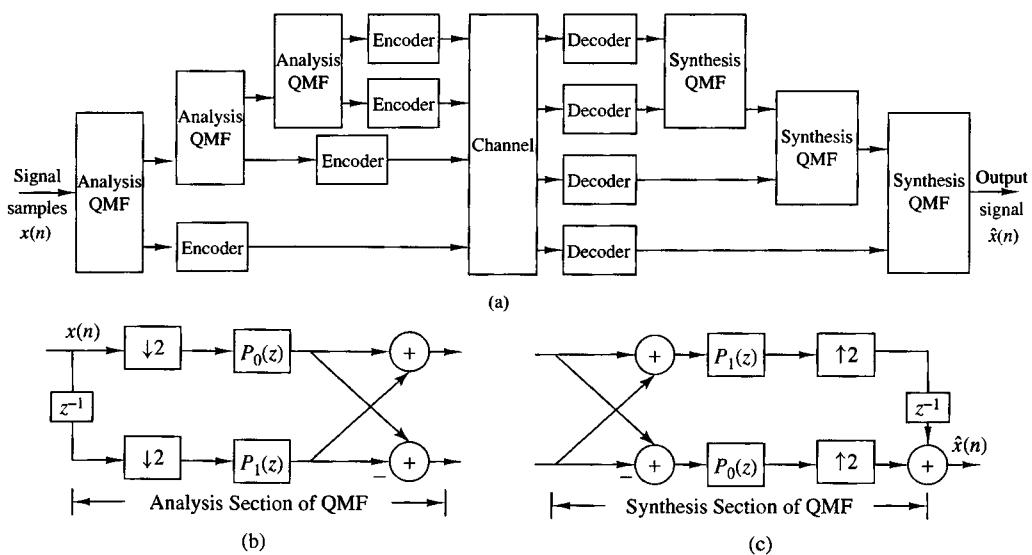


Figure 11.11.6 System for subband coding using two-channel QMF banks.

11.12 M -Channel QMF Bank

In this section, we consider the generalization of the QMF bank to M channels. Figure 11.12.1 illustrates the structure of an M -channel QMF bank, where $x(n)$ is the input to the analysis section, $x_k^{(a)}(n)$, $0 \leq k \leq M - 1$, are the outputs of the analysis filters, $x_k^{(s)}(n)$, $0 \leq k \leq M - 1$, are the inputs to the synthesis filters and $\hat{x}(n)$ is the output of the synthesis section.

The M outputs from the analysis filters may be expressed in the z -transform domain as

$$X_k^{(a)}(z) = \frac{1}{M} \sum_{m=0}^{M-1} H_k(z^{1/M} W_M^m) X(z^{1/M} W_M^m), \quad 0 \leq k \leq M - 1 \quad (11.12.1)$$

where $W_M = e^{-j2\pi/M}$. The output from the synthesis section is

$$\hat{X}(z) = \sum_{k=0}^{M-1} X_k^{(s)}(z^M) G_k(z) \quad (11.12.2)$$

As in the case of the two-channel QMF bank, we set $X_k^{(a)}(z) = X_k^{(s)}(z)$. Then, if we substitute (11.12.1) into (11.12.2), we obtain

$$\begin{aligned} \hat{X}(z) &= \sum_{k=0}^{M-1} G_k(z) \left[\frac{1}{M} \sum_{m=0}^{M-1} H_k(z W_M^m) X(z W_M^m) \right] \\ &= \sum_{m=0}^{M-1} \left[\frac{1}{M} \sum_{k=0}^{M-1} G_k(z) H_k(z W_M^m) \right] X(z W_M^m) \end{aligned} \quad (11.12.3)$$

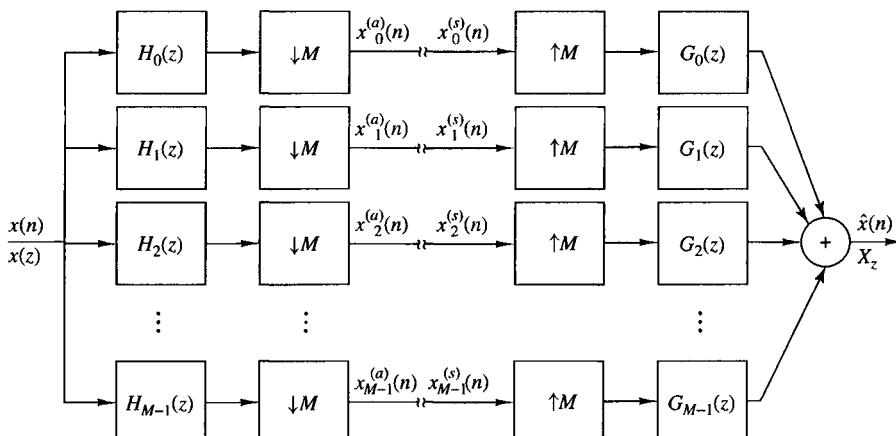


Figure 11.12.1 An M -channel QMF bank.

It is convenient to define the term in the brackets as

$$R_m(z) = \frac{1}{M} \sum_{k=0}^{M-1} G_k(z) H_k(z W_M^m), \quad 0 \leq m \leq M-1 \quad (11.12.4)$$

Then, (11.12.3) may be expressed as

$$\begin{aligned} \hat{X}(z) &= \sum_{m=0}^{M-1} R_m(z) X(z W_N^m) \\ &= R_0(z) X(z) + \sum_{m=1}^{M-1} R_m(z) X(z W_M^m) \end{aligned} \quad (11.12.5)$$

We note the first term in (11.12.5) is the alias-free component of the QMF bank and the second term is the aliasing component.

11.12.1 Alias-Free and Perfect Reconstruction Condition

From (11.12.5), it is clear that aliasing is eliminated by forcing the condition

$$R_m(z) = 0, \quad 1 \leq m \leq M-1 \quad (11.12.6)$$

With the elimination of the alias terms, the M -channel QMF bank becomes a linear time-invariant system that satisfies the input-output relation

$$\hat{X}(z) = R_0(z) X(z) \quad (11.12.7)$$

where

$$R_0(z) = \frac{1}{M} \sum_{k=0}^{M-1} H_k(z) G_k(z) \quad (11.12.8)$$

Then, the condition for a perfect reconstruction M -channel QMF bank becomes

$$R_0(z) = C z^{-k} \quad (11.12.9)$$

where C and k are positive constants.

11.12.2 Polyphase Form of the M -Channel QMF Bank

An efficient implementation of the M -channel QMF bank is achieved by employing polyphase filters. To obtain the polyphase form for the analysis filter bank, the k th filter $H_k(z)$ is represented as

$$H_k(z) = \sum_{m=0}^{M-1} z^{-m} P_{km}(z), \quad 0 \leq k \leq M-1 \quad (11.12.10)$$

We may express the equations for the M polyphase filter in matrix form as

$$\mathbf{H}(z) = \mathbf{P}(z^M)\mathbf{a}(z) \quad (11.12.11)$$

where

$$\begin{aligned} \mathbf{H}(z) &= [H_0(z) \ H_1(z) \ \cdots \ H_{M-1}(z)]^t \\ \mathbf{a}(z) &= [1 \ z^{-1} \ z^{-2} \ \cdots \ z^{-(M-1)}]^t \end{aligned} \quad (11.12.12)$$

and

$$\mathbf{P}(z) = \begin{bmatrix} P_{00}(z) & P_{01}(z) & \cdots & P_{0M-1}(z) \\ P_{10}(z) & P_{11}(z) & \cdots & P_{1M-1}(z) \\ \vdots & & & \\ P_{M-1\ 0}(z) & P_{M-1\ 1}(z) & \cdots & P_{M-1\ M-1}(z) \end{bmatrix} \quad (11.12.13)$$

The polyphase form of the analysis filter bank is shown in Figure 11.12.2(a), and after applying the first noble identity we obtain the structure shown in Figure 11.12.2(b).

The synthesis section can be constructed in a similar manner. Suppose we use a type II (transpose) form (see Problem 11.15) for the polyphase representation of the filters $\{G_k(z)\}$. Thus,

$$G_k(z) = \sum_{m=0}^{M-1} z^{-(M-1-m)} Q_{km}(z^M), \quad 0 \leq k \leq M-1 \quad (11.12.14)$$

When expressed in matrix form, (11.12.14) becomes

$$\mathbf{G}(z) = z^{-(M-1)} \mathbf{Q}^t(z^M) \mathbf{a}(z^{-1}) \quad (11.12.15)$$

where $\mathbf{a}(z)$ is defined in (11.12.12) and

$$\begin{aligned} \mathbf{G}(z) &= [G_0(z) \ G_1(z) \ \cdots \ G_{M-1}(z)]^t \\ \mathbf{Q}(z) &= \begin{bmatrix} Q_{00}(z) & Q_{01}(z) & \cdots & Q_{0\ M-1}(z) \\ Q_{10}(z) & Q_{11}(z) & \cdots & Q_{1\ M-1}(z) \\ \vdots & & & \\ Q_{M-1\ 0}(z) & Q_{M-1\ 1}(z) & \cdots & Q_{M-1\ M-1}(z) \end{bmatrix} \end{aligned} \quad (11.12.16)$$

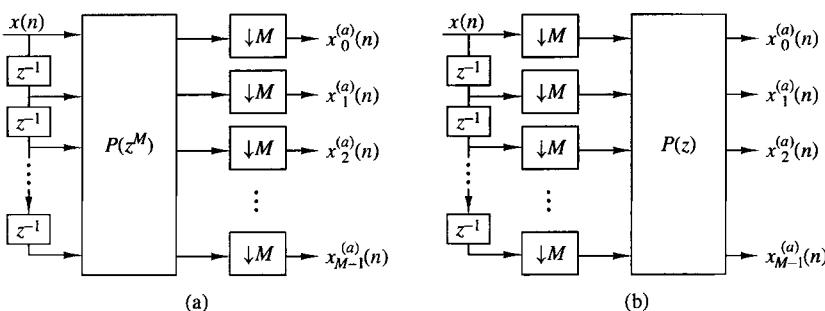


Figure 11.12.2 Polyphase structure of the analysis section of an M -channel QMF bank (a) before and (b) after applying the first noble identity.

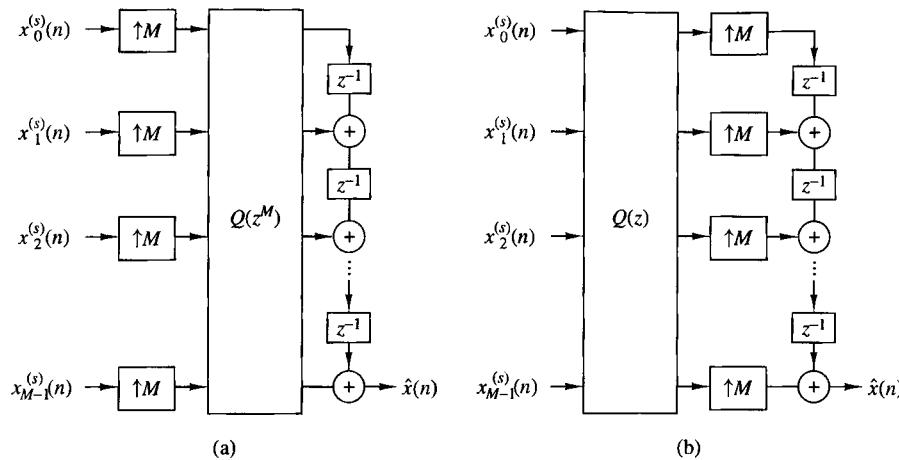


Figure 11.12.3 Polyphase structure of the synthesis section of an M -channel QMF bank (a) before and (b) after applying the first noble identity.

Therefore, the synthesis section of the M -channel QMF bank is realized as shown in Figure 11.11.3. By combining Figures 11.12.2(b) and 11.12.3(b), we obtain the polyphase structure of the complete M -channel QMF bank shown in Figure 11.12.4.

From the structure of the M -channel QMF bank shown in Figure 11.12.4, we observe that the perfect reconstruction condition can be restated as

$$\mathbf{Q}(z)\mathbf{P}(z) = Cz^{-k}\mathbf{I} \quad (11.12.17)$$

where \mathbf{I} is the $M \times M$ identity matrix. Hence, if the polyphase matrix $\mathbf{P}(z)$ is known, then the polyphase synthesis matrix $\mathbf{Q}(z)$ is

$$\mathbf{Q}(z) = Cz^{-k}[\mathbf{P}(z)]^{-1} \quad (11.12.18)$$

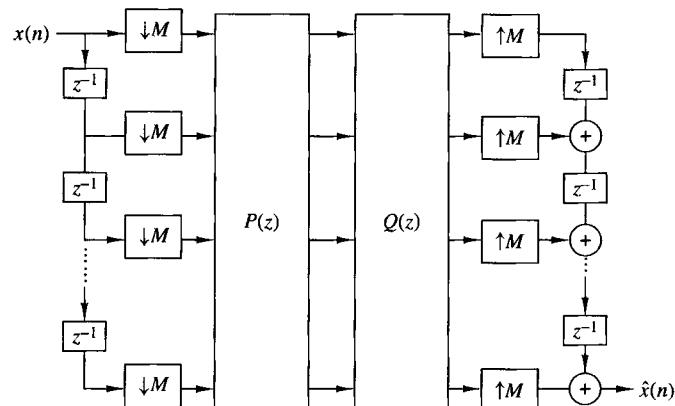


Figure 11.12.4
Polyphase realization of the M -channel QMF bank

EXAMPLE 11.12.1

Suppose the polyphase matrix for a three-channel perfect reconstruction FIR QMF bank is

$$\mathbf{P}(z^3) = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 3 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

Determine the analysis and the synthesis filters in the QMF bank.

Solution. The analysis filters are given by (11.12.11) as

$$\begin{bmatrix} H_0(z) \\ H_1(z) \\ H_2(z) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 3 & 1 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ z^{-1} \\ z^{-2} \end{bmatrix}$$

Hence,

$$H_0(z) = 1 + z^{-1} + 2z^{-2}, \quad H_1(z) = 2 + 3z^{-1} + z^{-2}, \quad H_2(z) = 1 + 2z^{-1} + z^{-2}$$

The inverse of $\mathbf{P}(z^3)$ is

$$[\mathbf{P}(z^3)]^{-1} = \frac{1}{2} \begin{bmatrix} 1 & 3 & -5 \\ -1 & -1 & 3 \\ 1 & -1 & 1 \end{bmatrix}$$

We may scale this inverse by the factor 2, so that

$$\mathbf{Q}(z^3) = 2[\mathbf{P}(z^3)]^{-1}$$

Then, by applying (11.12.15), we obtain the synthesis filters as

$$\begin{bmatrix} G_0(z) \\ G_1(z) \\ G_2(z) \end{bmatrix} = z^{-2} \begin{bmatrix} 1 & -1 & 1 \\ 3 & -1 & -1 \\ -5 & 3 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ z \\ z^2 \end{bmatrix}$$

Hence,

$$G_0(z) = 1 - z^{-1} + z^{-2} \quad G_1(z) = -1 - z^{-1} + 3z^{-2} \quad G_2(z) = 1 + 3z^{-1} - 5z^{-2}$$

Vaidyanathan (1992) treats the design of *M*-channel perfect reconstruction QMF banks by selecting the analysis filters $H_k(z)$ to be FIR with a paraunitary polyphase structure, i.e.,

$$\tilde{\mathbf{P}}(z)\mathbf{P}(z) = d\mathbf{I}, \quad d > 0 \quad (11.12.19)$$

where $\tilde{\mathbf{P}}(z)$ is the paraconjugate of $\mathbf{P}(z)$. That is, $\tilde{\mathbf{P}}(z)$ is formed by the transpose of $\mathbf{P}(1/z)$, with the coefficients of $\mathbf{P}(z)$ replaced by their complex conjugates. Then, the polyphase filters in the synthesis section are designed to satisfy

$$\mathbf{Q}(z) = Cz^{-k}\tilde{\mathbf{P}}(z), \quad C > 0, k > 0 \quad (11.12.20)$$

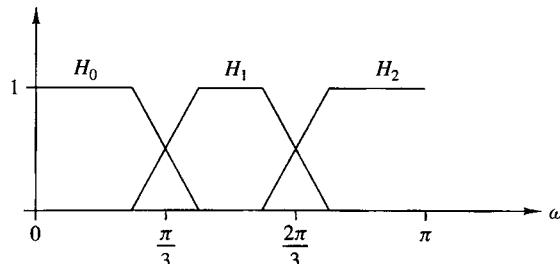


Figure 11.12.5
Magnitude response for analysis filters in an $M = 3$ QMF bank.

It can be shown (see Vaidyanathan et al. (1989)) that any causal L th-degree FIR paraunitary matrix $\mathbf{P}(z)$ can be expressed in a product form as

$$\mathbf{P}(z) = \mathbf{V}_L(z)\mathbf{V}_{L-1}(z) \cdots \mathbf{V}_1(z)\mathbf{U} \quad (11.12.21)$$

where \mathbf{U} is a unitary matrix and $\{\mathbf{V}_m(z)\}$ are paraunitary matrices that have the form

$$\mathbf{V}_m(z) = \mathbf{I} - \mathbf{v}_m \mathbf{v}_m^H + z^{-1} \mathbf{v}_m \mathbf{v}_m^H \quad (11.12.22)$$

where \mathbf{v}_m is an M -dimensional vector of unit norm. Then, the design of the synthesis filters reduces to the optimization of the components of \mathbf{v}_m and \mathbf{u}_i by minimizing an objective function, which may be a generalization of the two-channel QMF objective function given by (11.11.28). In particular, we may minimize the objective function

$$J = \sum_{k=0}^{M-1} \int_{k\text{th stopband}} |H_k(\omega)|^2 d\omega \quad (11.12.23)$$

by employing a nonlinear optimization technique to determine the components of \mathbf{v}_m and \mathbf{u}_i . Thus, the vectors \mathbf{v}_m and \mathbf{u}_i completely determine $\mathbf{P}(z)$ and, hence, the analysis filters $H_k(z)$. The synthesis filters are then determined from (11.12.20).

For example, consider the design of a perfect reconstruction three-channel QMF bank with magnitude responses as shown in Fig. 11.12.5. The design procedure described above yields the magnitude responses for the analysis filters that are shown in Fig. 11.12.6. The filter length is $N = 14$. We observe that the stopband attenuation of the filters is approximately 20 dB.

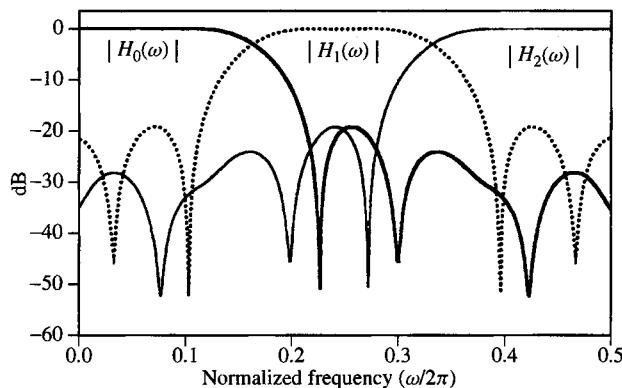


Figure 11.12.6
Magnitude response of optimized analysis filters for an $M = 3$ FIR perfect reconstruction QMF bank.
(From *Multirate Systems and Filter Banks*, by P.P. Vaidyanathan, ©1993 by Prentice Hall. Reprinted with permission of the publisher.)

11.13 Summary and References

The need for sampling rate conversion arises frequently in digital signal processing applications. In this chapter we first treated sampling rate reduction (decimation) and sampling rate increase (interpolation) by integer factors and then demonstrated how the two processes can be combined to obtain sampling rate conversion by any rational factor. Then, we described a method to achieve sampling rate conversion by an arbitrary factor. In the special case where the signal to be resampled is a bandpass signal, we described methods for performing the sampling rate conversion.

In general, the implementation of sampling rate conversion requires the use of a linear time-variant filter. We described methods for implementing such filters, including the class of polyphase filter structures, which are especially simple to implement. We also described the use of multistage implementations of multirate conversion as a means of simplifying the complexity of the filter required to meet the specifications.

We also described a number of applications that employ multirate signal processing, including the implementation of narrowband filters, phase shifters, filter banks, subband speech coders, quadrature mirror filters and transmultiplexers. These are just a few of the many applications encountered in practice where multirate signal processing is used.

The first comprehensive treatment of multirate signal processing was given in the book by Crochiere and Rabiner (1983). In the technical literature, we cite the papers by Schafer and Rabiner (1973), and Crochiere and Rabiner (1975, 1976, 1981, 1983). The use of interpolation methods to achieve sampling rate conversion by an arbitrary factor is treated in a paper by Ramstad (1984). A thorough tutorial treatment of multirate digital filters and filter banks, including quadrature mirror filters, is given by Vetterli (1987), and by Vaidyanathan (1990, 1993), where many references on various applications are cited. A comprehensive survey of digital transmultiplexing methods is found in the paper by Scheuermann and Gockler (1981). Subband coding of speech has been considered in many publications. The pioneering work on this topic was done by Crochiere (1977, 1981) and by Garland and Esteban (1980). Subband coding has also been applied to coding of images. We mention the papers by Vetterli (1984), Woods and O'Neil (1986), Smith and Eddins (1988), and Safranek et al. (1988) as just a few examples. In closing, we wish to emphasize that multirate signal processing continues to be a very active research area.

Problems

- 11.1** An analog signal $x_a(t)$ is bandlimited to the range $900 \leq F \leq 1100$ Hz. It is used as an input to the system shown in Fig. P11.1. In this system, $H(\omega)$ is an ideal lowpass filter with cutoff frequency $F_c = 125$ Hz.

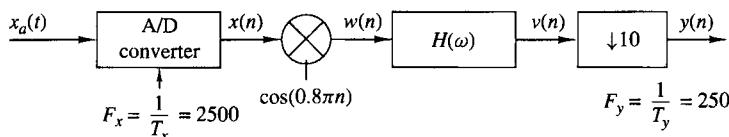


Figure P11.1

- (a) Determine and sketch the spectra for the signals $x(n)$, $w(n)$, $v(n)$, and $y(n)$.
 (b) Show that it is possible to obtain $y(n)$ by sampling $x_a(t)$ with period $T = 4$ milliseconds.

11.2 Consider the signal $x(n) = a^n u(n)$, $|a| < 1$.

- (a) Determine the spectrum $X(\omega)$.
 (b) The signal $x(n)$ is applied to a decimator that reduces the rate by a factor of 2. Determine the output spectrum.
 (c) Show that the spectrum in part (b) is simply the Fourier transform of $x(2n)$.

11.3 The sequence $x(n)$ is obtained by sampling an analog signal with period T . From this signal a new signal is derived having the sampling period $T/2$ by use of a linear interpolation method described by the equation

$$y(n) = \begin{cases} x(n/2), & n \text{ even} \\ \frac{1}{2} \left[x\left(\frac{n-1}{2}\right) + x\left(\frac{n+1}{2}\right) \right], & n \text{ odd} \end{cases}$$

- (a) Show that this linear interpolation scheme can be realized by basic digital signal processing elements.
 (b) Determine the spectrum of $y(n)$ when the spectrum of $x(n)$ is

$$X(\omega) = \begin{cases} 1, & 0 \leq |\omega| \leq 0.2\pi \\ 0, & \text{otherwise} \end{cases}$$

- (c) Determine the spectrum of $y(n)$ when the spectrum of $x(n)$ is

$$X(\omega) = \begin{cases} 1, & 0.7\pi \leq |\omega| \leq 0.9\pi \\ 0, & \text{otherwise} \end{cases}$$

11.4 Consider a signal $x(n)$ with Fourier transform

$$\begin{aligned} X(\omega) &= 0, & \omega_n < |\omega| \leq \pi \\ f_m &< |f| \leq \frac{1}{2} \end{aligned}$$

- (a) Show that the signal $x(n)$ can be recovered from its samples $x(mD)$ if the sampling frequency $\omega_s = 2\pi/D \leq 2\omega_m$ ($f_s = 1/D \geq 2f_m$).
 (b) Show that $x(n)$ can be reconstructed using the formula

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

where

$$h_r(n) = \frac{\sin(2\pi f_c n)}{2\pi n}, \quad f_m < f_c < f_s - f_m \\ \omega_m < \omega_c < \omega_s - \omega_m$$

- (c) Show that the bandlimited interpolation in part (b) can be thought of as a two-step process, first, increasing the sampling rate by a factor of D by inserting $(D-1)$ zero samples between successive samples of the decimated signal $x_a(n) = x(nD)$, and second, filtering the resulting signal using an ideal lowpass filter with cutoff frequency ω_c .

- 11.5** In this problem we illustrate the concepts of sampling and decimation for discrete-time signals. To this end consider a signal $x(n)$ with Fourier transform $X(\omega)$ as in Fig. P11.5.

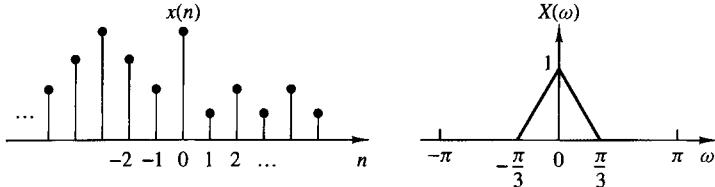


Figure P11.5

- (a)** Sampling $x(n)$ with a sampling period $D = 2$ results in the signal

$$x_s(n) = \begin{cases} x(n), & n = 0, \pm 2, \pm 4, \dots \\ 0, & n = \pm 1, \pm 3, \pm 5, \dots \end{cases}$$

Compute and sketch the signal $x_s(n)$ and its Fourier transform $X_s(\omega)$. Can we reconstruct $x(n)$ from $x_s(n)$? How?

- (b)** Decimating $x(n)$ by a factor of $D = 2$ produces the signal

$$x_d(n) = x(2n), \quad \text{all } n$$

Show that $X_d(\omega) = X_s(\omega/2)$. Plot the signal $x_d(n)$ and its transform $X_d(\omega)$. Do we lose any information when we decimate the sampled signal $x_s(n)$?

- 11.6** Design a decimator that downsamples an input signal $x(n)$ by a factor $D = 5$. Use the Remez algorithm to determine the coefficients of the FIR filter that has a 0.1-dB ripple in the passband ($0 \leq \omega \leq \pi/5$) and is down by at least 30 dB in the stopband. Also determine the corresponding polyphase filter structure for implementing the decimator.
- 11.7** Design an interpolator that increases the input sampling rate by a factor of $I = 2$. Use the Remez algorithm to determine the coefficients of the FIR filter that has a 0.1-dB ripple in the passband ($0 \leq \omega \leq \pi/2$) and is down by at least 30 dB in the stopband. Also, determine the corresponding polyphase filter structure for implementing the interpolator.
- 11.8** Design a sample rate converter that reduces the sampling rate by a factor $\frac{2}{5}$. Use the Remez algorithm to determine the coefficients of the FIR filter that has a 0.1-dB ripple in the passband and is down by at least 30 dB in the stopband. Specify the sets of time-variant coefficients $g(n, m)$ and the corresponding coefficients in the polyphase filter realization of the sample rate converter.

- 11.9** Consider the two different ways of cascading a decimator with an interpolator shown in Fig. P11.9.

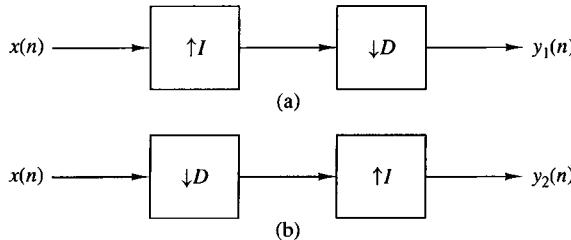


Figure P11.9

(a) If $D = I$, show that the outputs of the two configurations are different. Hence, in general, the two systems are not identical.

(b) Show that the two systems are identical if and only if D and I are relatively prime.

- 11.10** Prove the equivalence of the two decimator and interpolator configurations shown in Fig. P11.10 (noble identities) (see Vaidyanathan, 1990).

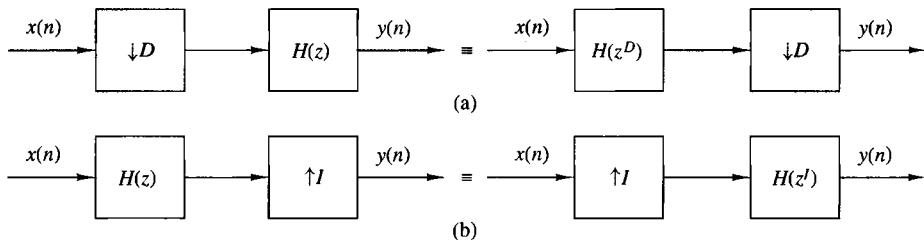


Figure P11.10

- 11.11** Consider an arbitrary digital filter with transfer function

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

(a) Perform a two-component polyphase decomposition of $H(z)$ by grouping the even-numbered samples $h_0(n) = h(2n)$ and the odd-numbered samples $h_1(n) = h(2n + 1)$. Thus show that $H(z)$ can be expressed as

$$H(z) = H_0(z^2) + z^{-1}H_1(z^2)$$

and determine $H_0(z)$ and $H_1(z)$.

- (b)** Generalize the result in part (a) by showing that $H(z)$ can be decomposed into a D -component polyphase filter structure with transfer function

$$H(z) = \sum_{k=0}^{D-1} z^{-k} H_k(z^D)$$

Determine $H_k(z)$.

- (c)** For the IIR filter with transfer function

$$H(z) = \frac{1}{1 - az^{-1}}$$

determine $H_0(z)$ and $H_1(z)$ for the two-component decomposition.

- 11.12** A sequence $x(n)$ is upsampled by $I = 2$, it passes through an LTI system $H_1(z)$, and then it is downsampled by $D = 2$. Can we replace this process with a single LTI system $H_2(z)$? If the answer is positive, determine the system function of this system.
- 11.13** Plot the signals and their corresponding spectra for rational sampling rate conversion by (a) $I/D = 5/3$ and (b) $I/D = 3/5$. Assume that the spectrum of the input signal $x(n)$ occupies the entire range $-\pi \leq \omega_x \leq \pi$.
- 11.14** We wish to design an efficient nonrecursive decimator for $D = 8$ using the factorization

$$H(z) = [(1 + z^{-1})(1 + z^{-2})(1 + z^{-4}) \dots (1 + z^{-2^{K-1}})]^5$$

- (a)** Derive an efficient implementation using filters with system function $H_k(z) = (1 + z^{-1})^5$.
- (b)** Show that each stage of the obtained decimator can be implemented more efficiently using a polyphase decomposition.
- 11.15** Let us consider an alternative polyphase decomposition by defining new polyphase filters $Q_m(z^N)$ as

$$H(z) = \sum_{m=0}^{N-1} z^{-(N-1-m)} Q_m(z^N)$$

This polyphase decomposition is called a type II form to distinguish it from the conventional decomposition based on polyphase filters $P_m(z^N)$.

- (a)** Show that the type II form polyphase filters $Q_m(z)$ are related to the polyphase filters $P_m(z^N)$ as follows:

$$Q_m(z^N) = P_{N-1-m}(z^N)$$

- (b)** Sketch the polyphase filter structure for $H(z)$ based in the polyphase filters $Q_m(z^N)$ and, thus, show that this structure is an alternative transpose form.

- 11.16** Use the result in Problem 11.15 to determine the type II form of the $I = 3$ interpolator in Figure 11.5.9.
- 11.17** Design a two-stage decimator for the following specifications:

$$D = 100$$

Passband: $0 \leq F \leq 50$

Transition band: $50 \leq F \leq 55$

Input sampling rate: 10,000 Hz

Ripple: $\delta_1 = 10^{-1}, \delta_2 = 10^{-3}$

- 11.18** Design a linear-phase FIR filter that satisfies the following specifications based on a single-stage and a two-stage multirate structure:

Sampling rate: 10,000 Hz

Passband: $0 \leq F \leq 60$

Transition band: $60 \leq F \leq 65$

Ripple: $\delta_1 = 10^{-1}, \delta_2 = 10^{-3}$

- 11.19** Prove that the half-band filter that satisfies (11.11.35) is always odd and the even coefficients are zero.

- 11.20** Design one-stage and two-stage interpolators to meet the following specifications:

$$I = 20$$

Input sampling rate: 10,000 Hz

Passband: $0 \leq F \leq 90$

Transition band: $90 \leq F \leq 100$

Ripple: $\delta_1 = 10^{-2}, \delta_2 = 10^{-3}$

- 11.21** By using (11.10.15) derive the equations corresponding to the structure for the polyphase synthesis section shown in Fig. 11.10.6.

- 11.22** Show that the transpose of an L -stage interpolator for increasing the sampling rate by an integer factor I is equivalent to an L -stage decimator that decreases the sampling rate by a factor $D = I$.

- 11.23** Sketch the polyphase filter structure for achieving a time advance of $(k/I)T_x$ in a sequence $x(n)$.

- 11.24** Prove the following expressions for an interpolator of order I .

- (a) The impulse response $h(n)$ can be expressed as

$$h(n) = \sum_{k=0}^{I-1} p_k(n - k)$$

where

$$p_k(n) = \begin{cases} p_k(n/I), & n = 0, \pm I, \pm 2I, \dots \\ 0, & \text{otherwise} \end{cases}$$

(b) $H(z)$ may be expressed as

$$H(z) = \sum_{k=0}^{I-1} z^{-k} p_k(z)$$

$$(c) P_k(z) = \frac{1}{I} \sum_{n=-\infty}^{\infty} \sum_{l=0}^{I-1} h(n) e^{j2\pi l(n-k)/I} z^{-(n-k)/I}$$

$$P_k(\omega) = \frac{1}{I} \sum_{l=0}^{I-1} H\left(\omega - \frac{2\pi l}{I}\right) e^{j(\omega - 2\pi l)k/I}$$

- 11.25** Consider the interpolation of a signal by a factor I . The interpolation filter with transfer function $H(z)$ is implemented by a polyphase filter structure based on an alternative (type II) decomposition, namely

$$H(z) = \sum_{m=0}^{I-1} z^{-(I-1-m)} Q_m(z^I)$$

Determine and sketch the structure of the interpolator that employs the polyphase filters $Q_m(z)$, $0 \leq m \leq I - 1$.

- 11.26** In the polyphase filter structures for a uniform DFT filter bank described in Section 11.10.1, the polyphase filters in the analysis section were defined by equation (11.10.10). Instead of this definition, suppose we define the N -band polyphase filters for the lowpass prototype filter, $H_0(z)$, as

$$H_0(z) = \sum_{i=0}^{N-1} z^{-i} P_i(z^N)$$

where

$$P_i(z) = \sum_{n=0}^{\infty} h_0(nN + i) z^{-n}, \quad 0 \leq i \leq N - 1$$

Then,

$$H_k(z) = H_0(z e^{-j2\pi k/N}) = H_0(z W_N^k)$$

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

- (a) Show that the filters $H_k(z)$, $1 \leq k \leq N - 1$, can be expressed as

$$H_k(z) = \begin{bmatrix} 1 & W_N^{-k} & W_N^{-2k} & \dots & W_N^{-(N-1)k} \end{bmatrix} \begin{bmatrix} P_0(z^N) \\ z^{-1} P_1(z^N) \\ \vdots \\ z^{-(N-1)} P_{N-1}(z^N) \end{bmatrix}$$

(b) Show that

$$\begin{bmatrix} H_0(z) \\ H_1(z) \\ \vdots \\ H_{N-1}(z) \end{bmatrix} = N\mathbf{W}^{-1} \begin{bmatrix} P_0(z^N) \\ z^{-1}P_1(z^N) \\ \vdots \\ z^{-(N-1)}P_{N-1}(z^N) \end{bmatrix}$$

where \mathbf{W} is the DFT matrix

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{(N-1)} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & W_N^{(N-1)} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix}$$

(c) Sketch the analysis section of the uniform DFT filter bank.

(d) Determine and sketch the synthesis section of the uniform DFT filter bank.

- 11.27** The prototype filter in a four-channel uniform DFT filter bank is characterized by the transfer function

$$H_0(z) = 1 + z^{-1} + 3z^{-2} + 4z^{-4}$$

- (a)** Determine the transfer functions of the filters $H_1(z)$, $H_2(z)$ and $H_3(z)$ in the analysis section.
(b) Determine the transfer functions of the filters in the synthesis section.
(c) Sketch the analysis and synthesis sections of the uniform DFT filter bank.

- 11.28** Consider the following FIR filter transfer function:

$$H(z) = -3 + 19z^{-2} + 32z^{-3} + 19z^{-4} - 3z^{-6}$$

- (a)** Show that $H(z)$ is a linear-phase filter.
(b) Show that $H(z)$ is a half-band filter.
(c) Plot the magnitude and phase responses of the filter.

- 11.29** The analysis filter $H_0(z)$ in a two-channel QMF has the transfer function

$$H_0(z) = 1 + z^{-1}$$

- (a)** Determine the polyphase filters $P_0(z^2)$ and $P_1(z^2)$.
(b) Determine the analysis filter $H_1(z)$ and sketch the two-channel analysis section that employs polyphase filters.
(c) Determine the synthesis filters $G_0(z)$ and $G_1(z)$ and sketch the entire two-channel QMF based on polyphase filters.
(d) Show that the QMF bank results in perfect reconstruction.

11.30 The analysis filters in a three-channel QMF bank have the transfer functions

$$H_0(z) = 1 + z^{-1} + z^{-2}$$

$$H_1(z) = 1 - z^{-1} + z^{-2}$$

$$H_2(z) = 1 - z^{-2}$$

- (a) Determine the polyphase matrix $\mathbf{P}(z^3)$ and express the analysis filters in the form (11.12.11).
- (b) Determine the synthesis filters $G_0(z)$, $G_1(z)$, and $G_2(z)$ that result in perfect reconstruction.
- (c) Sketch the three-channel QMF bank that employs polyphase filters in the analysis and synthesis sections.

11.31 *Zoom-frequency analysis* Consider the system in Fig. P11.31(a).

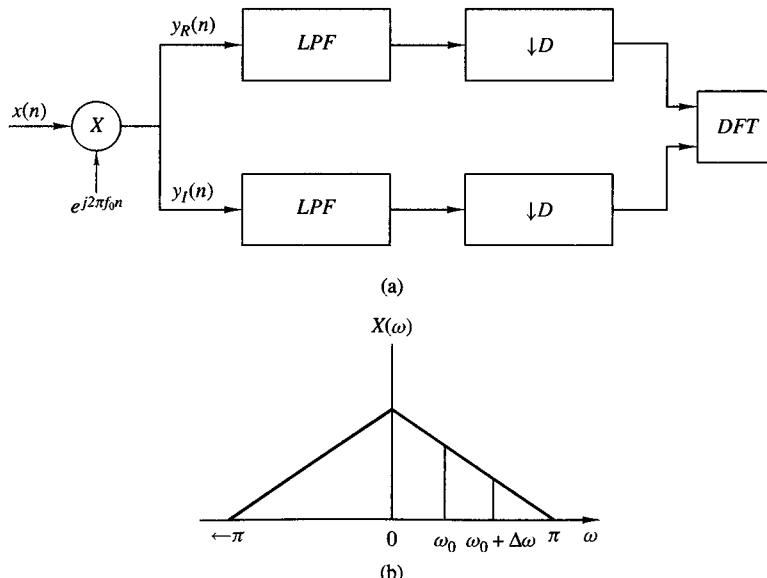


Figure P11.31

- (a) Sketch the spectrum of the signal $y(n) = y_R(n) + jy_I(n)$ if the input signal $x(n)$ has the spectrum shown in Fig. P11.31(b).
- (b) Suppose that we are interested in the analysis of the frequencies in the band $f_0 \leq f \leq f_0 + \Delta f$, where $f_0 = 1/6$ and $\Delta f = 1/3$. Determine the cutoff of a lowpass filter and the decimation factor D required to retain the information contained in this band of frequencies.

(c) Assume that

$$x(n) = \sum_{k=0}^{p-1} \left(1 - \frac{k}{2p}\right) \cos 2\pi f_k n$$

where $p = 40$ and $f_k = k/p$, $k = 0, 1, \dots, p - 1$. Compute and plot the 1024-point DFT of $x(n)$.

- (d) Repeat part (b) for the signal $x(n)$ given in part (c) by using an appropriately designed lowpass linear phase FIR filter to determine the decimated signal $s(n) = s_R(n) + js_I(n)$.
- (e) Compute the 1024-point DFT of $s(n)$ and investigate to see if you have obtained the expected results.

12

Linear Prediction and Optimum Linear Filters

The design of filters to perform signal estimation is a problem that frequently arises in the design of communication systems, control systems, in geophysics, and in many other applications and disciplines. In this chapter we treat the problem of optimum filter design from a statistical viewpoint. The filters are constrained to be linear and the optimization criterion is based on the minimization of the mean-square error. As a consequence, only the second-order statistics (autocorrelation and crosscorrelation functions) of a stationary process are required in the determination of the optimum filters.

Included in this treatment is the design of optimum filters for linear prediction. Linear prediction is a particularly important topic in digital signal processing, with applications in a variety of areas, such as speech signal processing, image processing, and noise suppression in communication systems. As we shall observe, determination of the optimum linear filter for prediction requires the solution of a set of linear equations that have some special symmetry. To solve these linear equations, we describe two algorithms, the Levinson–Durbin algorithm and the Schur algorithm, which provide the solution to the equations through computationally efficient procedures that exploit the symmetry properties.

The last section of the chapter treats an important class of optimum filters called Wiener filters. Wiener filters are widely used in many applications involving the estimation of signals corrupted with additive noise.

12.1 Random Signals, Correlation Functions, and Power Spectra

We begin with a brief review of the characterization of random signals in terms of statistical averages expressed in both the time domain and the frequency domain. The

reader is assumed to have a background in probability theory and random processes, at the level given in the books of Helstrom (1990), Peebles (1987), and Stark and Woods (1994).

12.1.1 Random Processes

Many physical phenomena encountered in nature are best characterized in statistical terms. For example, meteorological phenomena such as air temperature and air pressure fluctuate randomly as a function of time. Thermal noise voltages generated in the resistors of electronic devices, such as a radio or television receiver, are also randomly fluctuating phenomena. These are just a few examples of random signals. Such signals are usually modeled as infinite-duration infinite-energy signals.

Suppose that we take the set of waveforms corresponding to the air temperature in different cities around the world. For each city there is a corresponding waveform that is a function of time, as illustrated in Fig. 12.1.1. The set of all possible waveforms

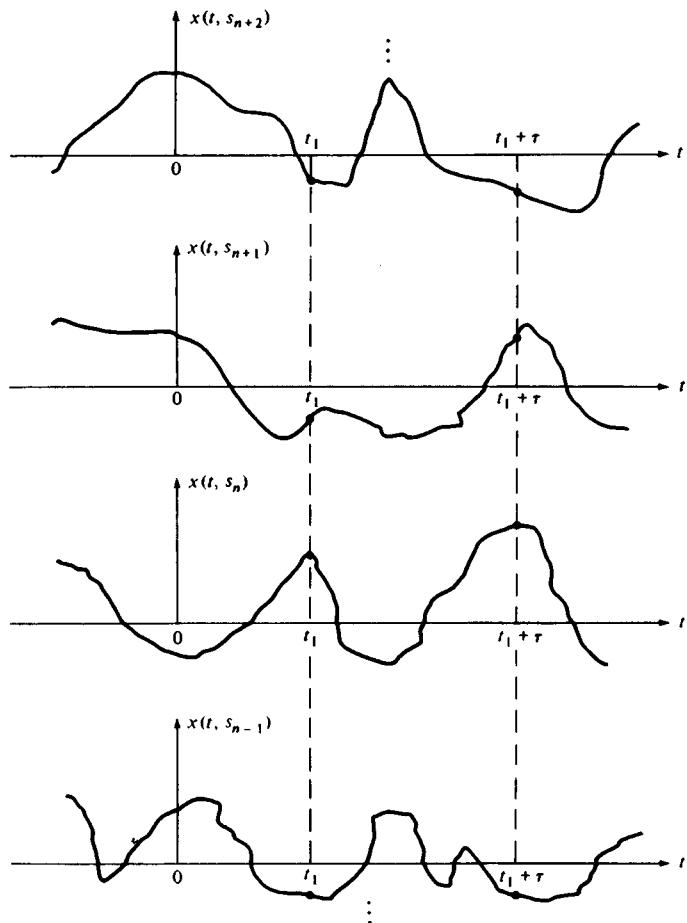


Figure 12.1.1 Sample functions of a random process.

is called an *ensemble* of time functions or, equivalently, a *random process*. The waveform for the temperature in any particular city is a *single realization* or a *sample function* of the random process.

Similarly, the thermal noise voltage generated in a resistor is a single realization or a sample function of the random process consisting of all noise voltage waveforms generated by the set of all resistors.

The set (ensemble) of all possible noise waveforms of a random process is denoted as $X(t, S)$, where t represents the time index and S represents the set (sample space) of all possible sample functions. A single waveform in the ensemble is denoted by $x(t, s)$. Usually, we drop the variable s (or S) for notational convenience, so that the random process is denoted as $X(t)$ and a single realization is denoted as $x(t)$.

Having defined a random process $X(t)$ as an ensemble of sample functions, let us consider the values of the process for any set of time instants $t_1 > t_2 > \dots > t_n$, where n is any positive integer. In general, the samples $X_{t_i} \equiv x(t_i)$, $i = 1, 2, \dots, n$ are n random variables characterized statistically by their joint probability density function (PDF) denoted as $p(x_{t_1}, x_{t_2}, \dots, x_{t_n})$ for any n .

12.1.2 Stationary Random Processes

Suppose that we have n samples of the random process $X(t)$ at $t = t_i$, $i = 1, 2, \dots, n$, and another set of n samples displaced in time from the first set by an amount τ . Thus the second set of samples are $X_{t_i+\tau} \equiv X(t_i + \tau)$, $i = 1, 2, \dots, n$, as shown in Fig. 12.1.1. This second set of n random variables is characterized by the joint probability density function $p(x_{t_1+\tau}, \dots, x_{t_n+\tau})$. The joint PDFs of the two sets of random variables may or may not be identical. When they are identical, then

$$p(x_{t_1}, x_{t_2}, \dots, x_{t_n}) = p(x_{t_1+\tau}, x_{t_2+\tau}, \dots, x_{t_n+\tau}) \quad (12.1.1)$$

for all τ and all n , and the random process is said to be *stationary in the strict sense*. In other words, the statistical properties of a stationary random process are invariant to a translation of the time axis. On the other hand, when the joint PDFs are different, the random process is nonstationary.

12.1.3 Statistical (Ensemble) Averages

Let us consider a random process $X(t)$ sampled at time instant $t = t_i$. Thus $X(t_i)$ is a random variable with PDF $p(x_{t_i})$. The l th *moment* of the random variable is defined as the *expected value* of $X^l(t_i)$, that is,

$$E(X_{t_i}^l) = \int_{-\infty}^{\infty} x_{t_i}^l p(x_{t_i}) dx_{t_i} \quad (12.1.2)$$

In general, the value of the l th moment depends on the time instant t_i , if the PDF of X_{t_i} depends on t_i . When the process is stationary, however, $p(x_{t_i+\tau}) = p(x_{t_i})$ for all τ . Hence the PDF is independent of time and, consequently, the l th moment is independent of time (a constant).

Next, let us consider the two random variables $X_{t_i} = X(t_i)$, $i = 1, 2$, corresponding to samples of $X(t)$ taken at $t = t_1$ and $t = t_2$. The statistical (ensemble) correlation between X_{t_1} and X_{t_2} is measured by the joint moment

$$E(X_{t_1} X_{t_2}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_{t_1} x_{t_2} p(x_{t_1} x_{t_2}) dx_1 dx_2 \quad (12.1.3)$$

Since the joint moment depends on the time instants t_1 and t_2 , it is denoted as $\gamma_{xx}(t_1, t_2)$ and is called the *autocorrelation function* of the random process. When the process $X(t)$ is stationary, the joint PDF of the pair (X_{t_1}, X_{t_2}) is identical to the joint PDF of the pair $(X_{t_1+\tau}, X_{t_2+\tau})$ for any arbitrary τ . This implies that the autocorrelation function of $X(t)$ depends on the time difference $t_1 - t_2 = \tau$. Hence for a stationary real-valued random process the autocorrelation function is

$$\gamma_{xx}(\tau) = E[X_{t_1+\tau} X_{t_1}] \quad (12.1.4)$$

On the other hand,

$$\gamma_{xx}(-\tau) = E(X_{t_1-\tau} X_{t_1}) = E(X'_{t_1} X'_{t_1+\tau}) = \gamma_{xx}(\tau) \quad (12.1.5)$$

Therefore, $\gamma_{xx}(\tau)$ is an even function. We also note that $\gamma_{xx}(0) = E(X_{t_1}^2)$ is the *average power* of the random process.

There exist nonstationary processes with the property that the mean value of the process is a constant and the autocorrelation function satisfies the property $\gamma_{xx}(t_1, t_2) = \gamma_{xx}(t_1 - t_2)$. Such a process is called *wide-sense stationary*. Clearly, wide-sense stationarity is a less stringent condition than strict-sense stationarity. In our treatment we shall require only that the processes be wide-sense stationary.

Related to the autocorrelation function is the autocovariance function, which is defined as

$$\begin{aligned} c_{xx}(t_1, t_2) &= E\{[X_{t_1} - m(t_1)][X_{t_2} - m(t_2)]\} \\ &= \gamma_{xx}(t_1, t_2) - m(t_1)m(t_2) \end{aligned} \quad (12.1.6)$$

where $m(t_1) = E(X_{t_1})$ and $m(t_2) = E(X_{t_2})$ are the mean values of X_{t_1} and X_{t_2} , respectively. When the process is stationary,

$$c_{xx}(t_1, t_2) = c_{xx}(t_1 - t_2) = c_{xx}(\tau) = \gamma_{xx}(\tau) - m_x^2 \quad (12.1.7)$$

where $\tau = t_1 - t_2$. Furthermore, the variance of the process is $\sigma_x^2 = c_{xx}(0) = \gamma_{xx}(0) - m_x^2$.

12.1.4 Statistical Averages for Joint Random Processes

Let $X(t)$ and $Y(t)$ be two random processes and let $X_{t_i} \equiv X(t_i)$, $i = 1, 2, \dots, n$, and $Y_{t'_j} \equiv Y(t'_j)$, $j = 1, 2, \dots, m$, represent the random variables at times $t_1 > t_2 >$

$\dots > t_n$ and $t'_1 > t'_2 > \dots > t'_m$, respectively. The two sets of random variables are characterized statistically by the joint PDF

$$p(x_{t_1}, x_{t_2}, \dots, x_{t_n}, y_{t'_1}, y_{t'_2}, \dots, y_{t'_m})$$

for any set of time instants $\{t_i\}$ and $\{t'_j\}$ and for any positive integer values of m and n .

The *crosscorrelation function* of $X(t)$ and $Y(t)$, denoted as $\gamma_{xy}(t_1, t_2)$, is defined by the joint moment

$$\gamma_{xy}(t_1, t_2) \equiv E(X_{t_1} Y_{t_2}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_{t_1} y_{t_2} p(x_{t_1}, y_{t_2}) dx_{t_1} dy_{t_2} \quad (12.1.8)$$

and the crosscovariance is

$$c_{xy}(t_1, t_2) = \gamma_{xy}(t_1, t_2) - m_x(t_1)m_y(t_2) \quad (12.1.9)$$

When the random processes are jointly and individually stationary, we have $\gamma_{xy}(t_1, t_2) = \gamma_{xy}(t_1 - t_2)$ and $c_{xy}(t_1, t_2) = c_{xy}(t_1 - t_2)$. In this case

$$\gamma_{xy}(-\tau) = E(X_{t_1} Y_{t_1+\tau}) = E(X_{t'_1-\tau} Y_{t'_1}) = \gamma_{yx}(\tau) \quad (12.1.10)$$

The random processes $X(t)$ and $Y(t)$ are said to be statistically independent if and only if

$$p(x_{t_1}, x_{t_2}, \dots, x_{t_n}, y_{t'_1}, y_{t'_2}, \dots, y_{t'_m}) = p(x_{t_1}, \dots, x_{t_n}) p(y_{t'_1}, \dots, y_{t'_m})$$

for all choices of t_i, t'_i and for all positive integers n and m . The processes are said to be uncorrelated if

$$\gamma_{xy}(t_1, t_2) = E(X_{t_1})E(Y_{t_2}) \quad (12.1.11)$$

so that $c_{xy}(t_1, t_2) = 0$.

A complex-valued random process $Z(t)$ is defined as

$$Z(t) = X(t) + jY(t) \quad (12.1.12)$$

where $X(t)$ and $Y(t)$ are random processes. The joint PDF of the complex-valued random variables $Z_{t_i} \equiv Z(t_i)$, $i = 1, 2, \dots$, is given by the joint PDF of the components (X_{t_i}, Y_{t_i}) , $i = 1, 2, \dots, n$. Thus the PDF that characterizes Z_{t_i} , $i = 1, 2, \dots, n$ is

$$p(x_{t_1}, x_{t_2}, \dots, x_{t_n}, y_{t_1}, y_{t_2}, \dots, y_{t_n})$$

A complex-valued random process $Z(t)$ is encountered in the representation of the in-phase and quadrature components of the lowpass equivalent of a narrowband random signal or noise. An important characteristic of such a process is its autocorrelation function, which is defined as

$$\begin{aligned} \gamma_{zz}(t_1, t_2) &= E(Z_{t_1} Z_{t_2}^*) \\ &= E[(X_{t_1} + jY_{t_1})(X_{t_2} - jY_{t_2})] \\ &= \gamma_{xx}(t_1, t_2) + \gamma_{yy}(t_1, t_2) + j[\gamma_{yx}(t_1, t_2) - \gamma_{xy}(t_1, t_2)] \end{aligned} \quad (12.1.13)$$

When the random processes $X(t)$ and $Y(t)$ are jointly and individually stationary, the autocorrelation function of $Z(t)$ becomes

$$\gamma_{zz}(t_1, t_2) = \gamma_{zz}(t_1 - t_2) = \gamma_{zz}(\tau)$$

where $\tau = t_1 - t_2$. The complex conjugate of (12.1.13) is

$$\gamma_{zz}^*(\tau) = E(Z_{t_1}^* Z_{t_1-\tau}) = \gamma_{zz}(-\tau) \quad (12.1.14)$$

Now, suppose that $Z(t) = X(t) + jY(t)$ and $W(t) = U(t) + jV(t)$ are two complex-valued random processes. Their crosscorrelation function is defined as

$$\begin{aligned} \gamma_{zw}(t_1, t_2) &= E(Z_{t_1} W_{t_2}^*) \\ &= E[(X_{t_1} + jY_{t_1})(U_{t_2} - jV_{t_2})] \\ &= \gamma_{xu}(t_1, t_2) + j\gamma_{yu}(t_1, t_2) + j[\gamma_{yu}(t_1, t_2) - \gamma_{xv}(t_1, t_2)] \end{aligned} \quad (12.1.15)$$

When $X(t)$, $Y(t)$, $U(t)$, and $V(t)$ are pairwise stationary, the crosscorrelation functions in (12.1.15) become functions of the time difference $\tau = t_1 - t_2$. In addition, we have

$$\gamma_{zw}^*(\tau) = E(Z_{t_1}^* W_{t_1-\tau}) = E(Z_{t_1+\tau}^* W_{t_1}) = \gamma_{wz}(-\tau) \quad (12.1.16)$$

12.1.5 Power Density Spectrum

A stationary random process is an infinite-energy signal and hence its Fourier transform does not exist. The spectral characteristic of a random process is obtained according to the Wiener–Khintchine theorem, by computing the Fourier transform of the autocorrelation function. That is, the distribution of power with frequency is given by the function

$$\Gamma_{xx}(F) = \int_{-\infty}^{\infty} \gamma_{xx}(\tau) e^{-j2\pi F\tau} d\tau \quad (12.1.17)$$

The inverse Fourier transform is given as

$$\gamma_{xx}(\tau) = \int_{-\infty}^{\infty} \Gamma_{xx}(F) e^{j2\pi F\tau} dF \quad (12.1.18)$$

We observe that

$$\begin{aligned} \gamma_{xx}(0) &= \int_{-\infty}^{\infty} \Gamma_{xx}(F) dF \\ &= E(X_t^2) \geq 0 \end{aligned} \quad (12.1.19)$$

Since $E(X_t^2) = \gamma_{xx}(0)$ represents the average power of the random process, which is the area under $\Gamma_{xx}(F)$, it follows that $\Gamma_{xx}(F)$ is the distribution of power as a function of frequency. For this reason, $\Gamma_{xx}(F)$ is called the *power density spectrum* of the random process.

If the random process is real, $\gamma_{xx}(\tau)$ is real and even and hence $\Gamma_{xx}(F)$ is real and even. If the random process is complex valued, $\gamma_{xx}(\tau) = \gamma_{xx}^*(-\tau)$ and, hence

$$\begin{aligned}\Gamma_{xx}^*(F) &= \int_{-\infty}^{\infty} \gamma_{xx}^*(\tau) e^{j2\pi F\tau} d\tau = \int_{-\infty}^{\infty} \gamma_{xx}^*(-\tau) e^{-j2\pi F\tau} d\tau \\ &= \int_{-\infty}^{\infty} \gamma_{xx}(\tau) e^{-j2\pi F\tau} d\tau = \Gamma_{xx}(F)\end{aligned}$$

Therefore, $\Gamma_{xx}(F)$ is always real.

The definition of the power density spectrum can be extended to two jointly stationary random processes $X(t)$ and $Y(t)$, which have a crosscorrelation function $\gamma_{xy}(\tau)$. The Fourier transform of $\gamma_{xy}(\tau)$ is

$$\Gamma_{xy}(F) = \int_{-\infty}^{\infty} \gamma_{xy}(\tau) e^{-j2\pi F\tau} d\tau \quad (12.1.20)$$

which is called the *cross-power density spectrum*. It is easily shown that $\Gamma_{xy}^*(F) = \Gamma_{yx}(-F)$. For real random processes, the condition is $\Gamma_{yx}(F) = \Gamma_{xy}(-F)$.

12.1.6 Discrete-Time Random Signals

This characterization of continuous-time random signals can be easily carried over to discrete-time signals. Such signals are usually obtained by uniformly sampling a continuous-time random process.

A discrete-time random process $X(n)$ consists of an ensemble of sample sequences $x(n)$. The statistical properties of $X(n)$ are similar to the characterization of $X(t)$, with the restriction that n is now an integer (time) variable. To be specific, we state the form for the important moments that we use in this text.

The l th moment of $X(n)$ is defined as

$$E(X_n^l) = \int_{-\infty}^{\infty} x_n^l p(x_n) dx_n \quad (12.1.21)$$

and the autocorrelation sequence is

$$\gamma_{xx}(n, k) = E(X_n X_k) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x_n x_k p(x_n, x_k) dx_n dx_k \quad (12.1.22)$$

Similarly, the autocovariance is

$$c_{xx}(n, k) = \gamma_{xx}(n, k) - E(X_n)E(X_k) \quad (12.1.23)$$

For a stationary process, we have the special forms ($m = n - k$)

$$\begin{aligned}\gamma_{xx}(n - k) &= \gamma_{xx}(m) \\ c_{xx}(n - k) &= c_{xx}(m) = \gamma_{xx}(m) - m_x^2\end{aligned} \quad (12.1.24)$$

where $m_x = E(X_n)$ is the mean of the random process. The variance is defined as $\sigma^2 = c_{xx}(0) = \gamma_{xx}(0) - m_x^2$.

For a complex-valued stationary process $Z(n) = X(n) + jY(n)$, we have

$$\gamma_{zz}(m) = \gamma_{xx}(m) + \gamma_{yy}(m) + j[\gamma_{yx}(m) - \gamma_{xy}(m)] \quad (12.1.25)$$

and the crosscorrelation sequence of two complex-valued stationary sequences is

$$\gamma_{zw}(m) = \gamma_{xu}(m) + \gamma_{yu}(m) + j[\gamma_{yu}(m) - \gamma_{xv}(m)] \quad (12.1.26)$$

As in the case of a continuous-time random process, a discrete-time random process has infinite energy but a finite average power and is given as

$$E(X_n^2) = \gamma_{xx}(0) \quad (12.1.27)$$

By use of the Wiener-Khintchine theorem, we obtain the power density spectrum of the discrete-time random process by computing the Fourier transform of the autocorrelation sequence $\gamma_{xx}(m)$, that is,

$$\Gamma_{xx}(f) = \sum_{m=-\infty}^{\infty} \gamma_{xx}(m) e^{-j2\pi fm} \quad (12.1.28)$$

The inverse transform relationship is

$$\gamma_{xx}(m) = \int_{-1/2}^{1/2} \Gamma_{xx}(f) e^{j2\pi fm} df \quad (12.1.29)$$

We observe that the average power is

$$\gamma_{xx}(0) = \int_{-1/2}^{1/2} \Gamma_{xx}(f) df \quad (12.1.30)$$

so that $\Gamma_{xx}(f)$ is the distribution of power as a function of frequency, that is, $\Gamma_{xx}(f)$ is the power density spectrum of the random process $X(n)$. The properties we have stated for $\Gamma_{xx}(F)$ also hold for $\Gamma_{xx}(f)$.

12.1.7 Time Averages for a Discrete-Time Random Process

Although we have characterized a random process in terms of statistical averages, such as the mean and the autocorrelation sequence, in practice, we usually have available a single realization of the random process. Let us consider the problem of obtaining the averages of the random process from a single realization. To accomplish this, the random process must be *ergodic*.

By definition, a random process $X(n)$ is ergodic if, with probability 1, all the statistical averages can be determined from a single sample function of the process. In effect, the random process is ergodic if time averages obtained from a single

realization are equal to the statistical (ensemble) averages. Under this condition we can attempt to estimate the ensemble averages using time averages from a single realization.

To illustrate this point, let us consider the estimation of the mean and the autocorrelation of the random process from a single realization $x(n)$. Since we are interested only in these two moments, we define ergodicity with respect to these parameters. For additional details on the requirements for mean ergodicity and autocorrelation ergodicity which are given below, the reader is referred to the book of Papoulis (1984).

12.1.8 Mean-Ergodic Process

Given a stationary random process $X(n)$ with mean

$$m_x = E(X_n)$$

let us form the *time average*

$$\hat{m}_x = \frac{1}{2N+1} \sum_{n=-N}^N x(n) \quad (12.1.31)$$

In general, we view \hat{m}_x in (12.1.31) as an estimate of the statistical mean whose value will vary with the different realizations of the random process. Hence \hat{m}_x is a random variable with a PDF $p(\hat{m}_x)$. Let us compute the expected value of \hat{m}_x over all possible realizations of $X(n)$. Since the summation and the expectation are linear operations, we can interchange them, so that

$$E(\hat{m}_x) = \frac{1}{2N+1} \sum_{n=-N}^N E[x(n)] = \frac{1}{2N+1} \sum_{n=-N}^N m_x = m_x \quad (12.1.32)$$

Since the mean value of the estimate is equal to the statistical mean, we say that the estimate \hat{m}_x is unbiased.

Next, we compute the variance of \hat{m}_x . We have

$$\text{var}(\hat{m}_x) = E(|\hat{m}_x|^2) - |m_x|^2$$

But

$$\begin{aligned} E(|\hat{m}_x|^2) &= \frac{1}{(2N+1)^2} \sum_{n=-N}^N \sum_{k=-N}^N E[x^*(n)x(k)] \\ &= \frac{1}{(2N+1)^2} \sum_{n=-N}^N \sum_{k=-N}^N \gamma_{xx}(k-n) \\ &= \frac{1}{2N+1} \sum_{m=-2N}^{2N} \left(1 - \frac{|m|}{2N+1}\right) \gamma_{xx}(m) \end{aligned}$$

Therefore,

$$\begin{aligned}\text{var}(\hat{m}_x) &= \frac{1}{2N+1} \sum_{m=-2N}^{2N} \left(1 - \frac{|m|}{2N+1}\right) \gamma_{xx} - |\bar{m}_x|^2 \\ &= \frac{1}{2N+1} \sum_{m=-2N}^{2N} \left(1 - \frac{|m|}{2N+1}\right) c_{xx}(m)\end{aligned}\quad (12.1.33)$$

If $\text{var}(\hat{m}_x) \rightarrow 0$ as $N \rightarrow \infty$, the estimate converges with probability 1 to the statistical mean \bar{m}_x . Therefore, the process $X(n)$ is mean ergodic if

$$\lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{m=-2N}^{2N} \left(1 - \frac{|m|}{2N+1}\right) c_{xx}(m) = 0 \quad (12.1.34)$$

Under this condition, the estimate \hat{m}_x in the limit as $N \rightarrow \infty$ becomes equal to the statistical mean, that is,

$$\bar{m}_x = \lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^N x(n) \quad (12.1.35)$$

Thus the time-average mean, in the limit as $N \rightarrow \infty$, is equal to the ensemble mean.

A sufficient condition for (12.1.34) to hold is if

$$\sum_{m=-\infty}^{\infty} |c_{xx}(m)| < \infty \quad (12.1.36)$$

which implies that $c_{xx}(m) \rightarrow 0$ as $m \rightarrow \infty$. This condition holds for most zero-mean processes encountered in the physical world.

12.1.9 Correlation-Ergodic Processes

Now, let us consider the estimate of the autocorrelation $\gamma_{xx}(m)$ from a single realization of the process. Following our previous notation, we denote the estimate (for a complex-valued signal, in general) as

$$r_{xx}(m) = \frac{1}{2N+1} \sum_{n=-N}^N x^*(n)x(n+m) \quad (12.1.37)$$

Again, we regard $r_{xx}(m)$ as a random variable for any given lag m , since it is a function of the particular realization. The expected value (mean value over all realizations) is

$$\begin{aligned}E[r_{xx}(m)] &= \frac{1}{2N+1} \sum_{n=-N}^N E[x^*(n)x(n+m)] \\ &= \frac{1}{2N+1} \sum_{n=-N}^N \gamma_{xx}(m) = \gamma_{xx}(m)\end{aligned}\quad (12.1.38)$$

Therefore, the expected value of the time-average autocorrelation is equal to the statistical average. Hence we have an unbiased estimate of $\gamma_{xx}(m)$.

To determine the variance of the estimate $r_{xx}(m)$, we compute the expected value of $|r_{xx}(m)|^2$ and subtract the square of the mean value. Thus

$$\text{var}[r_{xx}(m)] = E[|r_{xx}(m)|^2] - |\gamma_{xx}(m)|^2 \quad (12.1.39)$$

But

$$E[|r_{xx}(m)|^2] = \frac{1}{(2N+1)^2} \sum_{n=-N}^N \sum_{k=-N}^N E[x^*(n)x(n+m)x(k)x^*(k+m)] \quad (12.1.40)$$

The expected value of the term $x^*(n)x(n+m)x(k)x^*(k+m)$ is just the autocorrelation sequence of a random process defined as

$$v_m(n) = x^*(n)x(n+m)$$

Hence (12.1.40) may be expressed as

$$\begin{aligned} E[|r_{xx}(m)|^2] &= \frac{1}{(2N+1)^2} \sum_{n=-N}^N \sum_{k=-N}^N \gamma_{vv}^{(m)}(n-k) \\ &= \frac{1}{2N+1} \sum_{n=-2N}^{2N} \left(1 - \frac{|n|}{2N+1}\right) \gamma_{vv}^{(m)}(n) \end{aligned} \quad (12.1.41)$$

and the variance is

$$\text{var}[r_{xx}(m)] = \frac{1}{2N+1} \sum_{n=-2N}^{2N} \left(1 - \frac{|n|}{2N+1}\right) \gamma_{vv}^{(m)}(n) - |\gamma_{xx}(m)|^2 \quad (12.1.42)$$

If $\text{var}[r_{xx}(m)] \rightarrow 0$ as $N \rightarrow \infty$, the estimate $r_{xx}(m)$ converges with probability 1 to the statistical autocorrelation $\gamma_{xx}(m)$. Under these conditions, the process is correlation ergodic and the time-average correlation is identical to the statistical average, that is,

$$\lim_{N \rightarrow \infty} \frac{1}{2N+1} \sum_{n=-N}^N x^*(n)x(n+m) = \gamma_{xx}(m) \quad (12.1.43)$$

In our treatment of random signals, we assume that the random processes are mean ergodic and correlation ergodic, so that we can deal with time averages of the mean and the autocorrelation obtained from a single realization of the process.

12.2 Innovations Representation of a Stationary Random Process

In this section we demonstrate that a wide-sense stationary random process can be represented as the output of a causal and causally invertible linear system excited by a white noise process. The condition that the system is causally invertible also allows us to represent the wide-sense stationary random process by the output of the inverse system, which is a white noise process.

Let us consider a wide-sense stationary process $\{x(n)\}$ with autocorrelation sequence $\{\gamma_{xx}(m)\}$ and power spectral density $\Gamma_{xx}(f)$, $|f| \leq \frac{1}{2}$. We assume that $\Gamma_{xx}(f)$ is real and continuous for all $|f| \leq \frac{1}{2}$. The z -transform of the autocorrelation sequence $\{\gamma_{xx}(m)\}$ is

$$\Gamma_{xx}(z) = \sum_{m=-\infty}^{\infty} \gamma_{xx}(m) z^{-m} \quad (12.2.1)$$

from which we obtain the power spectral density by evaluating $\Gamma_{xx}(z)$ on the unit circle [i.e., by substituting $z = \exp(j2\pi f)$].

Now, let us assume that $\log \Gamma_{xx}(z)$ is analytic (possesses derivatives of all orders) in an annular region in the z -plane that includes the unit circle (i.e., $r_1 < |z| < r_2$ where $r_1 < 1$ and $r_2 > 1$). Then, $\log \Gamma_{xx}(z)$ can be expanded in a Laurent series of the form

$$\log \Gamma_{xx}(z) = \sum_{m=-\infty}^{\infty} v(m) z^{-m} \quad (12.2.2)$$

where the $\{v(m)\}$ are the coefficients in the series expansion. We can view $\{v(m)\}$ as the sequence with z -transform $V(z) = \log \Gamma_{xx}(z)$. Equivalently, we can evaluate $\log \Gamma_{xx}(z)$ on the unit circle,

$$\log \Gamma_{xx}(f) = \sum_{m=-\infty}^{\infty} v(m) e^{-j2\pi fm} \quad (12.2.3)$$

so that the $\{v(m)\}$ are the Fourier coefficients in the Fourier series expansion of the periodic function $\log \Gamma_{xx}(f)$. Hence

$$v(m) = \int_{-\frac{1}{2}}^{\frac{1}{2}} [\log \Gamma_{xx}(f)] e^{j2\pi fm} df, \quad m = 0, \pm 1, \dots \quad (12.2.4)$$

We observe that $v(m) = v(-m)$, since $\Gamma_{xx}(f)$ is a real and even function of f .

From (12.2.2) it follows that

$$\begin{aligned} \Gamma_{xx}(z) &= \exp \left[\sum_{m=-\infty}^{\infty} v(m) z^{-m} \right] \\ &= \sigma_w^2 H(z) H(z^{-1}) \end{aligned} \quad (12.2.5)$$

where, by definition, $\sigma_w^2 = \exp[v(0)]$ and

$$H(z) = \exp \left[\sum_{m=1}^{\infty} v(m)z^{-m} \right], \quad |z| > r_1 \quad (12.2.6)$$

If (12.2.5) is evaluated on the unit circle, we have the equivalent representation of the power spectral density as

$$\Gamma_{xx}(f) = \sigma_w^2 |H(f)|^2 \quad (12.2.7)$$

We note that

$$\begin{aligned} \log \Gamma_{xx}(f) &= \log \sigma_w^2 + \log H(f) + \log H^*(f) \\ &= \sum_{m=-\infty}^{\infty} v(m)e^{-j2\pi fm} \end{aligned}$$

From the definition of $H(z)$ given by (12.2.6), it is clear that the causal part of the Fourier series in (12.2.3) is associated with $H(z)$ and the anticausal part is associated with $H(z^{-1})$. The Fourier series coefficients $\{v(m)\}$ are the *cepstral coefficients* and the sequence $\{v(m)\}$ is called the cepstrum of the sequence $\{\gamma_{xx}(m)\}$, as defined in Section 4.2.7.

The filter with system function $H(z)$ given by (12.2.6) is analytic in the region $|z| > r_1 < 1$. Hence, in this region, it has a Taylor series expansion as a causal system of the form

$$H(z) = \sum_{m=0}^{\infty} h(n)z^{-n} \quad (12.2.8)$$

The output of this filter in response to a white noise input sequence $w(n)$ with power spectral density σ_w^2 is a stationary random process $\{x(n)\}$ with power spectral density $\Gamma_{xx}(f) = \sigma_w^2 |H(f)|^2$. Conversely, the stationary random process $\{x(n)\}$ with power spectral density $\Gamma_{xx}(f)$ can be transformed into a white noise process by passing $\{x(n)\}$ through a linear filter with system function $1/H(z)$. We call this filter a noise *whitening filter*. Its output, denoted as $\{w(n)\}$ is called the *innovations process* associated with the stationary random process $\{x(n)\}$. These two relationships are illustrated in Fig. 12.2.1.

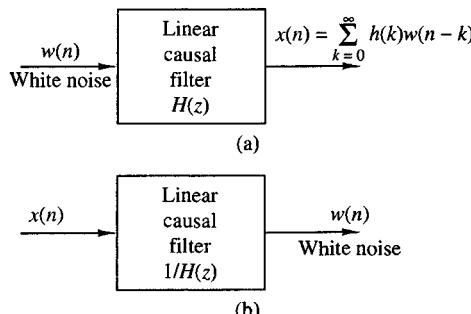


Figure 12.2.1
Filters for generating
(a) the random process
 $x(n)$ from white noise and
(b) the inverse filter.

The representation of the stationary stochastic process $\{x(n)\}$ as the output of an IIR filter with system function $H(z)$ given by (12.2.8) and excited by a white noise sequence $\{w(n)\}$ is called the *Wold representation*.

12.2.1 Rational Power Spectra

Let us now restrict our attention to the case where the power spectral density of the stationary random process $\{x(n)\}$ is a rational function, expressed as

$$\Gamma_{xx}(z) = \sigma_w^2 \frac{B(z)B(z^{-1})}{A(z)A(z^{-1})}, \quad r_1 < |z| < r_2 \quad (12.2.9)$$

where the polynomials $B(z)$ and $A(z)$ have roots that fall inside the unit circle in the z -plane. Then the linear filter $H(z)$ for generating the random process $\{x(n)\}$ from the white noise sequence $\{w(n)\}$ is also rational and is expressed as

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^q b_k z^{-k}}{1 + \sum_{k=1}^p a_k z^{-k}} \quad |z| > r_1 \quad (12.2.10)$$

where $\{b_k\}$ and $\{a_k\}$ are the filter coefficients that determine the location of the zeros and poles of $H(z)$, respectively. Thus $H(z)$ is causal, stable, and minimum phase. Its reciprocal $1/H(z)$ is also a causal, stable, and minimum-phase linear system. Therefore, the random process $\{x(n)\}$ uniquely represents the statistical properties of the innovations process $\{w(n)\}$, and vice versa.

For the linear system with the rational system function $H(z)$ given by (12.2.10), the output $x(n)$ is related to the input $w(n)$ by the difference equation

$$x(n) + \sum_{k=1}^p a_k x(n-k) = \sum_{k=0}^q b_k w(n-k) \quad (12.2.11)$$

We will distinguish among three specific cases.

Autoregressive (AR) process. $b_0 = 1, b_k = 0, k > 0$. In this case, the linear filter $H(z) = 1/A(z)$ is an all-pole filter and the difference equation for the input-output relationship is

$$x(n) + \sum_{k=1}^p a_k x(n-k) = w(n) \quad (12.2.12)$$

In turn, the noise-whitening filter for generating the innovations process is an all-zero filter.

Moving average (MA) process. $a_k = 0, k \geq 1$. In this case, the linear filter $H(z) = B(z)$ is an all-zero filter and the difference equation for the input–output relationship is

$$x(n) = \sum_{k=0}^q b_k w(n-k) \quad (12.2.13)$$

The noise-whitening filter for the MA process is an all-pole filter.

Autoregressive, moving average (ARMA) process. In this case, the linear filter $H(z) = B(z)/A(z)$ has both finite poles and zeros in the z -plane and the corresponding difference equation is given by (12.2.11). The inverse system for generating the innovations process from $x(n)$ is also a pole–zero system of the form $1/H(z) = A(z)/B(z)$.

12.2.2 Relationships Between the Filter Parameters and the Autocorrelation Sequence

When the power spectral density of the stationary random process is a rational function, there is a basic relationship between the autocorrelation sequence $\{\gamma_{xx}(m)\}$ and the parameters $\{a_k\}$ and $\{b_k\}$ of the linear filter $H(z)$ that generates the process by filtering the white noise sequence $w(n)$. This relationship can be obtained by multiplying the difference equation in (12.2.11) by $x^*(n-m)$ and taking the expected value of both sides of the resulting equation. Thus we have

$$\begin{aligned} E[x(n)x^*(n-m)] &= - \sum_{k=1}^p a_k E[x(n-k)x^*(n-m)] \\ &\quad + \sum_{k=0}^q b_k E[w(n-k)x^*(n-m)] \end{aligned} \quad (12.2.14)$$

Hence

$$\gamma_{xx}(m) = - \sum_{k=1}^p a_k \gamma_{xx}(m-k) + \sum_{k=0}^q b_k \gamma_{wx}(m-k) \quad (12.2.15)$$

where $\gamma_{wx}(m)$ is the crosscorrelation sequence between $w(n)$ and $x(n)$.

The crosscorrelation $\gamma_{wx}(m)$ is related to the filter impulse response. That is,

$$\begin{aligned} \gamma_{wx}(m) &= E[x^*(n)w(n+m)] \\ &= E \left[\sum_{k=0}^{\infty} h(k)w^*(n-k)w(n+m) \right] \\ &= \sigma_w^2 h(-m) \end{aligned} \quad (12.2.16)$$

where, in the last step, we have used the fact that the sequence $w(n)$ is white. Hence

$$\gamma_{wx}(m) = \begin{cases} 0, & m > 0 \\ \sigma_w^2 h(-m), & m \leq 0 \end{cases} \quad (12.2.17)$$

By combining (12.2.17) with (12.2.15), we obtain the desired relationship:

$$\gamma_{xx}(m) = \begin{cases} -\sum_{k=1}^p a_k \gamma_{xx}(m-k), & m > q \\ -\sum_{k=1}^p a_k \gamma_{xx}(m-k) + \sigma_w^2 \sum_{k=0}^{q-m} h(k) b_{k+m}, & 0 \leq m \leq q \\ \gamma_{xx}^*(-m), & m < 0 \end{cases} \quad (12.2.18)$$

This represents a nonlinear relationship between $\gamma_{xx}(m)$ and the parameters $\{a_k\}$, $\{b_k\}$.

The relationship in (12.2.18) applies, in general, to the ARMA process. For an AR process, (12.2.18) simplifies to

$$\gamma_{xx}(m) = \begin{cases} -\sum_{k=1}^p a_k \gamma_{xx}(m-k), & m > 0 \\ -\sum_{k=1}^p a_k \gamma_{xx}(m-k) + \sigma_w^2, & m = 0 \\ \gamma_{xx}^*(-m), & m < 0 \end{cases} \quad (12.2.19)$$

Thus we have a linear relationship between $\gamma_{xx}(m)$ and the $\{a_k\}$ parameters. These equations, called the *Yule-Walker* equations, can be expressed in the matrix form

$$\begin{bmatrix} \gamma_{xx}(0) & \gamma_{xx}(-1) & \gamma_{xx}(-2) & \cdots & \gamma_{xx}(-p) \\ \gamma_{xx}(1) & \gamma_{xx}(0) & \gamma_{xx}(-1) & \cdots & \gamma_{xx}(-p+1) \\ \vdots & \vdots & \vdots & & \vdots \\ \gamma_{xx}(p) & \gamma_{xx}(p-1) & \gamma_{xx}(p-2) & \cdots & \gamma_{xx}(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma_w^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (12.2.20)$$

This correlation matrix is Toeplitz, and hence it can be efficiently inverted by use of the algorithms described in Section 12.4.

Finally, by setting $a_k = 0$, $1 \leq k \leq p$, and $h(k) = b_k$, $0 \leq k \leq q$, in (12.2.18), we obtain the relationship for the autocorrelation sequence in the case of a MA process, namely,

$$\gamma_{xx}(m) = \begin{cases} \sigma_w^2 \sum_{k=0}^q b_k b_{k+m}, & 0 \leq m \leq q \\ 0, & m > q \\ \gamma_{xx}^*(-m), & m < 0 \end{cases} \quad (12.2.21)$$

12.3 Forward and Backward Linear Prediction

Linear prediction is an important topic in digital signal processing with many practical applications. In this section we consider the problem of linearly predicting the value of a stationary random process either forward in time or backward in time. This formulation leads to lattice filter structures and to some interesting connections to parametric signal models.

12.3.1 Forward Linear Prediction

Let us begin with the problem of predicting a future value of a stationary random process from observation of past values of the process. In particular, we consider the *one-step forward linear predictor*, which forms the prediction of the value $x(n)$ by a weighted linear combination of the past values $x(n-1), x(n-2), \dots, x(n-p)$. Hence the linearly predicted value of $x(n)$ is

$$\hat{x}(n) = - \sum_{k=1}^p a_p(k)x(n-k) \quad (12.3.1)$$

where the $\{-a_p(k)\}$ represent the weights in the linear combination. These weights are called the *prediction coefficients* of the one-step forward linear predictor of *order p*. The negative sign in the definition of $\hat{x}(n)$ is for mathematical convenience and conforms with current practice in the technical literature.

The difference between the value $x(n)$ and the predicted value $\hat{x}(n)$ is called the *forward prediction error*, denoted as $f_p(n)$:

$$\begin{aligned} f_p(n) &= x(n) - \hat{x}(n) \\ &= x(n) + \sum_{k=1}^p a_p(k)x(n-k) \end{aligned} \quad (12.3.2)$$

We view linear prediction as equivalent to linear filtering where the predictor is embedded in the linear filter, as shown in Fig. 12.3.1. This is called a *prediction-error filter* with input sequence $\{x(n)\}$ and output sequence $\{f_p(n)\}$. An equivalent realization for the prediction-error filter is shown in Fig. 12.3.2. This realization is a direct-form FIR filter with system function

$$A_p(z) = \sum_{k=0}^p a_p(k)z^{-k} \quad (12.3.3)$$

where, by definition, $a_p(0) = 1$.

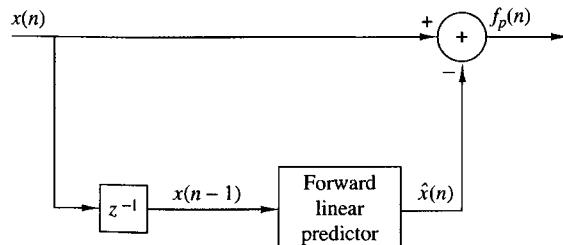


Figure 12.3.1
Forward linear prediction.

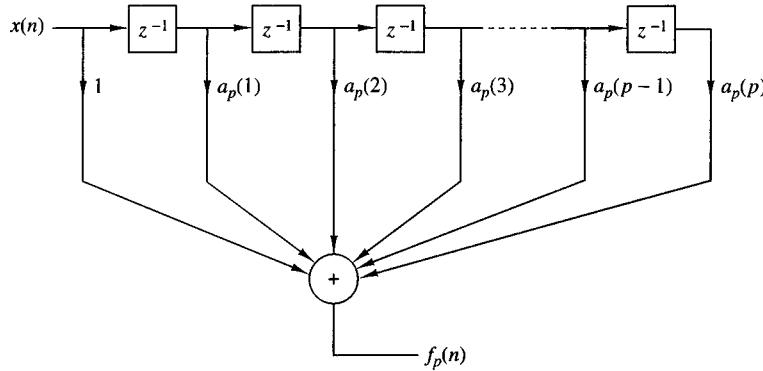


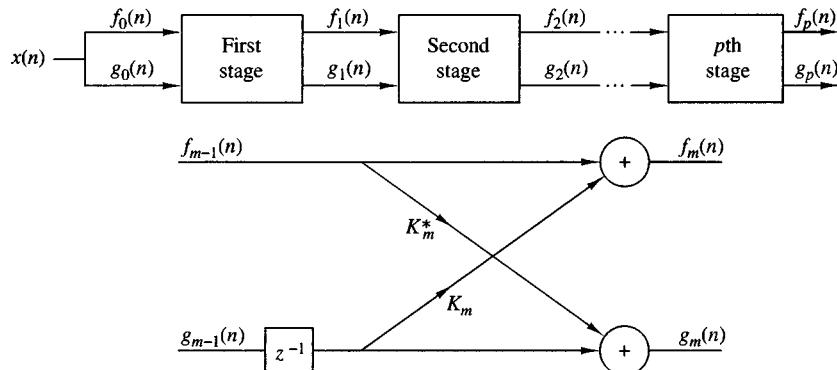
Figure 12.3.2 Prediction-error filter.

As shown in Section 9.2.4, the direct-form FIR filter is equivalent to an all-zero lattice filter. The lattice filter is generally described by the following set of *order-recursive equations*:

$$\begin{aligned} f_0(n) &= g_0(n) = x(n) \\ f_m(n) &= f_{m-1}(n) + K_m g_{m-1}(n-1) \quad m = 1, 2, \dots, p \\ g_m(n) &= K_m^* f_{m-1}(n) + g_{m-1}(n-1) \end{aligned} \quad (12.3.4)$$

where $\{K_m\}$ are the reflection coefficients and $g_m(n)$ is the backward prediction error defined in the following section. Note that for complex-valued data, the conjugate of K_m is used in the equation for $g_m(n)$. Figure 12.3.3 illustrates a p -stage lattice filter in block diagram form along with a typical stage showing the computations given by (12.3.4).

As a consequence of the equivalence between the direct-form prediction-error FIR filter and the FIR lattice filter, the output of the p -stage lattice filter is expressed

Figure 12.3.3 p -stage lattice filter.

as

$$f_p(n) = \sum_{k=0}^p a_p(k)x(n-k), \quad a_p(0) = 1 \quad (12.3.5)$$

Since (12.3.5) is a convolution sum, the z -transform relationship is

$$F_p(z) = A_p(z)X(z) \quad (12.3.6)$$

or, equivalently,

$$A_p(z) = \frac{F_p(z)}{X(z)} = \frac{F_p(z)}{F_0(z)} \quad (12.3.7)$$

The mean-square value of the forward linear prediction error $f_p(n)$ is

$$\begin{aligned} \mathcal{E}_p^f &= E[|f_p(n)|^2] \\ &= \gamma_{xx}(0) + 2\Re \left[\sum_{k=1}^p a_p^*(k)\gamma_{xx}(k) \right] + \sum_{k=1}^p \sum_{l=1}^p a_p^*(l)a_p(k)\gamma_{xx}(l-k) \end{aligned} \quad (12.3.8)$$

\mathcal{E}_p^f is a quadratic function of the predictor coefficients and its minimization leads to the set of linear equations

$$\gamma_{xx}(l) = - \sum_{k=1}^p a_p(k)\gamma_{xx}(l-k), \quad l = 1, 2, \dots, p \quad (12.3.9)$$

These are called the *normal equations* for the coefficients of the linear predictor. The minimum mean-square prediction error is simply

$$\min[\mathcal{E}_p^f] \equiv E_p^f = \gamma_{xx}(0) + \sum_{k=1}^p a_p(k)\gamma_{xx}(-k) \quad (12.3.10)$$

In the following section we extend the development above to the problem of predicting the value of a time series in the opposite direction, namely, backward in time.

12.3.2 Backward Linear Prediction

Let us assume that we have the data sequence $x(n), x(n-1), \dots, x(n-p+1)$ from a stationary random process and we wish to predict the value $x(n-p)$ of the process. In this case we employ a *one-step backward linear predictor* of order p . Hence

$$\hat{x}(n-p) = - \sum_{k=0}^{p-1} b_p(k)x(n-k) \quad (12.3.11)$$

The difference between the value $x(n - p)$ and the estimate $\hat{x}(n - p)$ is called the *backward prediction error*, denoted as $g_p(n)$:

$$\begin{aligned} g_p(n) &= x(n - p) + \sum_{k=0}^{p-1} b_p(k)x(n - k) \\ &= \sum_{k=0}^p b_p(k)x(n - k), \quad b_p(p) = 1 \end{aligned} \tag{12.3.12}$$

The backward linear predictor can be realized either by a direct-form FIR filter structure similar to the structure shown in Fig. 12.3.1 or as a lattice structure. The lattice structure shown in Fig. 12.3.3 provides the backward linear predictor as well as the forward linear predictor.

The weighting coefficients in the backward linear predictor are the complex conjugates of the coefficients for the forward linear predictor, but they occur in reverse order. Thus we have

$$b_p(k) = a_p^*(p - k), \quad k = 0, 1, \dots, p \tag{12.3.13}$$

In the z -domain, the convolution sum in (12.3.12) becomes

$$G_p(z) = B_p(z)X(z) \tag{12.3.14}$$

or, equivalently,

$$B_p(z) = \frac{G_p(z)}{X(z)} = \frac{G_p(z)}{G_0(z)} \tag{12.3.15}$$

where $B_p(z)$ represents the system function of the FIR filter with coefficients $b_p(k)$.

Since $b_p(k) = a_p^*(p - k)$, $G_p(z)$ is related to $A_p(z)$

$$\begin{aligned} B_p(z) &= \sum_{k=0}^p b_p(k)z^{-k} \\ &= \sum_{k=0}^p a_p^*(p - k)z^{-k} \\ &= z^{-p} \sum_{k=0}^p a_p^*(k)z^k \\ &= z^{-p} A_p^*(z^{-1}) \end{aligned} \tag{12.3.16}$$

The relationship in (12.3.16) implies that the zeros of the FIR filter with system function $B_p(z)$ are simply the (conjugate) reciprocals of the zeros of $A_p(z)$. Hence $B_p(z)$ is called the reciprocal or *reverse polynomial* of $A_p(z)$.

Now that we have established these interesting relationships between the forward and backward one-step predictors, let us return to the recursive lattice equations in (12.3.4) and transform them to the z -domain. Thus we have

$$\begin{aligned} F_0(z) &= G_0(z) = X(z) \\ F_m(z) &= F_{m-1}(z) + K_m z^{-1} G_{m-1}(z), \quad m = 1, 2, \dots, p \\ G_m(z) &= K_m^* F_{m-1}(z) + z^{-1} G_{m-1}(z), \quad m = 1, 2, \dots, p \end{aligned} \quad (12.3.17)$$

If we divide each equation by $X(z)$, we obtain the desired results in the form

$$\begin{aligned} A_0(z) &= B_0(z) = 1 \\ A_m(z) &= A_{m-1}(z) + K_m z^{-1} B_{m-1}(z), \quad m = 1, 2, \dots, p \\ B_m(z) &= K_m^* A_{m-1}(z) + z^{-1} B_{m-1}(z), \quad m = 1, 2, \dots, p \end{aligned} \quad (12.3.18)$$

Thus a lattice filter is described in the z -domain by the matrix equation

$$\begin{bmatrix} A_m(z) \\ B_m(z) \end{bmatrix} = \begin{bmatrix} 1 & K_m z^{-1} \\ K_m^* & z^{-1} \end{bmatrix} \begin{bmatrix} A_{m-1}(z) \\ B_{m-1}(z) \end{bmatrix} \quad (12.3.19)$$

The relations in (12.3.18) for $A_m(z)$ and $B_m(z)$ allow us to obtain the direct-form FIR filter coefficients $\{a_m(k)\}$ from the reflection coefficients $\{K_m\}$, and vice versa. These relationships were given in Section 9.2.4 by (9.2.51) through (9.2.53).

The lattice structure with parameters K_1, K_2, \dots, K_p corresponds to a class of p direct-form FIR filters with system functions $A_1(z), A_2(z), \dots, A_p(z)$. It is interesting to note that a characterization of this class of p FIR filters in direct form requires $p(p+1)/2$ filter coefficients. In contrast, the lattice-form characterization requires only the p reflection coefficients $\{K_i\}$. The reason the lattice provides a more compact representation for the class of p FIR filters is because appending stages to the lattice does not alter the parameters of the previous stages. On the other hand, appending the p th stage to a lattice with $(p-1)$ stages is equivalent to increasing the length of an FIR filter by one coefficient. The resulting FIR filter with system function $A_p(z)$ has coefficients totally different from the coefficients of the lower-order FIR filter with system function $A_{p-1}(z)$.

The formula for determining the filter coefficients $\{a_p(k)\}$ recursively is easily derived from polynomial relationships (12.3.18). We have

$$\begin{aligned} A_m(z) &= A_{m-1}(z) + K_m z^{-1} B_{m-1}(z) \\ \sum_{k=0}^m a_m(k) z^{-k} &= \sum_{k=0}^{m-1} a_{m-1}(k) z^{-k} + K_m \sum_{k=0}^{m-1} a_{m-1}^*(m-1-k) z^{-(k+1)} \end{aligned} \quad (12.3.20)$$

By equating the coefficients of equal powers of z^{-1} and recalling that $a_m(0) = 1$ for $m = 1, 2, \dots, p$, we obtain the desired recursive equation for the FIR filter

coefficients in the form

$$\begin{aligned} a_m(0) &= 1 \\ a_m(m) &= K_m \\ a_m(k) &= a_{m-1}(k) + K_m a_{m-1}^*(m-k) \\ &= a_{m-1}(k) + a_m(m) a_{m-1}^*(m-k), \quad \begin{matrix} 1 \leq k \leq m-1 \\ m = 1, 2, \dots, p \end{matrix} \end{aligned} \quad (12.3.21)$$

The conversion formula from the direct-form FIR filter coefficients $\{a_p(k)\}$ to the lattice reflection coefficients $\{K_i\}$ is also very simple. For the p -stage lattice we immediately obtain the reflection coefficient $K_p = a_p(p)$. To obtain K_{p-1}, \dots, K_1 , we need the polynomials $A_m(z)$ for $m = p-1, \dots, 1$. From (12.3.19) we obtain

$$A_{m-1}(z) = \frac{A_m(z) - K_m B_m(z)}{1 - |K_m|^2}, \quad m = p, \dots, 1 \quad (12.3.22)$$

which is just a step-down recursion. Thus we compute all lower-degree polynomials $A_m(z)$ beginning with $A_{p-1}(z)$ and obtain the desired lattice reflection coefficients from the relation $K_m = a_m(m)$. We observe that the procedure works as long as $|K_m| \neq 1$ for $m = 1, 2, \dots, p-1$. From this step-down recursion for the polynomials, it is relatively easy to obtain a formula for recursively and directly computing K_m , $m = p-1, \dots, 1$. For $m = p-1, \dots, 1$, we have

$$\begin{aligned} K_m &= a_m(m) \\ a_{m-1}(k) &= \frac{a_m(k) - K_m b_m(k)}{1 - |K_m|^2} \\ &= \frac{a_m(k) - a_m(m) a_m^*(m-k)}{1 - |a_m(m)|^2} \end{aligned} \quad (12.3.23)$$

which is just the recursion in the Schur–Cohn stability test for the polynomial $A_m(z)$.

As just indicated, the recursive equation in (12.3.23) breaks down if any of the lattice parameters $|K_m| = 1$. If this occurs, it is indicative that the polynomial $A_{m-1}(z)$ has a root located on the unit circle. Such a root can be factored out from $A_{m-1}(z)$ and the iterative process in (12.3.23) carried out for the reduced-order system.

Finally, let us consider the minimization of the mean-square error in a backward linear predictor. The backward prediction error is

$$\begin{aligned} g_p(n) &= x(n-p) + \sum_{k=0}^{p-1} b_p(k) x(n-k) \\ &= x(n-p) + \sum_{k=1}^p a_p^*(k) x(n-p+k) \end{aligned} \quad (12.3.24)$$

and its mean-square value is

$$\mathcal{E}_p^b = E[|g_p(n)|^2] \quad (12.3.25)$$

The minimization of \mathcal{E}_p^b with respect to the prediction coefficients yields the same set of linear equations as in (12.3.9). Hence the minimum mean-square error (MSE) is

$$\min[\mathcal{E}_p^b] \equiv E_p^b = E_p^f \quad (12.3.26)$$

which is given by (12.3.10).

12.3.3 The Optimum Reflection Coefficients for the Lattice Forward and Backward Predictors

In Sections 12.3.1 and 12.3.2 we derived the set of linear equations which provide the predictor coefficients that minimize the mean-square value of the prediction error. In this section we consider the problem of optimizing the reflection coefficients in the lattice predictor and expressing the reflection coefficients in terms of the forward and backward prediction errors.

The forward prediction error in the lattice filter is expressed as

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1) \quad (12.3.27)$$

The minimization of $E[|f_m(n)|^2]$ with respect to the reflection coefficient K_m yields the result

$$K_m = \frac{-E[f_{m-1}(n)g_{m-1}^*(n-1)]}{E[|g_{m-1}(n-1)|^2]} \quad (12.3.28)$$

or, equivalently,

$$K_m = \frac{-E[f_{m-1}(n)g_{m-1}^*(n-1)]}{\sqrt{E_{m-1}^f E_{m-1}^b}} \quad (12.3.29)$$

where $E_{m-1}^f = E_{m-1}^b = E[|g_{m-1}(n-1)|^2] = E[|f_{m-1}(n)|^2]$.

We observe that the optimum choice of the reflection coefficients in the lattice predictor is the negative of the (normalized) crosscorrelation coefficients between the forward and backward errors in the lattice.¹ Since it is apparent from (12.3.28) that $|K_m| \leq 1$, it follows that the minimum mean-square value of the prediction error, which can be expressed recursively as

$$E_m^f = (1 - |K_m|^2)E_{m-1}^f \quad (12.3.30)$$

is a monotonically decreasing sequence.

¹The normalized crosscorrelation coefficients between the forward and backward error in the lattice (i.e., $\{-K_m\}$) are often called the partial correlation (PARCOR) coefficients.

12.3.4 Relationship of an AR Process to Linear Prediction

The parameters of an $\text{AR}(p)$ process are intimately related to a predictor of order p for the same process. To see the relationship, we recall that in an $\text{AR}(p)$ process, the autocorrelation sequence $\{\gamma_{xx}(m)\}$ is related to the parameters $\{a_k\}$ by the Yule–Walker equations given in (12.2.19) or (12.2.20). The corresponding equations for the predictor of order p are given by (12.3.9) and (12.3.10).

A direct comparison of these two sets of relations reveals that there is a one-to-one correspondence between the parameters $\{a_k\}$ of the $\text{AR}(p)$ process and the predictor coefficients $\{a_p(k)\}$ of the p th-order predictor. In fact, if the underlying process $\{x(n)\}$ is $\text{AR}(p)$, the prediction coefficients of the p th-order predictor are identical to $\{a_k\}$. Furthermore, the minimum mean-square error (MMSE) in the p th-order predictor E_p^f is identical to σ_w^2 , the variance of the white noise process. In this case, the prediction-error filter is a noise-whitening filter which produces the innovations sequence $\{w(n)\}$.

12.4 Solution of the Normal Equations

In the preceding section we observed that the minimization of the mean-square value of the forward prediction error resulted in a set of linear equations for the coefficients of the predictor given by (12.3.9). These equations, called the **normal equations**, may be expressed in the compact form

$$\sum_{k=0}^p a_p(k) \gamma_{xx}(l-k) = 0 \quad l = 1, 2, \dots, p \quad (12.4.1)$$

$$a_p(0) = 1$$

The resulting minimum MSE (MMSE) is given by (12.3.10). If we augment (12.3.10) to the normal equations given by (12.4.1) we obtain the set of *augmented normal equations*, which may be expressed as

$$\sum_{k=0}^p a_p(k) \gamma_{xx}(l-k) = \begin{cases} E_p^f, & l = 0 \\ 0, & l = 1, 2, \dots, p \end{cases} \quad (12.4.2)$$

We also noted that if the random process is an $\text{AR}(p)$ process, the MMSE $E_p^f = \sigma_w^2$.

In this section we describe two computationally efficient algorithms for solving the normal equations. One algorithm, originally due to Levinson (1947) and modified by Durbin (1959), is called the Levinson–Durbin algorithm. This algorithm is suitable for serial processing and has a computation complexity of $O(p^2)$. The second algorithm, due to Schur (1917), also computes the reflection coefficients in $O(p^2)$ operations but with parallel processors, the computations can be performed in $O(p)$ time. Both algorithms exploit the Toeplitz symmetry property inherent in the autocorrelation matrix.

We begin by describing the Levinson–Durbin algorithm.

12.4.1 The Levinson–Durbin Algorithm

The Levinson–Durbin algorithm is a computationally efficient algorithm for solving the normal equations in (12.4.1) for the prediction coefficients. This algorithm exploits the special symmetry in the autocorrelation matrix

$$\Gamma_p = \begin{bmatrix} \gamma_{xx}(0) & \gamma_{xx}^*(1) & \cdots & \gamma_{xx}^*(p-1) \\ \gamma_{xx}(1) & \gamma_{xx}(0) & \cdots & \gamma_{xx}^*(p-2) \\ \vdots & \vdots & & \\ \gamma_{xx}(p-1) & \gamma_{xx}(p-2) & \cdots & \gamma_{xx}(0) \end{bmatrix} \quad (12.4.3)$$

Note that $\Gamma_p(i, j) = \Gamma_p(i - j)$, so that the autocorrelation matrix is a *Toeplitz matrix*. Since $\Gamma_p(i, j) = \Gamma_p^*(j, i)$, the matrix is also Hermitian.

The key to the Levinson–Durbin method of solution, which exploits the Toeplitz property of the matrix, is to proceed recursively, beginning with a predictor of order $m = 1$ (one coefficient), and then to increase the order recursively, using the lower-order solutions to obtain the solution to the next-higher order. Thus the solution to the first-order predictor obtained by solving (12.4.1) is

$$a_1(1) = -\frac{\gamma_{xx}(1)}{\gamma_{xx}(0)} \quad (12.4.4)$$

and the resulting MMSE is

$$\begin{aligned} E_1^f &= \gamma_{xx}(0) + a_1(1)\gamma_{xx}(-1) \\ &= \gamma_{xx}(0)[1 - |a_1(1)|^2] \end{aligned} \quad (12.4.5)$$

Recall that $a_1(1) = K_1$, the first reflection coefficient in the lattice filter.

The next step is to solve for the coefficients $\{a_2(1), a_2(2)\}$ of the second-order predictor and express the solution in terms of $a_1(1)$. The two equations obtained from (12.4.1) are

$$\begin{aligned} a_2(1)\gamma_{xx}(0) + a_2(2)\gamma_{xx}^*(1) &= -\gamma_{xx}(1) \\ a_2(1)\gamma_{xx}(1) + a_2(2)\gamma_{xx}(0) &= -\gamma_{xx}(2) \end{aligned} \quad (12.4.6)$$

By using the solution in (12.4.4) to eliminate $\gamma_{xx}(1)$, we obtain the solution

$$\begin{aligned} a_2(2) &= -\frac{\gamma_{xx}(2) + a_1(1)\gamma_{xx}(1)}{\gamma_{xx}(0)[1 - |a_1(1)|^2]} \\ &= -\frac{\gamma_{xx}(2) + a_1(1)\gamma_{xx}(1)}{E_1^f} \\ a_2(1) &= a_1(1) + a_2(2)a_1^*(1) \end{aligned} \quad (12.4.7)$$

Thus we have obtained the coefficients of the second-order predictor. Again, we note that $a_2(2) = K_2$, the second reflection coefficient in the lattice filter.

Proceeding in this manner, we can express the coefficients of the m th-order predictor in terms of the coefficients of the $(m - 1)$ st-order predictor. Thus we can write the coefficient vector \mathbf{a}_m as the sum of two vectors, namely,

$$\mathbf{a}_m = \begin{bmatrix} a_m(1) \\ a_m(2) \\ \vdots \\ a_m(m) \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{m-1} \\ \dots \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{d}_{m-1} \\ \dots \\ K_m \end{bmatrix} \quad (12.4.8)$$

where \mathbf{a}_{m-1} is the predictor coefficient vector of the $(m - 1)$ st-order predictor and the vector \mathbf{d}_{m-1} and the scalar K_m are to be determined. Let us also partition the $m \times m$ autocorrelation matrix Γ_{xx} as

$$\boldsymbol{\Gamma}_m = \begin{bmatrix} \boldsymbol{\Gamma}_{m-1} & \boldsymbol{\gamma}_{m-1}^{b*} \\ \boldsymbol{\gamma}_{m-1}^b & \gamma_{xx}(0) \end{bmatrix} \quad (12.4.9)$$

where $\boldsymbol{\gamma}_{m-1}^b = [\gamma_{xx}(m - 1) \ \gamma_{xx}(m - 2) \ \dots \ \gamma_{xx}(1)] = (\boldsymbol{\gamma}_{m-1}^b)^t$, the asterisk (*) denotes the complex conjugate, and $\boldsymbol{\gamma}_m^t$ denotes the transpose of $\boldsymbol{\gamma}_m$. The superscript b on $\boldsymbol{\gamma}_{m-1}$ denotes the vector $\boldsymbol{\gamma}_{m-1}^t = [\gamma_{xx}(1) \ \gamma_{xx}(2) \ \dots \ \gamma_{xx}(m - 1)]$ with elements taken in reverse order.

The solution to the equation $\boldsymbol{\Gamma}_m \mathbf{a}_m = -\boldsymbol{\gamma}_m$ can be expressed as

$$\begin{bmatrix} \boldsymbol{\Gamma}_{m-1} & \boldsymbol{\gamma}_{m-1}^{b*} \\ \boldsymbol{\gamma}_{m-1}^b & \gamma_{xx}(0) \end{bmatrix} \left\{ \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{d}_{m-1} \\ K_m \end{bmatrix} \right\} = -\begin{bmatrix} \boldsymbol{\gamma}_{m-1} \\ \gamma_{xx}(m) \end{bmatrix} \quad (12.4.10)$$

This is the key step in the Levinson–Durbin algorithm. From (12.4.10) we obtain two equations, namely,

$$\boldsymbol{\Gamma}_{m-1} \mathbf{a}_{m-1} + \boldsymbol{\Gamma}_{m-1} \mathbf{d}_{m-1} + K_m \boldsymbol{\gamma}_{m-1}^{b*} = -\boldsymbol{\gamma}_{m-1} \quad (12.4.11)$$

$$\boldsymbol{\gamma}_{m-1}^b \mathbf{a}_{m-1} + \boldsymbol{\gamma}_{m-1}^{b*} \mathbf{d}_{m-1} + K_m \gamma_{xx}(0) = -\gamma_{xx}(m) \quad (12.4.12)$$

Since $\boldsymbol{\Gamma}_{m-1} \mathbf{a}_{m-1} = -\boldsymbol{\gamma}_{m-1}$, (12.4.11) yields the solution

$$\mathbf{d}_{m-1} = -K_m \boldsymbol{\Gamma}_{m-1}^{-1} \boldsymbol{\gamma}_{m-1}^{b*} \quad (12.4.13)$$

But $\boldsymbol{\gamma}_{m-1}^{b*}$ is just $\boldsymbol{\gamma}_{m-1}$ with elements taken in reverse order and conjugated. Therefore, the solution in (12.4.13) is simply

$$\mathbf{d}_{m-1} = K_m \mathbf{a}_{m-1}^{b*} = K_m \begin{bmatrix} a_{m-1}^{*(m-1)} \\ a_{m-1}^{*(m-2)} \\ \vdots \\ a_{m-1}^{*(1)} \end{bmatrix} \quad (12.4.14)$$

The scalar equation (12.4.12) can now be used to solve for K_m . If we eliminate \mathbf{d}_{m-1} in (12.4.12) by using (12.4.14), we obtain

$$K_m [\gamma_{xx}(0) + \boldsymbol{\gamma}_{m-1}^{b*} \mathbf{a}_{m-1}^{b*}] + \boldsymbol{\gamma}_{m-1}^b \mathbf{a}_{m-1} = -\gamma_{xx}(m)$$

Hence

$$K_m = -\frac{\gamma_{xx}(m) + \gamma_{m-1}^{bt} \mathbf{a}_{m-1}}{\gamma_{xx}(0) + \gamma_{m-1}^{bt} \mathbf{a}_{m-1}^{b*}} \quad (12.4.15)$$

Therefore, by substituting the solutions in (12.4.14) and (12.4.15) into (12.4.8), we obtain the desired recursion for the predictor coefficients in the Levinson–Durbin algorithm as

$$a_m(m) = K_m = -\frac{\gamma_{xx}(m) + \gamma_{m-1}^{bt} \mathbf{a}_{m-1}}{\gamma_{xx}(0) + \gamma_{m-1}^{bt} \mathbf{a}_{m-1}^{b*}} = -\frac{\gamma_{xx}(m) + \gamma_{m-1}^{bt} \mathbf{a}_{m-1}}{E_{m-1}^f} \quad (12.4.16)$$

$$\begin{aligned} a_m(k) &= a_{m-1}(k) + K_m a_{m-1}^*(m-k) \\ &= a_{m-1}(k) + a_m(m) a_{m-1}^*(m-k), \quad k = 1, 2, \dots, m-1 \\ &\quad m = 1, 2, \dots, p \end{aligned} \quad (12.4.17)$$

The reader should note that the recursive relation in (12.4.17) is identical to the recursive relation in (12.3.21) for the predictor coefficients, obtained from the polynomials $A_m(z)$ and $B_m(z)$. Furthermore, K_m is the reflection coefficient in the m th stage of the lattice predictor. This development clearly illustrates that the Levinson–Durbin algorithm produces the reflection coefficients for the optimum lattice prediction filter as well as the coefficients of the optimum direct-form FIR predictor.

Finally, let us determine the expression for the MMSE. For the m th-order predictor, we have

$$\begin{aligned} E_m^f &= \gamma_{xx}(0) + \sum_{k=1}^m a_m(k) \gamma_{xx}(-k) \\ &= \gamma_{xx}(0) + \sum_{k=1}^m [a_{m-1}(k) + a_m(m) a_{m-1}^*(m-k)] \gamma_{xx}(-k) \\ &= E_{m-1}^f [1 - |a_m(m)|^2] = E_{m-1}^f (1 - |K_m|^2), \quad m = 1, 2, \dots, p \end{aligned} \quad (12.4.18)$$

where $E_0^f = \gamma_{xx}(0)$. Since the reflection coefficients satisfy the property that $|K_m| \leq 1$, the MMSE for the sequence of predictors satisfies the condition

$$E_0^f \geq E_1^f \geq E_2^f \geq \dots \geq E_p^f \quad (12.4.19)$$

This concludes the derivation of the Levinson–Durbin algorithm for solving the linear equations $\Gamma_m \mathbf{a}_m = -\gamma_m$, for $m = 0, 1, \dots, p$. We observe that the linear equations have the special property that the vector on the right-hand side also appears as a vector in Γ_m . In the more general case, where the vector on the right-hand side is some other vector, say \mathbf{c}_m , the set of linear equations can be solved recursively by introducing a second recursive equation to solve the more general linear

equations $\Gamma_m \mathbf{b}_m = \mathbf{c}_m$. The result is a *generalized Levinson–Durbin* algorithm (see Problem 12.12).

The Levinson–Durbin recursion given by (12.4.17) requires $O(m)$ multiplications and additions (operations) to go from stage m to stage $m + 1$. Therefore, for p stages it takes on the order of $1 + 2 + 3 + \dots + p(p + 1)/2$, or $O(p^2)$, operations to solve for the prediction filter coefficients, or the reflection coefficients, compared with $O(p^3)$ operations if we did not exploit the Toeplitz property of the correlation matrix.

If the Levinson–Durbin algorithm is implemented on a serial computer or signal processor, the required computation time is on the order of $O(p^2)$ time units. On the other hand, if the processing is performed on a parallel processing machine utilizing as many processors as necessary to exploit the full parallelism in the algorithm, the multiplications as well as the additions required to compute (12.4.17) can be carried out simultaneously. Therefore, this computation can be performed in $O(p)$ time units. However, the computation in (12.4.16) for the reflection coefficients takes additional time. Certainly, the inner products involving the vectors \mathbf{a}_{m-1} and γ_{m-1}^b can be computed simultaneously by employing parallel processors. However, the addition of these products cannot be done simultaneously, but instead, requires $O(\log p)$ time units. Hence, the computations in the Levinson–Durbin algorithm, when performed by p parallel processors, can be accomplished in $O(p \log p)$ time.

In the following section we describe another algorithm, due to Schur (1917), that avoids the computation of inner products, and therefore is more suitable for parallel computation of the reflection coefficients.

12.4.2 The Schur Algorithm

The Schur algorithm is intimately related to a recursive test for determining the positive definiteness of a correlation matrix. To be specific, let us consider the auto-correlation matrix Γ_{p+1} associated with the augmented normal equations given by (12.4.2). From the elements of this matrix we form the function

$$R_0(z) = \frac{\gamma_{xx}(1)z^{-1} + \gamma_{xx}(2)z^{-2} + \dots + \gamma_{xx}(p)z^{-p}}{\gamma_{xx}(0) + \gamma_{xx}(1)z^{-1} + \dots + \gamma_{xx}(p)z^{-p}} \quad (12.4.20)$$

and the sequence of functions $R_m(z)$ defined recursively as

$$R_m(z) = \frac{R_{m-1}(z) - R_{m-1}(\infty)}{z^{-1}[1 - R_{m-1}^*(\infty)R_{m-1}(z)]}, \quad m = 1, 2, \dots \quad (12.4.21)$$

Schur's theorem states that a necessary and sufficient condition for the correlation matrix to be positive definite is that $|R_m(\infty)| < 1$ for $m = 1, 2, \dots, p$.

Let us demonstrate that the condition for positive definiteness of the autocorrelation matrix Γ_{p+1} is equivalent to the condition that the reflection coefficients in the equivalent lattice filter satisfy the condition $|K_m| < 1$, $m = 1, 2, \dots, p$.

First, we note that $R_0(\infty) = 0$. Then, from (12.4.21) we have

$$R_1(z) = \frac{\gamma_{xx}(1) + \gamma_{xx}(2)z^{-1} + \dots + \gamma_{xx}(p)z^{-p+1}}{\gamma_{xx}(0) + \gamma_{xx}(1)z^{-1} + \dots + \gamma_{xx}(p)z^{-p}} \quad (12.4.22)$$

Hence $R_1(\infty) = \gamma_{xx}(1)/\gamma_{xx}(0)$. We observe that $R_1(\infty) = -K_1$.

Second, we compute $R_2(z)$ according to (12.4.21) and evaluate the result at $z = \infty$. Thus we obtain

$$R_2(\infty) = \frac{\gamma_{xx}(2) + K_1\gamma_{xx}(1)}{\gamma_{xx}(0)(1 - |K_1|^2)}$$

Again, we observe that $R_2(\infty) = -K_2$. By continuing this procedure, we find that $R_m(\infty) = -K_m$, for $m = 1, 2, \dots, p$. Hence the condition $|R_m(\infty)| < 1$ for $m = 1, 2, \dots, p$, is identical to the condition $|K_m| < 1$ for $m = 1, 2, \dots, p$, and ensures the positive definiteness of the autocorrelation matrix Γ_{p+1} .

Since the reflection coefficients can be obtained from the sequence of functions $R_m(z)$, $m = 1, 2, \dots, p$, we have another method for solving the normal equations. We call this method the *Schur algorithm*.

Schur algorithm. Let us first rewrite $R_m(z)$ as

$$R_m(z) = \frac{P_m(z)}{Q_m(z)}, \quad m = 0, 1, \dots, p \quad (12.4.23)$$

where

$$\begin{aligned} P_0(z) &= \gamma_{xx}(1)z^{-1} + \gamma_{xx}(2)z^{-2} + \cdots + \gamma_{xx}(p)z^{-p} \\ Q_0(z) &= \gamma_{xx}(0) + \gamma_{xx}(1)z^{-1} + \cdots + \gamma_{xx}(p)z^{-p} \end{aligned} \quad (12.4.24)$$

Since $K_0 = 0$ and $K_m = -R_m(\infty)$ for $m = 1, 2, \dots, p$, the recursive equation (12.4.21) implies the following recursive equations for the polynomials $P_m(z)$ and $Q_m(z)$:

$$\begin{bmatrix} P_m(z) \\ Q_m(z) \end{bmatrix} = \begin{bmatrix} 1 & K_{m-1} \\ K_{m-1}^* z^{-1} & z^{-1} \end{bmatrix} \begin{bmatrix} P_{m-1}(z) \\ Q_{m-1}(z) \end{bmatrix}, \quad m = 1, 2, \dots, p \quad (12.4.25)$$

Thus we have

$$\begin{aligned} P_1(z) &= P_0(z) = \gamma_{xx}(1)z^{-1} + \gamma_{xx}(2)z^{-2} + \cdots + \gamma_{xx}(p)z^{-p} \\ Q_1(z) &= z^{-1}Q_0(z) = \gamma_{xx}(0)z^{-1} + \gamma_{xx}(1)z^{-2} + \cdots + \gamma_{xx}(p)z^{-p-1} \end{aligned} \quad (12.4.26)$$

and

$$K_1 = -\left. \frac{P_1(z)}{Q_1(z)} \right|_{z=\infty} = -\frac{\gamma_{xx}(1)}{\gamma_{xx}(0)} \quad (12.4.27)$$

Next the reflection coefficient K_2 is obtained by determining $P_2(z)$ and $Q_2(z)$ from (12.4.25), dividing $P_2(z)$ by $Q_2(z)$ and evaluating the result at $z = \infty$. Thus we find

that

$$\begin{aligned}
 P_2(z) &= P_1(z) + K_1 Q_1(z) \\
 &= [\gamma_{xx}(2) + K_1 \gamma_{xx}(1)]z^{-2} + \dots \\
 &\quad + [\gamma_{xx}(p) + K_1 \gamma_{xx}(p-1)]z^{-p} \\
 Q_2(z) &= z^{-1}[Q_1(z) + K_1^* P_1(z)] \\
 &= [\gamma_{xx}(0) + K_1^* \gamma_{xx}(1)]z^{-2} + \dots \\
 &\quad + [\gamma_{xx}(p-2) + K_1^* \gamma_{xx}(p-1)]z^{-p}
 \end{aligned} \tag{12.4.28}$$

where the terms involving z^{-p-1} have been dropped. Thus we observe that the recursive equation in (12.4.25) is equivalent to (12.4.21).

Based on these relationships, the Schur algorithm is described by the following recursive procedure.

Initialization. Form the $2x(p+1)$ generator matrix

$$\mathbf{G}_0 = \begin{bmatrix} 0 & \gamma_{xx}(1) & \gamma_{xx}(2) & \cdots & \gamma_{xx}(p) \\ \gamma_{xx}(0) & \gamma_{xx}(1) & \gamma_{xx}(2) & \cdots & \gamma_{xx}(p) \end{bmatrix} \tag{12.4.29}$$

where the elements of the first row are the coefficients of $P_0(z)$ and the elements of the second row are the coefficients of $Q_0(z)$.

Step 1. Shift the second row of the generator matrix to the right by one place and discard the last element of this row. A zero is placed in the vacant position. Thus we obtain a new generator matrix,

$$\mathbf{G}_1 = \begin{bmatrix} 0 & \gamma_{xx}(1) & \gamma_{xx}(1) & \gamma_{xx}(2) & \cdots & \gamma_{xx}(p) \\ 0 & \gamma_{xx}(0) & \gamma_{xx}(1) & \gamma_{xx}(1) & \cdots & \gamma_{xx}(p-1) \end{bmatrix} \tag{12.4.30}$$

The (negative) ratio of the elements in the second column yields the reflection coefficient $K_1 = -\gamma_{xx}(1)/\gamma_{xx}(0)$.

Step 2. Multiply the generator matrix by the 2×2 matrix

$$\mathbf{V}_1 = \begin{bmatrix} 1 & K_1 \\ K_1^* & 1 \end{bmatrix} \tag{12.4.31}$$

Thus we obtain

$$\mathbf{V}_1 \mathbf{G}_1 = \begin{bmatrix} 0 & 0 & \gamma_{xx}(2) + K_1 \gamma_{xx}(1) & \gamma_{xx}(p) + K_1 \gamma_{xx}(p-1) \\ 0 & \gamma_{xx}(0) + K_1^* \gamma_{xx}(1) & \cdots & \gamma_{xx}(p-1) + K_1^* \gamma_{xx}(p) \end{bmatrix} \tag{12.4.32}$$

Step 3. Shift the second row of $\mathbf{V}_1 \mathbf{G}_1$ by one place to the right and thus form the new generator matrix

$$\mathbf{G}_2 = \begin{bmatrix} 0 & 0 & \gamma_{xx}(2) + K_1 \gamma_{xx}(1) & \cdots & \gamma_{xx}(p) + K_1 \gamma_{xx}(p-1) \\ 0 & 0 & \gamma_{xx}(0) + K_1^* \gamma_{xx}(1) & \cdots & \gamma_{xx}(p-2) + K_1^* \gamma_{xx}(p-1) \end{bmatrix} \tag{12.4.33}$$

The negative ratio of the elements in the third column of G_2 yields K_2 .

Steps 2 and 3 are repeated until we have solved for all p reflection coefficients. In general, the 2×2 matrix in Step 2 is

$$\mathbf{V}_m = \begin{bmatrix} 1 & K_m \\ K_m^* & 1 \end{bmatrix} \quad (12.4.34)$$

and multiplication of \mathbf{V}_m by \mathbf{G}_m yields $\mathbf{V}_m \mathbf{G}_m$. In Step 3 we shift the second row of $\mathbf{V}_m \mathbf{G}_m$ one place to the right and obtain the new generator matrix \mathbf{G}_{m+1} .

We observe that the shifting operation of the second row in each iteration is equivalent to multiplication by the delay operator z^{-1} in the second recursive equation in (12.4.25). We also note that the division of the polynomial $P_m(z)$ by the polynomial $Q_m(z)$ and the evaluation of the quotient at $z = \infty$ is equivalent to dividing the elements in the $(m + 1)$ st column of \mathbf{G}_m . The computation of the p reflection coefficients can be accomplished by use of parallel processors in $O(p)$ time units. Below we describe a pipelined architecture for performing these computations.

Another way of demonstrating the relationship of the Schur algorithm to the Levinson–Durbin algorithm and the corresponding lattice predictor is to determine the output of the lattice filter obtained when the input sequence is the correlation sequence $\{\gamma_{xx}(m), m = 0, 1, \dots\}$. Thus, the first input to the lattice filter is $\gamma_{xx}(0)$, the second input is $\gamma_{xx}(1)$, and so on [i.e., $f_0(n) = \gamma_{xx}(n)$]. After the delay in the first stage, we have $g_0(n - 1) = \gamma_{xx}(n - 1)$. Hence, for $n = 1$, the ratio $f_0(1)/g_0(0) = \gamma_{xx}(1)/\gamma_{xx}(0)$, which is the negative of the reflection coefficient K_1 . Alternatively, we can express this relationship as

$$f_0(1) + K_1 g_0(0) = \gamma_{xx}(1) + K_1 \gamma_{xx}(0) = 0$$

Furthermore, $g_0(0) = \gamma_{xx}(0) = E_0^f$. At time $n = 2$, the input to the second stage is, according to (12.3.4),

$$f_1(2) = f_0(2) + K_1 g_0(1) = \gamma_{xx}(2) + K_1 \gamma_{xx}(1)$$

and after the unit of delay in the second stage, we have

$$g_1(1) = K_1^* f_0(1) + g_0(0) = K_1^* \gamma_{xx}(1) + \gamma_{xx}(0)$$

Now the ratio $f_1(2)/g_1(1)$ is

$$\frac{f_1(2)}{g_1(1)} = \frac{\gamma_{xx}(2) + K_1 \gamma_{xx}(1)}{\gamma_{xx}(0) + K_1^* \gamma_{xx}(1)} = \frac{\gamma_{xx}(2) + K_1 \gamma_{xx}(1)}{E_1^f} = -K_2$$

Hence

$$f_1(2) + K_2 g_1(1) = 0$$

$$g_1(1) = E_1^f$$

By continuing in this manner, we can show that at the input to the m th lattice stage, the ratio $f_{m-1}(m)/g_{m-1}(m-1) = -K_m$ and $g_{m-1}(m-1) = E_{m-1}^f$. Consequently, the lattice filter coefficients obtained from the Levinson algorithm are identical to the coefficients obtained in the Schur algorithm. Furthermore, the lattice filter structure provides a method for computing the reflection coefficients in the lattice predictor.

A pipelined architecture for implementing the Schur algorithm. Kung and Hu (1983) developed a pipelined lattice-type processor for implementing the Schur algorithm. The processor consists of a cascade of p lattice-type stages, where each stage consists of two processing elements (PEs), which we designate as upper PEs denoted as A_1, A_2, \dots, A_p , and lower PEs denoted as B_1, B_2, \dots, B_p , as shown in Fig. 12.4.1. The PE designated as A_1 is assigned the task of performing divisions. The remaining PEs perform one multiplication and one addition per iteration (one clock cycle).

Initially, the upper PEs are loaded with the elements of the first row of the generator matrix \mathbf{G}_0 , as illustrated in Fig. 12.4.1. The lower PEs are loaded with the elements of the second row of the generator matrix \mathbf{G}_0 . The computational process begins with the division PE, A_1 , which computes the first reflection coefficient as $K_1 = -\gamma_{xx}(1)/\gamma_{xx}(0)$. The value of K_1 is sent simultaneously to all the PEs in the upper branch and lower branch.

The second step in the computation is to update the contents of all processing elements simultaneously. The contents of the upper and lower PEs are updated as follows:

$$\begin{aligned} \text{PE } A_m: A_m &\leftarrow A_m + K_1 B_m, \quad m = 2, 3, \dots, p \\ \text{PE } B_m: B_m &\leftarrow B_m + K_1^* A_m, \quad m = 1, 2, \dots, p \end{aligned}$$

The third step involves the shifting of the contents of the upper PEs one place to the left. Thus we have

$$\text{PE } A_m: A_{m-1} \leftarrow A_m, \quad m = 2, 3, \dots, p$$

At this point, PE A_1 contains $\gamma_{xx}(2) + K_1 \gamma_{xx}(1)$ while PE B_1 contains $\gamma_{xx}(0) + K_1^* \gamma_{xx}(1)$. Hence the processor A_1 is ready to begin the second cycle by computing the second reflection coefficient $K_2 = -A_1/B_1$. The three computational steps beginning with the division A_1/B_1 are repeated until all p reflection coefficients are computed. Note that PE B_1 provides the minimum mean-square error E_m^f for each iteration.

If τ_d denotes the time for PE A_1 to perform a (complex) division and τ_{ma} is the time required to perform a (complex) multiplication and an addition, the time required to compute all p reflection coefficients is $p(\tau_d + \tau_{ma})$ for the Schur algorithm.

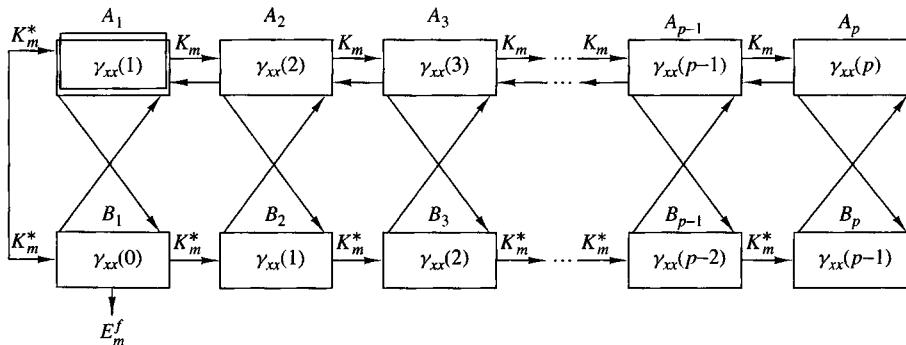


Figure 12.4.1 Pipelined parallel processor for computing the reflection coefficients.

12.5 Properties of the Linear Prediction-Error Filters

Linear prediction filters possess several important properties, which we now describe. We begin by demonstrating that the forward prediction-error filter is minimum phase.

Minimum-phase property of the forward prediction-error filter. We have already demonstrated that the reflection coefficients $\{K_i\}$ are correlation coefficients, and consequently, $|K_i| \leq 1$ for all i . This condition and the relation $E_m^f = (1 - |K_m|^2)E_{m-1}^f$ can be used to show that the zeros of the prediction-error filter are either all inside the unit circle or they are all on the unit circle.

First, we show that if $E_p^f > 0$, the zeros $|z_i| < 1$ for every i . The proof is by induction. Clearly, for $p = 1$ the system function for the prediction-error filter is

$$A_1(z) = 1 + K_1 z^{-1} \quad (12.5.1)$$

Hence $z_1 = -K_1$ and $E_1^f = (1 - |K_1|^2)E_0^f > 0$. Now, suppose that the hypothesis is true for $p - 1$. Then, if z_i is a root of $A_p(z)$, we have from (12.3.16) and (12.3.18),

$$\begin{aligned} A_p(z_i) &= A_{p-1}(z_i) + K_p z_i^{-1} B_{p-1}(z_i) \\ &= A_{p-1}(z_i) + K_p z_i^{-p} A_{p-1}^*(\frac{1}{z_i}) = 0 \end{aligned} \quad (12.5.2)$$

Hence

$$\frac{1}{K_p} = -\frac{z_i^{-p} A_{p-1}^*(1/z_i)}{A_{p-1}(z_i)} \equiv Q(z_i) \quad (12.5.3)$$

We note that the function $Q(z)$ is all pass. In general, an all-pass function of the form

$$P(z) = \prod_{k=1}^N \frac{zz_k^* + 1}{z + z_k}, \quad |z_k| < 1 \quad (12.5.4)$$

satisfies the property that $|P(z)| > 1$ for $|z| < 1$, $|P(z)| = 1$ for $|z| = 1$, and $|P(z)| < 1$ for $|z| > 1$. Since $Q(z) = -P(z)/z$, it follows that $|z_i| < 1$ if $|Q(z)| > 1$. Clearly, this is the case since $Q(z_i) = 1/K_p$ and $E_p^f > 0$.

On the other hand, suppose that $E_{p-1}^f > 0$ and $E_p^f = 0$. In this case $|K_p| = 1$ and $|Q(z_i)| = 1$. Since the MMSE is zero, the random process $x(n)$ is called *predictable* or *deterministic*. Specifically, a purely sinusoidal random process of the form

$$x(n) = \sum_{k=1}^M \alpha_k e^{j(n\omega_k + \theta_k)} \quad (12.5.5)$$

where the phases $\{\theta_k\}$ are statistically independent and uniformly distributed over $(0, 2\pi)$, has the autocorrelation

$$\gamma_{xx}(m) = \sum_{k=1}^M \alpha_k^2 e^{jm\omega_k} \quad (12.5.6)$$

and the power density spectrum

$$\Gamma_{xx}(f) = \sum_{k=1}^M \alpha_k^2 \delta(f - f_k), \quad f_k = \frac{\omega_k}{2\pi} \quad (12.5.7)$$

This process is predictable with a predictor of order $p \geq M$.

To demonstrate the validity of the statement, consider passing this process through a prediction error filter of order $p \geq M$. The MSE at the output of this filter is

$$\begin{aligned} \mathcal{E}_p^f &= \int_{-1/2}^{1/2} \Gamma_{xx}(f) |A_p(f)|^2 df \\ &= \int_{-1/2}^{1/2} \left[\sum_{k=1}^M \alpha_k^2 \delta(f - f_k) \right] |A_p(f)|^2 df \\ &= \sum_{k=1}^M \alpha_k^2 |A_p(f_k)|^2 \end{aligned} \quad (12.5.8)$$

By choosing M of the p zeros of the prediction-error filter to coincide with the frequencies $\{f_k\}$, the MSE \mathcal{E}_p^f can be forced to zero. The remaining $p - M$ zeros can be selected arbitrarily to be anywhere inside the unit circle.

Finally, the reader can prove that if a random process consists of a mixture of a continuous power spectral density and a discrete spectrum, the prediction-error filter must have all its roots inside the unit circle.

Maximum-phase property of the backward prediction-error filter. The system function for the backward prediction-error filter of order p is

$$B_p(z) = z^{-p} A_p^*(z^{-1}) \quad (12.5.9)$$

Consequently, the roots of $B_p(z)$ are the reciprocals of the roots of the forward prediction-error filter with system function $A_p(z)$. Hence if $A_p(z)$ is minimum phase, then $B_p(z)$ is maximum phase. However, if the process $x(n)$ is predictable, all the roots of $B_p(z)$ lie on the unit circle.

Whitening property. Suppose that the random process $x(n)$ is an AR(p) stationary random process that is generated by passing white noise with variance σ_w^2 through an all-pole filter with system function

$$H(z) = \frac{1}{1 + \sum_{k=1}^p a_k z^{-k}} \quad (12.5.10)$$

Then the prediction-error filter of order p has the system function

$$A_p(z) = 1 + \sum_{k=1}^p a_p(k)z^{-k} \quad (12.5.11)$$

where the predictor coefficients $a_p(k) = a_k$. The response of the prediction-error filter is a white noise sequence $\{w(n)\}$. In this case the prediction-error filter whitens the input random process $x(n)$ and is called a whitening filter, as indicated in Section 12.3.4.

More generally, even if the input process $x(n)$ is not an AR process, the prediction-error filter attempts to remove the correlation among the signal samples of the input process. As the order of the predictor is increased, the predictor output $\hat{x}(n)$ becomes a closer approximation to $x(n)$ and hence the difference $f(n) = \hat{x}(n) - x(n)$ approaches a white noise sequence.

Orthogonality of the backward prediction errors. The backward prediction errors $\{g_m(k)\}$ from different stages in the FIR lattice filter are orthogonal. That is,

$$E[g_m(n)g_l^*(n)] = \begin{cases} 0, & 0 \leq l \leq m-1 \\ E_m^b, & l = m \end{cases} \quad (12.5.12)$$

This property is easily proved by substituting for $g_m(n)$ and $g_l^*(n)$ into (12.5.12) and carrying out the expectation. Thus

$$\begin{aligned} E[g_m(n)g_l^*(n)] &= \sum_{k=0}^m b_m(k) \sum_{j=0}^l b_l^*(j) E[x(n-k)x^*(n-j)] \\ &= \sum_{j=0}^l b_l^*(j) \sum_{k=0}^m b_m(k) \gamma_{xx}(j-k) \end{aligned} \quad (12.5.13)$$

But the normal equations for the backward linear predictor require that

$$\sum_{k=0}^m b_m(k) \gamma_{xx}(j-k) = \begin{cases} 0, & j = 1, 2, \dots, m-1 \\ E_m^b, & j = m \end{cases} \quad (12.5.14)$$

Therefore,

$$E[g_m(n)g_l^*(n)] = \begin{cases} E_m^b = E_m^f, & m = l \\ 0, & 0 \leq l \leq m-1 \end{cases} \quad (12.5.15)$$

Additional properties. There are a number of other interesting properties regarding the forward and backward prediction errors in the FIR lattice filter. These are given here for real-valued data. Their proof is left as an exercise for the reader.

- (a) $E[f_m(n)x(n-i)] = 0, \quad 1 \leq i \leq m$
- (b) $E[g_m(n)x(n-i)] = 0, \quad 0 \leq i \leq m-1$
- (c) $E[f_m(n)x(n)] = E[g_m(n)x(n-m)] = E_m$
- (d) $E[f_i(n)f_j(n)] = E_{\max}(i, j)$
- (e) $E[f_i(n)f_j(n-t)] = 0, \text{ for } \begin{cases} 1 \leq t \leq i-j, & i > j \\ -1 \geq t \geq i-j, & i < j \end{cases}$
- (f) $E[g_i(n)g_j(n-t)] = 0, \text{ for } \begin{cases} 0 \leq t \leq i-j, & i > j \\ 0 \geq t \geq i-j+1, & i < j \end{cases}$
- (g) $E[f_i(n+i)f_j(n+j)] = \begin{cases} E_i, & i = j \\ 0, & i \neq j \end{cases}$
- (h) $E[g_i(n+i)g_j(n+j)] = E_{\max}(i, j)$
- (i) $E[f_i(n)g_j(n)] = \begin{cases} K_j E_i, & i \geq j, \quad i, j \geq 0, \\ 0, & i < j \end{cases} \quad K_0 = 1$
- (j) $E[f_i(n)g_i(n-1)] = -K_{i+1}E_i$
- (k) $E[g_i(n-1)x(n)] = E[f_i(n+1)x(n-1)] = -K_{i+1}E_i$
- (l) $E[f_i(n)g_j(n-1)] = \begin{cases} 0, & i > j \\ -K_{j+1}E_i, & i \leq j \end{cases}$

12.6 AR Lattice and ARMA Lattice-Ladder Filters

In Section 12.3 we showed the relationship of the all-zero FIR lattice to linear prediction. The linear predictor with transfer function,

$$A_p(z) = 1 + \sum_{k=1}^p a_p(k)z^{-k} \quad (12.6.1)$$

when excited by an input random process $\{x(n)\}$, produces an output that approaches a white noise sequence as $p \rightarrow \infty$. On the other hand, if the input process is an AR(p), the output of $A_p(z)$ is white. Since $A_p(z)$ generates an MA(p) process when excited with a white noise sequence, the all-zero lattice is sometimes called an MA lattice.

In the following section, we develop the lattice structure for the inverse filter $1/A_p(z)$, called the AR lattice, and the lattice-ladder structure for an ARMA process.

12.6.1 AR Lattice Structure

Let us consider an all-pole system with system function

$$H(z) = \frac{1}{1 + \sum_{k=1}^p a_p(k)z^{-k}} \quad (12.6.2)$$

The difference equation for this IIR system is

$$y(n) = -\sum_{k=1}^p a_p(k)y(n-k) + x(n) \quad (12.6.3)$$

Now suppose that we interchange the roles of the input and output [i.e., interchange $x(n)$ with $y(n)$ in (12.6.3)], obtaining the difference equation

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

or, equivalently,

$$y(n) = x(n) + \sum_{k=1}^p a_p(k)x(n-k) \quad (12.6.4)$$

We observe that (12.6.4) is a difference equation for an FIR system with system function $A_p(z)$. Thus an all-pole IIR system can be converted to an all-zero system by interchanging the roles of the input and output.

Based on this observation, we can obtain the structure of an AR(p) lattice from an MA(p) lattice by interchanging the input with the output. Since the MA(p) lattice has $y(n) = f_p(n)$ as its output and $x(n) = f_0(n)$ is the input, we let

$$\begin{aligned} x(n) &= f_p(n) \\ y(n) &= f_0(n) \end{aligned} \quad (12.6.5)$$

These definitions dictate that the quantities $\{f_m(n)\}$ be computed in descending order. This computation can be accomplished by rearranging the recursive equation for $\{f_m(n)\}$ in (12.3.4) and solving for $f_{m-1}(n)$ in terms of $f_m(n)$. Thus we obtain

$$f_{m-1}(n) = f_m(n) - K_m g_{m-1}(n-1), \quad m = p, p-1, \dots, 1$$

The equation for $g_m(n)$ remains unchanged. The result of these changes is the set of equations

$$\begin{aligned} x(n) &= f_p(n) \\ f_{m-1}(n) &= f_m(n) - K_m g_{m-1}(n-1) \\ g_m(n) &= K_m^* f_{m-1}(n) + g_{m-1}(n-1) \\ y(n) &= f_0(n) = g_0(n) \end{aligned} \quad (12.6.6)$$

The corresponding structure for the AR(p) lattice is shown in Fig. 12.6.1. Note that the all-pole lattice structure has an all-zero path with input $g_0(n)$ and output $g_p(n)$, which is identical to the all-zero path in the MA(p) lattice structure. This is not surprising, since the equation for $g_m(n)$ is identical in the two lattice structures.

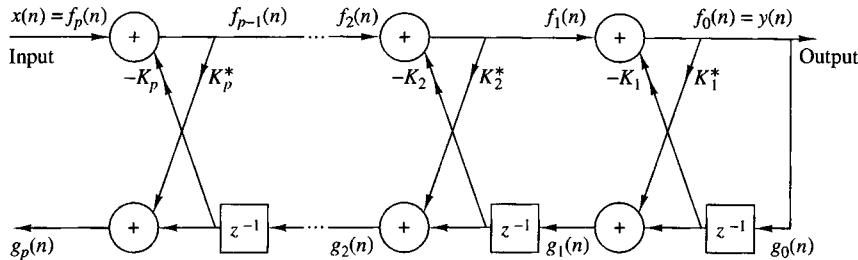


Figure 12.6.1 Lattice structure for an all-pole system.

We also observe that the AR(p) and MA(p) lattice structures are characterized by the same parameters, namely, the reflection coefficients $\{K_i\}$. Consequently, the equations given in (12.3.21) and (12.3.23) for converting between the system parameters $\{a_p(k)\}$ in the direct-form realizations of the all-zero system $A_p(z)$ and the lattice parameters $\{K_i\}$ of the MA(p) lattice structure apply as well to the all-pole structures.

12.6.2 ARMA Processes and Lattice-Ladder Filters

The all-pole lattice provides the basic building block for lattice-type structures that implement IIR systems that contain both poles and zeros. To construct the appropriate structure, let us consider an IIR system with system function

$$H(z) = \frac{\sum_{k=0}^q c_q(k)z^{-k}}{1 + \sum_{k=1}^p a_p(k)z^{-k}} = \frac{C_q(z)}{A_p(z)} \quad (12.6.7)$$

Without loss of generality, we assume that $p \ge q$.

This system is described by the difference equations

$$\begin{aligned} v(n) &= - \sum_{k=1}^p a_p(k)v(n-k) + x(n) \\ y(n) &= \sum_{k=0}^q c_q(k)v(n-k) \end{aligned} \quad (12.6.8)$$

obtained by viewing the system as a cascade of an all-pole system followed by an all-zero system. From (12.6.8) we observe that the output $y(n)$ is simply a linear combination of delayed outputs from the all-pole system.

Since zeros result from forming a linear combination of previous outputs, we can carry over this observation to construct a pole-zero system using the all-pole lattice structure as the basic building block. We have clearly observed that $g_m(n)$ in the

all-pole lattice can be expressed as a linear combination of present and past outputs. In fact, the system

$$H_b(z) \equiv \frac{G_m(z)}{Y(z)} = B_m(z) \quad (12.6.9)$$

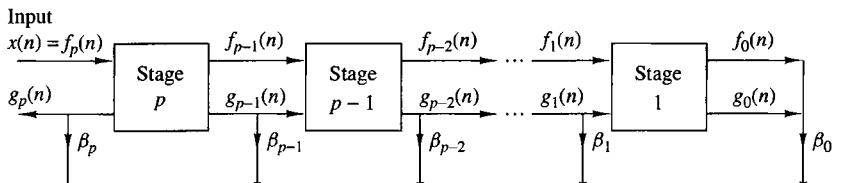
is an all-zero system. Therefore, any linear combination of $\{g_m(n)\}$ is also an all-zero filter.

Let us begin with an all-pole lattice filter with coefficients K_m , $1 \leq m \leq p$, and add a *ladder* part by taking as the output a weighted linear combination of $\{g_m(n)\}$. The result is a pole–zero filter that has the *lattice-ladder* structure shown in Fig. 12.6.2. Its output is

$$y(n) = \sum_{k=0}^q \beta_k g_k(n) \quad (12.6.10)$$

where $\{\beta_k\}$ are the parameters that determine the zeros of the system. The system function corresponding to (12.6.10) is

$$H(z) = \frac{Y(z)}{X(z)} = \sum_{k=0}^q \beta_k \frac{G_k(z)}{X(z)} \quad (12.6.11)$$



(a) Pole–zero system

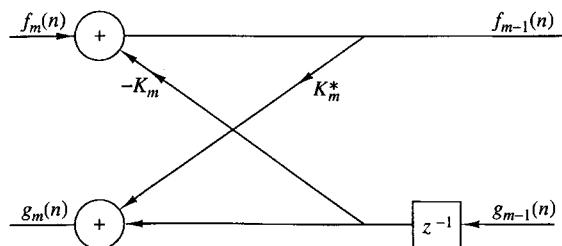
(b) m th stage of lattice

Figure 12.6.2 Lattice-ladder structure for pole–zero system.

Since $X(z) = F_p(z)$ and $F_0(z) = G_0(z)$, (12.6.11), can be expressed as

$$\begin{aligned} H(z) &= \sum_{k=0}^q \beta_k \frac{G_k(z)}{G_0(z)} \frac{F_0(z)}{F_p(z)} \\ &= \frac{1}{A_p(z)} \sum_{k=0}^q \beta_k B_k(z) \end{aligned} \quad (12.6.12)$$

Therefore,

$$C_q(z) = \sum_{k=0}^q \beta_k B_k(z) \quad (12.6.13)$$

This is the desired relationship that can be used to determine the weighting coefficients $\{\beta_k\}$ as previously shown in Section 9.3.5.

Given the polynomials $C_q(z)$ and $A_p(z)$, where $p \geq q$, the reflection coefficients $\{K_i\}$ are determined first from the coefficients $\{a_p(k)\}$. By means of the step-down recursive relation given by (12.3.22), we also obtain the polynomials $B_k(z)$, $k = 1, 2, \dots, p$. Then the ladder parameters can be obtained from (12.6.13), which can be expressed as

$$\begin{aligned} C_m(z) &= \sum_{k=0}^{m-1} \beta_k B_k(z) + \beta_m B_m(z) \\ &= C_{m-1}(z) + \beta_m B_m(z) \end{aligned} \quad (12.6.14)$$

or, equivalently,

$$C_{m-1}(z) = C_m(z) - \beta_m B_m(z), \quad m = p, p-1, \dots, 1 \quad (12.6.15)$$

By running this recursive relation backward, we can generate all the lower-degree polynomials, $C_m(z)$, $m = p-1, \dots, 1$. Since $b_m(m) = 1$, the parameters β_m are determined from (12.6.15) by setting

$$\beta_m = c_m(m), \quad m = p, p-1, \dots, 1, 0$$

When excited by a white noise sequence, this lattice-ladder filter structure generates an ARMA(p, q) process that has a power density spectrum

$$\Gamma_{xx}(f) = \sigma_w^2 \frac{|C_q(f)|^2}{|A_p(f)|^2} \quad (12.6.16)$$

and an autocorrelation function that satisfies (12.2.18), where σ_w^2 is the variance of the input white noise sequence.

12.7 Wiener Filters for Filtering and Prediction

In many practical applications we are given an input signal $\{x(n)\}$, consisting of the sum of a desired signal $\{s(n)\}$ and an undesired noise or interference $\{w(n)\}$, and we are asked to design a filter that suppresses the undesired interference component. In such a case, the objective is to design a system that filters out the additive interference while preserving the characteristics of the desired signal $\{s(n)\}$.

In this section we treat the problem of signal estimation in the presence of an additive noise disturbance. The estimator is constrained to be a linear filter with impulse response $\{h(n)\}$, designed so that its output approximates some specified desired signal sequence $\{d(n)\}$. Figure 12.7.1 illustrates the linear estimation problem.

The input sequence to the filter is $x(n) = s(n) + w(n)$, and its output sequence is $y(n)$. The difference between the desired signal and the filter output is the error sequence $e(n) = d(n) - y(n)$.

We distinguish three special cases:

1. If $d(n) = s(n)$, the linear estimation problem is referred to as *filtering*.
2. If $d(n) = s(n + D)$, where $D > 0$, the linear estimation problem is referred to as signal *prediction*. Note that this problem is different than the prediction considered earlier in this chapter, where $d(n) = x(n + D)$, $D \geq 0$.
3. If $d(n) = s(n - D)$, where $D > 0$, the linear estimation problem is referred to as signal *smoothing*.

Our treatment will concentrate on filtering and prediction.

The criterion selected for optimizing the filter impulse response $\{h(n)\}$ is the minimization of the mean-square error. This criterion has the advantages of simplicity and mathematical tractability.

The basic assumptions are that the sequences $\{s(n)\}$, $\{w(n)\}$, and $\{d(n)\}$ are zero mean and wide-sense stationary. The linear filter will be assumed to be either FIR or IIR. If it is IIR, we assume that the input data $\{x(n)\}$ are available over the infinite past. We begin with the design of the optimum FIR filter. The optimum linear filter, in the sense of minimum mean-square error (MMSE), is called a *Wiener filter*.

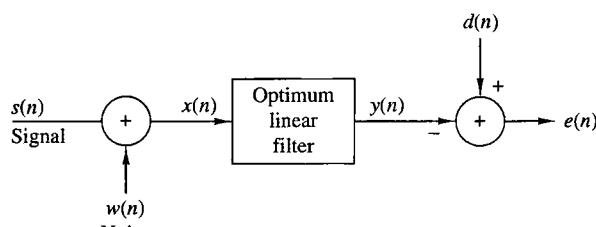


Figure 12.7.1
Model for linear estimation problem.

12.7.1 FIR Wiener Filter

Suppose that the filter is constrained to be of length M with coefficients $\{h_k, 0 \leq k \leq M - 1\}$. Hence its output $y(n)$ depends on the finite data record $x(n), x(n - 1), \dots, x(n - M + 1)$,

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (12.7.1)$$

The mean-square value of the error between the desired output $d(n)$ and $y(n)$ is

$$\begin{aligned} \mathcal{E}_M &= E|e(n)|^2 \\ &= E \left| d(n) - \sum_{k=0}^{M-1} h(k)x(n-k) \right|^2 \end{aligned} \quad (12.7.2)$$

Since this is a quadratic function of the filter coefficients, the minimization of \mathcal{E}_M yields the set of linear equations

$$\sum_{k=0}^{M-1} h(k)\gamma_{xx}(l-k) = \gamma_{dx}(l), \quad l = 0, 1, \dots, M-1 \quad (12.7.3)$$

where $\gamma_{xx}(k)$ is the autocorrelation of the input sequence $\{x(n)\}$ and $\gamma_{dx}(k) = E[d(n)x^*(n-k)]$ is the crosscorrelation between the desired sequence $\{d(n)\}$ and the input sequence $\{x(n), 0 \leq n \leq M-1\}$. The set of linear equations that specify the optimum filter is called the *Wiener-Hopf equation*. These equations are also called the normal equations, encountered earlier in the chapter in the context of linear one-step prediction.

In general, the equations in (12.7.3) can be expressed in matrix form as

$$\boldsymbol{\Gamma}_M \mathbf{h}_M = \boldsymbol{\gamma}_d \quad (12.7.4)$$

where $\boldsymbol{\Gamma}_M$ is an $M \times M$ (Hermitian) Toeplitz matrix with elements $\Gamma_{lk} = \gamma_{xx}(l-k)$ and $\boldsymbol{\gamma}_d$ is the $M \times 1$ crosscorrelation vector with elements $\gamma_{dx}(l), l = 0, 1, \dots, M-1$. The solution for the optimum filter coefficients is

$$\mathbf{h}_{\text{opt}} = \boldsymbol{\Gamma}_M^{-1} \boldsymbol{\gamma}_d \quad (12.7.5)$$

and the resulting minimum MSE achieved by the Wiener filter is

$$\text{MMSE}_M = \min_{\mathbf{h}_M} \mathcal{E}_M = \sigma_d^2 - \sum_{k=0}^{M-1} h_{\text{opt}}(k)\gamma_{dx}^*(k) \quad (12.7.6)$$

or, equivalently,

$$\text{MMSE}_M = \sigma_d^2 - \boldsymbol{\gamma}_d^* \boldsymbol{\Gamma}_M^{-1} \boldsymbol{\gamma}_d \quad (12.7.7)$$

where $\sigma_d^2 = E|d(n)|^2$.

Let us consider some special cases of (12.7.3). If we are dealing with filtering, the $d(n) = s(n)$. Furthermore, if $s(n)$ and $w(n)$ are uncorrelated random sequences, as is usually the case in practice, then

$$\begin{aligned}\gamma_{xx}(k) &= \gamma_{ss}(k) + \gamma_{ww}(k) \\ \gamma_{dx}(k) &= \gamma_{ss}(k)\end{aligned}\quad (12.7.8)$$

and the normal equations in (12.7.3) become

$$\sum_{k=0}^{M-1} h(k)[\gamma_{ss}(l-k) + \gamma_{ww}(l-k)] = \gamma_{ss}(l), \quad l = 0, 1, \dots, M-1 \quad (12.7.9)$$

If we are dealing with prediction, then $d(n) = s(n+D)$ where $D > 0$. Assuming that $s(n)$ and $w(n)$ are uncorrelated random sequences, we have

$$\gamma_{dx}(k) = \gamma_{ss}(l+D) \quad (12.7.10)$$

Hence the equations for the Wiener prediction filter become

$$\sum_{k=0}^{M-1} h(k)[\gamma_{ss}(l-k) + \gamma_{ww}(l-k)] = \gamma_{ss}(l+D), \quad l = 0, 1, \dots, M-1 \quad (12.7.11)$$

In all these cases, the correlation matrix to be inverted is Toeplitz. Hence the (generalized) Levinson–Durbin algorithm may be used to solve for the optimum filter coefficients.

EXAMPLE 12.7.1

Let us consider a signal $x(n) = s(n) + w(n)$, where $s(n)$ is an AR(1) process that satisfies the difference equation

$$s(n) = 0.6s(n-1) + v(n)$$

where $\{v(n)\}$ is a white noise sequence with variance $\sigma_v^2 = 0.64$, and $\{w(n)\}$ is a white noise sequence with variance $\sigma_w^2 = 1$. We will design a Wiener filter of length $M = 2$ to estimate $\{s(n)\}$.

Solution. Since $\{s(n)\}$ is obtained by exciting a single-pole filter by white noise, the power spectral density of $s(n)$ is

$$\begin{aligned}\Gamma_{ss}(f) &= \sigma_v^2 |H(f)|^2 \\ &= \frac{0.64}{|1 - 0.6e^{-j2\pi f}|^2} \\ &= \frac{0.64}{1.36 - 1.2 \cos 2\pi f}\end{aligned}$$

The corresponding autocorrelation sequence $\{\gamma_{ss}(m)\}$ is

$$\gamma_{ss}(m) = (0.6)^{|m|}$$

The equations for the filter coefficients are

$$2h(0) + 0.6h(1) = 1$$

$$0.6h(0) + 2h(1) = 0.6$$

Solution of these equations yields the result

$$h(0) = 0.451, \quad h(1) = 0.165$$

The corresponding minimum MSE is

$$\begin{aligned} \text{MMSE}_2 &= 1 - h(0)\gamma_{ss}(0) - h(1)\gamma_{ss}(1) \\ &= 1 - 0.451 - (0.165)(0.6) \\ &= 0.45 \end{aligned}$$

This error can be reduced further by increasing the length of the Wiener filter (see Problem 12.35).

12.7.2 Orthogonality Principle in Linear Mean-Square Estimation

The normal equations for the optimum filter coefficients given by (12.7.3) can be obtained directly by applying the orthogonality principle in linear mean-square estimation. Simply stated, the mean-square error \mathcal{E}_M in (12.7.2) is a minimum if the filter coefficients $\{h(k)\}$ are selected such that the error is orthogonal to each of the data points in the estimate,

$$E[e(n)x^*(n-l)] = 0, \quad l = 0, 1, \dots, M-1 \quad (12.7.12)$$

where

$$e(n) = d(n) - \sum_{k=0}^{M-1} h(k)x(n-k) \quad (12.7.13)$$

Conversely, if the filter coefficients satisfy (12.7.12), the resulting MSE is a minimum.

When viewed geometrically, the output of the filter, which is the estimate

$$\hat{d}(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (12.7.14)$$

is a vector in the subspace spanned by the data $\{x(k), 0 \leq k \leq M-1\}$. The error $e(n)$ is a vector from $d(n)$ to $\hat{d}(n)$ [i.e., $d(n) = e(n) + \hat{d}(n)$], as shown in Fig. 12.7.2. The orthogonality principle states that the length $\mathcal{E}_M = E|e(n)|^2$ is a minimum when $e(n)$ is perpendicular to the data subspace [i.e., $e(n)$ is orthogonal to each data point $x(k), 0 \leq k \leq M-1$].

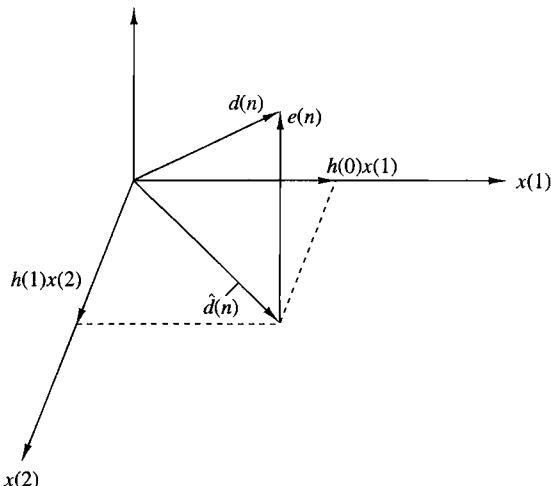


Figure 12.7.2
Geometric interpretation of linear MSE problem.

We note that the solution obtained from the normal equations in (12.7.3) is unique if the data $\{x(n)\}$ in the estimate $\hat{d}(n)$ are *linearly independent*. In this case, the correlation matrix Γ_M is nonsingular. On the other hand, if the data are linearly dependent, the rank of Γ_M is less than M and therefore the solution is not unique. In this case, the estimate $\hat{d}(n)$ can be expressed as a linear combination of a reduced set of linearly independent data points equal to the rank of Γ_M .

Since the MSE is minimized by selecting the filter coefficients to satisfy the orthogonality principle, the residual minimum MSE is simply

$$\text{MMSE}_M = E[e(n)d^*(n)] \quad (12.7.15)$$

which yields the result given in (12.7.6).

12.7.3 IIR Wiener Filter

In the preceding section we constrained the filter to be FIR and obtained a set of M linear equations for the optimum filter coefficients. In this section we allow the filter to be infinite in duration (IIR) and the data sequence to be infinite as well. Hence the filter output is

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k) \quad (12.7.16)$$

The filter coefficients are selected to minimize the mean-square error between the desired output $d(n)$ and $y(n)$, that is,

$$\begin{aligned} \mathcal{E}_{\infty} &= E|e(n)|^2 \\ &= E \left| d(n) - \sum_{k=0}^{\infty} h(k)x(n-k) \right|^2 \end{aligned} \quad (12.7.17)$$

Application of the orthogonality principle leads to the Wiener–Hopf equation,

$$\sum_{k=0}^{\infty} h(k) \gamma_{xx}(l-k) = \gamma_{dx}(l), \quad l \geq 0 \quad (12.7.18)$$

The residual MMSE is simply obtained by application of the condition given by (12.7.15). Thus we obtain

$$\text{MMSE}_{\infty} = \min_{\mathbf{h}} \mathcal{E}_{\infty} = \sigma_d^2 - \sum_{k=0}^{\infty} h_{\text{opt}}(k) \gamma_{dx}^*(k) \quad (12.7.19)$$

The Wiener–Hopf equation given by (12.7.18) cannot be solved directly with z -transform techniques, because the equation holds only for $l \geq 0$. We shall solve for the optimum IIR Wiener filter based on the innovations representation of the stationary random process $\{x(n)\}$.

Recall that a stationary random process $\{x(n)\}$ with autocorrelation $\gamma_{xx}(k)$ and power spectral density $\Gamma_{xx}(f)$ can be represented by an equivalent innovations process, $\{i(n)\}$, by passing $\{x(n)\}$ through a noise-whitening filter with system function $1/G(z)$, where $G(z)$ is the minimum-phase part obtained from the spectral factorization of $\Gamma_{xx}(z)$:

$$\Gamma_{xx}(z) = \sigma_i^2 G(z) G(z^{-1}) \quad (12.7.20)$$

Hence $G(z)$ is analytic in the region $|z| > r_1$, where $r_1 < 1$.

Now, the optimum Wiener filter can be viewed as the cascade of the whitening filter $1/G(z)$ with a second filter, say $Q(z)$, whose output $y(n)$ is identical to the output of the optimum Wiener filter. Since

$$y(n) = \sum_{k=0}^{\infty} q(k) i(n-k) \quad (12.7.21)$$

and $e(n) = d(n) - y(n)$, application of the orthogonality principle yields the new Wiener–Hopf equation as

$$\sum_{k=0}^{\infty} q(k) \gamma_{ii}(l-k) = \gamma_{di}(l), \quad l \geq 0 \quad (12.7.22)$$

But since $\{i(n)\}$ is white, it follows that $\gamma_{ii}(l-k) = 0$ unless $l = k$. Thus we obtain the solution as

$$q(l) = \frac{\gamma_{di}(l)}{\gamma_{ii}(0)} = \frac{\gamma_{di}(l)}{\sigma_i^2}, \quad l \geq 0 \quad (12.7.23)$$

The z -transform of the sequence $\{q(l)\}$ is

$$\begin{aligned} Q(z) &= \sum_{k=0}^{\infty} q(k) z^{-k} \\ &= \frac{1}{\sigma_i^2} \sum_{k=0}^{\infty} \gamma_{di}(k) z^{-k} \end{aligned} \quad (12.7.24)$$

If we denote the z -transform of the two-sided crosscorrelation sequence $\gamma_{di}(k)$ by $\Gamma_{di}(z)$

$$\Gamma_{di}(z) = \sum_{k=-\infty}^{\infty} \gamma_{di}(k)z^{-k} \quad (12.7.25)$$

and define $[\Gamma_{di}(z)]_+$ as

$$[\Gamma_{di}(z)]_+ = \sum_{k=0}^{\infty} \gamma_{di}(k)z^{-k} \quad (12.7.26)$$

then

$$Q(z) = \frac{1}{\sigma_i^2} [\Gamma_{di}(z)]_+ \quad (12.7.27)$$

To determine $[\Gamma_{di}(z)]_+$, we begin with the output of the noise-whitening filter, which can be expressed as

$$i(n) = \sum_{k=0}^{\infty} v(k)x(n-k) \quad (12.7.28)$$

where $\{v(k), k \geq 0\}$ is the impulse response of the noise-whitening filter,

$$\frac{1}{G(z)} \equiv V(z) = \sum_{k=0}^{\infty} v(k)z^{-k} \quad (12.7.29)$$

Then

$$\begin{aligned} \gamma_{di}(k) &= E[d(n)i^*(n-k)] \\ &= \sum_{m=0}^{\infty} v(m)E[d(n)x^*(n-m-k)] \\ &= \sum_{m=0}^{\infty} v(m)\gamma_{dx}(k+m) \end{aligned} \quad (12.7.30)$$

The z -transform of the crosscorrelation $\gamma_{di}(k)$ is

$$\begin{aligned} \Gamma_{di}(z) &= \sum_{k=-\infty}^{\infty} \left[\sum_{m=0}^{\infty} v(m)\gamma_{dx}(k+m) \right] z^{-k} \\ &= \sum_{m=0}^{\infty} v(m) \sum_{k=-\infty}^{\infty} \gamma_{dx}(k+m)z^{-k} \\ &= \sum_{m=0}^{\infty} v(m)z^m \sum_{k=-\infty}^{\infty} \gamma_{dx}(k)z^{-k} \\ &= V(z^{-1})\Gamma_{dx}(z) = \frac{\Gamma_{dx}(z)}{G(z^{-1})} \end{aligned} \quad (12.7.31)$$

Therefore,

$$Q(z) = \frac{1}{\sigma_i^2} \left[\frac{\Gamma_{dx}(z)}{G(z^{-1})} \right]_+ \quad (12.7.32)$$

Finally, the optimum IIR Wiener filter has the system function

$$\begin{aligned} H_{\text{opt}}(z) &= \frac{Q(z)}{G(z)} \\ &= \frac{1}{\sigma_i^2 G(z)} \left[\frac{\Gamma_{dx}(z)}{G(z^{-1})} \right]_+ \end{aligned} \quad (12.7.33)$$

In summary, the solution for the optimum IIR Wiener filter requires that we perform a spectral factorization of $\Gamma_{xx}(z)$ to obtain $G(z)$, the minimum-phase component, and then we solve for the causal part of $\Gamma_{dx}(z)/G(z^{-1})$. The following example illustrates the procedure.

EXAMPLE 12.7.2

Let us determine the optimum IIR Wiener filter for the signal given in Example 12.7.1.

Solution. For this signal we have

$$\Gamma_{xx}(z) = \Gamma_{ss}(z) + 1 = \frac{1.8(1 - \frac{1}{3}z^{-1})(1 - \frac{1}{3}z)}{(1 - 0.6z^{-1})(1 - 0.6z)}$$

where $\sigma_i^2 = 1.8$ and

$$G(z) = \frac{1 - \frac{1}{3}z^{-1}}{1 - 0.6z^{-1}}$$

The z -transform of the crosscorrelation $\gamma_{dx}(m)$ is

$$\Gamma_{dx}(z) = \Gamma_{ss}(z) = \frac{0.64}{(1 - 0.6z^{-1})(1 - 0.6z)}$$

Hence

$$\begin{aligned} \left[\frac{\Gamma_{dx}(z)}{G(z^{-1})} \right]_+ &= \left[\frac{0.64}{(1 - \frac{1}{3}z)(1 - 0.6z^{-1})} \right]_+ \\ &= \left[\frac{0.8}{1 - 0.6z^{-1}} + \frac{0.266z}{1 - \frac{1}{3}z} \right]_+ \\ &= \frac{0.8}{1 - 0.6z^{-1}} \end{aligned}$$

The optimum IIR filter has the system function

$$\begin{aligned} H_{\text{opt}}(z) &= \frac{1}{1.8} \left(\frac{1 - 0.6z^{-1}}{1 - \frac{1}{3}z^{-1}} \right) \left(\frac{0.8}{1 - 0.6z^{-1}} \right) \\ &= \frac{\frac{4}{9}}{1 - \frac{1}{3}z^{-1}} \end{aligned}$$

and an impulse response

$$h_{\text{opt}}(n) = \frac{4}{9} \left(\frac{1}{3}\right)^n, \quad n \geq 0$$

We conclude this section by expressing the minimum MSE given by (12.7.19) in terms of the frequency-domain characteristics of the filter. First, we note that $\sigma_d^2 \equiv E|d(n)|^2$ is simply the value of the autocorrelation sequence $\{\gamma_{dd}(k)\}$ evaluated at $k = 0$. Since

$$\gamma_{dd}(k) = \frac{1}{2\pi j} \oint_C \Gamma_{dd}(z) z^{k-1} dz \quad (12.7.34)$$

it follows that

$$\sigma_d^2 = \gamma_{dd}(0) = \frac{1}{2\pi j} \oint_C \frac{\Gamma_{dd}(z)}{z} dz \quad (12.7.35)$$

where the contour integral is evaluated along a closed path encircling the origin in the region of convergence of $\Gamma_{dd}(z)$.

The second term in (12.7.19) is also easily transformed to the frequency domain by application of Parseval's theorem. Since $h_{\text{opt}}(k) = 0$ for $k < 0$, we have

$$\sum_{k=-\infty}^{\infty} h_{\text{opt}}(k) \gamma_{dx}^*(k) = \frac{1}{2\pi j} \oint_C H_{\text{opt}}(z) \Gamma_{dx}(z^{-1}) z^{-1} dz \quad (12.7.36)$$

where C is a closed contour encircling the origin that lies within the common region of convergence of $H_{\text{opt}}(z)$ and $\Gamma_{dx}(z^{-1})$.

By combining (12.7.35) with (12.7.36), we obtain the desired expression for the MMSE $_{\infty}$ in the form

$$\text{MMSE}_{\infty} = \frac{1}{2\pi j} \oint_C [\Gamma_{dd}(z) - H_{\text{opt}}(z) \Gamma_{dx}(z^{-1})] z^{-1} dz \quad (12.7.37)$$

EXAMPLE 12.7.3

For the optimum Wiener filter derived in Example 12.7.2, the minimum MSE is

$$\text{MMSE}_{\infty} = \frac{1}{2\pi j} \oint_C \left[\frac{0.3555}{(z - \frac{1}{3})(1 - 0.6z)} \right] dz$$

There is a single pole inside the unit circle at $z = \frac{1}{3}$. By evaluating the residue at the pole, we obtain

$$\text{MMSE}_{\infty} = 0.444$$

We observe that this MMSE is only slightly smaller than that for the optimum two-tap Wiener filter in Example 12.7.1.

12.7.4 Noncausal Wiener Filter

In the preceding section we constrained the optimum Wiener filter to be causal [i.e., $h_{\text{opt}}(n) = 0$ for $n < 0$]. In this section we drop this condition and allow the filter to include both the infinite past and the infinite future of the sequence $\{x(n)\}$ in forming the output $y(n)$, that is,

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \quad (12.7.38)$$

The resulting filter is physically unrealizable. It can also be viewed as a *smoothing filter* in which the infinite future signal values are used to smooth the estimate $\hat{d}(n) = y(n)$ of the desired signal $d(n)$.

Application of the orthogonality principle yields the Wiener–Hopf equation for the noncausal filter in the form

$$\sum_{k=-\infty}^{\infty} h(k)\gamma_{xx}(l-k) = \gamma_{dx}(l), \quad -\infty < l < \infty \quad (12.7.39)$$

and the resulting MMSE_{nc} as

$$\text{MMSE}_{\text{nc}} = \sigma_d^2 - \sum_{k=-\infty}^{\infty} h(k)\gamma_{dx}^*(k) \quad (12.7.40)$$

Since (12.7.39) holds for $-\infty < l < \infty$, this equation can be transformed directly to yield the optimum noncausal Wiener filter as

$$H_{\text{nc}}(z) = \frac{\Gamma_{dx}(z)}{\Gamma_{xx}(z)} \quad (12.7.41)$$

The MMSE_{nc} can also be simply expressed in the z -domain as

$$\text{MMSE}_{\text{nc}} = \frac{1}{2\pi j} \oint_C [\Gamma_{dd}(z) - H_{\text{nc}}(z)\Gamma_{dx}(z^{-1})]z^{-1}dz \quad (12.7.42)$$

In the following example we compare the form of the optimal noncausal filter with the optimal causal filter obtained in the previous section.

EXAMPLE 12.7.4

The optimum noncausal Wiener filter for the signal characteristics given in Example 12.7.1 is given by (12.7.41), where

$$\Gamma_{dx}(z) = \Gamma_{ss}(z) = \frac{0.64}{(1 - 0.6z^{-1})(1 - 0.6z)}$$

and

$$\Gamma_{xx}(z) = \Gamma_{ss}(z) + 1$$

$$= \frac{2(1 - 0.3z^{-1} - 0.3z)}{(1 - 0.6z^{-1})(1 - 0.6z)}$$

Then,

$$H_{nc}(z) = \frac{0.3555}{(1 - \frac{1}{3}z^{-1})(1 - \frac{1}{3}z)}$$

This filter is clearly noncausal.

The minimum MSE achieved by this filter is determined from evaluating (12.7.42). The integrand is

$$\frac{1}{z} \Gamma_{ss}(z)[1 - H_{nc}(z)] = \frac{0.3555}{(z - \frac{1}{3})(1 - \frac{1}{3}z)}$$

The only pole inside the unit circle is $z = \frac{1}{3}$. Hence the residue is

$$\left. \frac{0.3555}{1 - \frac{1}{3}z} \right|_{z=\frac{1}{3}} = \frac{0.3555}{8/9} = 0.40$$

Hence the minimum achievable MSE obtained with the optimum noncausal Wiener filter is

$$\text{MMSE}_{nc} = 0.40$$

Note that this is lower than the MMSE for the causal filter, as expected.

12.8 Summary and References

The major focal point in this chapter is the design of optimum linear systems for linear prediction and filtering. The criterion for optimality is the minimization of the mean-square error between a specified desired filter output and the actual filter output.

In the development of linear prediction, we demonstrated that the equations for the forward and backward prediction errors specified a lattice filter whose parameters, the reflection coefficients $\{K_m\}$, were simply related to the filter coefficients $\{a_m(k)\}$ of the direct-form FIR linear predictor and the associated prediction-error filter. The optimum filter coefficients $\{K_m\}$ and $\{a_m(k)\}$ are easily obtained from the solution of the normal equations.

We described two computationally efficient algorithms for solving the normal equations, the Levinson–Durbin algorithm and the Schur algorithm. Both algorithms are suitable for solving a Toeplitz system of linear equations and have a computational complexity of $O(p^2)$ when executed on a single processor. However, with full parallel processing, the Schur algorithm solves the normal equations in $O(p)$ time, whereas the Levinson–Durbin algorithm requires $O(p \log p)$ time.

In addition to the all-zero lattice filter resulting from linear prediction, we also derived the AR lattice (all-pole) filter structure and the ARMA lattice-ladder (pole-zero) filter structure. Finally, we described the design of the class of optimum linear filters, called Wiener filters.

Linear estimation theory has had a long and rich history of development over the past four decades. Kailath (1974) presents a historical account of the first three

decades. The pioneering work of Wiener (1949) on optimum linear filtering for statistically stationary signals is especially significant. The generalization of the Wiener filter theory to dynamical systems with random inputs was developed by Kalman (1960) and Kalman and Bucy (1961). Kalman filters are treated in the books by Meditch (1969), Brown (1983), and Chui and Chen (1987). The monograph by Kailath (1981) treats both Wiener and Kalman filters.

There are numerous references on linear prediction and lattice filters. Tutorial treatments on these subjects have been published in the journal papers by Makhoul (1975, 1978) and Friedlander (1982a, b). The books by Haykin (1991), Markel and Gray 1976), and Tretter (1976) provide comprehensive treatments of these subjects. Applications of linear prediction to spectral analysis are found in the books by Kay (1988) and Marple (1987), to geophysics in the book by Robinson and Treitel (1980), and to adaptive filtering in the book by Haykin (1991).

The Levinson–Durbin algorithm for solving the normal equations recursively was given by Levinson (1947) and later modified by Durbin (1959). Variations of this classical algorithm, called *split Levinson algorithms*, have been developed by Delsarte and Genin (1986) and by Krishna (1988). These algorithms exploit additional symmetries in the Toeplitz correlation matrix and save about a factor of 2 in the number of multiplications.

The Schur algorithm was originally described by Schur (1917) in a paper published in German. An English translation of this paper appears in the book edited by Gohberg (1986). The Schur algorithm is intimately related to the polynomials $\{A_m(z)\}$, which can be interpreted as orthogonal polynomials. A treatment of orthogonal polynomials is given in the books by Szegö (1967), Grenander and Szegö (1958), and Geronimus (1958). The thesis of Vieira (1977) and the papers by Kailath et al. (1978), Delsarte et al. (1978), and Youla and Kazanjian (1978) provide additional results on orthogonal polynomials. Kailath (1985, 1986) provides tutorial treatments of the Schur algorithm and its relationship to orthogonal polynomials and the Levinson–Durbin algorithm. The pipelined parallel processing structure for computing the reflection coefficients based on the Schur algorithm and the related problem of solving Toeplitz systems of linear equations is described in the paper by Kung and Hu (1983). Finally, we should mention that some additional computational efficiency can be achieved in the Schur algorithm, by further exploiting symmetry properties of Toeplitz matrices, as described by Krishna (1988). This leads to the so-called split-Schur algorithm, which is analogous to the split-Levinson algorithm.

Problems

- 12.1** The power density spectrum of an AR process $\{x(n)\}$ is given as

$$\begin{aligned}\Gamma_{xx}(\omega) &= \frac{\sigma_w^2}{|A(\omega)|^2} \\ &= \frac{25}{|1 - e^{-j\omega} + \frac{1}{2}e^{-j2\omega}|^2}\end{aligned}$$

where σ_w^2 is the variance of the input sequence.

(a) Determine the difference equation for generating the AR process when the excitation is white noise.

(b) Determine the system function for the whitening filter.

An ARMA process has an autocorrelation $\{\gamma_{xx}(m)\}$ whose z -transform is given as

$$\Gamma_{xx}(z) = 9 \frac{(z - \frac{1}{3})(z - 3)}{(z - \frac{1}{2})(z - 2)}, \quad \frac{1}{2} < |z| < 2$$

(a) Determine the filter $H(z)$ for generating $\{x(n)\}$ from a white noise input sequence. Is $H(z)$ unique? Explain.

(b) Determine a stable linear whitening filter for the sequence $\{x(n)\}$.

Consider the ARMA process generated by the difference equation

$$x(n) = 1.6x(n-1) - 0.63x(n-2) + w(n) + 0.9w(n-1)$$

(a) Determine the system function of the whitening filter and its poles and zeros.

(b) Determine the power density spectrum of $\{x(n)\}$.

Determine the lattice coefficients corresponding to the FIR filter with system function

$$H(z) = A_3(z) = 1 + \frac{13}{24}z^{-1} + \frac{5}{8}z^{-2} + \frac{1}{3}z^{-3}$$

Determine the reflection coefficients $\{K_m\}$ of the lattice filter corresponding to the FIR filter described by the system function

$$H(z) = A_2(z) = 1 + 2z^{-1} + \frac{1}{3}z^{-2}$$

(a) Determine the zeros and sketch the zero pattern for the FIR lattice filter with reflection coefficients

$$K_1 = \frac{1}{2}, \quad K_2 = -\frac{1}{3}, \quad K_3 = 1$$

(b) Repeat part (a) but with $K_3 = -1$.

(c) You should have found that the zeros lie on the unit circle. Can this result be generalized? How?

Determine the impulse response of the FIR filter that is described by the lattice coefficients $K_1 = 0.6$, $K_2 = 0.3$, $K_3 = 0.5$, and $K_4 = 0.9$.

In Section 12.3.4 we indicated that the noise-whitening filter $A_p(z)$ for a causal AR(p) process is a forward linear prediction-error filter of order p . Show that the backward linear prediction-error filter of order p is the noise-whitening filter of the corresponding anticausal AR(p) process.

Use the orthogonality principle to determine the normal equations and the resulting minimum MSE for a forward predictor of order p that predicts m samples ($m > 1$) into the future (m -step forward predictor). Sketch the prediction-error filter.

Repeat Problem 12.9 for an m -step backward predictor.

- 12.11** Determine a Levinson–Durbin recursive algorithm for solving for the coefficients of a backward prediction-error filter. Use the result to show that coefficients of the forward and backward predictors can be expressed recursively as

$$\mathbf{a}_m = \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix} + K_m \begin{bmatrix} \mathbf{b}_{m-1} \\ 1 \end{bmatrix}$$

$$\mathbf{b}_m = \begin{bmatrix} \mathbf{b}_{m-1} \\ 0 \end{bmatrix} + K_m^* \begin{bmatrix} \mathbf{a}_{m-1} \\ 1 \end{bmatrix}$$

- 12.12** The Levinson–Durbin algorithm described in Section 12.4.1 solved the linear equations

$$\boldsymbol{\Gamma}_m \mathbf{a}_m = -\boldsymbol{\gamma}_m$$

where the right-hand side of this equation has elements of the autocorrelation sequence that are also elements of the matrix $\boldsymbol{\Gamma}$. Let us consider the more general problem of solving the linear equations

$$\boldsymbol{\Gamma}_m \mathbf{b}_m = \mathbf{c}_m$$

where \mathbf{c}_m is an arbitrary vector. (The vector \mathbf{b}_m is not related to the coefficients of the backward predictor.) Show that the solution to $\boldsymbol{\Gamma}_m \mathbf{b}_m = \mathbf{c}_m$ can be obtained from a *generalized Levinson–Durbin* algorithm which is given recursively as

$$b_m(m) = \frac{c(m) - \gamma_{m-1}^{bt} \mathbf{b}_{m-1}}{E_{m-1}^f}$$

$$b_m(k) = b_{m-1}(k) - b_m(m) a_{m-1}^*(m-k), \quad \begin{array}{l} k = 1, 2, \dots, m-1 \\ m = 1, 2, \dots, p \end{array}$$

where $b_1(1) = c(1)/\gamma_{xx}(0) = c(1)/E_0^f$ and $a_m(k)$ is given by (12.4.17). Thus a second recursion is required to solve the equation $\boldsymbol{\Gamma}_m \mathbf{b}_m = \mathbf{c}_m$.

- 12.13** Use the generalized Levinson–Durbin algorithm to solve the normal equations recursively for the m -step forward and backward predictors.
- 12.14** Consider the AR(3) process generated by the equation

$$x(n) = \frac{14}{24}x(n-1) + \frac{9}{24}x(n-2) - \frac{1}{24}x(n-3) + w(n)$$

where $w(n)$ is a stationary white noise process with variance σ_w^2 .

- (a) Determine the coefficients of the optimum $p = 3$ linear predictor.
- (b) Determine the autocorrelation sequence $\gamma_{xx}(m)$, $0 \leq m \leq 5$.
- (c) Determine the reflection coefficients corresponding to the $p = 3$ linear predictor.

- 12.15** The z -transform of the autocorrelation $\gamma_{xx}(m)$ of an ARMA(1, 1) process is

$$\Gamma_{xx}(z) = \sigma_w^2 H(z) H(z^{-1})$$

$$\Gamma_{xx}(z) = \frac{4\sigma_w^2}{9} \frac{5 - 2z - 2z^{-1}}{10 - 3z^{-1} - 3z}$$

- (a) Determine the minimum-phase system function $H(z)$.
 (b) Determine the system function $H(z)$ for a mixed-phase stable system.

12.16 Consider an FIR filter with coefficient vector

$$[1 \quad -2r \cos \theta \quad r^2]$$

- (a) Determine the reflection coefficients for the corresponding FIR lattice filter.
 (b) Determine the values of the reflection coefficients in the limit as $r \rightarrow 1$.

12.17 An AR(3) process is characterized by the prediction coefficients

$$a_3(1) = -1.25, \quad a_3(2) = 1.25, \quad a_3(3) = -1$$

- (a) Determine the reflection coefficients.
 (b) Determine $\gamma_{xx}(m)$ for $0 \leq m \leq 3$.
 (c) Determine the mean-square prediction error.

12.18 The autocorrelation sequence for a random process is

$$\gamma_{xx}(m) = \begin{cases} 1, & m = 0 \\ -0.5, & m = \pm 1 \\ 0.625, & m = \pm 2 \\ -0.6875, & m = \pm 3 \\ 0 & \text{otherwise} \end{cases}$$

Determine the system functions $A_m(z)$ for the prediction-error filters for $m = 1, 2, 3$, the reflection coefficients $\{K_m\}$, and the corresponding mean-square prediction errors.

12.19 The autocorrelation sequence for an AR process $x(n)$ is

$$\gamma_{xx}(m) = (\frac{1}{4})^{|m|}$$

- (a) Determine the difference equation for $x(n)$.
 (b) Is your answer unique? If not, give any other possible solutions.

12.20 Repeat Problem 12.19 for an AR process with autocorrelation

$$\gamma_{xx}(m) = a^{|m|} \cos \frac{\pi m}{2}$$

where $0 < a < 1$.

12.21 Prove that a FIR filter with system function

$$A_p(z) = 1 + \sum_{k=1}^p a_p(k)z^{-k}$$

and reflection coefficients $|K_k| < 1$ for $1 \leq k \leq p - 1$ and $|K_p| > 1$ is maximum phase [all the roots of $A_p(z)$ lie outside the unit circle].

12.22 Show that the transformation

$$\mathbf{V}_m = \begin{bmatrix} 1 & K_m \\ K_m^* & 1 \end{bmatrix}$$

in the Schur algorithm satisfies the special property

$$\mathbf{V}_m \mathbf{J} \mathbf{V}_m^T = (1 - |K_m|^2) \mathbf{J}$$

where

$$\mathbf{J} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Thus \mathbf{V}_m is called a J -rotation matrix. Its role is to rotate or hyperbolate the row of \mathbf{G}_m to lie along the first coordinate direction (Kailath, 1985).

- 12.23** Prove the additional properties (a) through (l) of the prediction-error filters given in Section 12.5.
- 12.24** Extend the additional properties (a) through (l) of the prediction-error filters given in Section 12.5 to complex-valued signals.
- 12.25** Determine the reflection coefficient K_3 in terms of the autocorrelations $\{\gamma_{xx}(m)\}$ from the Schur algorithm and compare your result with the expression for K_3 obtained from the Levinson–Durbin algorithm.
- 12.26** Consider an infinite-length ($p = \infty$) one-step forward predictor for a stationary random process $\{x(n)\}$ with a power density spectrum of $\Gamma_{xx}(f)$. Show that the mean-square error of the prediction-error filter can be expressed as

$$E_\infty^f = 2\pi \exp\left\{\int_{-1/2}^{1/2} \ln \Gamma_{xx}(f) df\right\}$$

- 12.27** Determine the output of an infinite-length ($p = \infty$) m -step forward predictor and the resulting mean-square error when the input signal is a first-order autoregressive process of the form

$$x(n) = ax(n-1) + w(n)$$

- 12.28** An AR(3) process $\{x(n)\}$ is characterized by the autocorrelation sequence $\gamma_{xx}(0) = 1$, $\gamma_{xx}(1) = \frac{1}{2}$, $\gamma_{xx}(2) = \frac{1}{8}$, and $\gamma_{xx}(3) = \frac{1}{64}$.
 - (a)** Use the Schur algorithm to determine the three reflection coefficients K_1 , K_2 , and K_3 .
 - (b)** Sketch the lattice filter for synthesizing $\{x(n)\}$ from a white noise excitation.
- 12.29** The purpose of this problem is to show that the polynomials $\{A_m(z)\}$, which are the system functions of the forward prediction-error filters of order m , $m = 0, 1, \dots, p$, can be interpreted as orthogonal on the unit circle. Toward this end, suppose that $\Gamma_{xx}(f)$ is the power spectral density of a zero-mean random process $\{x(n)\}$ and let $\{A_m(z)\}$, $m = 0, 1, \dots, p\}$, be the system functions of the corresponding prediction-error filters. Show that the polynomials $\{A_m(z)\}$ satisfy the orthogonality property

$$\int_{-1/2}^{1/2} \Gamma_{xx}(f) A_m(f) A_n^*(f) df = E_m^f \delta_{mn}, \quad m, n = 0, 1, \dots, p$$

- 12.30** Determine the system function of the all-pole filter described by the lattice coefficients $K_1 = 0.6$, $K_2 = 0.3$, $K_3 = 0.5$, and $K_4 = 0.9$.
- 12.31** Determine the parameters and sketch the lattice-ladder filter structure for the system with system function

$$H(z) = \frac{1 - 0.8z^{-1} + 0.15z^{-2}}{1 + 0.1z^{-1} - 0.72z^{-2}}$$

- 12.32** Consider a signal $x(n) = s(n) + w(n)$, where $s(n)$ is an AR(1) process that satisfies the difference equation

$$s(n) = 0.8s(n-1) + v(n)$$

where $\{v(n)\}$ is a white noise sequence with variance $\sigma_v^2 = 0.49$ and $\{w(n)\}$ is a white noise sequence with variance $\sigma_w^2 = 1$. The processes $\{v(n)\}$ and $\{w(n)\}$ are uncorrelated.

- (a) Determine the autocorrelation sequences $\{\gamma_{ss}(m)\}$ and $\{\gamma_{xx}(m)\}$.
- (b) Design a Wiener filter of length $M = 2$ to estimate $\{s(n)\}$.
- (c) Determine the MMSE for $M = 2$.

- 12.33** Determine the optimum causal IIR Wiener filter for the signal given in Problem 12.32 and the corresponding MMSE_∞ .
- 12.34** Determine the system function for the noncausal IIR Wiener filter for the signal given in Problem 12.32 and the corresponding MMSE_{nc} .
- 12.35** Determine the optimum FIR Wiener filter of length $M = 3$ for the signal in Example 12.7.1 and the corresponding MMSE_3 . Compare MMSE_3 with MMSE_2 and comment on the difference.
- 12.36** An AR(2) process is defined by the difference equation

$$x(n) = x(n-1) - 0.6x(n-2) + w(n)$$

where $\{w(n)\}$ is a white noise process with variance σ_w^2 . Use the Yule–Walker equations to solve for the values of the autocorrelation $\gamma_{xx}(0)$, $\gamma_{xx}(1)$, and $\gamma_{xx}(2)$.

- 12.37** An observed random process $\{x(n)\}$ consists of the sum of an AR(p) process of the form

$$s(n) = -\sum_{k=1}^p a_p(k)s(n-k) + v(n)$$

and a white noise process $\{w(n)\}$ with variance σ_w^2 . The random process $\{v(n)\}$ is also white with variance σ_v^2 . The sequences $\{v(n)\}$ and $\{w(n)\}$ are uncorrelated.

Show that the observed process $\{x(n) = s(n) + w(n)\}$ is ARMA(p, p) and determine the coefficients of the numerator polynomial (MA component) in the corresponding system function.

Adaptive Filters

In contrast to filter design techniques described in Chapter 12, which were based on knowledge of the second-order statistics of the signals, there are many digital signal processing applications in which these statistics cannot be specified a priori. Such applications include channel equalization, echo cancellation, and system modeling among others, as described in this chapter. In these applications, filters with adjustable coefficients, called *adaptive filters*, are employed. Such filters incorporate algorithms that allow the filter coefficients to adapt to the signal statistics.

Adaptive filters have received considerable attention from researchers over the past 25 years. As a result, many computationally efficient algorithms for adaptive filtering have been developed. In this chapter, we describe two basic algorithms: the least-mean-square (LMS) algorithm, which is based on a gradient optimization for determining the coefficients, and the class of recursive least-squares algorithms, which includes both direct-form FIR and lattice realizations. Before we describe the algorithms, we present several practical applications in which adaptive filters have been successfully used in the estimation of signals corrupted by noise and other interference.

13.1 Applications of Adaptive Filters

Adaptive filters have been used widely in communication systems, control systems, and various other systems in which the statistical characteristics of the signals to be filtered are either unknown a priori or, in some cases, are slowly time-variant (non-stationary signals). Numerous applications of adaptive filters have been described in the literature. Some of the more noteworthy applications include: (1) adaptive antenna systems, in which adaptive filters are used for beam steering and for providing

nulls in the beam pattern to remove undesired interference (Widrow, Mantey, and Griffiths (1967)); (2) digital communication receivers, in which adaptive filters are used to provide equalization of intersymbol interference and for channel identification (Lucky (1965), Proakis and Miller (1969), Gershoff (1969), George, Bowen, and Storey (1971), Proakis (1970; 1975), Magee and Proakis (1973), Picinbono (1978) and Nichols, Giordano, and Proakis (1977)); (3) adaptive noise cancelling techniques, in which adaptive filters are used to estimate and eliminate a noise component in a desired signal (Widrow et al. (1975), Hsu and Giordano (1978), and Ketchum and Proakis (1982)); (4) system modeling, in which adaptive filters are used as models to estimate the characteristics of an unknown system. These are just a few of the best-known examples of the use of adaptive filters.

Although both IIR and FIR filters have been considered for adaptive filtering, the FIR filter is by far the most practical and widely used. The reason for this preference is quite simple: the FIR filter has only adjustable zeros; hence, it is free of stability problems associated with adaptive IIR filters, which have adjustable poles as well as zeros. We should not conclude, however, that adaptive FIR filters are always stable. On the contrary, the stability of the filter depends critically on the algorithm for adjusting its coefficients, as will be demonstrated in Sections 13.2 and 13.3.

Of the various FIR filter structures that are possible, the direct form and the lattice form are the ones used in adaptive filtering applications. The direct-form FIR filter structure with adjustable coefficients $h(n)$ is illustrated in Figure 13.1.1. On the other hand, the adjustable parameters in an FIR lattice structure are the reflection coefficients K_n .

An important consideration in the use of an adaptive filter is the criterion for optimizing the adjustable filter parameters. The criterion must not only provide a meaningful measure of filter performance, but it must also result in a practically realizable algorithm.

For example, a desirable performance index in a digital communication system is the average probability of error. Consequently, in implementing an adaptive equalizer, we might consider the selection of the equalizer coefficients to minimize the average probability of error as the basis for our optimization criterion. Unfortunately,

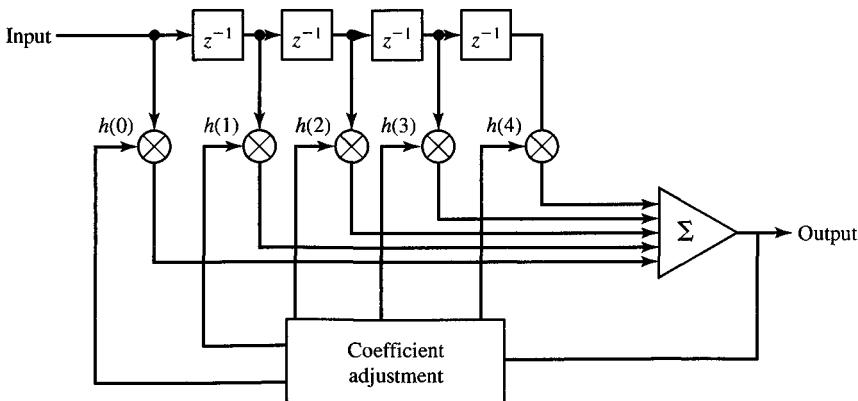


Figure 13.1.1 Direct-form adaptive FIR filter.

however, the performance index (average probability of error) for this criterion is a highly nonlinear function of the filter coefficients and the signal statistics. As a consequence, the implementation of an adaptive filter that optimizes such a performance index is complex and impractical.

In some cases, a performance index that is a nonlinear function of the filter parameters possesses many relative minima (or maxima), so that one is not certain whether the adaptive filter has converged to the optimum solution or to one of the relative minima (or maxima). For such reasons, some desirable performance indices, such as the average probability of error in a digital communication system, must be rejected on the grounds that they are impractical to implement.

Two criteria that provide good measures of performance in adaptive filtering applications are the least-squares criterion and its counterpart in a statistical formulation of the problem, namely, the mean-square-error (MSE) criterion. The least-squares (and MSE) criterion results in a quadratic performance index as a function of the filter coefficients and, hence, it possesses a single minimum. The resulting algorithms for adjusting the coefficients of the filter are relatively easy to implement, as we demonstrate in Sections 13.2 and 13.3.

In the following section, we describe several applications of adaptive filters that serve as a motivation for the mathematical development of algorithms derived in Sections 13.2 and 13.3. We find it convenient to use the direct-form FIR structure in these examples. Although we will not develop the recursive algorithms for automatically adjusting the filter coefficients in this section, it is instructive to formulate the optimization of the filter coefficients as a least-squares optimization problem. This development will serve to establish a common framework for the algorithms derived in the next two sections.

13.1.1 System Identification or System Modeling

In the formulation of this problem we have an unknown system, called a *plant*, that we wish to identify. The system is modeled by an FIR filter with M adjustable coefficients. Both the plant and model are excited by an input sequence $x(n)$. If $y(n)$ denotes the output of the plant and $\hat{y}(n)$ denotes the output of the model,

$$\hat{y}(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (13.1.1)$$

We may form the error sequence

$$e(n) = y(n) - \hat{y}(n), \quad n = 0, 1, \dots \quad (13.1.2)$$

and select the coefficients $h(k)$ to minimize

$$\mathcal{E}_M = \sum_{n=0}^N \left[y(n) - \sum_{k=0}^{M-1} h(k)x(n-k) \right]^2 \quad (13.1.3)$$

where $N + 1$ is the number of observations.

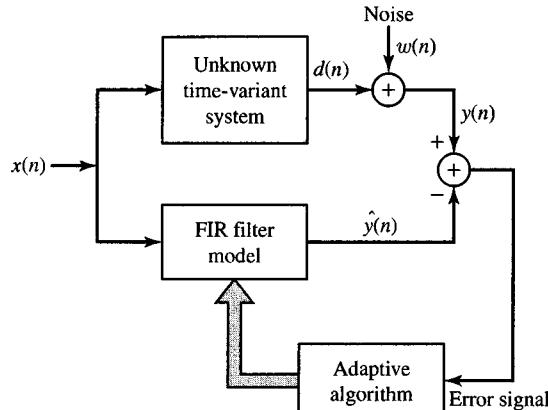


Figure 13.1.2
Application of adaptive
filtering to system
identification.

The least-squares criterion leads to the set of linear equations for determining the filter coefficients, namely,

$$\sum_{k=0}^{M-1} h(k)r_{xx}(l-k) = r_{yx}(l), \quad l = 0, 1, \dots, M-1 \quad (13.1.4)$$

In (13.1.4), $r_{xx}(l)$ is the autocorrelation of the sequence $x(n)$ and $r_{yx}(l)$ is the cross-correlation of the system output with the input sequence.

By solving (13.1.4), we obtain the filter coefficients for the model. Since the filter parameters are obtained directly from measurement data at the input and output of the system, without prior knowledge of the plant, we call the FIR filter model an adaptive filter.

If our only objective were to identify the system by use of the FIR model, the solution of (13.1.4) would suffice. In control systems applications, however, the system being modeled may be time variant, changing slowly with time, and our purpose for having a model is to ultimately use it for designing a controller that controls the plant. Furthermore, measurement noise is usually present at the output of the plant. This noise introduces uncertainty in the measurements and corrupts the estimates of the filter coefficients in the model. Such a scenario is illustrated in Figure 13.1.2. In this case, the adaptive filter must identify and track the time-variant characteristics of the plant in the presence of measurement noise at the output of the plant. The algorithms to be described in Sections 13.2 and 13.3 are applicable to this system identification problem.

13.1.2 Adaptive Channel Equalization

Figure 13.1.3 shows a block diagram of a digital communication system in which an adaptive equalizer is used to compensate for the distortion caused by the transmission medium (channel). The digital sequence of information symbols $a(n)$ is fed to the transmitting filter, whose output is

$$s(t) = \sum_{k=0}^{\infty} a(k)p(t - kT_s) \quad (13.1.5)$$

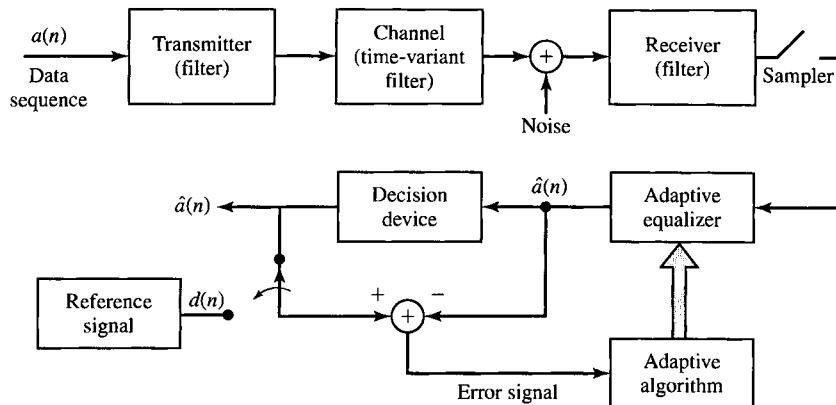


Figure 13.1.3 Application of adaptive filtering to adaptive channel equalization.

where $p(t)$ is the impulse response of the filter at the transmitter and T_s is the time interval between information symbols; that is, $1/T_s$ is the symbol rate. For purposes of this discussion, we may assume that $a(n)$ is a multilevel sequence that takes on values from the set $\pm 1, \pm 3, \pm 5, \dots, \pm(K - 1)$, where K is the number of possible symbol values.

Typically, the pulse $p(t)$ is designed to have the characteristics illustrated in Figure 13.1.4. Note that $p(t)$ has amplitude $p(0) = 1$ at $t = 0$ and $p(nT_s) = 0$ at $t = nT_s$, $n = \pm 1, \pm 2, \dots$. As a consequence, successive pulses transmitted sequentially every T_s seconds do not interfere with one another when sampled at the time instants $t = nT_s$. Thus, $a(n) = s(nT_s)$.

The channel, which is usually well modeled as a linear filter, distorts the pulse and, thus, causes intersymbol interference. For example, in telephone channels, filters are used throughout the system to separate signals in different frequency ranges. These filters cause phase and amplitude distortion. Figure 13.1.5 illustrates the effect of channel distortion on the pulse $p(t)$ as it might appear at the output of a telephone channel. Now, we observe that the samples taken every T_s seconds are corrupted by

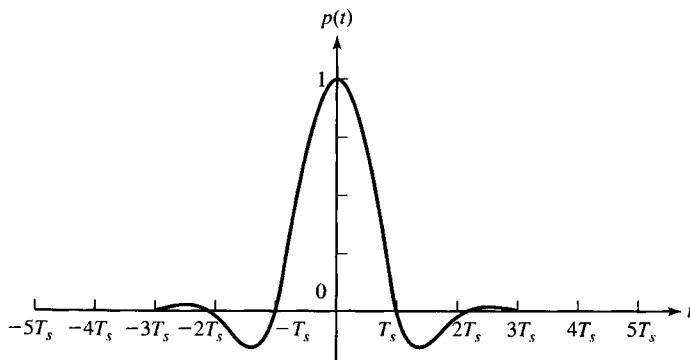


Figure 13.1.4 Pulse shape for digital transmission of symbols at a rate of $1/T_s$ symbols per second.

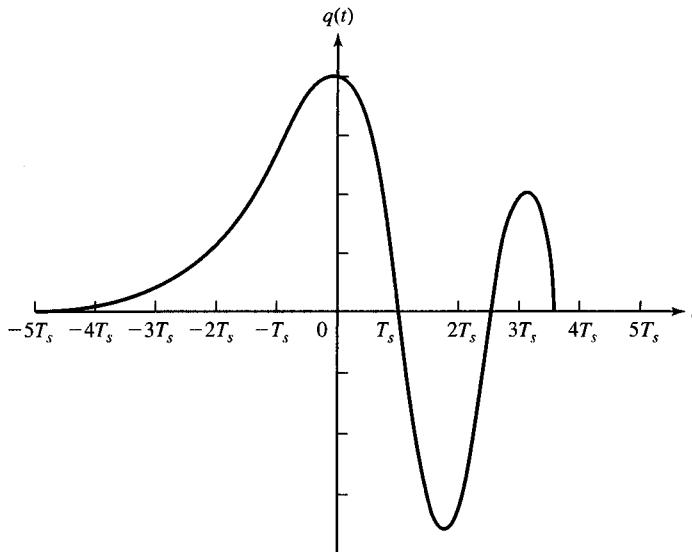


Figure 13.1.5 Effect of channel distortion on the signal pulse in Figure 13.1.4.

interference from several adjacent symbols. The distorted signal is also corrupted by additive noise, which is usually wideband.

At the receiving end of the communication system, the signal is first passed through a filter that is designed primarily to eliminate the noise outside of the frequency band occupied by the signal. We may assume that this filter is a linear-phase FIR filter that limits the bandwidth of the noise but causes negligible additional distortion on the channel-corrupted signal.

Samples of the received signal at the output of this filter reflect the presence of intersymbol interference and additive noise. If we ignore for the moment the possible time variations in the channel, we may express the sampled output at the receiver as

$$\begin{aligned}
 x(nT_s) &= \sum_{k=0}^{\infty} a(k)q(nT_s - kT_s) + w(nT_s) \\
 &= a(n)q(0) + \sum_{\substack{k=0 \\ k \neq n}}^{\infty} a(k)q(nT_s - kT_s) + w(nT_s)
 \end{aligned} \tag{13.1.6}$$

where $w(t)$ represents the additive noise and $q(t)$ represents the distorted pulse at the output of the receiver filter.

To simplify our discussion, we assume that the sample $q(0)$ is normalized to unity by means of an automatic gain control (AGC) contained in the receiver. Then, the

sampled signal given in (13.1.6) may be expressed as

$$x(n) = a(n) + \sum_{\substack{k=0 \\ k \neq n}}^{\infty} a(k)q(n-k) + w(n) \quad (13.1.7)$$

where $x(n) \equiv x(nT_s)$, $q(n) \equiv q(nT_s)$, and $w(n) \equiv w(nT_s)$. The term $a(n)$ in (13.1.7) is the desired symbol at the n th sampling instant. The second term,

$$\sum_{\substack{k=0 \\ k \neq n}}^{\infty} a(k)q(n-k)$$

constitutes the intersymbol interference due to the channel distortion, and $w(n)$ represents the additive noise in the system.

In general, the channel distortion effects embodied through the sampled values $q(n)$ are unknown at the receiver. Furthermore, the channel may vary slowly with time such that the intersymbol interference effects are time variant. The purpose of the adaptive equalizer is to compensate the signal for the channel distortion, so that the resulting signal can be detected reliably. Let us assume that the equalizer is an FIR filter with M adjustable coefficients $h(n)$. Its output may be expressed as

$$\hat{a}(n) = \sum_{k=0}^{M-1} h(k)x(n+D-k) \quad (13.1.8)$$

where D is some nominal delay in processing the signal through the filter and $\hat{a}(n)$ represents an estimate of the n th information symbol. Initially, the equalizer is trained by transmitting a known data sequence $d(n)$. Then, the equalizer output, $\hat{a}(n)$, is compared with $d(n)$ and an error is generated that is used to optimize the filter coefficients.

If we again adopt the least-squares error criterion, we select the coefficients $h(k)$ to minimize the quantity

$$\mathcal{E}_M = \sum_{n=0}^N [d(n) - \hat{a}(n)]^2 = \sum_{n=0}^N \left[d(n) - \sum_{k=0}^{M-1} h(k)x(n+D-k) \right]^2 \quad (13.1.9)$$

The result of the optimization is a set of linear equations of the form

$$\sum_{k=0}^{M-1} h(k)r_{xx}(l-k) = r_{dx}(l-D), \quad l = 0, 1, 2, \dots, M-1 \quad (13.1.10)$$

where $r_{xx}(l)$ is the autocorrelation of the sequence $x(n)$ and $r_{dx}(l)$ is the crosscorrelation between the desired sequence $d(n)$ and the received sequence $x(n)$.

Although the solution of (13.1.10) is obtained recursively in practice (as demonstrated in the following two sections), in principle, we observe that these equations result in values of the coefficients for the initial adjustment of the equalizer. After the short training period, which usually lasts less than one second for most channels, the transmitter begins to transmit the information sequence $a(n)$. In order to track the possible time variations in the channel, the equalizer coefficients must continue to be adjusted in an adaptive manner while receiving data. As illustrated in Figure 13.1.3, this is usually accomplished by treating the decisions at the output of the decision device as correct, and using the decisions in place of the reference $d(n)$ to generate the error signal. This approach works quite well when decision errors occur infrequently (for example, less than one decision error per hundred symbols). The occasional decision errors cause only small misadjustments in the equalizer coefficients. In Sections 13.2 and 13.3, we describe the adaptive algorithms for recursively adjusting the equalizer coefficients.

13.1.3 Echo Cancellation in Data Transmission over Telephone Channels

In the transmission of data over telephone channels, modems (modulator/demodulators) are used to provide an interface between the digital data sequence and the analog channel. Shown in Figure 13.1.6 is a block diagram of a communication system in which two terminals, labeled A and B, transmit data by using modems A and B to interface to a telephone channel. As shown, a digital sequence $a(n)$ is transmitted from terminal A to terminal B while another digital sequence $b(n)$ is transmitted from terminal B to A. This simultaneous transmission in both directions is called *full-duplex transmission*.

As described, the two transmitted signals may be represented as

$$s_A(t) = \sum_{k=0}^{\infty} a(k) p(t - kT_s) \quad (13.1.11)$$

$$s_B(t) = \sum_{k=0}^{\infty} b(k) p(t - kT_s) \quad (13.1.12)$$

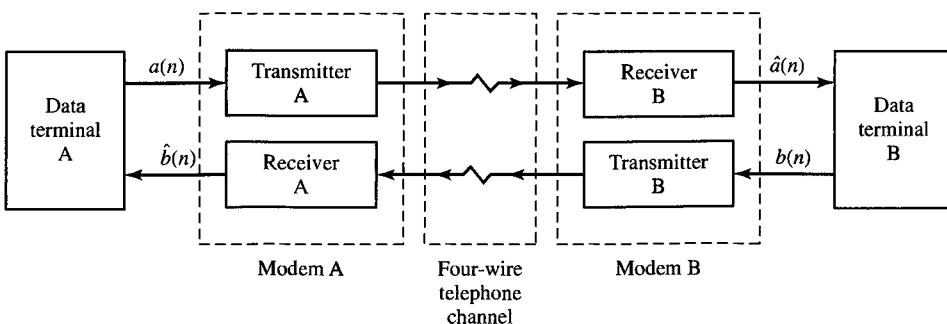


Figure 13.1.6 Full-duplex data transmission over telephone channels.

where $p(t)$ is a pulse as shown in Figure 13.1.4.

When a subscriber leases a private line from a telephone company for the purpose of transmitting data between terminals A and B, the telephone line provided is a four-wire line, which is equivalent to having two dedicated telephone (two-wire) channels, one (pair of wires) for transmitting data in one direction and one (pair of wires) for receiving data from the other direction. In such a case the two transmission paths are isolated and, consequently, there is no “crosstalk” or mutual interference between the two signal paths. Channel distortion is compensated by use of an adaptive equalizer, as previously described, at the receiver of each modem.

The major problem with the system shown in Figure 13.1.6 is the cost of leasing a four-wire telephone channel. If the volume of traffic is high and the telephone channel is used either continuously or frequently, as in banking transactions systems or airline reservation systems, the system pictured in Figure 13.1.6 may be cost effective. Otherwise, it will not be.

An alternative solution for low-volume, infrequent transmission of data is to use the dial-up switched telephone network. In this case, the local communication link between the subscriber and the local central telephone office is a two-wire line, called the *local loop*. At the central office, the subscriber two-wire line is connected to the main four-wire telephone channels that interconnect different central offices, called *trunk lines*, by a device called a *hybrid*. By using transformer coupling, the hybrid is tuned to provide isolation between the transmission and reception channels in full-duplex operation. However, due to impedance mismatch between the hybrid and the telephone channel, the level of isolation is often insufficient and, consequently, some of the signal on the transmitter side leaks back and corrupts the signal on the receiver side, causing an “echo” that is often heard in voice communications over telephone channels.

To mitigate the echoes in voice transmissions, the telephone companies employ a device called an *echo suppressor*. In data transmission, the solution is to use an *echo canceller* within each modem. The echo cancellers are implemented as adaptive filters with automatically adjustable coefficients, just as in the case of transversal equalizers.

With the use of hybrids to couple a two-wire to a four-wire channel, and echo cancellers at each modem to estimate and subtract the echoes, the data communication system for the dial-up switched network takes the form shown in Figure 13.1.7. A hybrid is needed at each modem to isolate the transmitter from the receiver and to couple to the two-wire local loop. Hybrid A is physically located at the central office of subscriber A while hybrid B is located at the central office to which subscriber B is connected. The two central offices are connected by a four-wire line, one pair used for transmission from A to B and the other pair used for transmission in the reverse direction, from B to A. An echo at terminal A due to the hybrid A is called a *near-end echo*, while an echo at terminal A due to the hybrid B is termed a *far-end echo*. Both types of echoes are usually present in data transmission and must be removed by the echo canceller.

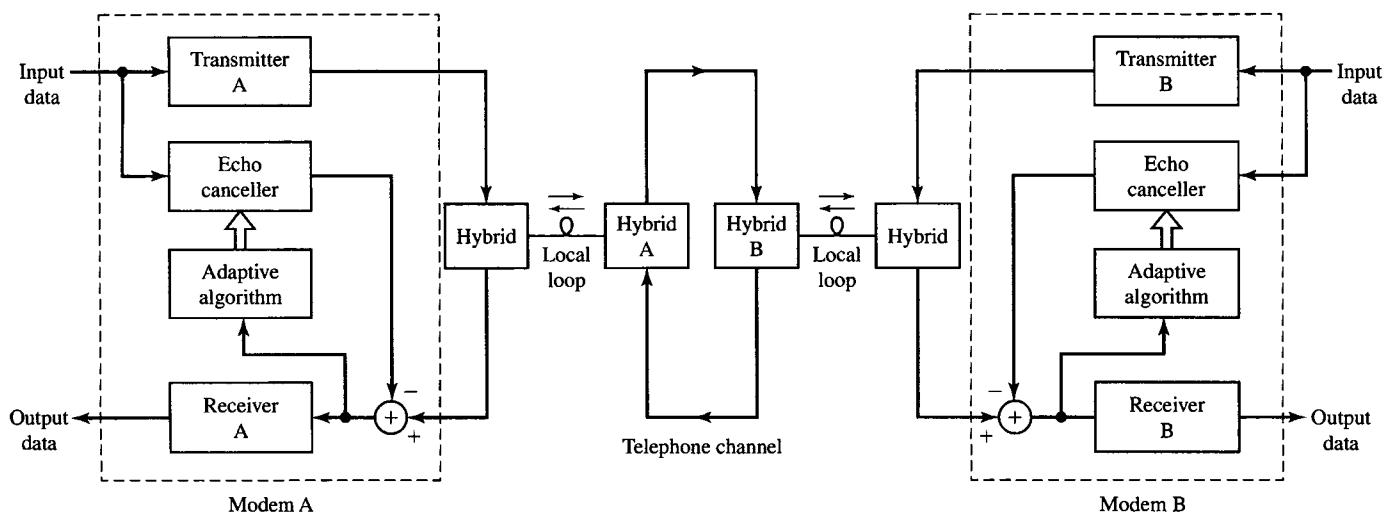


Figure 13.1.7 Block diagram model of a digital communication system that uses echo cancellers in the modems.

Suppose we neglect the channel distortion for purposes of this discussion, and deal with the echoes only. The signal received at modem A may be expressed as

$$s_{RA}(t) = A_1 s_B(t) + A_2 s_A(t - d_1) + A_3 s_A(t - d_2) \quad (13.1.13)$$

where $s_B(t)$ is the desired signal to be demodulated at modem A; $s_A(t - d_1)$ is the near-end echo due to hybrid A; $s_A(t - d_2)$ is the far-end echo due to hybrid B; A_i , $i = 1, 2, 3$, are the corresponding amplitudes of the three signal components; and d_1 and d_2 are the delays associated with the echo components. A further disturbance that corrupts the received signal is additive noise, so that the received signal at modem A is

$$r_A(t) = s_{RA}(t) + w(t) \quad (13.1.14)$$

where $w(t)$ represents the additive noise process.

The adaptive echo canceller attempts to estimate adaptively the two echo components. If its coefficients are $h(n)$, $n = 0, 1, \dots, M - 1$, its output is

$$\hat{s}_A(n) = \sum_{k=0}^{M-1} h(k) a(n - k) \quad (13.1.15)$$

which is an estimate of the echo signal components. This estimate is subtracted from the sampled received signal, and the resulting error signal can be minimized in the least-squares sense to optimally adjust the coefficients of the echo canceller. There are several possible configurations for placement of the echo canceller in the modem, and for forming the corresponding error signal. Figure 13.1.8 illustrates one configuration, in which the canceller output is subtracted from the sampled output of the receiver filter with input $r_A(t)$. Figure 13.1.9 illustrates a second configuration, in which the echo canceller is generating samples at the Nyquist rate instead of the symbol rate; in this case the error signal used to adjust the coefficients is simply the difference between $r_A(n)$, the sampled received signal, and the canceller output. Finally, Figure 13.1.10 illustrates the canceller operating in combination with an adaptive equalizer.

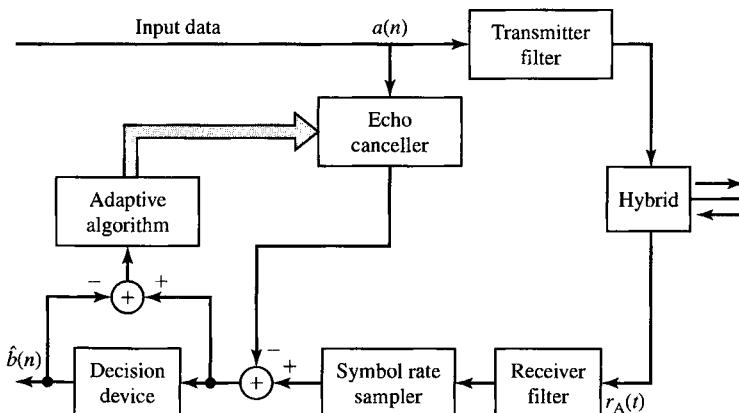


Figure 13.1.8 Symbol rate echo canceller.

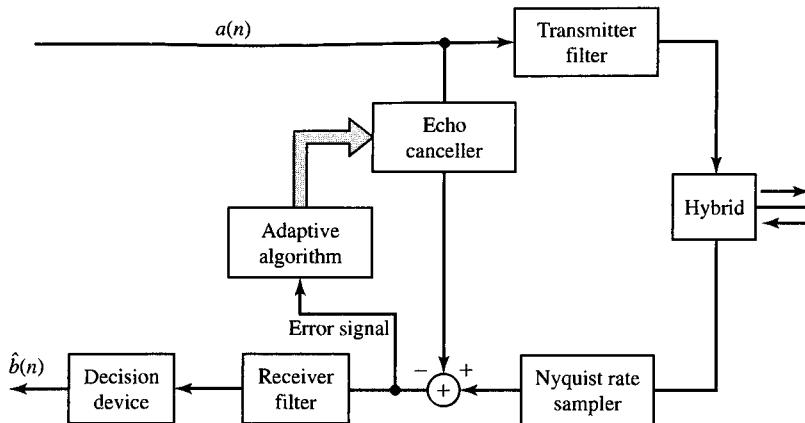


Figure 13.1.9 Nyquist rate echo canceller.

Application of the least-squares criterion in any of the configurations shown in Figures 13.1.8–13.1.10 leads to a set of linear equations for the coefficients of the echo canceller. The reader is encouraged to derive the equations corresponding to the three configurations.

13.1.4 Suppression of Narrowband Interference in a Wideband Signal

We now discuss a problem that arises in practice, especially in signal detection and in digital communications. Let us assume that we have a signal sequence $v(n)$ that

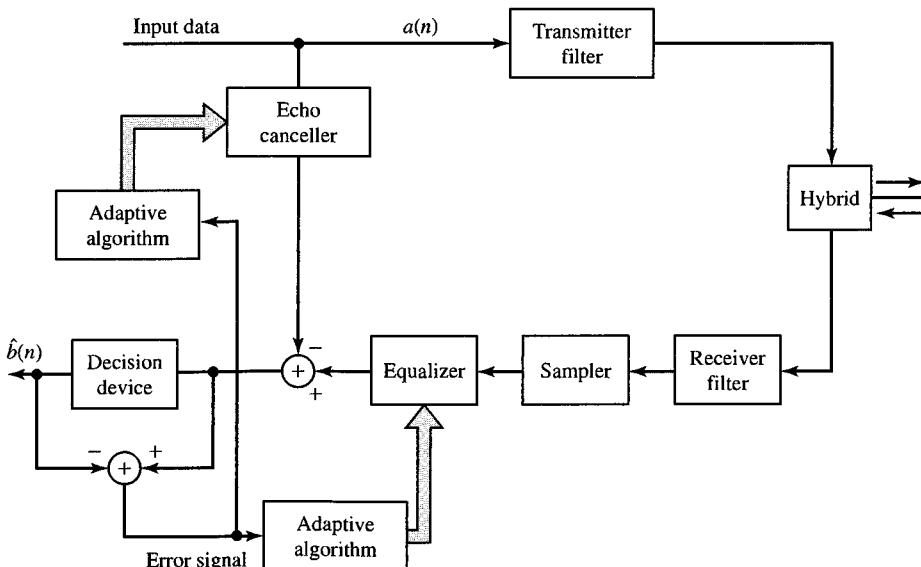


Figure 13.1.10 Modem with adaptive equalizer and echo canceller.

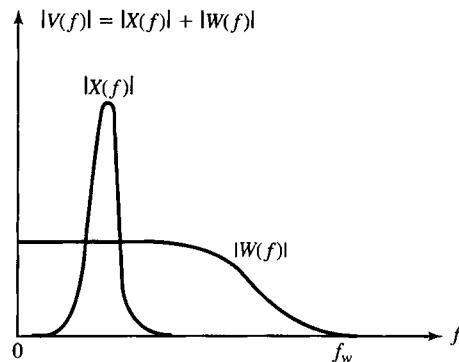


Figure 13.1.11
Strong narrowband interference $X(f)$ in a wideband signal $W(f)$.

consists of a desired wideband signal sequence $w(n)$ corrupted by an additive narrowband interference sequence $x(n)$. The two sequences are uncorrelated. These sequences result from sampling an analog signal $v(t)$ at the Nyquist rate (or faster) of the wideband signal $w(t)$. Figure 13.1.11 illustrates the spectral characteristics of $w(n)$ and $x(n)$. Usually, the interference $|X(f)|$ is much larger than $|W(f)|$ within the narrow frequency band that it occupies.

In digital communications and signal detection problems that fit the above model, the desired signal sequence $w(n)$ is often a *spread-spectrum signal*, while the narrowband interference represents a signal from another user of the frequency band, or intentional interference from a jammer who is trying to disrupt the communications or detection system.

Our objective from a filtering viewpoint is to employ a filter that suppresses the narrowband interference. In effect, such a filter will have a notch in the frequency band occupied by $|X(f)|$, and in practice, the band occupied by $|X(f)|$ is unknown. Moreover, if the interference is nonstationary, its frequency band occupancy may vary with time. Hence, an adaptive filter is desired.

From another viewpoint, the narrowband characteristics of the interference allow us to estimate $x(n)$ from past samples of the sequence $v(n)$ and to subtract the estimate from $v(n)$. Since the bandwidth of $x(n)$ is narrow compared to the bandwidth of the sequence $w(n)$, the samples $x(n)$ are highly correlated due to the high sampling rate. On the other hand, the samples $w(n)$ are not highly correlated, since the samples are taken at the Nyquist rate of $w(n)$. By exploiting the high correlation between $x(n)$ and past samples of the sequence $v(n)$, it is possible to obtain an estimate of $x(n)$, which can be subtracted from $v(n)$.

The general configuration is illustrated in Figure 13.1.12. The signal $v(n)$ is delayed by D samples, where D is selected sufficiently large so that the wideband signal components $w(n)$ and $w(n - D)$ contained in $v(n)$ and $v(n - D)$, respectively, are uncorrelated. Usually, a choice of $D = 1$ or 2 is adequate. The delayed signal sequence $v(n - D)$ is passed through an FIR filter, which is best characterized as a linear predictor of the value $x(n)$ based on M samples $v(n - D - k)$, $k = 0, 1, \dots, M - 1$. The output of the linear predictor is

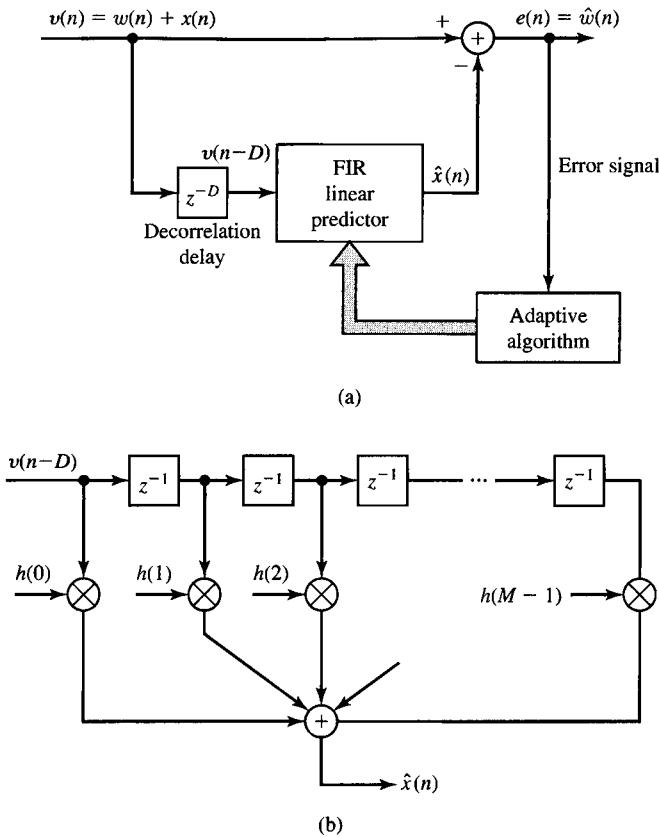


Figure 13.1.12 Adaptive filter for estimating and suppressing a narrowband interference in a wideband signal.

$$\hat{x}(n) = \sum_{k=0}^{M-1} h(k)v(n - D - k) \quad (13.1.16)$$

This predicted value of $x(n)$ is subtracted from $v(n)$ to yield an estimate of $w(n)$, as illustrated in Figure 13.1.12. Clearly, the quality of the estimate $\hat{x}(n)$ determines how well the narrowband interference is suppressed. It is also apparent that the delay D must be kept as small as possible in order to obtain a good estimate of $x(n)$, but must be sufficiently large so that $w(n)$ and $w(n - D)$ are uncorrelated.

Let us define the error sequence

$$\begin{aligned} e(n) &= v(n) - \hat{x}(n) \\ &= v(n) - \sum_{k=0}^{M-1} h(k)v(n - D - k) \end{aligned} \quad (13.1.17)$$

If we apply the least-squares criterion to optimally select the prediction coefficients,

we obtain the set of linear equations

$$\sum_{k=0}^{M-1} h(k)r_{vv}(l-k) = r_{vv}(l+D), \quad l = 0, 1, \dots, M-1 \quad (13.1.18)$$

where $r_{vv}(l)$ is the autocorrelation sequence of $v(n)$. Note, however, that the right-hand side of (13.1.18) may be expressed as

$$\begin{aligned} r_{vv}(l+D) &= \sum_{n=0}^N v(n)v(n-l-D) \\ &= \sum_{n=0}^N [w(n) + x(n)][w(n-l-D) + x(n-l-D)] \\ &= r_{ww}(l+D) + r_{xx}(l+D) + r_{wx}(l+D) + r_{xw}(l+D) \end{aligned} \quad (13.1.19)$$

The correlations in (13.1.19) are time-average correlation sequences. The expected value of $r_{ww}(l+D)$ is

$$E[r_{ww}(l+D)] = 0, \quad l = 0, 1, \dots, M-1 \quad (13.1.20)$$

because $w(n)$ is wideband and D is large enough that $w(n)$ and $w(n-D)$ are uncorrelated. Also,

$$E[r_{xw}(l+D)] = E[r_{wx}(l+D)] = 0 \quad (13.1.21)$$

by assumption. Finally,

$$E[r_{xx}(l+D)] = \gamma_{xx}(l+D) \quad (13.1.22)$$

Therefore, the expected value of $r_{vv}(l+D)$ is simply the statistical autocorrelation of the narrowband signal $x(n)$. Furthermore, if the wideband signal is weak relative to the interference, the autocorrelation $r_{vv}(l)$ in the left-hand side of (13.1.18) is approximately $r_{xx}(l)$. The major influence of $w(n)$ is to the diagonal elements of $r_{vv}(l)$. Consequently, the values of the filter coefficients determined from the linear equations in (13.1.18) are a function of the statistical characteristics of the interference $x(n)$.

The overall filter structure in Figure 13.1.12 is an adaptive FIR prediction-error filter with coefficients

$$h'(k) = \begin{cases} 1, & k = 0 \\ -h(k-D), & k = D, D+1, \dots, D+M-1 \\ 0, & \text{otherwise} \end{cases} \quad (13.1.23)$$

and a frequency response

$$H(\omega) = \sum_{k=0}^{M-1} h'(k+D)e^{-j\omega k} \quad (13.1.24)$$

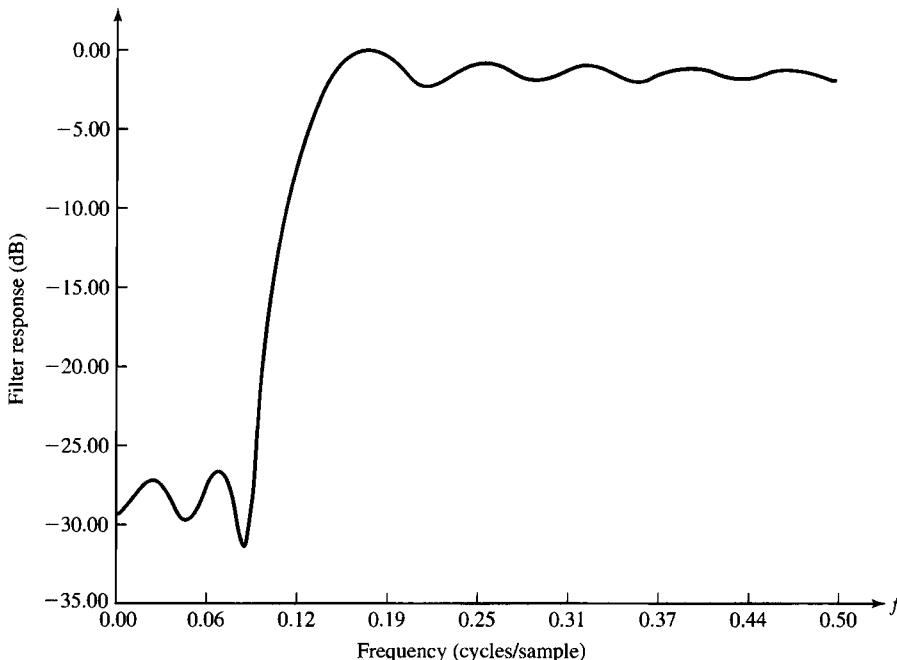


Figure 13.1.13 Frequency response characteristics of an adaptive notch filter.

This overall filter acts as a notch filter for the interference. For example, Figure 13.1.13 illustrates the magnitude of the frequency response of an adaptive filter with $M = 15$ coefficients, which attempts to suppress a narrowband interference that occupies 20 percent of the frequency band of a desired spread-spectrum signal sequence. The data were generated pseudorandomly by adding a narrowband interference consisting of 100 randomly phased, equal-amplitude sinusoids to a pseudonoise spread-spectrum signal. The coefficients of the filter were obtained by solving the equations in (13.1.18), with $D = 1$, where the correlation $r_{vv}(l)$ was obtained from the data. We observe that the overall interference suppression filter has the characteristics of a notch filter. The depth of the notch depends on the power of the interference relative to the wideband signal. The stronger the interference, the deeper the notch.

The algorithms presented in Sections 13.2 and 13.3 are appropriate for estimating the predictor coefficients continuously, in order to track a nonstationary narrowband interference signal.

13.1.5 Adaptive Line Enhancer

In the preceding example, the adaptive linear predictor was used to estimate the narrowband interference for the purpose of suppressing the interference from the input sequence $v(n)$. An adaptive line enhancer (ALE) has the same configuration as the interference suppression filter in Figure 13.1.12, except that the objective is different.

In the adaptive line enhancer, $x(n)$ is the desired signal and $w(n)$ represents

a wideband noise component that masks $x(n)$. The desired signal $x(n)$ is either a spectral line or a relatively narrowband signal. The linear predictor shown in Figure 13.1.12(b) operates in exactly the same fashion as that in Figure 13.1.12(a), and provides an estimate of the narrowband signal $x(n)$. It is apparent that the ALE (i.e., the FIR prediction filter) is a self-tuning filter that has a peak in its frequency response at the frequency of the sinusoid or, equivalently, in the frequency band of the narrowband signal $x(n)$. By having a narrow bandwidth, the noise $w(n)$ outside of the band is suppressed and, thus, the spectral line is enhanced in amplitude relative to the noise power in $w(n)$. This explains why the FIR predictor is called an ALE. Its coefficients are determined by the solution of (13.1.18).

13.1.6 Adaptive Noise Cancelling

Echo cancellation, the suppression of narrowband interference in a wideband signal, and the ALE are related to another form of adaptive filtering called *adaptive noise cancelling*. A model for the adaptive noise canceller is illustrated in Figure 13.1.14.

The primary input signal consists of a desired signal sequence $x(n)$ corrupted by an additive noise sequence $w_1(n)$ and an additive interference (noise) $w_2(n)$. The additive interference (noise) is also observable after it has been filtered by some unknown linear system that yields $v_2(n)$ and is further corrupted by an additive noise sequence $w_3(n)$. Thus, we have available a secondary signal sequence, which may be expressed as $v(n) = v_2(n) + w_3(n)$. The sequences $w_1(n)$, $w_2(n)$, and $w_3(n)$ are assumed to be mutually uncorrelated and zero mean.

As shown in Figure 13.1.14, an adaptive FIR filter is used to estimate the interference sequence $w_2(n)$ from the secondary signal $v(n)$ and subtract the estimate $\hat{w}_2(n)$ from the primary signal. The output sequence, which represents an estimate of the desired signal $x(n)$, is the error signal

$$\begin{aligned} e(n) &= y(n) - \hat{w}_2(n) \\ &= y(n) - \sum_{k=0}^{M-1} h(k)v(n-k) \end{aligned} \quad (13.1.25)$$

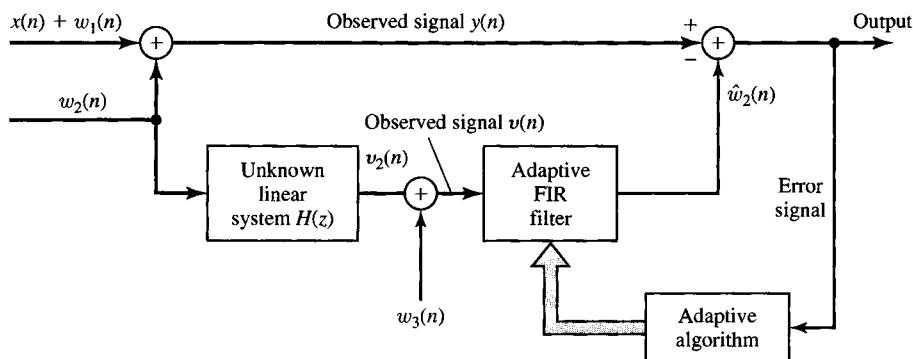


Figure 13.1.14 Example of an adaptive noise-cancelling system.

This error sequence is used to adaptively adjust the coefficients of the FIR filter.

If the least-squares criterion is used to determine the filter coefficients, the result of the optimization is the set of linear equations

$$\sum_{k=0}^{M-1} h(k)r_{vv}(l-k) = r_{yv}(l), \quad l = 0, 1, \dots, M-1 \quad (13.1.26)$$

where $r_{vv}(l)$ is the sample (time-average) autocorrelation of the sequence $v(n)$ and $r_{yv}(l)$ is the sample crosscorrelation of the sequences $y(n)$ and $v(n)$. Clearly, the noise cancelling problem is similar to the last three adaptive filtering applications described above.

13.1.7 Linear Predictive Coding of Speech Signals

Various methods have been developed over the past four decades for digital encoding of speech signals. In the telephone system, for example, two commonly used methods for speech encoding are pulse code modulation (PCM) and differential PCM (DPCM). These are examples of *waveform coding* methods. Other waveform coding methods have also been developed, such as delta modulation (DM) and adaptive DPCM.

Since the digital speech signal is ultimately transmitted from the source to a destination, a primary objective in devising speech encoders is to minimize the number of bits required to represent the speech signal, while maintaining speech intelligibility. This objective has led to the development of a class of low bit-rate (10,000 bits per second and below) speech encoding methods, which are based on constructing a model of the speech source and transmitting the model parameters. Adaptive filtering finds application in these model-based speech coding systems. We describe one very effective method called *linear predictive coding* (LPC).

In LPC, the vocal tract is modeled as a linear all-pole filter having the system function

$$H(z) = \frac{G}{1 - \sum_{k=1}^p a_k z^{-k}} \quad (13.1.27)$$

where p is the number of poles, G is the filter gain, and a_k are the parameters that determine the poles. There are two mutually exclusive excitation functions, used to model voiced and unvoiced speech sounds. On a short-time basis, voiced speech is periodic with a fundamental frequency F_0 , or a pitch period $1/F_0$, which depends on the speaker. Thus, voiced speech is generated by exciting the all-pole filter model by a periodic impulse train with a period equal to the desired pitch period. Unvoiced speech sounds are generated by exciting the all-pole filter model by the output of a random-noise generator. This model is shown in Figure 13.1.15.

Given a short-time segment of a speech signal, the speech encoder at the transmitter must determine the proper excitation function, the pitch period for voiced speech, the gain parameter G , and the coefficients $\{a_k\}$. A block diagram that illustrates the source encoding system is given in Figure 13.1.16. The parameters of the model are determined adaptively from the data. Then, the speech samples are

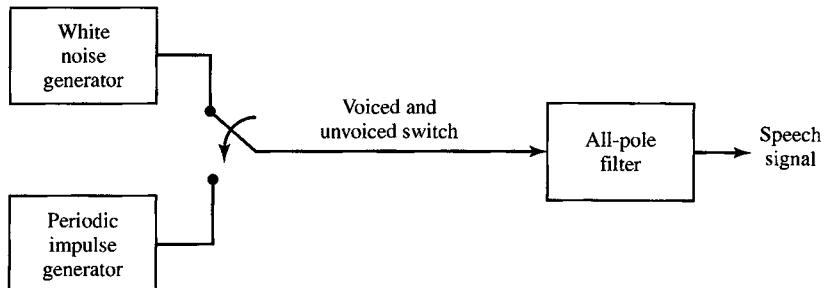


Figure 13.1.15 Block diagram model for the generation of a speech signal.

synthesized by using the model, and an error signal sequence is generated (as shown in Figure 13.1.16) by taking the difference between the actual and the synthesized sequence. The error signal and the model parameters are encoded into a binary sequence and transmitted to the destination. At the receiver, the speech signal is synthesized from the model and the error signal.

The parameters of the all-pole filter model are easily determined from the speech samples by means of linear prediction. To be specific, consider the system shown in Figure 13.1.17 and assume that we have N signal samples. The output of the FIR filter is

$$\hat{x}(n) = \sum_{k=1}^p a_k x(n-k) \quad (13.1.28)$$

and the corresponding error between the observed sample $x(n)$ and the estimate $\hat{x}(n)$ is

$$e(n) = x(n) - \sum_{k=1}^p a_k x(n-k) \quad (13.1.29)$$

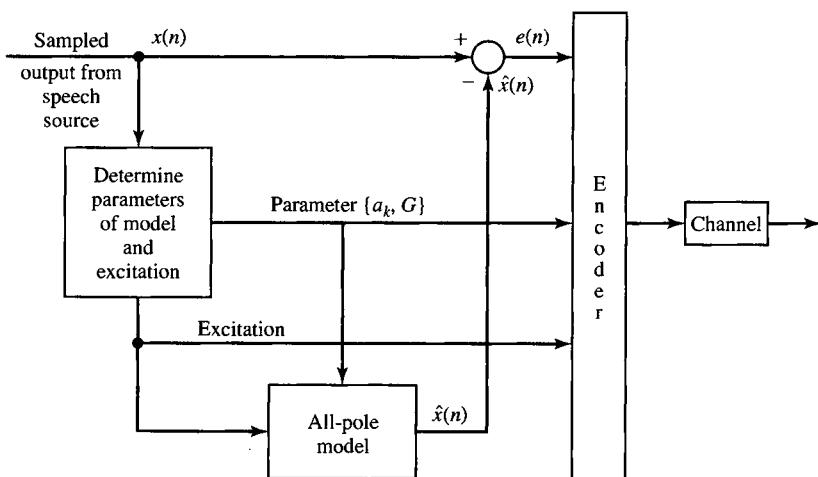


Figure 13.1.16 Source encoder for a speech signal.

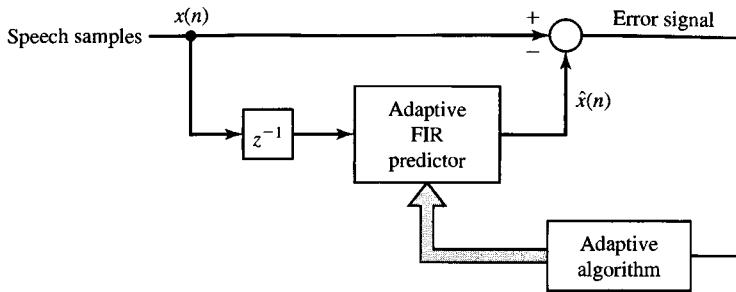


Figure 13.1.17 Estimation of pole parameters in LPC.

By applying the least-squares criterion, we can determine the model parameters a_k . The result of this optimization is a set of linear equations

$$\sum_{k=1}^p a_k r_{xx}(l-k) = r_{xx}(l), \quad l = 1, 2, \dots, p \quad (13.1.30)$$

where $r_{xx}(l)$ is the time-average autocorrelation of the sequence $x(n)$. The gain parameter for the filter can be obtained by noting that its input-output equation is

$$x(n) = \sum_{k=1}^p a_k x(n-k) + Gv(n) \quad (13.1.31)$$

where $v(n)$ is the input sequence. Clearly,

$$\begin{aligned} Gv(n) &= x(n) - \sum_{k=1}^p a_k x(n-k) \\ &= e(n) \end{aligned}$$

Then,

$$G^2 \sum_{n=0}^{N-1} v^2(n) = \sum_{n=0}^{N-1} e^2(n) \quad (13.1.32)$$

If the input excitation is normalized to unit energy by design, then

$$\begin{aligned} G^2 &= \sum_{n=0}^{N-1} e^2(n) \\ &= r_{xx}(0) - \sum_{k=1}^p a_k r_{xx}(k) \end{aligned} \quad (13.1.33)$$

Thus, G^2 is set equal to the residual energy resulting from the least-squares optimization.

In this development, we have described the use of linear prediction to adaptively determine the pole parameters and the gain of an all-pole filter model for speech generation. In practice, due to the nonstationary character of speech signals, this model is applied to short-time segments (10 to 20 milliseconds) of a speech signal. Usually, a new set of parameters is determined for each short-time segment. However, it is often advantageous to use the model parameters measured from previous segments to smooth out sharp discontinuities that usually exist in estimates of model parameters obtained from segment to segment. Although our discussion was totally in terms of the FIR filter structure, we should mention that speech synthesis is usually performed by using the FIR lattice structure and the reflection coefficients K_i . Since the dynamic range of the K_i is significantly smaller than that of the a_k , the reflection coefficients require fewer bits to represent them. Hence, the K_i are transmitted over the channel. Consequently, it is natural to synthesize the speech at the destination using the all-pole lattice structure.

In our treatment of LPC for speech coding, we have not considered algorithms for the estimation of the excitation and the pitch period. A discussion of appropriate algorithms for these parameters of the model would take us too far afield and, hence, is omitted. The interested reader is referred to Rabiner and Schafer (1978) and Deller, Hansen and Proakis (2000) for a detailed treatment of speech analysis and synthesis methods.

13.1.8 Adaptive Arrays

In the previous examples, we considered adaptive filtering performed on a single data sequence. However, adaptive filtering has also been widely applied to multiple data sequences that result from antenna, hydrophone, and seismometer arrays, where the sensors (antennas, hydrophones, or seismometers) are arranged in some spatial configuration. Each element of the array of sensors provides a signal sequence. By properly combining the signals from the various sensors, it is possible to change the directivity pattern of the array. For example, consider a linear antenna array consisting of five elements, as shown in Figure 13.1.18(a). If the signals are simply linearly summed, we obtain the sequence

$$x(n) = \sum_{k=1}^5 x_k(n) \quad (13.1.34)$$

which results in the antenna directivity pattern shown in Figure 13.1.18(a). Now, suppose that an interference signal is received from a direction corresponding to one of the sidelobes in the array. By properly weighting the sequences $x_k(n)$ prior to combining, it is possible to alter the sidelobe pattern such that the array contains a null in the direction of the interference, as shown in Figure 13.1.18(b). Thus, we obtain

$$x(n) = \sum_{k=1}^5 h_k x_k(n) \quad (13.1.35)$$

where the h_k are the weights.

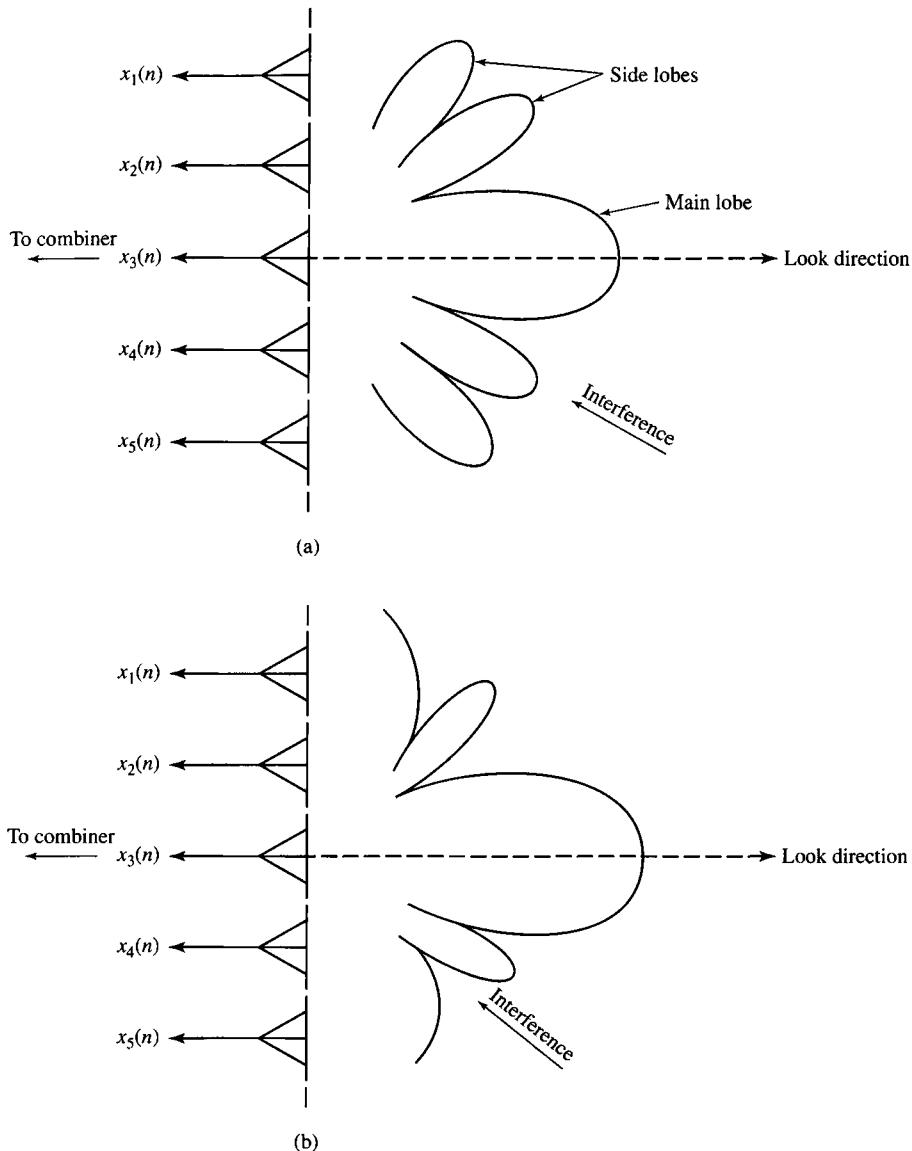


Figure 13.1.18 Linear antenna array: (a) linear antenna array with antenna pattern; (b) linear antenna array with a null placed in the direction of the interference.

We may also change or steer the direction of the main antenna lobe by simply introducing delays in the output of the sensor signals prior to combining. Hence, from K sensors we have a combined signal of the form

$$x(n) = \sum_{k=1}^K h_k x_k(n - n_k) \quad (13.1.36)$$

where the h_k are the weights and n_k corresponds to an n_k -sample delay in the signal $x(n)$. The choice of weights may be used to place nulls in specific directions.

More generally, we may simply filter each sequence prior to combining. In such a case, the output sequence has the general form

$$\begin{aligned} y(n) &= \sum_{k=1}^K y_k(n) \\ &= \sum_{k=1}^K \sum_{l=0}^{M-1} h_k(l)x_k(n - n_k - l) \end{aligned} \quad (13.1.37)$$

where $h_k(l)$ is the impulse response of the filter for processing the k th sensor output and the n_k are the delays that steer the beam pattern.

The LMS algorithm described in Section 13.2.2 is frequently used in adaptively selecting the weights h_k or the impulse responses $h_k(l)$. The more powerful recursive least-squares algorithms described in Section 13.3 can also be applied to the multisensor (multichannel) data problem.

13.2 Adaptive Direct-Form FIR Filters—The LMS Algorithm

From the examples of the previous section, we observed that there is a common framework in all the adaptive filtering applications. The least-squares criterion that we have adopted leads to a set of linear equations for the filter coefficients, which may be expressed as

$$\sum_{k=0}^{M-1} h(k)r_{xx}(l-k) = r_{dx}(l+D), \quad l = 0, 1, 2, \dots, M-1 \quad (13.2.1)$$

where $r_{xx}(l)$ is the autocorrelation of the sequence $x(n)$ and $r_{dx}(l)$ is the crosscorrelation of the sequences $d(n)$ and $x(n)$. The delay parameter D is zero in some cases and nonzero in others.

We observe that the autocorrelation $r_{xx}(l)$ and the crosscorrelation $r_{dx}(l)$ are obtained from the data and, hence, represent estimates of the true (statistical) autocorrelation and crosscorrelation sequences. As a result, the coefficients $h(k)$ obtained from (13.2.1) are estimates of the true coefficients. The quality of the estimates depends on the length of the data record that is available for estimating $r_{xx}(l)$ and $r_{dx}(l)$. This is one problem that must be considered in the implementation of an adaptive filter.

A second problem that must be considered is that the underlying random process $x(n)$ is usually nonstationary. For example, in channel equalization, the frequency response characteristics of the channel may vary with time. As a consequence, the statistical autocorrelation and crosscorrelation sequences—and, hence, their estimates—vary with time. This implies that the coefficients of the adaptive filter must change with time to incorporate the time-variant statistical characteristics

of the signal into the filter. This also implies that the quality of the estimates cannot be made arbitrarily high by simply increasing the number of signal samples used in the estimation of the autocorrelation and crosscorrelation sequences.

There are several ways by which the coefficients of the adaptive filter can be varied with time to track the time-variant statistical characteristics of the signal. The most popular method is to adapt the filter recursively on a sample-by-sample basis, as each new signal sample is received. A second approach is to estimate $r_{xx}(l)$ and $r_{dx}(l)$ on a block-by-block basis, with no attempt to maintain continuity in the values of the filter coefficients from one block of data to another. In such a scheme, the block size must be relatively small, encompassing a time interval that is short compared to the time interval over which the statistical characteristics of the data change significantly. In addition to this block processing method, other block processing schemes can be devised that incorporate some block-to-block continuity in the filter coefficients.

In our treatment of adaptive filtering algorithms, we consider only time-recursive algorithms that update the filter coefficients on a sample-by-sample basis. In particular, we consider two types of algorithms, the LMS algorithm, which is based on a gradient-type search for tracking the time-variant signal characteristics, and the class of recursive least-squares algorithms, which are significantly more complex than the LMS algorithm, but which provide faster convergence to changes in signal statistics.

13.2.1 Minimum Mean-Square-Error Criterion

The LMS algorithm that is described in the following subsection is most easily obtained by formulating the optimization of the FIR filter coefficients as an estimation problem based on the minimization of the mean-square error. Let us assume that we have available the (possibly complex-valued) data sequence $x(n)$, which consists of samples from a stationary random process with autocorrelation sequence

$$\gamma_{xx}(m) = E[x(n)x^*(n - m)] \quad (13.2.2)$$

From these samples, we form an estimate of the desired sequence $d(n)$ by passing the observed data $x(n)$ through an FIR filter with coefficients $h(n)$, $0 \leq n \leq M - 1$. The filter output may be expressed as

$$\hat{d}(n) = \sum_{k=0}^{M-1} h(k)x(n - k) \quad (13.2.3)$$

where $\hat{d}(n)$ represents an estimate of $d(n)$. The estimation error is defined as

$$\begin{aligned} e(n) &= d(n) - \hat{d}(n) \\ &= d(n) - \sum_{k=0}^{M-1} h(k)x(n - k) \end{aligned} \quad (13.2.4)$$

The mean-square error as a function of the filter coefficients is

$$\begin{aligned}
 \mathcal{E}_M &= E[|e(n)|^2] \\
 &= E \left[\left| d(n) - \sum_{k=0}^{M-1} h(k)x(n-k) \right|^2 \right] \\
 &= E \left\{ |d(n)|^2 - 2\operatorname{Re} \left[\sum_{k=0}^{M-1} h^*(l)d(n)x^*(n-l) \right] + \sum_{k=0}^{M-1} \sum_{l=0}^{M-1} h^*(l)h(k)x^*(n-l)x(n-k) \right\} \\
 &= \sigma_d^2 - 2\operatorname{Re} \left[\sum_{l=0}^{M-1} h^*(l)\gamma_{dx}(l) \right] + \sum_{l=0}^{M-1} \sum_{k=0}^{M-1} h^*(l)h(k)\gamma_{xx}(l-k)
 \end{aligned} \tag{13.2.5}$$

where, by definition, $\sigma_d^2 = E[|d(n)|^2]$.

We observe that the MSE is a quadratic function of the filter coefficients. Consequently, the minimization of \mathcal{E}_M with respect to the coefficients leads to the set of M linear equations,

$$\sum_{k=0}^{M-1} h(k)\gamma_{xx}(l-k) = \gamma_{dx}(l), \quad l = 0, 1, \dots, M-1 \tag{13.2.6}$$

The filter with coefficients obtained from (13.2.6), which is the Wiener–Hopf equation previously derived in Section 12.7.1, is called the *Wiener filter*.

If we compare (13.2.6) with (13.2.1), it is apparent that these equations are similar in form. In (13.2.1), we use estimates of the autocorrelation and crosscorrelation to determine the filter coefficients, whereas in (13.2.6) the statistical autocorrelation and crosscorrelation are employed. Hence, (13.2.6) yields the optimum (Wiener) filter coefficients in the MSE sense, whereas (13.2.1) yields estimates of the optimum coefficients.

The equations in (13.2.6) may be expressed in matrix form as

$$\boldsymbol{\Gamma}_M \mathbf{h}_M = \boldsymbol{\gamma}_d \tag{13.2.7}$$

where \mathbf{h}_M denotes the vector of coefficients, $\boldsymbol{\Gamma}_M$ is an $M \times M$ (Hermitian) Toeplitz matrix with elements $\Gamma_{lk} = \gamma_{xx}(l-k)$, and $\boldsymbol{\gamma}_d$ is an $M \times 1$ crosscorrelation vector with elements $\gamma_{dx}(l)$, $l = 0, 1, \dots, M-1$. The complex-conjugate of \mathbf{h}_M is denoted as \mathbf{h}_M^* and the transpose as \mathbf{h}_M^T . The solution for the optimum filter coefficients is

$$\mathbf{h}_{\text{opt}} = \boldsymbol{\Gamma}_M^{-1} \boldsymbol{\gamma}_d \tag{13.2.8}$$

and the resulting minimum MSE achieved with the optimum coefficients given by (13.2.8) is

$$\begin{aligned}
 \mathcal{E}_{M \text{ min}} &= \sigma_d^2 - \sum_{k=0}^{M-1} h_{\text{opt}}(k)\gamma_{dx}^*(k) \\
 &= \sigma_d^2 - \boldsymbol{\gamma}_d^H \boldsymbol{\Gamma}_M^{-1} \boldsymbol{\gamma}_d
 \end{aligned} \tag{13.2.9}$$

where the exponent H denotes the conjugate transpose.

Recall that the set of linear equations in (13.2.6) can also be obtained by invoking the orthogonality principle in mean-square estimation (see Section 12.7.2). According to the orthogonality principle, the mean-square estimation error is minimized when the error $e(n)$ is orthogonal, in the statistical sense, to the estimate $\hat{d}(n)$, that is,

$$E[e(n)\hat{d}^*(n)] = 0 \quad (13.2.10)$$

But the condition in (13.2.10) implies that

$$E\left[\sum_{k=0}^{M-1} h(k)e(n)x^*(n-k)\right] = \sum_{k=0}^{M-1} h(k)E[e(n)x^*(n-k)] = 0$$

or, equivalently,

$$E[e(n)x^*(n-l)] = 0, \quad l = 0, 1, \dots, M-1 \quad (13.2.11)$$

If we substitute for $e(n)$ in (13.2.11) using the expression for $e(n)$ given in (13.2.4), and perform the expectation operation, we obtain the equations given in (13.2.6).

Since $\hat{d}(n)$ is orthogonal to $e(n)$, the residual (minimum) mean-square error is

$$\begin{aligned} \mathcal{E}_M \text{ min} &= E[e(n)d^*(n)] \\ &= E[|d(n)|^2] - \sum_{k=0}^{M-1} h_{\text{opt}}(k)\gamma_d^*(k) \end{aligned} \quad (13.2.12)$$

which is the result given in (13.2.9).

The optimum filter coefficients given by (13.2.8) can be solved efficiently by using the Levinson-Durbin algorithm. However, we shall consider the use of a gradient method for solving for \mathbf{h}_{opt} , iteratively. This development leads to the LMS algorithm for adaptive filtering.

13.2.2 The LMS Algorithm

There are various numerical methods that can be used to solve the set of linear equations given by (13.2.6) or (13.2.7) for the optimum FIR filter coefficients. In the following, we consider recursive methods that have been devised for finding the minimum of a function of several variables. In our problem, the performance index is the MSE given by (13.2.5), which is a quadratic function of the filter coefficients. Hence, this function has a unique minimum, which we shall determine by an iterative search.

For the moment, let us assume that the autocorrelation matrix Γ_M and the cross-correlation vector γ_d are known. Hence, \mathcal{E}_M is a known function of the coefficients $h(n)$, $0 \leq n \leq M-1$. Algorithms for recursively computing the filter coefficients and, thus, searching for the minimum of \mathcal{E}_M , have the form

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \frac{1}{2}\Delta(n)\mathbf{S}(n), \quad n = 0, 1, \dots \quad (13.2.13)$$

where $\mathbf{h}_M(n)$ is the vector of filter coefficients at the n th iteration, $\Delta(n)$ is a step size at the n th iteration, and $\mathbf{S}(n)$ is a direction vector for the n th iteration. The initial vector $\mathbf{h}_M(0)$ is chosen arbitrarily. In this treatment we exclude methods that require the computations of Γ_M^{-1} , such as Newton's method, and consider only search methods based on the use of gradient vectors.

The simplest method for finding the minimum of \mathcal{E}_M recursively is based on a steepest-descent search (see Murray (1972)). In the method of steepest descent, the direction vector $\mathbf{S}(n) = -\mathbf{g}(n)$, where $\mathbf{g}(n)$ is the gradient vector at the n th iteration, defined as

$$\begin{aligned}\mathbf{g}(n) &= \frac{d\mathcal{E}_M(n)}{d\mathbf{h}_M(n)} \\ &= 2[\Gamma_M \mathbf{h}_M(n) - \gamma_d], \quad n = 0, 1, 2, \dots\end{aligned}\tag{13.2.14}$$

Hence, we compute the gradient vector at each iteration and change the values of $\mathbf{h}_M(n)$ in a direction opposite the gradient. Thus, the recursive algorithm based on the method of steepest descent is

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) - \frac{1}{2} \Delta(n) \mathbf{g}(n)\tag{13.2.15}$$

or, equivalently,

$$\mathbf{h}_M(n+1) = [\mathbf{I} - \Delta(n) \Gamma_M] \mathbf{h}_M(n) + \Delta(n) \gamma_d\tag{13.2.16}$$

We state without proof that the algorithm leads to the convergence of $\mathbf{h}_M(n)$ to \mathbf{h}_{opt} in the limit as $n \rightarrow \infty$, provided that the sequence of step sizes $\Delta(n)$ is absolutely summable, with $\Delta(n) \rightarrow 0$ as $n \rightarrow \infty$. It follows that as $n \rightarrow \infty$, $\mathbf{g}(n) \rightarrow \mathbf{0}$.

Other candidate algorithms that provide faster convergence are the *conjugate-gradient* algorithm and the Fletcher-Powell algorithm. In the conjugate-gradient algorithm, the direction vectors are given as

$$\mathbf{S}(n) = \beta(n-1) \mathbf{S}(n-1) - \mathbf{g}(n)\tag{13.2.17}$$

where $\beta(n)$ is a scalar function of the gradient vectors (see Beckman (1960)). In the Fletcher-Powell algorithm, the direction vectors are given as

$$\mathbf{S}(n) = -\mathbf{H}(n) \mathbf{g}(n)\tag{13.2.18}$$

where $\mathbf{H}(n)$ is an $M \times M$ positive definite matrix, computed iteratively, that converges to the inverse of Γ_M (see Fletcher and Powell (1963)). Clearly, the three algorithms differ in the manner in which the direction vectors are computed.

These three algorithms are appropriate when Γ_M and γ_d are known. However, this is not the case in adaptive filtering applications, as we have previously indicated. In the absence of knowledge of Γ_M and γ_d , we may substitute estimates $\hat{\mathbf{S}}(n)$ of the direction vectors in place of the actual vectors $\mathbf{S}(n)$. We consider this approach for the steepest-descent algorithm.

First, we note that the gradient vector given by (13.2.14) may also be expressed in terms of the orthogonality conditions given by (13.2.11). In fact, the conditions in (13.2.11) are equivalent to the expression

$$E[e(n)\mathbf{X}_M^*(n)] = \boldsymbol{\gamma}_d - \boldsymbol{\Gamma}_M \mathbf{h}_M(n) \quad (13.2.19)$$

where $\mathbf{X}_M(n)$ is the vector with elements $x(n-l)$, $l = 0, 1, \dots, M-1$. Therefore, the gradient vector is simply

$$\mathbf{g}(n) = -2E[e(n)\mathbf{X}_M^*(n)] \quad (13.2.20)$$

Clearly, the gradient vector $\mathbf{g}(n) = \mathbf{0}$ when the error is orthogonal to the data in the estimate $\hat{d}(n)$.

An unbiased estimate of the gradient vector at the n th iteration is simply obtained from (13.2.20) as

$$\hat{\mathbf{g}}(n) = -2e(n)\mathbf{X}_M^*(n) \quad (13.2.21)$$

where $e(n) = d(n) - \hat{d}(n)$, and $\mathbf{X}_M(n)$ is the set of M signal samples in the filter at the n th iteration. Thus, with $\hat{\mathbf{g}}(n)$ substituted for $\mathbf{g}(n)$, we have the algorithm

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \Delta(n)e(n)\mathbf{X}_M^*(n) \quad (13.2.22)$$

This is called a *stochastic-gradient-descent algorithm*. As given by (13.2.22), it has a variable step size.

It has become common practice in adaptive filtering to use a fixed-step-size algorithm for two reasons. The first is that a fixed-step-size algorithm is easily implemented in either hardware or software. The second is that a fixed step size is appropriate for tracking time-variant signal statistics, whereas if $\Delta(n) \rightarrow 0$ as $n \rightarrow \infty$, adaptation to signal variations cannot occur. For these reasons, (13.2.22) is modified to the algorithm

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \Delta e(n)\mathbf{X}_M^*(n) \quad (13.2.23)$$

where Δ is now the fixed step size. This algorithm was first proposed by Widrow and Hoff (1960) and is now widely known as the *LMS (least-mean-squares) algorithm*. Clearly, it is a stochastic-gradient algorithm.

The LMS algorithm is relatively simple to implement. For this reason, it has been widely used in many adaptive filtering applications. Its properties and limitations have also been thoroughly investigated. In the following section, we provide a brief treatment of its important properties concerning convergence, stability, and the noise resulting from the use of estimates of the gradient vectors. Subsequently, we compare its properties with the more complex recursive least-squares algorithms.

13.2.3 Related Stochastic Gradient Algorithms

Several variations of the basic LMS algorithm have been proposed in the literature and implemented in adaptive filtering applications. One variation is obtained if we

average the gradient vectors over several iterations prior to making adjustments of the filter coefficients. For example, the average over K gradient vectors is

$$\bar{\mathbf{g}}(nK) = -\frac{2}{K} \sum_{k=0}^{K-1} e(nK + k) \mathbf{X}_M^*(nK + k) \quad (13.2.24)$$

and the corresponding recursive equation for updating the filter coefficients once every K iterations is

$$\mathbf{h}_M((n+1)K) = \mathbf{h}_M(nK) - \frac{1}{2} \Delta \bar{\mathbf{g}}(nK) \quad (13.2.25)$$

In effect, the averaging operation performed in (13.2.24) reduces the noise in the estimate of the gradient vector, as shown by Gardner (1984).

An alternative approach is to filter the gradient vectors by a lowpass filter and use the output of the filter as an estimate of the gradient vector. For example, a simple lowpass filter for the gradients yields as an output

$$\hat{\mathbf{S}}(n) = \beta \hat{\mathbf{S}}(n-1) - \hat{\mathbf{g}}(n), \quad \mathbf{S}(0) = -\hat{\mathbf{g}}(0) \quad (13.2.26)$$

where the choice of $0 \leq \beta < 1$ determines the bandwidth of the lowpass filter. When β is close to unity, the filter bandwidth is small and the effective averaging is performed over many gradient vectors. On the other hand, when β is small, the lowpass filter has a large bandwidth and, hence, it provides little averaging of the gradient vectors. With the filtered gradient vectors given by (13.2.26) in place of $\hat{\mathbf{g}}(n)$, we obtain the filtered version of the LMS algorithm given by

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \frac{1}{2} \Delta \hat{\mathbf{S}}(n) \quad (13.2.27)$$

An analysis of the filtered-gradient LMS algorithm is given in Proakis (1974).

Three other variations of the basic LMS algorithm given in (13.2.23) are obtained by using sign information contained in the error signal sequence $e(n)$ and/or in the components of the signal vector $\mathbf{X}_M(n)$. Hence, the three possible variations are

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \Delta \text{csgn}[e(n)] \mathbf{X}_M^*(n) \quad (13.2.28)$$

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \Delta e(n) \text{csgn}[\mathbf{X}_M^*(n)] \quad (13.2.29)$$

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \Delta \text{csgn}[e(n)] \text{csgn}[\mathbf{X}_M^*(n)] \quad (13.2.30)$$

where $\text{csgn}[x]$ is the complex sign function defined as

$$\text{csgn}[x] = \begin{cases} 1+j, & \text{if } \text{Re}(x) > 0 \text{ and } \text{Im}(x) > 0 \\ 1-j, & \text{if } \text{Re}(x) > 0 \text{ and } \text{Im}(x) < 0 \\ -1+j, & \text{if } \text{Re}(x) < 0 \text{ and } \text{Im}(x) > 0 \\ -1-j, & \text{if } \text{Re}(x) < 0 \text{ and } \text{Im}(x) < 0 \end{cases}$$

and $\text{csgn}[\mathbf{X}]$ denotes the complex sign function applied to each element of the vector \mathbf{X} . These three variations of the LMS algorithm may be called reduced complexity LMS algorithms, since multiplications are completely avoided in (13.2.30), and can be completely avoided in (13.2.28) and (13.2.29) by selecting Δ to be a power of $1/2$. The price paid for the reduction in computational complexity is a slower convergence of the filter coefficients to their optimum values.

Another version of the LMS algorithms, called a *normalized LMS*(NLMS) algorithm, that is frequently used in practice is given as

$$\mathbf{h}_M(n+1) = \mathbf{h}_M(n) + \frac{\Delta}{||\mathbf{X}_M(n)||^2} e(n) \mathbf{X}_M^*(n) \quad (13.2.31)$$

By dividing the step size by the norm of the data vector $\mathbf{X}_M(n)$, the NLMS algorithm is equivalent to employing a variable step size of the form

$$\Delta(n) = \frac{\Delta}{||\mathbf{X}_M(n)||^2} \quad (13.2.32)$$

Thus, the step size at each iteration is inversely proportional to the energy in the received data vector $\mathbf{X}_M(n)$. This scaling is advantageous in adaptive filtering applications where the dynamic range of the input to the adaptive filter is large, as would be the case, for example, in the implementation of adaptive equalizers for slowly fading communication channels. In such applications, it may be advantageous to add a small positive constant to the denominator of (13.2.32) to avoid numerical instabilities that may result when the norm of $\mathbf{X}_M(n)$ is small. Thus, another version of the NLMS algorithm may employ a variable step size of the form

$$\Delta(n) = \frac{\Delta}{\delta + ||\mathbf{X}_M(n)||^2} \quad (13.2.33)$$

where δ is a small positive number.

13.2.4 Properties of the LMS Algorithm

In this section, we consider the basic properties of the LMS algorithm given by (13.2.23). In particular, we focus on its convergence properties, its stability, and the excess noise generated as a result of using noisy gradient vectors in place of the actual gradient vectors. The use of noisy estimates of the gradient vectors implies that the filter coefficients will fluctuate randomly and, hence, an analysis of the characteristics of the algorithm should be performed in statistical terms.

The convergence and stability of the LMS algorithm may be investigated by determining how the mean value of $\mathbf{h}_M(n)$ converges to the optimum coefficients \mathbf{h}_{opt} . If we take the expected value of (13.2.23), we obtain

$$\begin{aligned} \bar{\mathbf{h}}_M(n+1) &= \bar{\mathbf{h}}_M(n) + \Delta E[e(n) \mathbf{X}_M^*(n)] \\ &= \bar{\mathbf{h}}_M(n) + \Delta [\gamma_d - \Gamma_M \bar{\mathbf{h}}_M(n)] \\ &= (\mathbf{I} - \Delta \Gamma_M) \bar{\mathbf{h}}_M(n) + \Delta \gamma_d \end{aligned} \quad (13.2.34)$$

where $\bar{\mathbf{h}}_M(n) = \mathbf{E}[\mathbf{h}_M(n)]$, and \mathbf{I} is the identity matrix.

The recursive relation in (13.2.34) may be represented as a closed-loop control system, as shown in Figure 13.2.1. The convergence rate and the stability of this closed-loop system are governed by our choice of the step-size parameter Δ . To determine the convergence behavior, it is convenient to decouple the M simultaneous difference equations given in (13.2.34), by performing a linear transformation of the mean coefficient vector $\bar{\mathbf{h}}_M(n)$. The appropriate transformation is obtained by noting that the autocorrelation matrix Γ_M is Hermitian and, hence, it can be represented (see Gantmacher (1960)) as

$$\Gamma_M = \mathbf{U}\Lambda\mathbf{U}^H \quad (13.2.35)$$

where \mathbf{U} is the normalized modal matrix of Γ_M and Λ is a diagonal matrix with diagonal elements λ_k , $0 \leq k \leq M - 1$, equal to the eigenvalues of Γ_M .

When (13.2.35) is substituted into (13.2.34), the latter may be expressed as

$$\bar{\mathbf{h}}_M^0(n+1) = (\mathbf{I} - \Delta\Lambda)\bar{\mathbf{h}}_M^0(n) + \Delta\gamma_d^0 \quad (13.2.36)$$

where the transformed (orthogonalized) vectors are $\bar{\mathbf{h}}_M^0(n) = \mathbf{U}^H\bar{\mathbf{h}}_M(n)$ and $\gamma_d^0 = \mathbf{U}^H\gamma_d$. The set of M first-order difference equations in (13.2.36) are now decoupled. Their convergence and their stability are determined from the homogeneous equation

$$\bar{\mathbf{h}}_M^0(n+1) = (\mathbf{I} - \Delta\Lambda)\bar{\mathbf{h}}_M^0(n) \quad (13.2.37)$$

If we focus our attention on the solution of the k th equation in (13.2.37), we observe that

$$\bar{h}^0(k, n) = C(1 - \Delta\lambda_k)^n u(n), \quad k = 0, 1, 2, \dots, M - 1 \quad (13.2.38)$$

where C is an arbitrary constant and $u(n)$ is the unit step sequence. Clearly, $\bar{h}^0(k, n)$ converges to zero exponentially, provided that

$$|1 - \Delta\lambda_k| < 1$$

or, equivalently,

$$0 < \Delta < \frac{2}{\lambda_k}, \quad k = 0, 1, \dots, M - 1 \quad (13.2.39)$$

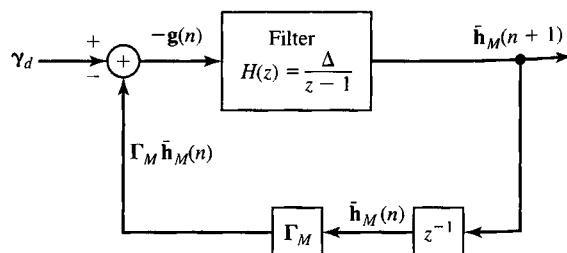


Figure 13.2.1

Closed-loop control system representation of recursive Equation (13.2.34).

The condition given by (13.2.39) for convergence of the homogeneous difference equation for the k th normalized filter coefficient (k th mode of the closed-loop system) must be satisfied for all $k = 0, 1, \dots, M - 1$. Therefore, the range of values of Δ that ensures the convergence of the mean of the coefficient vector in the LMS algorithm is

$$0 < \Delta < \frac{2}{\lambda_{\max}} \quad (13.2.40)$$

where λ_{\max} is the largest eigenvalue of Γ_M .

Since Γ_M is an autocorrelation matrix, its eigenvalues are nonnegative. Hence, an upper bound on λ_{\max} is

$$\lambda_{\max} < \sum_{k=0}^{M-1} \lambda_k = \text{trace } \Gamma_M = M\gamma_{xx}(0) \quad (13.2.41)$$

where $\gamma_{xx}(0)$ is the input signal power, which is easily estimated from the received signal. Therefore, an upper bound on the step size Δ is $2/M\gamma_{xx}(0)$.

From (13.2.38), we observe that rapid convergence of the LMS algorithm occurs when $|1 - \Delta\lambda_k|$ is small, that is, when the poles of the closed-loop system in Figure 13.2.1 are far from the unit circle. However, we cannot achieve this desirable condition and still satisfy the upper bound in (13.2.39) when there is a large difference between the largest and smallest eigenvalues of Γ_M . In other words, even if we select Δ to be $1/\lambda_{\max}$, the convergence rate of the LMS algorithm will be determined by the decay of the mode corresponding to the smallest eigenvalue λ_{\min} . For this mode, with $\Delta = 1/\lambda_{\max}$ substituted in (13.2.38), we have

$$h_M^0(k, n) = C \left(1 - \frac{\lambda_{\min}}{\lambda_{\max}}\right)^n u(n) \quad (13.2.42)$$

Consequently, the ratio $\lambda_{\min}/\lambda_{\max}$ ultimately determines the convergence rate. If $\lambda_{\min}/\lambda_{\max}$ is small (much smaller than unity), the convergence rate will be slow. On the other hand, if $\lambda_{\min}/\lambda_{\max}$ is close to unity, the convergence rate of the algorithm is fast.

The other important characteristic of the LMS algorithm is the noise resulting from the use of estimates of the gradient vectors. The noise in the gradient-vector estimates causes random fluctuations in the coefficients about their optimal values and, thus, leads to an increase in the MMSE at the output of the adaptive filter. Hence, the total MSE is $\mathcal{E}_{M\min} + \mathcal{E}_\Delta$, where \mathcal{E}_Δ is called the *excess mean-square error*.

For any given set of filter coefficients $\mathbf{h}_M(n)$, the total MSE at the output of the adaptive filter may be expressed as

$$\mathcal{E}_t(n) = \mathcal{E}_{M\min} + (\mathbf{h}_M(n) - \mathbf{h}_{\text{opt}})^T \Gamma_M (\mathbf{h}_M(n) - \mathbf{h}_{\text{opt}})^* \quad (13.2.43)$$

where \mathbf{h}_{opt} represents the optimum filter coefficients defined by (13.2.8). A plot of $\mathcal{E}_t(n)$ as a function of the iteration n is called a *learning curve*. If we substitute

(13.2.35) for Γ_M and perform the linear orthogonal transformation used previously, we obtain

$$\mathcal{E}_t(n) = \mathcal{E}_{M \min} + \sum_{k=0}^{M-1} \lambda_k |h^0(k, n) - h_{\text{opt}}^0(k)|^2 \quad (13.2.44)$$

where the term $h^0(k, n) - h_{\text{opt}}^0(k)$ represents the error in the k th filter coefficient (in the orthogonal coordinate system). The excess MSE is defined as the expected value of the second term in (13.2.44),

$$\mathcal{E}_\Delta = \sum_{k=0}^{M-1} \lambda_k E[|h^0(k, n) - h_{\text{opt}}^0(k)|^2] \quad (13.2.45)$$

To derive an expression for the excess MSE \mathcal{E}_Δ , we assume that the mean values of the filter coefficients $\mathbf{h}_M(n)$ have converged to their optimum values \mathbf{h}_{opt} . Then, the term $\Delta e(n)\mathbf{X}_M^*(n)$ in the LMS algorithm given by (13.2.23) is a zero-mean noise vector. Its covariance is

$$\text{cov}[\Delta e(n)\mathbf{X}_M^*(n)] = \Delta^2 E[|e(n)|^2 \mathbf{X}_M(n)\mathbf{X}_M^H(n)] \quad (13.2.46)$$

To a first approximation, we assume that $|e(n)|^2$ is uncorrelated with the signal vector. Although this assumption is not strictly true, it simplifies the derivation and yields useful results. (The reader may refer to Mazo (1979), Jones, Cavin, and Reed (1982), and Gardner (1984) for further discussion on this assumption). Then,

$$\begin{aligned} \text{cov}[\Delta e(n)\mathbf{X}_M^*(n)] &= \Delta^2 E[|e(n)|^2] E[\mathbf{X}_M(n)\mathbf{X}_M^H(n)] \\ &= \Delta^2 \mathcal{E}_{M \min} \Gamma_M \end{aligned} \quad (13.2.47)$$

For the orthogonalized coefficient vector $\mathbf{h}_M^0(n)$ with additive noise, we have the equation

$$\mathbf{h}_M^0(n+1) = (\mathbf{I} - \Delta \boldsymbol{\Lambda})\mathbf{h}_M^0(n) + \Delta \boldsymbol{\gamma}_d^0 + \mathbf{w}^0(n) \quad (13.2.48)$$

where $\mathbf{w}^0(n)$ is the additive noise vector, which is related to the noise vector $\Delta e(n)\mathbf{X}_M^*(n)$ through the transformation

$$\begin{aligned} \mathbf{w}^0(n) &= \mathbf{U}^H [\Delta e(n)\mathbf{X}_M^*(n)] \\ &= \Delta e(n)\mathbf{U}^H \mathbf{X}_M^*(n) \end{aligned} \quad (13.2.49)$$

It is easily seen that the covariance matrix of the noise vector is

$$\begin{aligned} \text{cov}[\mathbf{w}^0(n)] &= \Delta^2 \mathcal{E}_{M \ min} \mathbf{U}^H \boldsymbol{\Gamma}_M \mathbf{U} \\ &= \Delta^2 \mathcal{E}_{M \ min} \boldsymbol{\Lambda} \end{aligned} \quad (13.2.50)$$

Therefore, the M components of $\mathbf{w}^0(n)$ are uncorrelated and each component has the variance $\sigma_k^2 = \Delta^2 \mathcal{E}_{M \min} \lambda_k$, $k = 0, 1, \dots, M - 1$.

Since the noise components of $\mathbf{w}^0(n)$ are uncorrelated, we may consider the M uncoupled difference equations in (13.2.48) separately. Each first-order difference equation represents a filter with impulse response $(1 - \Delta \lambda_k)^n$. When such a filter is excited with a noise sequence $w_k^0(n)$, the variance of the noise at the output of the filter is

$$E[|h^0(k, n) - h_{\text{opt}}^0(k)|^2] = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} (1 - \Delta \lambda_k)^n (1 - \Delta \lambda_k)^m E[w_k^0(n) w_k^{0*}(m)] \quad (13.2.51)$$

We make the simplifying assumption that the noise sequence $w_k^0(n)$ is white. Then, (13.2.51) reduces to

$$E[|h^0(k, n) - h_{\text{opt}}^0(k)|^2] = \frac{\sigma_k^2}{1 - (1 - \Delta \lambda_k)^2} = \frac{\Delta^2 \mathcal{E}_{M \min} \lambda_k}{1 - (1 - \Delta \lambda_k)^2} \quad (13.2.52)$$

If we substitute the result of (13.2.52) into (13.2.45), we obtain the expression for the excess MSE as

$$\mathcal{E}_\Delta = \Delta^2 \mathcal{E}_{M \min} \sum_{k=0}^{M-1} \frac{\lambda_k^2}{1 - (1 - \Delta \lambda_k)^2} \quad (13.2.53)$$

This expression can be simplified if we assume that Δ is selected such that $\Delta \lambda_k \ll 1$ for all k . Then,

$$\begin{aligned} \mathcal{E}_\Delta &\approx \Delta^2 \mathcal{E}_{M \min} \sum_{k=0}^{M-1} \frac{\lambda_k^2}{2 \Delta \lambda_k} \\ &\approx \frac{1}{2} \Delta \mathcal{E}_{M \min} \sum_{k=0}^{M-1} \lambda_k \\ &\approx \frac{\Delta M \mathcal{E}_{M \min} \gamma_{xx}(0)}{2} \end{aligned} \quad (13.2.54)$$

where $\gamma_{xx}(0)$ is the power of the input signal.

The expression for \mathcal{E}_Δ indicates that the excess MSE is proportional to the step-size parameter Δ . Hence, our choice of Δ must be based on a compromise between fast convergence and a small excess MSE. In practice, it is desirable to have $\mathcal{E}_\Delta < \mathcal{E}_{M \min}$. Hence,

$$\frac{\mathcal{E}_\Delta}{\mathcal{E}_{M \min}} \approx \frac{\Delta M \gamma_{xx}(0)}{2} < 1$$

or, equivalently,

$$\Delta < \frac{2}{M\gamma_{xx}(0)} \quad (13.2.55)$$

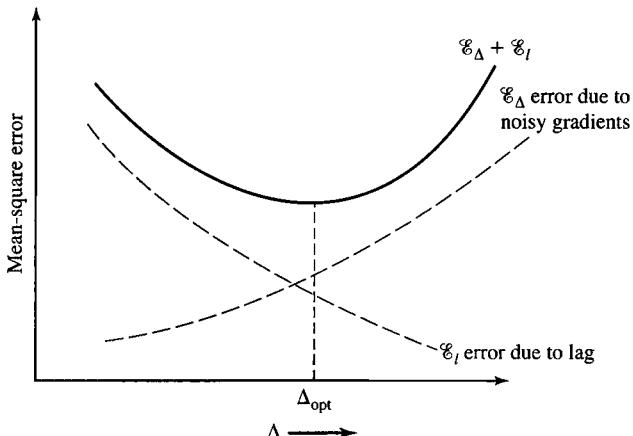
But, this is just the upper bound that we had obtained previously for λ_{\max} . In steady-state operation, Δ should satisfy the upper bound in (13.2.55); otherwise the excess MSE causes significant degradation in the performance of the adaptive filter.

The preceding analysis of the excess MSE is based on the assumption that the mean values of the filter coefficients have converged to the optimum solution \mathbf{h}_{opt} . Under this condition, the step size Δ should satisfy the bound in (13.2.55). On the other hand, we have determined that convergence of the mean coefficient vector requires that $\Delta < 2/\lambda_{\max}$. While a choice of Δ near the upper bound $2/\lambda_{\max}$ may lead to initial convergence of the deterministic (known) gradient algorithm, such a large value of Δ will usually result in instability of the stochastic-gradient LMS algorithm.

The initial convergence or transient behavior of the LMS algorithm has been investigated by several researchers. Their results clearly indicate that the step size must be reduced in direct proportion to the length of the adaptive filter, as in (13.2.55). The upper bound given in (13.2.55) is necessary to ensure the initial convergence of the stochastic-gradient LMS algorithm. In practice, a choice of $\Delta < 1/M\gamma_{xx}(0)$ is usually made. Sayed (2003), Gitlin and Weinstein (1979), and Ungerboeck (1972) have given an analysis of the transient behavior and the convergence properties of the LMS algorithm.

In a digital implementation of the LMS algorithm, the choice of the step-size parameter becomes even more critical. In an attempt to reduce the excess MSE, it is possible to reduce the step-size parameter to the point at which the total output MSE actually increases. This condition occurs when the estimated gradient components $e(n)x^*(n-l)$, $l = 0, 1, M-1$, after multiplication by the small step-size parameter Δ , are smaller than one-half of the least significant bit in the fixed-point representation of the filter coefficients. In such a case, adaptation ceases. Consequently, it is important for the step size to be large enough to bring the filter coefficients in the vicinity of \mathbf{h}_{opt} . If it is desired to decrease the step size significantly, it is necessary to increase the precision in the filter coefficients. Typically, sixteen bits of precision may be used for the filter coefficients, with the twelve most significant bits used for arithmetic operations in the filtering of the data. The four least significant bits are required to provide the necessary precision for the adaptation process. Thus, the scaled, estimated gradient components $\Delta e(n)x^*(n-l)$ usually affect only the least significant bits. In effect, the added precision also allows for the noise to be averaged out, since several incremental changes in the least significant bits are required before any change occurs in the upper, more significant bits used in arithmetic operations for filtering of the data. For an analysis of round-off errors in a digital implementation of the LMS algorithm, the reader is referred to Gitlin and Weinstein (1979), Gitlin, Meadors, and Weinstein (1982), and Caraiscos and Liu (1984).

As a final point, we should indicate that the LMS algorithm is appropriate for tracking slowly time-variant signal statistics. In such a case, the minimum MSE and the optimum coefficient vector will be time variant. In other words, $\mathcal{E}_M \min$ is a function of time, and the M -dimensional error surface is moving with the time

**Figure 13.2.2**

Excess mean-square error E_Δ and lag error E_l as a function of the step size Δ .

index n . The LMS algorithm attempts to follow the moving minimum $E_{M \text{ min}}$ in the M -dimensional space, but it is always lagging behind due to its use of (estimated) gradient vectors. As a consequence, the LMS algorithm incurs another form of error, called the *lag error*, whose mean-square value decreases with an increase in the step size Δ . The total MSE can now be expressed as

$$E_{\text{total}} = E_{M \text{ min}} + E_\Delta + E_l \quad (13.2.56)$$

where E_l denotes the MSE due to the lag.

In any given nonstationary adaptive filtering problem, if we plot the E_Δ and E_l as a function of Δ , we expect these errors to behave as illustrated in Figure 13.2.2. We observe that E_Δ increases with an increase in Δ , whereas E_l decreases with an increase in Δ . The total error will exhibit a minimum, which will determine the optimum choice of the step-size parameter.

When the statistical time variations of the signals occur rapidly, the lag error will dominate the performance of the adaptive filter. In such a case, $E_l \gg E_{M \text{ min}} + E_\Delta$, even when the largest possible value of Δ is used.

EXAMPLE 13.2.1

Learning curves for the LMS algorithm, when used to adaptively equalize a communication channel, are illustrated in Figure 13.2.3. The FIR equalizer was realized in direct form and had a length $M = 11$. The autocorrelation matrix Γ_M has an eigenvalue spread of $\lambda_{\max}/\lambda_{\min} = 11$. These three learning curves have been obtained with step sizes $\Delta = 0.045, 0.09$, and 0.115 , by averaging the (estimated) MSE in 200 simulation runs. The input signal power was normalized to unity. Hence, the upper bound in (13.2.55) is equal to 0.18. By selecting $\Delta = 0.09$ (one-half of the upper bound), we obtain a rapidly decaying learning curve, as shown in Figure 13.2.3. If we divide Δ by 2 to obtain 0.045, the convergence rate is reduced but the excess MSE is also reduced, so the algorithm performs better in the time-invariant signal environment. Finally, we note that a choice of $\Delta = 0.115$ causes large undesirable fluctuations in the output MSE of the algorithm. Note that $\Delta = 0.115$ is significantly lower than the upper bound given in (13.2.55).

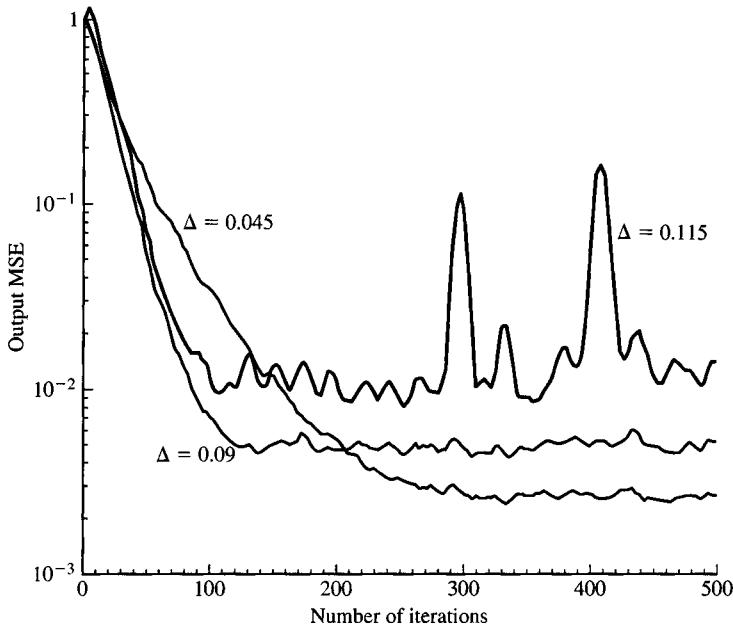


Figure 13.2.3 Learning curves for the LMS algorithm applied to an adaptive equalizer of length $M = 11$ and a channel with eigenvalue spread $\lambda_{\max}/\lambda_{\min} = 11$.

13.3 Adaptive Direct-Form Filters—RLS Algorithms

The major advantage of the LMS algorithm lies in its computational simplicity. However, the price paid for this simplicity is slow convergence, especially when the eigenvalues of the autocorrelation matrix Γ_M have a large spread, that is, when $\lambda_{\max}/\lambda_{\min} \gg 1$. From another point of view, the LMS algorithm has only a single adjustable parameter for controlling the convergence rate, namely, the step-size parameter Δ . Since Δ is limited for purposes of stability to be less than the upper bound in (13.2.55), the modes corresponding to the smaller eigenvalues converge very slowly.

To obtain faster convergence, it is necessary to devise more complex algorithms, which involve additional parameters. In particular, if the correlation matrix Γ_M has unequal eigenvalues $\lambda_0, \lambda_1, \dots, \lambda_{M-1}$, we should use an algorithm that contains M parameters, one for each of the eigenvalues. In deriving more rapidly converging adaptive filtering algorithms, we adopt the least-squares criterion instead of the statistical approach based on the MSE criterion. Thus, we deal directly with the data sequence $x(n)$ and obtain estimates of correlations from the data.

13.3.1 RLS Algorithm

It is convenient to express the least-squares algorithms in matrix form, in order to simplify the notation. Since the algorithms will be recursive in time, it is also necessary to introduce a time index in the filter-coefficient vector and in the error sequence.

Hence, we define the filter-coefficient vector at time n as

$$\mathbf{h}_M(n) = \begin{bmatrix} h(0, n) \\ h(1, n) \\ h(2, n) \\ \vdots \\ h(M-1, n) \end{bmatrix} \quad (13.3.1)$$

where the subscript M denotes the length of the filter. Similarly, the input signal vector to the filter at time n is denoted as

$$\mathbf{X}_M(n) = \begin{bmatrix} x(n) \\ x(n-1) \\ x(n-2) \\ \vdots \\ x(n-M+1) \end{bmatrix} \quad (13.3.2)$$

We assume that $x(n) = 0$ for $n < 0$. This is usually called *prewindowing* of the input data.

The recursive least-squares problem may now be formulated as follows. Suppose that we have observed the vectors $\mathbf{X}_M(l)$, $l = 0, 1, \dots, n$, and we wish to determine the filter-coefficient vector $\mathbf{h}_M(n)$ that minimizes the weighted sum of magnitude-squared errors

$$\mathcal{E}_M = \sum_{l=0}^n w^{n-l} |e_M(l, n)|^2 \quad (13.3.3)$$

where the error is defined as the difference between the desired sequence $d(l)$ and the estimate $\hat{d}(l, n)$,

$$\begin{aligned} e_M(l, n) &= d(l) - \hat{d}(l, n) \\ &= d(l) - \mathbf{h}_M^t(n) \mathbf{X}_M(l) \end{aligned} \quad (13.3.4)$$

and w is a weighting factor in the range $0 < w < 1$.

The purpose of the factor w is to weight the most recent data points more heavily and, thus, allow the filter coefficients to adapt to time-varying statistical characteristics of the data. This is accomplished by using the exponential weighting factor with the past data. Alternatively, we may use a finite-duration sliding window with uniform weighting over the window length. We find the exponential weighting factor more convenient, both mathematically and practically. For comparison, an exponentially weighted window sequence has an effective memory of

$$\bar{N} = \frac{\sum_{n=0}^{\infty} n w^n}{\sum_{n=0}^{\infty} w^n} = \frac{w}{1-w} \quad (13.3.5)$$

and, hence, should be approximately equivalent to a sliding window of length \bar{N} .

The minimization of \mathcal{E}_M with respect to the filter-coefficient vector $\mathbf{h}_M(n)$ yields the set of linear equations

$$\mathbf{R}_M(n)\mathbf{h}_M(n) = \mathbf{D}_M(n) \quad (13.3.6)$$

where $\mathbf{R}_M(n)$ is the signal (estimated) correlation matrix defined as

$$\mathbf{R}_M(n) = \sum_{l=0}^n w^{n-l} \mathbf{X}_M^*(l) \mathbf{X}_M^t(l) \quad (13.3.7)$$

and $\mathbf{D}_M(n)$ is the (estimated) crosscorrelation vector

$$\mathbf{D}_M(n) = \sum_{l=0}^n w^{n-l} \mathbf{X}_M^*(l) d(l) \quad (13.3.8)$$

The solution of (13.3.6) is

$$\mathbf{h}_M(n) = \mathbf{R}_M^{-1}(n) \mathbf{D}_M(n) \quad (13.3.9)$$

Clearly, the matrix $\mathbf{R}_M(n)$ is akin to the statistical autocorrelation matrix Γ_M , and the vector $\mathbf{D}_M(n)$ is akin to the crosscorrelation vector y_d , defined previously. We emphasize, however, that $\mathbf{R}_M(n)$ is not a Toeplitz matrix, whereas Γ_M is. We should also mention that for small values of n , $\mathbf{R}_M(n)$ may be ill conditioned, so that its inverse is not computable. In such a case, it is customary to initially add the matrix $\delta \mathbf{I}_M$ to $\mathbf{R}_M(n)$, where \mathbf{I}_M is an identity matrix and δ is a small positive constant. With exponential weighting into the past, the effect of adding $\delta \mathbf{I}_M$ dissipates with time.

Now, suppose that we have the solution of (13.3.9) at time $n - 1$ —that is, we have $\mathbf{h}_M(n - 1)$, and we wish to compute $\mathbf{h}_M(n)$. It is inefficient and, hence, impractical to solve the set of M linear equations for each new signal component. Instead, we may compute the matrix and vectors recursively. First, $\mathbf{R}_M(n)$ may be computed recursively as

$$\mathbf{R}_M(n) = w \mathbf{R}_M(n - 1) + \mathbf{X}_M^*(n) \mathbf{X}_M^t(n) \quad (13.3.10)$$

We call (13.3.10) the *time-update equation* for $\mathbf{R}_M(n)$.

Since the inverse of $\mathbf{R}_M(n)$ is needed, we use the matrix inversion lemma (see Householder (1964)),

$$\mathbf{R}_M^{-1}(n) = \frac{1}{w} \left[\mathbf{R}_M^{-1}(n - 1) - \frac{\mathbf{R}_M^{-1}(n - 1) \mathbf{X}_M^*(n) \mathbf{X}_M^t(n) \mathbf{R}_M^{-1}(n - 1)}{w + \mathbf{X}_M^t(n) \mathbf{R}_M^{-1}(n - 1) \mathbf{X}_M^*(n)} \right] \quad (13.3.11)$$

Thus, $\mathbf{R}_M^{-1}(n)$ may be computed recursively.

For convenience, we define $\mathbf{P}_M(n) = \mathbf{R}_M^{-1}(n)$. It is also convenient to define an M -dimensional vector $\mathbf{K}_M(n)$, sometimes called the *Kalman gain vector*, as

$$\mathbf{K}_M(n) = \frac{1}{w + \mu_M(n)} \mathbf{P}_M(n - 1) \mathbf{X}_M^*(n) \quad (13.3.12)$$

where $\mu_M(n)$ is a scalar defined as

$$\mu_M(n) = \mathbf{X}_M^t(n)\mathbf{P}_M(n-1)\mathbf{X}_M^*(n) \quad (13.3.13)$$

With these definitions, (13.3.11) becomes

$$\mathbf{P}_M(n) = \frac{1}{w} [\mathbf{P}_M(n-1) - \mathbf{K}_M(n)\mathbf{X}_M^t(n)\mathbf{P}_M(n-1)] \quad (13.3.14)$$

Let us postmultiply (13.3.14) by $\mathbf{X}_M^*(n)$. Then,

$$\begin{aligned} \mathbf{P}_M(n)\mathbf{X}_M^*(n) &= \frac{1}{w} [\mathbf{P}_M(n-1)\mathbf{X}_M^*(n) - \mathbf{K}_M(n)\mathbf{X}_M^t(n)\mathbf{P}_M(n-1)\mathbf{X}_M^*(n)] \\ &= \frac{1}{w} \{[w + \mu_M(n)]\mathbf{K}_M(n) - \mathbf{K}_M(n)\mu_M(n)\} = \mathbf{K}_M(n) \end{aligned} \quad (13.3.15)$$

Therefore, the Kalman gain vector may also be defined as $\mathbf{P}_M(n)\mathbf{X}_M^*(n)$.

Now, we may use the matrix inversion lemma to derive an equation for computing the filter coefficients recursively. Since

$$\mathbf{h}_M(n) = \mathbf{P}_M(n)\mathbf{D}_M(n) \quad (13.3.16)$$

and

$$\mathbf{D}_M(n) = w\mathbf{D}_M(n-1) + d(n)\mathbf{X}_M^*(n) \quad (13.3.17)$$

we have, upon substitution of (13.3.14) and (13.3.17) into (13.3.9),

$$\begin{aligned} \mathbf{h}_M(n) &= \frac{1}{w} [\mathbf{P}_M(n-1) - \mathbf{K}_M(n)\mathbf{X}_M^t(n)\mathbf{P}_M(n-1)] \\ &\quad \times [w\mathbf{D}_M(n-1) + d(n)\mathbf{X}_M^*(n)] \\ &= \mathbf{P}_M(n-1)\mathbf{D}_M(n-1) + \frac{1}{w} d(n)\mathbf{P}_M(n-1)\mathbf{X}_M^*(n) \\ &\quad - \mathbf{K}_M(n)\mathbf{X}_M^t(n)\mathbf{P}_M(n-1)\mathbf{D}_M(n-1) \\ &\quad - \frac{1}{w} d(n)\mathbf{K}_M(n)\mathbf{X}_M^t(n)\mathbf{P}_M(n-1)\mathbf{X}_M^*(n) \\ &= \mathbf{h}_M(n-1) + \mathbf{K}_M(n)[d(n) - \mathbf{X}_M^t(n)\mathbf{h}_M(n-1)] \end{aligned} \quad (13.3.18)$$

We observe that $\mathbf{X}_M^t(n)\mathbf{h}_M(n - 1)$ is the output of the adaptive filter at time n based on use of the filter coefficients at time $n - 1$. Since

$$\mathbf{X}_M^t(n)\mathbf{h}_M(n - 1) = \hat{d}(n, n - 1) \equiv \hat{d}(n) \quad (13.3.19)$$

and

$$e_M(n, n - 1) = d(n) - \hat{d}(n, n - 1) \equiv e_M(n) \quad (13.3.20)$$

it follows that the time-update equation for $\mathbf{h}_M(n)$ may be expressed as

$$\mathbf{h}_M(n) = \mathbf{h}_M(n - 1) + \mathbf{K}_M(n)e_M(n) \quad (13.3.21)$$

or, equivalently,

$$\mathbf{h}_M(n) = \mathbf{h}_M(n - 1) + \mathbf{P}_M(n)\mathbf{X}_M^*(n)e_M(n) \quad (13.3.22)$$

To summarize, suppose we have the optimum filter coefficients $\mathbf{h}_M(n - 1)$, the matrix $\mathbf{P}_M(n - 1)$, and the vector $\mathbf{X}_M(n - 1)$. When the new signal component $x(n)$ is obtained, we form the vector $\mathbf{X}_M(n)$ by dropping the term $x(n - M)$ from $\mathbf{X}_M(n - 1)$ and adding the term $x(n)$ as the first element. Then, the recursive computation for the filter coefficients proceeds as follows:

1. Compute the filter output:

$$\hat{d}(n) = \mathbf{X}_M^t(n)\mathbf{h}_M(n - 1) \quad (13.3.23)$$

2. Compute the error:

$$e_M(n) = d(n) - \hat{d}(n) \quad (13.3.24)$$

3. Compute the Kalman gain vector:

$$\mathbf{K}_M(n) = \frac{\mathbf{P}_M(n - 1)\mathbf{X}_M^*(n)}{w + \mathbf{X}_M^t(n)\mathbf{P}_M(n - 1)\mathbf{X}_M^*(n)} \quad (13.3.25)$$

4. Update the inverse of the correlation matrix

$$\mathbf{P}_M(n) = \frac{1}{w} [\mathbf{P}_M(n - 1) - \mathbf{K}_M(n)\mathbf{X}_M^t(n)\mathbf{P}_M(n - 1)] \quad (13.3.26)$$

5. Update the coefficient vector of the filter

$$\mathbf{h}_M(n) = \mathbf{h}_M(n - 1) + \mathbf{K}_M(n)e_M(n) \quad (13.3.27)$$

The recursive algorithm specified by (13.3.23) through (13.3.27) is called *the direct-form recursive least-squares (RLS) algorithm*. It is initialized by setting $\mathbf{h}_M(-1) = \mathbf{0}$ and $\mathbf{P}_M(-1) = 1/\delta \mathbf{I}_M$, where δ is a small positive number.

The residual MSE resulting from the preceding optimization is

$$\mathcal{E}_M \min = \sum_{l=0}^n w^{n-l} |d(l)|^2 - \mathbf{h}_M^t(n) \mathbf{D}_M^*(n) \quad (13.3.28)$$

From (13.3.27), we observe that the filter coefficients vary with time by an amount equal to the error $e_M(n)$ multiplied by the Kalman gain vector $\mathbf{K}_M(n)$. Since $\mathbf{K}_M(n)$ is an M -dimensional vector, each filter coefficient is controlled by one of the elements of $\mathbf{K}_M(n)$. Consequently, rapid convergence is obtained. In contrast, the time-update equation for the coefficients of the filter adjusted by use of the LMS algorithm is

$$\mathbf{h}_M(n) = \mathbf{h}_M(n-1) + \Delta \mathbf{X}^*(n) e_M(n) \quad (13.3.29)$$

which has only the single parameter Δ for controlling the adjustment rate of the coefficients.

13.3.2 The LDU Factorization and Square-Root Algorithms

The RLS algorithm is very susceptible to round-off noise in an implementation of the algorithm with finite-precision arithmetic. The major problem with round-off errors occurs in the updating of $\mathbf{P}_M(n)$. To remedy this problem, we may perform a decomposition of either the correlation matrix $\mathbf{R}_M(n)$ or its inverse $\mathbf{P}_M(n)$.

To be specific, let us consider an LDU (lower-triangular/diagonal/upper-triangular) decomposition of $\mathbf{P}_M(n)$. We may write

$$\mathbf{P}_M(n) = \mathbf{L}_M(n) \bar{\mathbf{D}}_M(n) \mathbf{L}_M^H(n) \quad (13.3.30)$$

where $\mathbf{L}_M(n)$ is a lower-triangular matrix with elements l_{ik} , $\bar{\mathbf{D}}_M(n)$ is a diagonal matrix with elements δ_k , and $\mathbf{L}_M^H(n)$ is an upper-triangular matrix. The diagonal elements of $\mathbf{L}_M(n)$ are set to unity (i.e., $l_{ii} = 1$). Now, instead of computing $\mathbf{P}_M(n)$ recursively, we can determine a formula for updating the factors $\mathbf{L}_M(n)$ and $\bar{\mathbf{D}}_M(n)$ directly, thus avoiding the computation of $\mathbf{P}_M(n)$.

The desired update formula is obtained by substituting the factored form of $\mathbf{P}_M(n)$ into (13.3.26). Thus, we have

$$\begin{aligned} & \mathbf{L}_M(n) \bar{\mathbf{D}}_M(n) \mathbf{L}_M^H(n) \\ &= \frac{1}{w} \mathbf{L}_M(n-1) \left[\bar{\mathbf{D}}_M(n-1) - \frac{1}{w + \mu_M(n)} \mathbf{V}_M(n-1) \mathbf{V}_M^H(n-1) \right] \mathbf{L}_M^H(n-1) \end{aligned} \quad (13.3.31)$$

where, by definition,

$$\mathbf{V}_M(n-1) = \bar{\mathbf{D}}_M(n-1)\mathbf{L}_M^H(n-1)\mathbf{X}_M^*(n) \quad (13.3.32)$$

The term inside the brackets in (13.3.31) is a Hermitian matrix and may be expressed in an LDU factored form as

$$\begin{aligned} & \hat{\mathbf{L}}_M(n-1)\hat{\mathbf{D}}_M(n-1)\hat{\mathbf{L}}_M^H(n-1) \\ &= \bar{\mathbf{D}}_M(n-1) - \frac{1}{w + \mu_M(n)}\mathbf{V}_M(n-1)\mathbf{V}_M^H(n-1) \end{aligned} \quad (13.3.33)$$

Then, if we substitute (13.3.33) into (13.3.31), we obtain

$$\mathbf{L}_M(n)\bar{\mathbf{D}}_M(n)\hat{\mathbf{L}}_M^H(n) = \frac{1}{w}[\mathbf{L}_M(n-1)\hat{\mathbf{L}}_M(n-1)\hat{\mathbf{D}}_M(n-1)\hat{\mathbf{L}}_M^H(n-1)\mathbf{L}_M^H(n-1)] \quad (13.3.34)$$

Consequently, the desired update relations are

$$\begin{aligned} \mathbf{L}_M(n) &= \mathbf{L}_M(n-1)\hat{\mathbf{L}}_M(n-1) \\ \bar{\mathbf{D}}_M(n) &= \frac{1}{w}\hat{\mathbf{D}}_M(n-1) \end{aligned} \quad (13.3.35)$$

To determine the factors $\hat{\mathbf{L}}_M(n-1)$ and $\hat{\mathbf{D}}_M(n-1)$, we need to factor the matrix on the right-hand side (RHS) of (13.3.33). This factorization may be expressed by the set of linear equations

$$\sum_{k=1}^j l_{ik}d_kl_{jk}^* = p_{ij}, \quad 1 \leq j \leq i-1, \quad i \geq 2 \quad (13.3.36)$$

where $\{d_k\}$ are the elements of $\hat{\mathbf{D}}_M(n-1)$, $\{l_{ik}\}$ are elements of $\hat{\mathbf{L}}_M(n-1)$ and $\{p_{ij}\}$ are the elements of the matrix on the RHS of (13.3.33). Then, $\{l_{ik}\}$ and $\{d_k\}$ are determined as follows:

$$\begin{aligned} d_1 &= p_{11} \\ l_{ij}d_j &= p_{ij} - \sum_{k=1}^{j-1} l_{ik}d_kl_{jk}^*, \quad 1 \leq j \leq i-1, \quad 2 \leq i \leq M \\ d_i &= p_{ii} - \sum_{k=1}^{i-1} |l_{ik}|^2 d_k, \quad 2 \leq i \leq M \end{aligned} \quad (13.3.37)$$

TABLE 13.1 LDU Form of Square-Root RLS Algorithm

```

for  $j = 1, \dots, 2, \dots, M$  do
     $f_j = x^*(n)$ 
end loop  $j$ 
for  $j = 1, 2, \dots, M - 1$  do
    for  $i = j + 1, j + 2, \dots, M$  do
         $f_j = f_j + l_{ij}(n - 1)f_i$ 
    end loop  $j$ 
    for  $j = 1, 2, \dots, M$  do
         $\bar{d}_j(n) = d_j(n - 1)/w$ 
         $v_j = \bar{d}_j(n)f_j$ 
    end loop  $j$ 
     $\alpha_M = 1 + v_M f_M^*$ 
     $d_M(n) = \bar{d}_M(n)/\alpha_M$ 
     $\bar{k}_M = v_M$ 
    for  $j = M - 1, M - 2, \dots, 1$  do
         $\bar{k}_j = v_j$ 
         $\alpha_j = \alpha_{j+1} + v_j f_j^*$ 
         $\lambda_j = f_j/\alpha_{j+1}$ 
         $d_j(n) = \bar{d}_j(n)\alpha_{j+1}/\alpha_1$ 
        for  $i = M, M - 1, \dots, j + 1$  do
             $l_{ij}(n) = l_{ij}(n - 1) + \bar{k}_i^* \lambda_j$ 
             $\bar{k}_i = \bar{k}_i + v_j l_{ij}^*(n - 1)$  down to  $j = 2$ 
        end loop  $i$ 
    end loop  $j$ 
     $\bar{\mathbf{K}}_M(n) = [\bar{k}_1, \bar{k}_2, \dots, \bar{k}_M]'$ 
     $e_M(n) = d(n) - \bar{d}(n)$ 
     $\mathbf{h}_M(n) = \mathbf{h}_M(n - 1) + [e_M(n)/\alpha_1]\bar{\mathbf{K}}_M(n)$ 

```

The resulting algorithm, obtained from the time-update equations in (13.3.35), depends directly on the data vector $\mathbf{X}_M(n)$ and not on the “square” of the data vector. Thus, the squaring operation of the data vector is avoided and, consequently, the effect of round-off errors is significantly reduced.

The RLS algorithms obtained from an LDU decomposition of either $\mathbf{R}_M(n)$ or $\mathbf{P}_M(n)$ are called *square-root RLS algorithms*. Bierman (1977), Carlson and Culmone (1979), and Hsu (1982) treat these types of algorithms. A square-root RLS algorithm based on the LDU decomposition of $\mathbf{P}_M(n)$, as just described, is given in Table 13.1. Its computational complexity is proportional to M^2 .

13.3.3 Fast RLS Algorithms

The RLS direct-form algorithm and the square-root algorithms have a computational complexity proportional to M^2 , as indicated. On the other hand, the RLS lattice algorithms derived in Section 13.4 have a computational complexity proportional to M . Basically, the lattice algorithms avoid the matrix multiplications involved in computing the Kalman gain vector $\mathbf{K}_M(n)$.

TABLE 13.2 Fast RLS Algorithm: Version A

$f_{M-1}(n) = x(n) + \mathbf{a}_{M-1}'(n-1)\mathbf{X}_{M-1}(n-1)$
$g_{M-1}(n) = x(n-M+1) + \mathbf{b}_{M-1}'(n-1)\mathbf{X}_{M-1}(n)$
$\mathbf{a}_{M-1}(n) = \mathbf{a}_{M-1}(n-1) - \mathbf{K}_{M-1}(n-1)f_{M-1}(n)$
$f_{M-1}(n, n) = x(n) + \mathbf{a}_{M-1}'(n)\mathbf{X}_{M-1}(n-1)$
$E_{M-1}^f(n) = wE_{M-1}^f(n-1) + f_{M-1}(n)f_{M-1}^*(n, n)$
$\begin{bmatrix} \mathbf{C}_{M-1}(n) \\ c_{MM}(n) \end{bmatrix} \equiv \mathbf{K}_M(n) = \begin{bmatrix} 0 \\ \mathbf{K}_{M-1}(n-1) \end{bmatrix} + \frac{f_{M-1}^*(n, n)}{E_{M-1}^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_{M-1}(n) \end{bmatrix}$
$\mathbf{K}_{M-1}(n) = \frac{\mathbf{C}_{M-1}(n) - c_{MM}(n)\mathbf{b}_{M-1}(n-1)}{1 - c_{MM}(n)g_{M-1}(n)}$
$\mathbf{b}_{M-1}(n) = \mathbf{b}_{M-1}(n-1) - \mathbf{K}_{M-1}(n)g_{M-1}(n)$
$\hat{d}(n) = \mathbf{h}_M'(n-1)\mathbf{X}_M(n)$
$e_M(n) = d(n) - \hat{d}(n)$
$\mathbf{h}_M(n) = \mathbf{h}_M(n-1) + \mathbf{K}_M(n)e_M(n)$
<i>Initialization</i>
$\mathbf{a}_{M-1}(-1) = \mathbf{b}_{M-1}(-1) = \mathbf{0}$
$\mathbf{K}_{M-1}(-1) = \mathbf{0}$
$\mathbf{h}_{M-1}(-1) = \mathbf{0}$
$E_{M-1}^f(-1) = \epsilon, \quad \epsilon > 0$

By using the forward and backward prediction formulas derived in Section 13.4 for the RLS lattice, it is possible to obtain time-update equations for the Kalman gain vector that completely avoid matrix multiplications. The resulting algorithms have a complexity that is proportional to M (multiplications and divisions) and, hence, they are called *fast RLS algorithms* for direct-form FIR filters.

There are several versions of fast algorithms, which differ in minor ways. Two versions are given in Tables 13.2 and 13.3 for complex-valued signals. The variables used in the fast algorithms listed in these tables are defined in Section 13.4. The computational complexity for Version A is $10M - 4$ (complex) multiplications and divisions, whereas Version B has a complexity of $9M + 1$ multiplications and divisions. Further reduction of computational complexity to $7M$ is possible. For example, Carayannis, Manolakis, and Kalouptsidis (1983) describe a fast RLS algorithm, termed the FAEST (fast a posteriori error sequential technique) algorithm, with a computational complexity of $7M$; this algorithm is given in Section 13.4. Other versions of these algorithms with a complexity of $7M$ have been proposed, but many of these algorithms are extremely sensitive to round-off noise and exhibit instability problems (Falconer and Ljung (1978), Carayannis, Manolakis, and Kalouptsidis (1983; 1986) and Cioffi and Kailath (1984)). Slock and Kailath (1988; 1991) have shown how to

TABLE 13.3 Fast RLS Algorithm: Version B

$f_{M-1}(n) = x(n) + \mathbf{a}'_{M-1}(n-1)\mathbf{X}_{M-1}(n-1)$
$g_{M-1}(n) = x(n-M+1) + \mathbf{b}'_{M-1}(n-1)\mathbf{X}_{M-1}(n)$
$\mathbf{a}_{M-1}(n) = \mathbf{a}_{M-1}(n-1) - \mathbf{K}_{M-1}(n-1)f_{M-1}(n)$
$f_{M-1}(n, n) = \alpha_{M-1}(n-1)f_{M-1}(n)$
$E_{M-1}^f(n) = wE_{M-1}^f(n-1) + \alpha_{M-1}(n-1) f_{M-1}(n) ^2$
$\begin{bmatrix} \mathbf{C}_{M-1}(n) \\ c_{\text{MM}}(n) \end{bmatrix} \equiv \mathbf{K}_M(n) = \begin{bmatrix} 0 \\ \mathbf{K}_{M-1}(n-1) \end{bmatrix} + \frac{f_{M-1}^*(n, n)}{E_{M-1}^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_{M-1}(n) \end{bmatrix}$
$\mathbf{K}_{M-1}(n) = \frac{\mathbf{C}_{M-1}(n) - c_{\text{MM}}(n)\mathbf{b}_{M-1}(n-1)}{1 - c_{\text{MM}}(n)g_{M-1}(n)}$
$\mathbf{b}_{M-1}(n) = \mathbf{b}_{M-1}(n-1) - \mathbf{K}_{M-1}(n)g_{M-1}(n)$
$\alpha_{M-1}(n) = \alpha_{M-1}(n-1) \left[\frac{1 - \frac{f_{M-1}(n)f_{M-1}^*(n, n)}{E_{M-1}^f(n)}}{1 - c_{\text{MM}}(n)g_{M-1}(n)} \right]$
$\hat{d}(n) = \mathbf{h}'_M(n-1)\mathbf{X}_M(n)$
$e_M(n) = d(n) - \hat{d}(n)$
$\mathbf{h}_M(n) = \mathbf{h}_M(n-1) + \mathbf{K}_M(n)e_M(n)$
<i>Initialization</i>
$\mathbf{a}_{M-1}(-1) = \mathbf{b}_{M-1}(-1) = \mathbf{0}$
$\mathbf{K}_{M-1}(-1) = \mathbf{0}, \quad \mathbf{h}_{M-1}(-1) = \mathbf{0}$
$E_{M-1}^f(-1) = \epsilon > 0$

stabilize these fast ($7M$) algorithms with a relatively small increase in the number of computations; two stabilized fast RLS algorithms are given in Section 13.4.

13.3.4 Properties of the Direct-Form RLS Algorithms

A major advantage of the direct-form RLS algorithms over the LMS algorithm is their faster convergence rate. This characteristic behavior is illustrated in Figure 13.3.1, which shows the convergence rate of the LMS and direct-form RLS algorithms for an adaptive FIR channel equalizer of length $M = 11$. The statistical autocorrelation matrix Γ_M for the received signal has an eigenvalue ratio of $\lambda_{\max}/\lambda_{\min} = 11$. All the equalizer coefficients were initially set to zero. The step size for the LMS algorithm was selected as $\Delta = 0.02$, which represents a good compromise between convergence rate and excess MSE.

The superiority of the RLS algorithm in achieving faster convergence is clearly evident. The algorithm converges in less than 70 iterations (70 signal samples) while

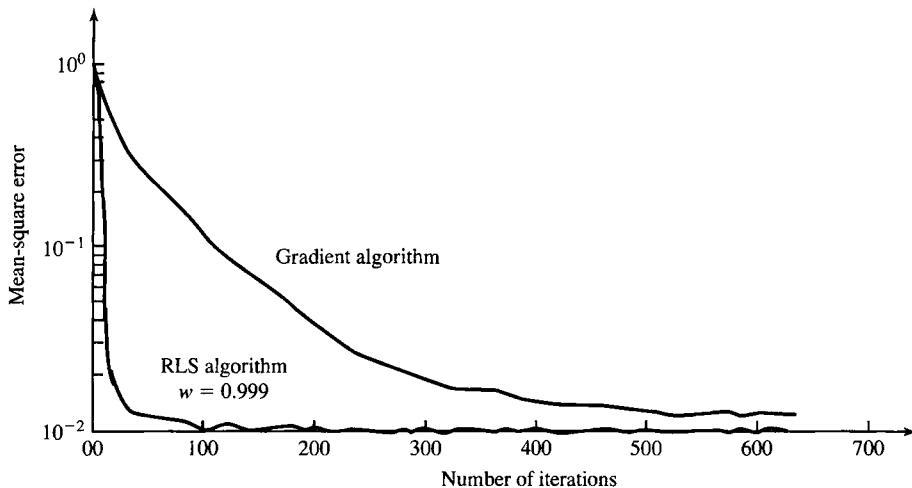


Figure 13.3.1 Learning curves for RLS and LMS algorithms for adaptive equalizer of length $M = 11$. The eigenvalue spread of the channel is $\lambda_{\max}/\lambda_{\min} = 11$. The step size for the LMS algorithm is $\Delta = 0.02$. (From *Digital Communication* by John G. Proakis, © 1983 by McGraw-Hill Book Company. Reprinted with permission of the publisher.)

the LMS algorithm has not converged in over 600 iterations. This rapid rate of convergence of the RLS algorithm is extremely important in applications in which the signal statistics vary rapidly with time. For example, the time variations of the characteristics of an ionospheric high-frequency (HF) radio channel result in frequent fading of the signal to the point where the signal strength is comparable to or even lower than the additive noise. During a signal fade, both the LMS and RLS algorithms are unable to track the channel characteristics. As the signal emerges from the fade, the channel characteristics are generally different than those prior to the fade. In such a case, the LMS algorithm is slow to adapt to the new channel characteristics. On the other hand, the RLS algorithm adapts sufficiently fast to track such rapid variations (Hsu (1982)).

Despite their superior convergence rate, the RLS algorithms for FIR adaptive filtering described in the previous section have two important disadvantages. One is their computational complexity. The square-root algorithms have a complexity proportional to M^2 . The fast RLS algorithms have a computational complexity proportional to M , but the proportionality factor is four to five times that of the LMS algorithm.

The second disadvantage of the algorithms is their sensitivity to round-off errors that accumulate as a result of the recursive computations. In some cases, the round-off errors cause these algorithms to become unstable.

The numerical properties of the RLS algorithms have been investigated by several researchers, including Ling and Proakis (1984a), Ljung and Ljung (1985), and Cioffi (1987b). For illustrative purposes, Table 13.4 includes simulation results on the steady-state (time-averaged) square error for the RLS square-root algorithm, the fast RLS algorithm in Table 13.2, and the LMS algorithm, for different word lengths.

TABLE 13.4 Numerical Accuracy of FIR Adaptive Filtering Algorithms (Least-Squares Error $\times 10^{-3}$)

Number of bits (including sign)	Algorithm		
	RLS square root	Fast RLS	LMS
16	2.17	2.17	2.30
13	2.33	2.21	2.30
11	6.14	3.34	19.0
10	17.6	^a	77.2
9	75.3	^a	311.0
8	^a	^a	1170.0

^a Algorithm does not converge to optimum coefficients.

The simulation was performed with a linear adaptive equalizer having $M = 11$ coefficients. The channel had an eigenvalue ratio of $\lambda_{\max}/\lambda_{\min} = 11$. The exponential weighting factor used in the RLS algorithms was $w = 0.975$ and the step size for the LMS algorithm was $\Delta = 0.025$. The additive noise has a variance of 0.001. The output MSE with infinite precision is 2.1×10^{-3} .

We should indicate that the direct-form RLS algorithm becomes unstable and, hence, does not work properly with 16-bit fixed-point arithmetic. For this algorithm, we found experimentally that approximately 20–24 bits of precision are needed for the algorithm to work properly. On the other hand, the square-root algorithm works down to about 9 bits, but the degradation in performance below 11 bits is significant. The fast RLS algorithm works well down to 11 bits for short durations of the order of 500 iterations. For a much larger number of iterations, the algorithm becomes unstable due to the accumulation of round-off errors. In such a case, several methods have been proposed to restart the algorithm in order to prevent overflow in the coefficients. The interested reader may refer to Eleftheriou and Falconer (1987), Cioffi and Kailath (1984), and Hsu (1982). Alternatively, one may modify the algorithm as proposed by Slock and Kailath (1988; 1991) and, thus, stabilize it.

We also observe from the results of Table 13.4 that the LMS algorithm is quite robust to round-off noise. It deteriorates as expected with a decrease in the precision of the filter coefficients, but no catastrophic failure (instability) occurs with 8 or 9 bits of precision. However, the degradation in performance below 12 bits is significant.

13.4 Adaptive Lattice-Ladder Filters

In Chapters 9 and 12, we demonstrated that an FIR filter may also be realized as a lattice structure in which the lattice parameters, called the *reflection coefficients*, are related to the filter coefficients in the direct-form FIR structure. The method for converting the FIR filter coefficients into the reflection coefficients (and vice versa) was also described.

In this section, we derive adaptive filtering algorithms in which the filter structure is a lattice or a lattice-ladder. These adaptive lattice-ladder filter algorithms,

based on the method of least squares, have several desirable properties, including computational efficiency and robustness to round-off errors. From the development of the RLS lattice-ladder algorithms, we obtain the fast RLS algorithms that were described in Section 13.3.3.

13.4.1 Recursive Least-Squares Lattice-Ladder Algorithms

In Chapter 12, we showed the relationship between the lattice filter structure and a linear predictor, and derived the equations that relate the predictor coefficients to the reflection coefficients of the lattice (and vice versa). We also established the relationship between the Levinson-Durbin recursions for the linear predictor coefficients and the reflection coefficients in the lattice filter. From these developments, we would expect to obtain the recursive least-squares lattice filter by formulating the least-squares estimation problem in terms of linear prediction. This is the approach that we take.

The recursive least-squares algorithms for the direct-form FIR structures described in Section 13.3.1 are recursive in time only. The length of the filter is fixed. A change (increase or decrease) in the filter length results in a new set of filter coefficients that are totally different from the previous set.

In contrast, the lattice filter is order recursive. As a consequence, the number of sections that it contains can be easily increased or decreased without affecting the reflection coefficients of the remaining sections. This and several other advantages (described in this and subsequent sections) make the lattice filter very attractive for adaptive filtering applications.

To begin, suppose that we observe the signal $x(n-l)$, $l = 1, 2, \dots, m$, and let us consider the linear prediction of $x(n)$. Let $f_m(l, n)$ denote the forward prediction error for an m th-order predictor, defined as

$$f_m(l, n) = x(l) + \mathbf{a}_m^t(n) \mathbf{X}_m(l-1) \quad (13.4.1)$$

where the vector $-\mathbf{a}_m(n)$ consists of the forward prediction coefficients, that is,

$$\mathbf{a}_m^t(n) = [a_m(1, n) \ a_m(2, n) \ \dots \ a_m(m, n)] \quad (13.4.2)$$

and the data vector $\mathbf{X}_m(l-1)$ is

$$\mathbf{X}_m^t(l-1) = [x(l-1) \ x(l-2) \ \dots \ x(l-m)] \quad (13.4.3)$$

The predictor coefficients $\mathbf{a}_m(n)$ are selected to minimize the time-averaged weighted squared error

$$\mathcal{E}_m^f(n) = \sum_{l=0}^n w^{n-l} |f_m(l, n)|^2 \quad (13.4.4)$$

The minimization of $\mathcal{E}_m^f(n)$ with respect to $\mathbf{a}_m(n)$ leads to the set of linear equations

$$\mathbf{R}_m(n-1) \mathbf{a}_m(n) = -\mathbf{Q}_m(n) \quad (13.4.5)$$

where $\mathbf{R}_m(n)$ is defined by (13.3.7) and $\mathbf{Q}_m(n)$ is defined as

$$\mathbf{Q}_m(n) = \sum_{l=0}^n w^{n-l} x(l) \mathbf{X}_m^*(l-1) \quad (13.4.6)$$

The solution of (13.4.5) is

$$\mathbf{a}_m(n) = -\mathbf{R}_m^{-1}(n-1) \mathbf{Q}_m(n) \quad (13.4.7)$$

The minimum value of $\mathcal{E}_m^f(n)$, obtained with the linear predictor specified by (13.4.7), is denoted as $E_m^f(n)$ and is given by

$$\begin{aligned} E_m^f(n) &= \sum_{l=0}^n w^{n-l} x^*(l) [x(l) + \mathbf{a}_m^t(n) \mathbf{X}_m(l-1)] \\ &= q(n) + \mathbf{a}_m^t(n) \mathbf{Q}_m^*(n) \end{aligned} \quad (13.4.8)$$

where $q(n)$ is defined as

$$q(n) = \sum_{l=0}^n w^{n-l} |x(l)|^2 \quad (13.4.9)$$

The linear equations in (13.4.5) and the equation for $E_m^f(n)$ in (13.4.8) can be combined in a single matrix equation of the form

$$\begin{bmatrix} q(n) & \mathbf{Q}_m^H(n) \\ \mathbf{Q}_m(n) & \mathbf{R}_m(n-1) \end{bmatrix} \begin{bmatrix} 1 \\ \mathbf{a}_m(n) \end{bmatrix} = \begin{bmatrix} E_m^f(n) \\ \mathbf{O}_m \end{bmatrix} \quad (13.4.10)$$

where \mathbf{O}_m is the m -dimensional null vector. It is interesting to note that

$$\begin{aligned} \mathbf{R}_{m+1}(n) &= \sum_{l=0}^n w^{n-l} \mathbf{X}_{m+1}^*(l) \mathbf{X}_{m+1}^t(l) \\ &= \sum_{l=0}^n w^{n-l} \begin{bmatrix} x^*(l) \\ \mathbf{X}_m^*(l-1) \end{bmatrix} [x(l) \mathbf{X}_m^t(l-1)] \\ &= \begin{bmatrix} q(n) & \mathbf{Q}_m^H(n) \\ \mathbf{Q}_m(n) & \mathbf{R}_m(n-1) \end{bmatrix} \end{aligned} \quad (13.4.11)$$

which is the matrix in (13.4.10).

In a completely parallel development to (13.4.1) through (13.4.11), we minimize the backward time-averaged weighted squared error for an m th-order backward predictor defined as

$$\mathcal{E}_m^b(n) = \sum_{l=0}^n w^{n-l} |g_m(l, n)|^2 \quad (13.4.12)$$

where the backward error is defined as

$$g_m(l, n) = x(l - m) + \mathbf{b}_m^t(n) \mathbf{X}_m(l) \quad (13.4.13)$$

and $\mathbf{b}_m^t(n) = [b_m(1, n) \ b_m(2, n) \dots b_m(m, n)]$ is the vector of coefficients for the backward predictor. The minimization of $\mathcal{E}_m^b(n)$ leads to the equation

$$\mathbf{R}_m(n) \mathbf{b}_m(n) = -\mathbf{V}_m(n) \quad (13.4.14)$$

and, hence, to the solution

$$\mathbf{b}_m(n) = -\mathbf{R}_m^{-1}(n) \mathbf{V}_m(n) \quad (13.4.15)$$

where

$$\mathbf{V}_m(n) = \sum_{l=0}^n w^{n-l} x(l - m) \mathbf{X}_m^*(l) \quad (13.4.16)$$

The minimum value of $\mathcal{E}_m^b(n)$, denoted as $E_m^b(n)$, is

$$\begin{aligned} E_m^b(n) &= \sum_{l=0}^n w^{n-l} [x(l - m) + \mathbf{b}_m^t(n) \mathbf{X}_m(l)] x^*(l - m) \\ &= v(n) + \mathbf{b}_m^t(n) \mathbf{V}_m^*(n) \end{aligned} \quad (13.4.17)$$

where the scalar quantity $v(n)$ is defined as

$$v(n) = \sum_{l=0}^n w^{n-l} |x(l - m)|^2 \quad (13.4.18)$$

If we combine (13.4.14) and (13.4.17) into a single equation, we obtain

$$\begin{bmatrix} \mathbf{R}_m(n) & \mathbf{V}_m(n) \\ \mathbf{V}_m^H(n) & v(n) \end{bmatrix} \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{O}_m \\ E_m^b(n) \end{bmatrix} \quad (13.4.19)$$

We also note that the (estimated) autocorrelation matrix $\mathbf{R}_{m+1}(n)$ can be expressed as

$$\begin{aligned} \mathbf{R}_{m+1}(n) &= \sum_{l=0}^n w^{n-l} \begin{bmatrix} \mathbf{X}_m^*(l) \\ x^*(l - m) \end{bmatrix} [\mathbf{X}_m^t(l) x(l - m)] \\ &= \begin{bmatrix} \mathbf{R}_m(n) & \mathbf{V}_m(n) \\ \mathbf{V}_m^H(n) & v(n) \end{bmatrix} \end{aligned} \quad (13.4.20)$$

Thus, we have obtained the equations for the forward and backward least-squares predictors of order m .

Next, we derive the order-update equations for these predictors, which will lead us to the lattice filter structure. In deriving the order-update equations for $\mathbf{a}_m(n)$ and

$\mathbf{b}_m(n)$, we will make use of the two matrix inversion identities for a matrix of the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \quad (13.4.21)$$

where \mathbf{A} , \mathbf{A}_{11} , and \mathbf{A}_{22} are square matrices. The inverse of \mathbf{A} is expressible in two different forms, namely,

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{A}_{11}^{-1} + \mathbf{A}_{11}^{-1}\mathbf{A}_{12}\tilde{\mathbf{A}}_{22}^{-1}\mathbf{A}_{21}\mathbf{A}_{11}^{-1} & -\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\tilde{\mathbf{A}}_{22}^{-1} \\ -\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{A}_{11}^{-1} & \tilde{\mathbf{A}}_{22}^{-1} \end{bmatrix} \quad (13.4.22)$$

and

$$\mathbf{A}^{-1} = \begin{bmatrix} \tilde{\mathbf{A}}_{11}^{-1} & -\tilde{\mathbf{A}}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22}^{-1} \\ -\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\tilde{\mathbf{A}}_{11}^{-1} & \mathbf{A}_{22}^{-1}\mathbf{A}_{21}\tilde{\mathbf{A}}_{11}^{-1}\mathbf{A}_{12}\mathbf{A}_{22}^{-1} + \mathbf{A}_{22}^{-1} \end{bmatrix} \quad (13.4.23)$$

where $\tilde{\mathbf{A}}_{11}$ and $\tilde{\mathbf{A}}_{12}$ are defined as

$$\begin{aligned} \tilde{\mathbf{A}}_{11} &= \mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21} \\ \tilde{\mathbf{A}}_{22} &= \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12} \end{aligned} \quad (13.4.24)$$

Order-Update Recursions. Now, let us use the formula in (13.4.22) to obtain the inverse of $\mathbf{R}_{m+1}(n)$ by using the form in (13.4.20). First, we have

$$\begin{aligned} \tilde{\mathbf{A}}_{22} &= v(n) - \mathbf{V}_m^H(n)\mathbf{R}_m^{-1}(n)\mathbf{V}_m(n) \\ &= v(n) + \mathbf{b}_m^t(n)\mathbf{V}_m^*(n) = E_m^b(n) \end{aligned} \quad (13.4.25)$$

and

$$\mathbf{A}_{11}^{-1}\mathbf{A}_{12} = \mathbf{R}_m^{-1}(n)\mathbf{V}_m(n) = -\mathbf{b}_m(n) \quad (13.4.26)$$

Hence,

$$\mathbf{R}_{m+1}^{-1}(n) \equiv \mathbf{P}_{m+1}(n) = \begin{bmatrix} \mathbf{P}_m(n) + \frac{\mathbf{b}_m(n)\mathbf{b}_m^H(n)}{E_m^b(n)} & \frac{\mathbf{b}_m(n)}{E_m^b(n)} \\ \frac{\mathbf{b}_m^H(n)}{E_m^b(n)} & \frac{1}{E_m^b(n)} \end{bmatrix}$$

or, equivalently,

$$\mathbf{P}_{m+1}(n) = \begin{bmatrix} \mathbf{P}_m(n) & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ \mathbf{b}_m^H(n) \end{bmatrix} \begin{bmatrix} \mathbf{b}_m^H(n) & 1 \end{bmatrix} \quad (13.4.27)$$

By substituting $n - 1$ for n in (13.4.27) and postmultiplying the result by $-\mathbf{Q}_{m+1}(n)$, we obtain the order update for $\mathbf{a}_m(n)$. Thus,

$$\begin{aligned} \mathbf{a}_{m+1}(n) &= -\mathbf{P}_{m+1}(n-1)\mathbf{Q}_{m+1}(n) \\ &= \begin{bmatrix} \mathbf{P}_m(n-1) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -\mathbf{Q}_m^{(n)} \\ \dots \end{bmatrix} \\ &\quad - \frac{1}{E_m^b(n-1)} \begin{bmatrix} \mathbf{b}_m(n-1) \\ 1 \end{bmatrix} \begin{bmatrix} \mathbf{b}_m^H(n-1) & 1 \end{bmatrix} \mathbf{Q}_{m+1}(n) \\ &= \begin{bmatrix} \mathbf{a}_m(n) \\ 0 \end{bmatrix} - \frac{k_{m+1}(n)}{E_m^b(n-1)} \begin{bmatrix} \mathbf{b}_m(n-1) \\ 1 \end{bmatrix} \end{aligned} \quad (13.4.28)$$

where the scalar quantity $k_{m+1}(n)$ is defined as

$$k_{m+1}(n) = [\mathbf{b}_m^H(n-1) \quad 1] \mathbf{Q}_{m+1}(n) \quad (13.4.29)$$

The reader should observe that (13.4.28) is a Levinson-type recursion for the predictor coefficients.

To obtain the corresponding order update for $\mathbf{b}_m(n)$, we use the matrix inversion formula in (13.4.23) for the inverse of $\mathbf{R}_{m+1}(n)$, along with the form in (13.4.11). In this case, we have

$$\begin{aligned} \tilde{\mathbf{A}}_{11} &= q(n) - \mathbf{Q}_m^H(n) \mathbf{R}_m^{-1}(n-1) \mathbf{Q}_m(n) \\ &= q(n) + \mathbf{a}_m^t(n) \mathbf{Q}_m^*(n) = E_m^f(n) \end{aligned} \quad (13.4.30)$$

and

$$\mathbf{A}_{22}^{-1} \mathbf{A}_{21} = \mathbf{R}_m^{-1}(n-1) \mathbf{Q}_m(n) = -\mathbf{a}_m(n) \quad (13.4.31)$$

Hence,

$$\mathbf{P}_{m+1}(n) = \begin{bmatrix} \frac{1}{E_m^f(n)} & \frac{\mathbf{a}_m^H(n)}{E_m^f(n)} \\ \frac{\mathbf{a}_m(n)}{E_m^f(n)} & \mathbf{P}_m(n-1) + \frac{\mathbf{a}_m(n)\mathbf{a}_m^H(n)}{E_m^f(n)} \end{bmatrix}$$

or, equivalently,

$$\mathbf{P}_{m+1}(n) = \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{P}_m(n-1) \end{bmatrix} + \frac{1}{E_m^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_m(n) \end{bmatrix} \begin{bmatrix} 1 & \mathbf{a}_m^H(n) \end{bmatrix} \quad (13.4.32)$$

Now, if we postmultiply (13.4.32) by $-\mathbf{V}_{m+1}(n)$, we obtain

$$\begin{aligned} \mathbf{b}_{m+1}(n) &= \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{P}_m(n-1) \end{bmatrix} \begin{bmatrix} \dots \\ -\mathbf{V}_m(n-1) \end{bmatrix} \\ &\quad - \frac{1}{E_m^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_m(n) \end{bmatrix} \begin{bmatrix} 1 & \mathbf{a}_m^H(n) \end{bmatrix} \mathbf{V}_{m+1}(n) \\ &= \begin{bmatrix} 0 \\ \mathbf{b}_m(n-1) \end{bmatrix} - \frac{k_{m+1}^*(n)}{E_m^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_m(n) \end{bmatrix} \end{aligned} \quad (13.4.33)$$

where

$$[1 \quad \mathbf{a}_m^H(n)] \mathbf{V}_{m+1}(n) = [\mathbf{b}_m^t(n-1) \quad 1] \mathbf{Q}_{m+1}^*(n) = k_{m+1}^*(n) \quad (13.4.34)$$

The proof of (13.4.34) and its relation to (13.4.29) is left as an exercise for the reader. Thus, (13.4.28) and (13.4.33) specify the order-update equations for $\mathbf{a}_m(n)$ and $\mathbf{b}_m(n)$, respectively.

The order-update equations for $E_m^f(n)$ and $E_m^b(n)$ may now be obtained. From the definition of $E_m^f(n)$ given by (13.4.8), we have

$$E_{m+1}^f(n) = q(n) + \mathbf{a}_{m+1}^t(n) \mathbf{Q}_{m+1}^*(n) \quad (13.4.35)$$

By substituting for $\mathbf{a}_{m+1}(n)$ from (13.4.28) into (13.4.35), we obtain

$$\begin{aligned} E_{m+1}^f(n) &= q(n) + [\mathbf{a}'_m(n) \quad 0] \begin{bmatrix} \mathbf{Q}_m^*(n) \\ \vdots \\ E_m^b(n-1) \end{bmatrix} - \frac{k_{m+1}(n)}{E_m^b(n-1)} [\mathbf{b}'_m(n-1) \quad 1] \mathbf{Q}_{m+1}^*(n) \\ &= E_m^f(n) - \frac{|k_{m+1}(n)|^2}{E_m^b(n-1)} \end{aligned} \quad (13.4.36)$$

Similarly, by using (13.4.17) and (13.4.33), we obtain the order update for $E_{m+1}^b(n)$, in the form

$$E_{m+1}^b(n) = E_m^b(n-1) - \frac{|k_{m+1}(n)|^2}{E_m^f(n)} \quad (13.4.37)$$

The lattice filter is specified by two coupled equations involving the forward and backward errors $f_m(n, n-1)$ and $g_m(n, n-1)$, respectively. From the definition of the forward error in (13.4.1), we have

$$f_{m+1}(n, n-1) = x(n) + \mathbf{a}_{m+1}^t(n-1) \mathbf{X}_{m+1}(n-1) \quad (13.4.38)$$

Substituting for $\mathbf{a}_{m+1}^t(n-1)$ from (13.4.28) into (13.4.38) yields

$$\begin{aligned} f_{m+1}(n, n-1) &= x(n) + [\mathbf{a}'_m(n-1) \quad 0] \begin{bmatrix} \mathbf{X}_m(n-1) \\ \vdots \\ E_m^b(n-2) \end{bmatrix} \\ &\quad - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} [\mathbf{b}'_m(n-2) \quad 1] \mathbf{X}_{m+1}(n-1) \\ &= f_m(n, n-1) - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} \times [x(n-m-1) + \mathbf{b}'_m(n-2) \mathbf{X}_m(n-1)] \\ &= f_m(n, n-1) - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} g_m(n-1, n-2) \end{aligned} \quad (13.4.39)$$

To simplify the notation, we define

$$\begin{aligned} f_m(n) &= f_m(n, n-1) \\ g_m(n) &= g_m(n, n-1) \end{aligned} \quad (13.4.40)$$

Then, (13.4.39) may be expressed as

$$f_{m+1}(n) = f_m(n) - \frac{k_{m+1}(n-1)}{E_m^b(n-2)} g_m(n-1) \quad (13.4.41)$$

Similarly, beginning with the definition of the backward error given by (13.4.13), we have

$$g_{m+1}(n, n-1) = x(n-m-1) + \mathbf{b}_{m+1}^t(n-1) \mathbf{X}_{m+1}(n) \quad (13.4.42)$$

Substituting for $b_{m+1}(n-1)$ from (13.4.33) and simplifying the result, we obtain

$$g_{m+1}(n, n-1) = g_m(n-1, n-2) - \frac{k_{m+1}^*(n-1)}{E_m^f(n-1)} f_m(n, n-1) \quad (13.4.43)$$

or, equivalently,

$$g_{m+1}(n) = g_m(n-1) - \frac{k_{m+1}^*(n-1)}{E_m^f(n-1)} f_m(n) \quad (13.4.44)$$

The two recursive equations in (13.4.41) and (13.4.44) specify the lattice filter illustrated in Figure 13.4.1 where, for notational convenience, we have defined the reflection coefficients for the lattice as

$$\begin{aligned} \mathcal{R}_m^f(n) &= \frac{-k_m(n)}{E_{m-1}^b(n-1)} \\ \mathcal{R}_m^b(n) &= \frac{-k_m^*(n)}{E_{m-1}^f(n)} \end{aligned} \quad (13.4.45)$$

The initial conditions on the order updates are

$$\begin{aligned} f_0(n) &= g_0(n) = x(n) \\ E_0^f(n) = E_0^b(n) &= \sum_{l=0}^n w^{n-l} |x(l)|^2 = w E_0^f(n-1) + |x(n)|^2 \end{aligned} \quad (13.4.46)$$

We note that (13.4.46) is also a time-update equation for $E_0^f(n)$ and $E_0^b(n)$.

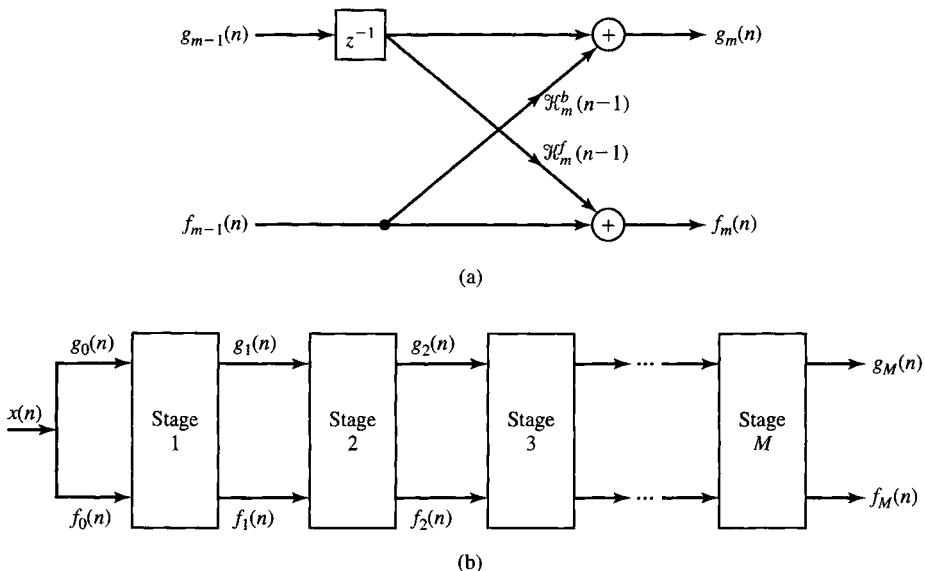


Figure 13.4.1 Least-squares lattice filter.

Time-Update Recursions. Our goal is to determine a time-update equation for $k_m(n)$, which is necessary if the lattice filter is to be adaptive. This derivation will require time-update equations for the prediction coefficients. We begin with the form

$$k_{m+1}(n) = -\mathbf{V}_{m+1}^H(n) \begin{bmatrix} 1 \\ \mathbf{a}_m(n) \end{bmatrix} \quad (13.4.47)$$

The time-update equation for $\mathbf{V}_{m+1}(n)$ is

$$\mathbf{V}_{m+1}(n) = w\mathbf{V}_{m+1}(n-1) + x(n-m-1)\mathbf{X}_{m+1}^*(n) \quad (13.4.48)$$

The time-update equations for the prediction coefficients are determined as follows. From (13.4.6), (13.4.7), and (13.3.14), we have

$$\begin{aligned} \mathbf{a}_m(n) &= -\mathbf{P}_m(n-1)\mathbf{Q}_m(n) \\ &= -\frac{1}{w} [\mathbf{P}_m(n-2) - \mathbf{K}_m(n-1)\mathbf{X}_m^t(n-1)\mathbf{P}_m(n-2)] \\ &\quad \times [w\mathbf{Q}_m(n-1) + x(n)\mathbf{X}_m^*(n-1)] \\ &= \mathbf{a}_m(n-1) - \mathbf{K}_m(n-1)[x(n) + \mathbf{a}_m^t(n-1)\mathbf{X}_m(n-1)] \end{aligned} \quad (13.4.49)$$

where $\mathbf{K}_m(n-1)$ is the Kalman gain vector at iteration $n-1$. But, from (13.4.38), we have

$$x(n) + \mathbf{a}_m^t(n-1)\mathbf{X}_m(n-1) = f_m(n, n-1) \equiv f_m(n)$$

Therefore, the time-update equation for $\mathbf{a}_m(n)$ is

$$\mathbf{a}_m(n) = \mathbf{a}_m(n-1) - \mathbf{K}_m(n-1)f_m(n) \quad (13.4.50)$$

In a parallel development, using (13.4.15), (13.4.16), and (13.3.14), we obtain the time-update equations for the coefficients of the backward predictor, in the form

$$\mathbf{b}_m(n) = \mathbf{b}_m(n-1) - \mathbf{K}_m(n)g_m(n) \quad (13.4.51)$$

Now, from (13.4.48) and (13.4.50), the time-update equation for $k_{m+1}(n)$ is

$$\begin{aligned} k_{m+1}(n) &= -[w\mathbf{V}_{m+1}^H(n-1) + x^*(n-m-1)\mathbf{X}_{m+1}^t(n)] \\ &\quad \times \left(\begin{bmatrix} 1 \\ \mathbf{a}_m(n-1) \end{bmatrix} - \begin{bmatrix} 0 \\ \mathbf{K}_m(n-1)f_m(n) \end{bmatrix} \right) \\ &= wk_{m+1}(n-1) - w\mathbf{V}_{m+1}^H(n-1) \begin{bmatrix} 0 \\ \mathbf{K}_m(n-1) \end{bmatrix} f_m(n) \\ &\quad + x^*(n-m-1)\mathbf{X}_{m+1}^t(n) \begin{bmatrix} 1 \\ \mathbf{a}_m(n-1) \end{bmatrix} \\ &\quad - x^*(n-m-1)\mathbf{X}_{m+1}^t(n) \begin{bmatrix} 0 \\ \mathbf{K}_m(n-1) \end{bmatrix} f_m(n) \end{aligned} \quad (13.4.52)$$

But

$$\mathbf{X}_{m+1}^t(n) \begin{bmatrix} 1 \\ \mathbf{a}_m(n-1) \end{bmatrix} = [x(n) \mathbf{X}_m^t(n-1)] \begin{bmatrix} 1 \\ \mathbf{a}_m(n-1) \end{bmatrix} = f_m(n) \quad (13.4.53)$$

and

$$\begin{aligned} \mathbf{V}_{m+1}^H(n-1) \begin{bmatrix} 0 \\ \mathbf{K}_m(n-1) \end{bmatrix} &= \mathbf{V}_m^H(n-2) \mathbf{K}_m(n-1) \\ &= \frac{\mathbf{V}_m^H(n-2) \mathbf{P}_m(n-2) \mathbf{X}_m^*(n-1)}{w + \mu_m(n-1)} \\ &= \frac{-\mathbf{b}_m^H(n-2) \mathbf{X}_m^*(n-1)}{w + \mu_m(n-1)} \\ &= -\frac{g_m^*(n-1) - x^*(n-m-1)}{w + \mu_m(n-1)} \end{aligned} \quad (13.4.54)$$

where $\mu_m(n)$ is as previously defined in (13.3.13). Finally,

$$\mathbf{X}_{m+1}^t(n) \begin{bmatrix} 0 \\ \mathbf{K}_m(n-1) \end{bmatrix} = \frac{\mathbf{X}_m^t(n-1) \mathbf{P}_m(n-2) \mathbf{X}_m^*(n-1)}{w + \mu_m(n-1)} = \frac{\mu_m(n-1)}{w + \mu_m(n-1)} \quad (13.4.55)$$

Substituting the results of (13.4.53), (13.4.54), and (13.4.55) into (13.4.52) we obtain the desired time-update equation in the form

$$k_{m+1}(n) = w k_{m+1}(n-1) + \frac{w}{w + \mu_m(n-1)} f_m(n) g_m^*(n-1) \quad (13.4.56)$$

It is convenient to define a new variable

$$\alpha_m(n) = \frac{w}{w + \mu_m(n)} \quad (13.4.57)$$

Clearly, $\alpha_m(n)$ is real-valued and has a range $0 < \alpha_m(n) < 1$. Then, the time-update equation (13.4.56) becomes

$$k_{m+1}(n) = w k_{m+1}(n-1) + \alpha_m(n-1) f_m(n) g_m^*(n-1) \quad (13.4.58)$$

Order Update for $\alpha_m(n)$. Although $\alpha_m(n)$ can be computed directly for each value of m and for each n , it is more efficient to use an order-update equation, which is determined as follows. First, from the definition of $\mathbf{K}_m(n)$ given in (13.3.12), it is easily seen that

$$\alpha_m(n) = 1 - \mathbf{X}_m^t(n) \mathbf{K}_m(n) \quad (13.4.59)$$

To obtain an order-update equation for $\alpha_m(n)$, we need an order-update equation for the Kalman gain vector $\mathbf{K}_m(n)$. But $\mathbf{K}_{m+1}(n)$ may be expressed as

$$\begin{aligned}\mathbf{K}_{m+1}(n) &= \mathbf{P}_{m+1}(n)\mathbf{X}_{m+1}^*(n) \\ &= \left(\begin{bmatrix} \mathbf{P}_m(n) & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix} \begin{bmatrix} \mathbf{b}_m^H(n) & 1 \end{bmatrix} \right) \begin{bmatrix} \mathbf{X}_m^*(n) \\ x^*(n-m) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{K}_m(n) \\ 0 \end{bmatrix} + \frac{g_m^*(n, n)}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix}\end{aligned}\quad (13.4.60)$$

The term $g_m(n, n)$ may also be expressed as

$$\begin{aligned}g_m(n, n) &= x(n-m) + \mathbf{b}_m^t(n)\mathbf{X}_m(n) \\ &= x(n-m) + [\mathbf{b}_m^t(n-1) - \mathbf{K}_m^t(n)g_m(n)]\mathbf{X}_m(n) \\ &= x(n-m) + \mathbf{b}_m^t(n-1)\mathbf{X}_m(n) - g_m(n)\mathbf{K}_m^t(n)\mathbf{X}_m(n) \\ &= g_m(n)[1 - \mathbf{K}_m^t(n)\mathbf{X}_m(n)] \\ &= \alpha_m(n)g_m(n)\end{aligned}\quad (13.4.61)$$

Hence, the order-update equation for $\mathbf{K}_m(n)$ in (13.4.60) may also be written as

$$\mathbf{K}_{m+1}(n) = \begin{bmatrix} \mathbf{K}_m(n) \\ 0 \end{bmatrix} + \frac{\alpha_m(n)g_m^*(n)}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix}\quad (13.4.62)$$

By using (13.4.62) and the relation in (13.4.59), we obtain the order-update equation for $\alpha_m(n)$ as follows:

$$\begin{aligned}\alpha_{m+1}(n) &= 1 - \mathbf{X}_{m+1}^t(n)\mathbf{K}_{m+1}(n) = 1 - [\mathbf{X}_m^t(n)x(n-m)] \\ &\quad \times \left(\begin{bmatrix} \mathbf{K}_m(n) \\ 0 \end{bmatrix} + \frac{\alpha_m(n)g_m^*(n)}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix} \right) \\ &= \alpha_m(n) - \frac{\alpha_m(n)g_m^*(n)}{E_m^b(n)} [\mathbf{X}_m^t(n)x(n-m)] \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix} \\ &= \alpha_m(n) - \frac{\alpha_m(n)g_m^*(n)}{E_m^b(n)} g_m(n, n) \\ &= \alpha_m(n) - \frac{\alpha_m^2(n)|g_m(n)|^2}{E_m^b(n)}\end{aligned}\quad (13.4.63)$$

Thus, we have obtained both the order-update and time-update equations for the basic least-squares lattice shown in Figure 13.4.1. The basic equations are (13.4.41) and (13.4.44) for the forward and backward errors, usually called the *residuals*, (13.4.36) and (13.4.37) for the corresponding least-squares errors, the time-update

Equation (13.4.58) for $k_m(n)$, and the order-update Equation (13.4.63) for the parameter $\alpha_m(n)$. Initially, we have

$$\begin{aligned} E_m^f(-1) &= E_m^b(-1) = E_m^b(-2) = \epsilon > 0 \\ f_m(-1) &= g_m(-1) = k_m(-1) = 0 \\ \alpha_m(-1) &= 1, \quad \alpha_{-1}(n) = \alpha_{-1}(n-1) = 1 \end{aligned} \quad (13.4.64)$$

Joint Process Estimation. The last step in the derivation is to obtain the least-squares estimate of the desired signal $d(n)$ from the lattice. Suppose that the adaptive filter has $m + 1$ coefficients, which are determined to minimize the average weighted squared error

$$\mathcal{E}_{m+1} = \sum_{l=0}^n w^{n-l} |e_{m+1}(l, n)|^2 \quad (13.4.65)$$

where

$$e_{m+1}(l, n) = d(l) - \mathbf{h}_{m+1}^t(n) \mathbf{X}_{m+1}(l) \quad (13.4.66)$$

The linear estimate

$$\hat{d}(l, n) = \mathbf{h}_{m+1}^t(n) \mathbf{X}_{m+1}(l) \quad (13.4.67)$$

which will be obtained from the lattice by using the residuals $g_m(n)$, is called the *joint process estimate*.

From the results of Section 13.3.1, we have already established that the coefficients of the adaptive filter that minimize (13.4.65) are given by the equation

$$\mathbf{h}_{m+1}(n) = \mathbf{P}_{m+1}(n) \mathbf{D}_{m+1}(n) \quad (13.4.68)$$

We have also established that $\mathbf{h}_m(n)$ satisfies the time-update equation given in (13.3.27).

Now, let us obtain an order-update equation for $\mathbf{h}_m(n)$. From (13.4.68) and (13.4.27), we have

$$\mathbf{h}_{m+1}(n) = \begin{bmatrix} \mathbf{P}_m(n) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{D}_m(n) \\ \dots \end{bmatrix} + \frac{1}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ \vdots \\ 1 \end{bmatrix} [\mathbf{b}_m^H(n) \quad 1] \mathbf{D}_{m+1}(n) \quad (13.4.69)$$

We define a complex-valued scalar quantity $\delta_m(n)$ as

$$\delta_m(n) = [\mathbf{b}_m^H(n) \quad 1] \mathbf{D}_{m+1}(n) \quad (13.4.70)$$

Then, (13.4.69) may be expressed as

$$\mathbf{h}_{m+1}(n) = \begin{bmatrix} \mathbf{h}_m(n) \\ 0 \end{bmatrix} + \frac{\delta_m(n)}{E_m^b(n)} \begin{bmatrix} \mathbf{b}_m(n) \\ 1 \end{bmatrix} \quad (13.4.71)$$

The scalar $\delta_m(n)$ satisfies a time-update equation that is obtained from the time-update equations for $\mathbf{b}_m(n)$ and $\mathbf{D}_m(n)$, given by (13.4.51) and (13.3.17), respectively. Thus,

$$\begin{aligned}\delta_m(n) &= [\mathbf{b}_m^H(n-1) - \mathbf{K}_m^H(n)g_m^*(n) \quad 1] [w\mathbf{D}_{m+1}(n-1) + d(n)\mathbf{X}_{m+1}^*(n)] \\ &= w\delta_m(n-1) + [\mathbf{b}_m^H(n-1) \quad 1] \mathbf{X}_{m+1}^*(n)d(n) \\ &\quad - w g_m^*(n) [\mathbf{K}_m^H(n) \quad 0] \mathbf{D}_{m+1}(n-1) - g_m^*(n)d(n) [\mathbf{K}_m^H(n) \quad 0] \mathbf{X}_{m+1}^*(n)\end{aligned}\quad (13.4.72)$$

But

$$[\mathbf{b}_m^H(n-1) \quad 1] \mathbf{X}_{m+1}^*(n) = x^*(n-m) + \mathbf{b}_m^H(n-1)\mathbf{X}_m^*(n) = g_m^*(n) \quad (13.4.73)$$

Also,

$$\begin{aligned}[\mathbf{K}_m^H(n) \quad 0] \mathbf{D}_{m+1}(n-1) &= \frac{1}{w + \mu_m(n)} [\mathbf{X}_m^t(n)\mathbf{P}_m(n-1) \quad 0] \begin{bmatrix} \mathbf{D}_m(n-1) \\ \vdots \end{bmatrix} \\ &= \frac{1}{w + \mu_m(n)} \mathbf{X}_m^t(n)\mathbf{h}_m(n-1)\end{aligned}\quad (13.4.74)$$

The last term in (13.4.72) may be expressed as

$$\begin{aligned}[\mathbf{K}_m^H(n) \quad 0] \begin{bmatrix} \mathbf{X}_m^*(n) \\ \vdots \end{bmatrix} &= \frac{1}{w + \mu_m(n)} \mathbf{X}_m^t(n)\mathbf{P}_m(n-1)\mathbf{X}_m^*(n) \\ &= \frac{\mu_m(n)}{w + \mu_m(n)}\end{aligned}\quad (13.4.75)$$

Upon substituting the results in (13.4.73–13.4.75) into (13.4.72), we obtain the desired time-update equation for $\delta_m(n)$ as

$$\delta_m(n) = w\delta_m(n-1) + \alpha_m(n)g_m^*(n)e_m(n) \quad (13.4.76)$$

Order-update equations for $\alpha_m(n)$ and $g_m(n)$ have already been derived. With $e_0(n) = d(n)$, the order-update equation for $e_m(n)$ is obtained as follows:

$$\begin{aligned}e_m(n) &= e_m(n, n-1) = d(n) - \mathbf{h}_m^t(n-1)\mathbf{X}_m(n) \\ &= d(n) - [\mathbf{h}_{m-1}^t(n-1) \quad 0] \begin{bmatrix} \mathbf{X}_{m-1}(n) \\ \vdots \end{bmatrix} \\ &\quad - \frac{\delta_{m-1}(n-1)}{E_{m-1}^b(n-1)} [\mathbf{b}_{m-1}^t(n-1) \quad 1] \mathbf{X}_m(n) \\ &= e_{m-1}(n) - \frac{\delta_{m-1}(n-1)g_{m-1}(n)}{E_{m-1}^b(n-1)}\end{aligned}\quad (13.4.77)$$

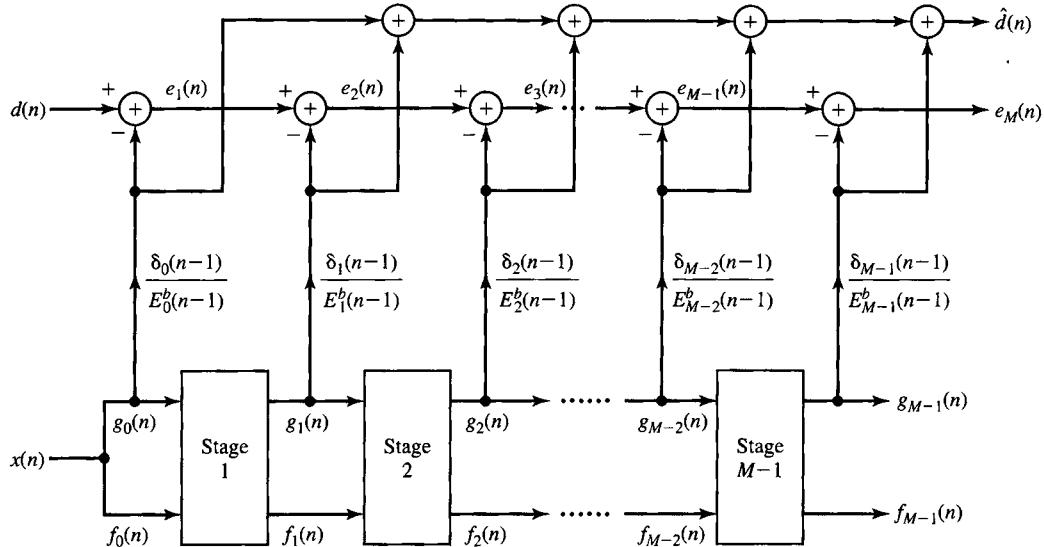


Figure 13.4.2 Adaptive RLS lattice-ladder filter.

Finally, the output estimate $d(n)$ of the least-squares lattice is

$$\hat{d}(n) = \mathbf{h}_{m+1}^t(n-1) \mathbf{X}_{m+1}(n) \quad (13.4.78)$$

But $\mathbf{h}_{m+1}^t(n-1)$ is not computed explicitly. By repeated substitution of the order-update equation for $\mathbf{h}_{m+1}(n)$ given by (13.4.71) into (13.4.78), we obtain the desired expression for $\hat{d}(n)$ in the form

$$\hat{d}(n) = \sum_{k=0}^{M-1} \frac{\delta_k(n-1)}{E_k^b(n-1)} g_k(n) \quad (13.4.79)$$

In other words, the output estimate $\hat{d}(n)$ is a linear weighted sum of the backward residuals $g_k(n)$.

The adaptive least-squares lattice/joint-process (ladder) estimator is illustrated in Figure 13.4.2. This lattice-ladder structure is mathematically equivalent to the RLS direct-form FIR filter. The recursive equations are summarized in Table 13.5. This is called the *a priori form* of the *RLS lattice-ladder algorithm* in order to distinguish it from another form of the algorithm, called the *a posteriori form*, in which the coefficient vector $\mathbf{h}_M(n)$ is used in place of $\mathbf{h}_M(n-1)$ to compute the estimate $d(n)$. In many adaptive filtering problems, such as channel equalization and echo cancellation, the *a posteriori* form cannot be used, because $\mathbf{h}_M(n)$ cannot be computed prior to the computation of $d(n)$.

We now describe a number of modifications that can be made to the “conventional” RLS lattice-ladder algorithm given in Table 13.5.

TABLE 13.5 A Priori Form of the RLS Lattice-Ladder Algorithm

Lattice predictor: Begin with $n = 1$ and compute the order updates for $m = 0, 1, \dots, M - 2$

$$k_{m+1}(n-1) = wk_{m+1}(n-2) + \alpha_m(n-2)f_m(n-1)g_m^*(n-2)$$

$$\mathcal{H}_{m+1}^f(n-1) = -\frac{k_{m+1}(n-1)}{E_m^b(n-2)}$$

$$\mathcal{H}_{m+1}^b(n-1) = -\frac{k_{m+1}^*(n-1)}{E_m^f(n-1)}$$

$$f_{m+1}(n) = f_m(n) + \mathcal{H}_{m+1}^f(n-1)g_m(n-1)$$

$$g_{m+1}(n) = g_m(n-1) + \mathcal{H}_{m+1}^b(n-1)f_m(n)$$

$$E_{m+1}^f(n-1) = E_m^f(n-1) - \frac{|k_{m+1}(n-1)|^2}{E_m^b(n-2)}$$

$$E_{m+1}^b(n-1) = E_m^b(n-2) - \frac{|k_{m+1}(n-1)|^2}{E_m^f(n-1)}$$

$$\alpha_{m+1}(n-1) = \alpha_m(n-1) - \frac{\alpha_m^2(n-1)|g_m(n-1)|^2}{E_m^b(n-1)}$$

Ladder filter: Begin with $n = 1$ and compute the order updates for $m = 0, 1, \dots, M - 1$

$$\delta_m(n-1) = w\delta_m(n-2) + \alpha_m(n-1)g_m^*(n-1)e_m(n-1)$$

$$\xi_m(n-1) = -\frac{\delta_m(n-1)}{E_m^b(n-1)}$$

$$e_{m+1}(n) = e_m(n) + \xi_m(n-1)g_m(n)$$

Initialization

$$\alpha_0(n-1) = 1, \quad e_0(n) = d(n), \quad f_0(n) = g_0(n) = x(n)$$

$$E_0^f(n) = E_0^b(n) = wE_0^f(n-1) + |x(n)|^2$$

$$\alpha_m(-1) = 1, \quad k_m(-1) = 0$$

$$E_m^b(-1) = E_m^f(0) = \epsilon > 0; \quad \delta_m(-1) = 0$$

Modified RLS Lattice Algorithms. The recursive equations in the RLS lattice algorithm given in Table 13.5 are by no means unique. Modifications can be made to some of the equations without affecting the optimality of the algorithm. However, some modifications result in algorithms that are more numerically robust when fixed-point arithmetic is used in the implementation of the algorithms. We give a number of basic relationships that are easily established from the above developments.

First, we have a relationship between the a priori and a posteriori error residuals.

A priori errors:

$$f_m(n, n-1) \equiv f_m(n) = x(n) + \mathbf{a}_m^t(n-1)\mathbf{X}_m(n-1) \quad (13.4.80)$$

$$g_m(n, n-1) \equiv g_m(n) = x(n-m) + \mathbf{b}_m^t(n-1)\mathbf{X}_m(n)$$

A posteriori errors:

$$\begin{aligned} f_m(n, n) &= x(n) + \mathbf{a}_m^t(n) \mathbf{X}_m(n-1) \\ g_m(n, n) &= x(n-m) + \mathbf{b}_m^t(n) \mathbf{X}_m(n) \end{aligned} \quad (13.4.81)$$

The basic relations between (13.4.80) and (13.4.81) are

$$\begin{aligned} f_m(n, n) &= \alpha_m(n-1) f_m(n) \\ g_m(n, n) &= \alpha_m(n) g_m(n) \end{aligned} \quad (13.4.82)$$

These relations follow easily by using (13.4.50) and (13.4.51) in (13.4.81).

Second, we may obtain time-update equations for the least-squares forward and backward errors. For example, from (13.4.8) and (13.4.50) we obtain

$$\begin{aligned} E_m^f(n) &= q(n) + \mathbf{a}_m^t(n) \mathbf{Q}_m^*(n) \\ &= q(n) + [\mathbf{a}_m^t(n-1) - \mathbf{K}_m^t(n-1) f_m(n)] [w \mathbf{Q}_m^*(n-1) + x^*(n) \mathbf{X}_m(n-1)] \\ &= w E_m^f(n-1) + \alpha_m(n-1) |f_m(n)|^2 \end{aligned} \quad (13.4.83)$$

Similarly, from (13.4.17) and (13.4.51) we obtain

$$E_m^b(n) = w E_m^b(n-1) + \alpha_m(n) |g_m(n)|^2 \quad (13.4.84)$$

Usually, (13.4.83) and (13.4.84) are used in place of the sixth and seventh equations in Table 13.5.

Third, we obtain a time-update equation for the Kalman gain vector, which is not explicitly used in the lattice algorithm, but which is used in the fast FIR filter algorithms. For this derivation, we also use the time-update equations for the forward and backward prediction coefficients given by (13.4.50) and (13.4.51). Thus, we have

$$\begin{aligned} \mathbf{K}_m(n) &= \mathbf{P}_m(n) \mathbf{X}_m^*(n) \\ &= \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{P}_{m-1}(n-1) \end{bmatrix} \begin{bmatrix} x^*(n) \\ \mathbf{X}_{m-1}^*(n-1) \end{bmatrix} \\ &\quad + \frac{1}{E_{m-1}^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_{m-1}(n) \end{bmatrix} [1 \quad \mathbf{a}_{m-1}^H(n)] \begin{bmatrix} x^*(n) \\ \mathbf{X}_{m-1}^*(n-1) \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ \mathbf{K}_{m-1}(n-1) \end{bmatrix} + \frac{f_{m-1}^*(n, n)}{E_{m-1}^f(n)} \begin{bmatrix} 1 \\ \mathbf{a}_{m-1}(n) \end{bmatrix} \\ &\equiv \begin{bmatrix} \mathbf{C}_{m-1}(n) \\ c_{mm}(n) \end{bmatrix} \end{aligned} \quad (13.4.85)$$

where, by definition, $\mathbf{C}_{m-1}(n)$ consists of the first $(m - 1)$ elements of $\mathbf{K}_m(n)$ and $c_{mm}(n)$ is the last element. From (13.4.60), we also have the order-update equation for $\mathbf{K}_m(n)$ as

$$\mathbf{K}_m(n) = \begin{bmatrix} \mathbf{K}_{m-1}(n) \\ 0 \end{bmatrix} + \frac{g_{m-1}^*(n, n)}{E_{m-1}^b(n)} \begin{bmatrix} \mathbf{b}_{m-1}(n) \\ 1 \end{bmatrix} \quad (13.4.86)$$

By equating (13.4.85) to (13.4.86), we obtain the result

$$c_{mm}(n) = \frac{g_{m-1}^*(n, n)}{E_{m-1}^b(n)} \quad (13.4.87)$$

and, hence,

$$\mathbf{K}_{m-1}(n) + c_{mm}(n)\mathbf{b}_{m-1}(n) = \mathbf{C}_{m-1}(n) \quad (13.4.88)$$

By substituting for $\mathbf{b}_{m-1}(n)$ from (13.4.51) into (13.4.88), we obtain the time-update equation for the Kalman gain vector in (13.4.85) as

$$\mathbf{K}_{m-1}(n) = \frac{\mathbf{C}_{m-1}(n) - c_{mm}(n)\mathbf{b}_{m-1}(n-1)}{1 - c_{mm}(n)g_{m-1}(n)} \quad (13.4.89)$$

There is also a time-update equation for the scalar $\alpha_m(n)$. From (13.4.63), we have

$$\begin{aligned} \alpha_m(n) &= \alpha_{m-1}(n) - \frac{\alpha_{m-1}^2(n)|g_{m-1}(n)|^2}{E_{m-1}^b(n)} \\ &= \alpha_{m-1}(n)[1 - c_{mm}(n)g_{m-1}(n)] \end{aligned} \quad (13.4.90)$$

A second relation is obtained by using (13.4.85) to eliminate $\mathbf{K}_{m-1}(n)$ in the expression for $\alpha_m(n)$. Then,

$$\begin{aligned} \alpha_m(n) &= 1 - \mathbf{X}_m^t(n)\mathbf{K}_m(n) \\ &= \alpha_{m-1}(n-1) \left[1 - \frac{f_{m-1}^*(n, n)f_{m-1}(n)}{E_{m-1}^f(n)} \right] \end{aligned} \quad (13.4.91)$$

By equating (13.4.90) to (13.4.91), we obtain the desired time-update equation for $\alpha_m(n)$ as

$$\alpha_{m-1}(n) = \alpha_{m-1}(n-1) \left[\frac{1 - \frac{f_{m-1}^*(n, n)f_{m-1}(n)}{E_{m-1}^f(n)}}{1 - c_{mm}(n)g_{m-1}(n)} \right] \quad (13.4.92)$$

Finally, we wish to distinguish between two different methods for updating the reflection coefficients in the lattice filter and the ladder part: the *conventional (indirect) method* and the *direct method*. In the conventional (indirect) method,

$$\mathcal{K}_{m+1}^f(n) = -\frac{k_{m+1}(n)}{E_m^b(n-1)} \quad (13.4.93)$$

$$\mathcal{K}_{m+1}^b(n) = -\frac{k_{m+1}^*(n)}{E_m^f(n)} \quad (13.4.94)$$

$$\xi_m(n) = -\frac{\delta_m(n)}{E_m^b(n)} \quad (13.4.95)$$

where $k_{m+1}(n)$ is time-updated from (13.4.58), $\delta_m(n)$ is updated according to (13.4.76), and $E_m^f(n)$ and $E_m^b(n)$ are updated according to (13.4.83) and (13.4.84). By substituting for $k_{m+1}(n)$ from (13.4.58) into (13.4.93), and using (13.4.84) and the eighth equation in Table 13.5, we obtain

$$\begin{aligned} \mathcal{K}_{m+1}^f(n) &= -\frac{k_{m+1}(n-1)}{E_m^b(n-2)} \left(\frac{wE_m^b(n-2)}{E_m^b(n-1)} \right) - \frac{\alpha_m(n-1)f_m(n)g_m^*(n-1)}{E_m^b(n-1)} \\ &= \mathcal{K}_{m+1}^f(n-1) \left(1 - \frac{\alpha_m(n-1)|g_m(n-1)|^2}{E_m^b(n-1)} \right) \\ &\quad - \frac{\alpha_m(n-1)f_m(n)g_m^*(n-1)}{E_m^b(n-1)} \\ &= \mathcal{K}_{m+1}^f(n-1) - \frac{\alpha_m(n-1)f_{m+1}(n)g_m^*(n-1)}{E_m^b(n-1)} \end{aligned} \quad (13.4.96)$$

which is a formula for directly updating the reflection coefficients in the lattice. Similarly, by substituting (13.4.58) into (13.4.94), and using (13.4.83) and the eighth equation in Table 13.5, we obtain

$$\mathcal{K}_{m+1}^b(n) = \mathcal{K}_{m+1}^b(n-1) - \frac{\alpha_m(n-1)f_m^*(n)g_{m+1}(n)}{E_m^f(n)} \quad (13.4.97)$$

Finally, the ladder gain can also be updated directly according to the relation

$$\xi_m(n) = \xi_m(n-1) - \frac{\alpha_m(n)g_m^*(n)e_{m+1}(n)}{E_m^b(n)} \quad (13.4.98)$$

The RLS lattice-ladder algorithm that uses the direct update relations in (13.4.96–13.4.98) and (13.4.83–13.4.84) is listed in Table 13.6.

An important characteristic of the algorithm in Table 13.6 is that the forward and backward residuals are fed back to time-update the reflection coefficients in the lattice stage, and $e_{m+1}(n)$ is fed back to update the ladder gain $\xi_m(n)$. For this reason, this RLS lattice-ladder algorithm has been called the *error-feedback form*. A similar form can be obtained for the a posteriori RLS lattice-ladder algorithm. For more details on the error-feedback form of RLS lattice-ladder algorithms, the interested reader is referred to Ling, Manolakis, and Proakis (1986).

TABLE 13.6 Direct Update (Error-Feedback) Form of the A Priori RLS Lattice-Ladder Algorithm

Lattice predictor: Begin with $n = 1$ and compute the order updates for $m = 0, 1, \dots, M - 2$

$$\mathcal{H}_{m+1}^f(n-1) = \mathcal{H}_{m+1}^f(n-2) - \frac{\alpha_m(n-2)f_{m+1}(n-1)g_m^*(n-2)}{E_m^b(n-2)}$$

$$\mathcal{H}_{m+1}^b(n-1) = \mathcal{H}_{m+1}^b(n-2) - \frac{\alpha_m(n-2)f_m^*(n-1)g_{m+1}(n-1)}{E_m^f(n-1)}$$

$$f_{m+1}(n) = f_m(n) + \mathcal{H}_{m+1}^f(n-1)g_m(n-1)$$

$$g_{m+1}(n) = g_m(n-1) + \mathcal{H}_{m+1}^b(n-1)f_m(n)$$

$$E_{m+1}^f(n-1) = wE_{m+1}^f(n-2) + \alpha_{m+1}(n-2)|f_{m+1}(n-1)|^2$$

$$\alpha_{m+1}(n-1) = \alpha_m(n-1) - \frac{\alpha_m^2(n-1)|g_m(n-1)|^2}{E_m^b(n-1)}$$

$$E_{m+1}^b(n-1) = wE_{m+1}^b(n-2) + \alpha_{m+1}(n-1)|g_{m+1}(n-1)|^2$$

Ladder filter: Begin with $n = 1$ and compute the order updates $m = 0, 1, \dots, M - 1$

$$\xi_m(n-1) = \xi_m(n-2) - \frac{\alpha_m(n-1)g_m^*(n-1)e_{m+1}(n-1)}{E_m^b(n-1)}$$

$$e_{m+1}(n) = e_m(n) + \xi_m(n-1)g_m(n)$$

Initialization

$$\alpha_0(n-1) = 1, \quad e_0(n) = d(n), \quad f_0(n) = g_0(n) = x(n)$$

$$E_0^f(n) = E_0^b(n) = wE_0^f(n-1) + |x(n)|^2$$

$$\alpha_m(-1) = 1, \quad \mathcal{H}_m^f(-1) = \mathcal{H}_m^b(-1) = 0$$

$$E_m^b(-1) = E_m^f(0) = \epsilon > 0$$

Fast RLS Algorithms. The two versions of the fast RLS algorithms given in Section 13.3.3 follow directly from the relationships that we have obtained in this section. In particular, we fix the size of the lattice and the associated forward and backward predictors at $M - 1$ stages. Thus, we obtain the first seven recursive equations in the two versions of the algorithm. The remaining problem is to determine the time-update equation for the Kalman gain vector, which was determined in (13.4.85–13.4.89). In version B of the algorithm, given in Table 13.3, we used the scalar $\alpha_m(n)$ to reduce the computations from $10M$ to $9M$. Version A of the algorithm, given in Table 13.2, avoids the use of this parameter. Since these algorithms provide a direct updating of the Kalman gain vector, they have been called *fast Kalman algorithms* (for reference, see Falconer and Ljung (1978) and Proakis (1989)).

Further reduction of computational complexity to $7M$ is possible by directly updating the following *alternative (Kalman) gain vector* (see Carayannis, Manolakis, and Kalouptsidis (1983)) defined as

$$\tilde{\mathbf{K}}_M(n) = \frac{1}{w}\mathbf{P}_M(n-1)\mathbf{X}_M^*(n) \quad (13.4.99)$$

Several fast algorithms using this gain vector have been proposed, with complexities ranging from $7M$ to $10M$. Table 13.7 lists the FAEST (Fast A Posteriori Error Sequential Technique) algorithm with a computational complexity $7M$ (for a derivation, see Carayannis, Manolakis, and Kalouptsidis (1983; 1986) and Problem 13.7).

In general, the $7M$ fast RLS algorithms and some variations are very sensitive to round-off noise and exhibit instability problems (Falconer and Ljung (1978), Carayannis, Manolakis, and Kalouptsidis (1983; 1986), and Cioffi and Kailath (1984)). The instability problem in the $7M$ algorithms has been addressed by Slock and Kailath (1988; 1991), and modifications have been proposed that stabilize these algorithms. The resulting stabilized algorithms have a computational complexity ranging from $8M$ to $9M$. Thus, their computational complexity is increased by a relatively small amount compared to the unstable $7M$ algorithms.

To understand the stabilized fast RLS algorithms, we begin by comparing the fast RLS algorithm given in Table 13.3 and the FAEST algorithm in Table 13.7. As

TABLE 13.7 FAEST Algorithm

$f_{M-1}(n) = x(n) + \mathbf{a}_{M-1}^t(n-1)\mathbf{X}_{M-1}(n-1)$
$\bar{f}_{M-1}(n, n) = \frac{f_{M-1}(n)}{\bar{\alpha}_{M-1}(n-1)}$
$\mathbf{a}_{M-1}(n) = \mathbf{a}_{M-1}(n-1) - \bar{\mathbf{K}}_{M-1}(n-1)\bar{f}_{M-1}(n, n)$
$E_{M-1}^f(n) = wE_{M-1}^f(n-1) + f_{M-1}(n)f_{M-1}^*(n, n)$
$\bar{\mathbf{K}}_M(n) \equiv \begin{bmatrix} \bar{\mathbf{C}}_{M-1}(n) \\ \bar{c}_{MM}(n) \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{\mathbf{K}}_{M-1}(n-1) \end{bmatrix} + \frac{f_{M-1}^*(n)}{wE_{M-1}^f(n-1)} \begin{bmatrix} 1 \\ \mathbf{a}_{M-1}(n-1) \end{bmatrix}$
$g_{M-1}(n) = -wE_{M-1}^b(n-1)\bar{c}_{MM}^*(n)$
$\bar{\mathbf{K}}_{M-1}(n) = \bar{\mathbf{C}}_{M-1}(n) - \mathbf{b}_{M-1}(n-1)\bar{c}_{MM}(n)$
$\bar{\alpha}_M(n) = \bar{\alpha}_{M-1}(n-1) + \frac{ f_{M-1}(n) ^2}{wE_{M-1}^f(n-1)}$
$\bar{\alpha}_{M-1}(n) = \bar{\alpha}_M(n) + g_{M-1}(n)\bar{c}_{MM}(n)$
$\bar{g}_{M-1}(n, n) = \frac{g_{M-1}(n)}{\bar{\alpha}_{M-1}(n)}$
$E_{M-1}^b(n) = wE_{M-1}^b(n-1) + g_{M-1}(n)\bar{g}_{M-1}^*(n, n)$
$\mathbf{b}_{M-1}(n) = \mathbf{b}_{M-1}(n-1) + \bar{\mathbf{K}}_{M-1}(n)\bar{g}_{M-1}(n, n)$
$e_M(n) = d(n) - \mathbf{h}_M^t(n-1)\mathbf{X}_M(n)$
$\bar{e}_M(n, n) = \frac{e_M(n)}{\bar{\alpha}_M(n)}$
$\mathbf{h}_M(n) = \mathbf{h}_M(n-1) + \bar{\mathbf{K}}_M(n)\bar{e}_M(n, n)$
<i>Initialization:</i> Set all vectors to zero
$E_{M-1}^f(-1) = E_{M-1}^b(-1) = \epsilon > 0$
$\bar{\alpha}_{M-1}(-1) = 1$

indicated, there are two major differences between these two algorithms. First, the FAEST algorithm uses the alternative (Kalman) gain vector instead of the Kalman gain vector. Second, the fast RLS algorithm computes the a priori backward prediction error $g_{M-1}(n)$ through FIR filtering using the backward prediction coefficient vector $\mathbf{b}_{m-1}(n-1)$, whereas the FAEST algorithm computes the same quantity through a scalar operation by noticing that the last element of the alternative gain vector, $\tilde{c}_{MM}(n)$, is equal to $-wE_{M-1}^b g_{M-1}(n)$. Since these two algorithms are algebraically equivalent, the backward prediction errors calculated in different ways should be identical if infinite precision is used in the computation. Practically, when finite-precision arithmetic is used, the backward prediction errors computed using different formulas are only approximately equal. In what follows, we denote them by $g_{M-1}^{(f)}$ and $g_{M-1}^{(s)}(n)$, respectively. The superscripts (f) and (s) indicate that they are computed using the filtering approach and scalar operation, respectively.

There are other quantities in the algorithms that can also be computed in different ways. In particular, the parameter $\alpha_{M-1}(n)$ can be computed from the vector quantities $\tilde{\mathbf{K}}_{M-1}(n)$ and $\mathbf{X}_{M-1}(n)$ as

$$\alpha_{M-1}(n) = 1 + \tilde{\mathbf{K}}_{M-1}^t(n) \mathbf{X}_{M-1}(n) \quad (13.4.100)$$

or from scalar quantities. We denote these values as $\tilde{\alpha}_{M-1}^{(f)}(n)$ and $\tilde{\alpha}_{M-1}^{(s)}(n)$, respectively. Finally, the last element of $\tilde{\mathbf{K}}_M(n)$, denoted as $\tilde{c}_{MM}^{(f)}(n)$, may be computed from the relation

$$\tilde{c}_{MM}^{(f)}(n) = \frac{-g_{M-1}^{(f)}(n)}{wE_{M-1}^b(n-1)} \quad (13.4.101)$$

The two quantities in each of the three pairs $[g_{M-1}^{(f)}(n), g_{M-1}^{(s)}(n)]$, $[\alpha_{M-1}^{(f)}(n), \alpha_{M-1}^{(s)}(n)]$, and $[\tilde{c}_{MM}^{(f)}(n), \tilde{c}_{MM}^{(s)}(n)]$ are algebraically equivalent. Hence, either of the two quantities or their linear combination (of the form $k\beta^{(s)} + (1-k)\beta^{(f)}$, where β represents any of the three parameters) are algebraically equivalent to the original quantities, and may be used in the algorithm. Slock and Kailath (1988; 1991) found that by using the appropriate quantity or its linear combination in the fast RLS algorithm, it was sufficient to correct for the positive feedback inherent in the fast RLS algorithms. Implementation of this basic notion leads to the stabilized fast RLS algorithm given in Table 13.8.

We observe from Table 13.8 that the stabilized fast RLS algorithm employs constants k_i , $i = 1, 2, \dots, 5$, to form five linear combinations of the three pairs of quantities just described. The best values of the k_i found by Slock and Kailath resulted from computer search, and are given as $k_1 = 1.5$, $k_2 = 2.5$, $k_3 = 1$, $k_4 = 0$, $k_5 = 1$. When $k_i = 0$ or 1, we use only one of the quantities in the linear combination. Hence, some of the parameters in the three pairs need not be computed. It was also found that the stability of the algorithm is only slightly affected if $\alpha_{M-1}^{(f)}(n)$ is not used. These simplifications result in the algorithm given in Table 13.9, which has a complexity of $8M$ and is numerically stable.

The performance of the stabilized fast RLS algorithms depends highly on proper initialization. On the other hand, an algorithm that uses $g_{M-1}^{(f)}(n)$ in its computations

TABLE 13.8 The Stabilized Fast RLS Algorithm

$f_{M-1}(n) = x(n) + \mathbf{a}_{M-1}^t(n-1)\mathbf{X}_{M-1}(n-1)$
$f_{M-1}(n, n) = \frac{f_{M-1}(n)}{\bar{\alpha}_{M-1}(n-1)}$
$\mathbf{a}_{M-1}(n) = \mathbf{a}_{M-1}(n-1) - \tilde{\mathbf{K}}_{M-1}(n-1)f_{M-1}(n, n)$
$\bar{c}_{M1}(n) = \frac{f_{M-1}^{(f)*}(n)}{wE_{M-1}^f(n-1)}$
$\begin{bmatrix} \tilde{\mathbf{C}}_{M-1}(n) \\ \bar{c}_{MM}^{(s)}(n) \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{\mathbf{K}}_{M-1}(n-1) \end{bmatrix} + \bar{c}_{M1}(n) \begin{bmatrix} 1 \\ \mathbf{a}_{M-1}(n-1) \end{bmatrix}$
$g_{M-1}^{(f)}(n) = x(n-M+1) + \mathbf{b}_{M-1}^t(n-1)\mathbf{X}_{M-1}(n)$
$\bar{c}_{MM}^{(f)}(n) = -\frac{g_{M-1}^{(f)*}(n)}{wE_{M-1}^b(n-1)}$
$\bar{c}_{MM}(n) = k_4\bar{c}_{MM}^{(f)}(n) + (1-k_4)\bar{c}_{MM}^{(s)}(n)$
$\tilde{\mathbf{K}}_M(n) = \begin{bmatrix} \tilde{\mathbf{C}}_{M-1}(n) \\ \bar{c}_{MM}(n) \end{bmatrix}$
$g_{M-1}^{(s)}(n) = -wE_{M-1}^b(n-1)\bar{c}_{MM}^{(s)*}(n)$
$g_{M-1}^{(i)}(n) = k_i g_{M-1}^{(f)}(n) + (1-k_i)g_{M-1}^{(s)}(n), \quad i = 1, 2, 5$
$\tilde{\mathbf{K}}_{M-1}(n) = \tilde{\mathbf{C}}_{M-1}(n) - \mathbf{b}_{M-1}(n-1)\bar{c}_{MM}(n)$
$\bar{\alpha}_M(n) = \bar{\alpha}_{M-1}(n-1) + \bar{c}_{M1}(n)f_{M-1}(n)$
$\bar{\alpha}_{M-1}^{(s)}(n) = \bar{\alpha}_M(n) + g_{M-1}^{(s)}(n)\bar{c}_{MM}^{(s)}(n)$
$\bar{\alpha}_{M-1}^{(f)}(n) = 1 + \tilde{\mathbf{K}}_{M-1}^t(n)\mathbf{X}_{M-1}(n)$
$\bar{\alpha}_{M-1}(n) = k_3\bar{\alpha}_{M-1}^{(f)}(n) + (1-k_3)\bar{\alpha}_{M-1}^{(s)}(n)$
$E_{M-1}^f(n) = wE_{M-1}^f(n-1) + f_{M-1}(n)f_{M-1}^{*(n, n)}$
$\text{or, } \frac{1}{E_{M-1}^f(n)} = \frac{1}{wE_{M-1}^f(n-1)} - \frac{ \bar{c}_{M1}(n) ^2}{\bar{\alpha}_{M-1}^{(s)}(n)}$
$g_{M-1}^{(i)}(n, n) = \frac{g_{M-1}^{(i)}(n)}{\bar{\alpha}_{M-1}(n)}, \quad i = 1, 2$
$\mathbf{b}_{M-1}(n) = \mathbf{b}_{M-1}(n-1) + \tilde{\mathbf{K}}_{M-1}(n)g_{M-1}^{(1)}(n, n)$
$E_{M-1}^b(n) = wE_{M-1}^b(n-1) + g_{M-1}^{(2)}(n)g_{M-1}^{(2)*}(n, n)$
$e_M(n) = d(n) - \mathbf{h}_M^t(n-1)\mathbf{X}_M(n)$
$e_M(n, n) = \frac{e_M(n)}{\bar{\alpha}_M(n)}$
$\mathbf{h}_M(n) = \mathbf{h}_M(n-1) + \tilde{\mathbf{K}}_M(n)e_M(n, n)$

TABLE 13.9 A Simplified Stabilized Fast RLS Algorithm

$f_{M-1}(n) = x(n) + \mathbf{a}_{M-1}'(n-1)\mathbf{X}_{M-1}(n-1)$
$f_{M-1}(n, n) = \frac{f_{M-1}(n)}{\bar{\alpha}_{M-1}(n-1)}$
$\mathbf{a}_{M-1}(n) = \mathbf{a}_{M-1}(n-1) - \tilde{\mathbf{K}}_{M-1}(n-1)f_{M-1}(n, n)$
$\bar{c}_{M1}(n) = \frac{f_{M-1}^*(n)}{wE_{M-1}^f(n-1)}$
$\tilde{\mathbf{K}}_M(n) \equiv \begin{bmatrix} \tilde{\mathbf{C}}_{M-1}(n) \\ \bar{c}_{MM}(n) \end{bmatrix} = \begin{bmatrix} 0 \\ \tilde{\mathbf{K}}_{M-1}(n-1) \end{bmatrix} + \frac{f_{M-1}^*(n)}{wE_{M-1}^f(n-1)} \begin{bmatrix} 1 \\ \mathbf{a}_{M-1}(n-1) \end{bmatrix}$
$g_{M-1}^{(f)}(n) = x(n-M+1) + \mathbf{b}_{M-1}'(n-1)\mathbf{X}_{M-1}(n)$
$g_{M-1}^{(s)}(n) = -wE_{M-1}^b(n-1)\bar{c}_{MM}^*(n)$
$g_{M-1}^{(i)}(n) = k_i g_{M-1}^{(f)}(n) + (1-k_i)g_{M-1}^{(s)}(n), \quad i = 1, 2$
$\tilde{\mathbf{K}}_{M-1}(n) = \tilde{\mathbf{C}}_{M-1}(n) - \mathbf{b}_{M-1}(n-1)\bar{c}_{MM}(n)$
$\bar{\alpha}_M(n) = \bar{\alpha}_{M-1}(n-1) + \bar{c}_{M1}(n)f_{M-1}(n)$
$\tilde{\alpha}_{M-1}(n) = \bar{\alpha}_M(n) + g_{M-1}^{(f)}(n)\bar{c}_{MM}(n)$
$E_{M-1}^f(n) = wE_{M-1}^f(n-1) + f_{M-1}(n)f_{M-1}^*(n, n)$
$g_{M-1}^{(i)}(n, n) = \frac{g_{M-1}^{(i)}(n)}{\tilde{\alpha}_{M-1}(n)}, \quad i = 1, 2$
$\mathbf{b}_{M-1}(n) = \mathbf{b}_{M-1}(n-1) + \tilde{\mathbf{K}}_{M-1}(n)g_{M-1}^{(1)}(n, n)$
$E_{M-1}^b(n) = wE_{M-1}^b(n-1) + g_{M-1}^{(2)}(n)g_{M-1}^{(2)*}(n, n)$
$e_M(n) = d(n) - \mathbf{h}_M'(n-1)\mathbf{X}_M(n)$
$e_M(n, n) = \frac{e_M(n)}{\tilde{\alpha}_M(n)}$
$\mathbf{h}_M(n) = \mathbf{h}_M(n-1) + \tilde{\mathbf{K}}_M(n)e_M(n, n)$

is not critically affected by proper initialization (although, eventually, it will diverge). Consequently, we may initially use $g_{M-1}^{(f)}(n)$ in place of $g_{M-1}^{(s)}(n)$ (or their linear combination) for the first few hundred iterations, and then switch to the form for the stabilized fast RLS algorithm. By doing so, we obtain a stabilized fast RLS algorithm that is also insensitive to initial conditions.

13.4.2 Other Lattice Algorithms

Another type of RLS lattice algorithm is obtained by normalizing the forward and backward prediction errors through division of the errors by $\sqrt{E_m^f(n)}$ and $\sqrt{E_m^b(n)}$, respectively, and multiplication by $\sqrt{\alpha_m(n-1)}$ and $\sqrt{\alpha_m(n)}$, respectively. The resulting lattice algorithm is called a square-root or angle-and-power normalized RLS

lattice algorithm. This algorithm has a more compact form than the other forms of RLS lattice algorithms. However, the algorithm requires many square-root operations, which can be computationally complex. This problem can be solved by using CORDIC processors, which compute a square root in N clock cycles, where N is the number of bits of the computer word length. A description of the square-root/normalized RLS lattice algorithm and the CORDIC algorithm is given in the book by Proakis et al. (2002).

It is also possible to simplify the computational complexity of the RLS algorithms described in the previous section at the expense of compromising the convergence rate. One such algorithm is called *the gradient-lattice algorithm*. In this algorithm each stage of the lattice filter is characterized by the output–input relations

$$\begin{aligned} f_m(n) &= f_{m-1}(n) - k_m(n)g_{m-1}(n-1) \\ g_m(n) &= g_{m-1}(n-1) - k_m^*(n)f_{m-1}(n) \end{aligned} \quad (13.4.102)$$

where $k_m(n)$ is the reflection coefficient in the m th stage of the lattice and $f_m(n)$ and $g_m(n)$ are the forward and backward residuals.

This form of the lattice filter is identical to the Levinson-Durbin algorithm, except that now $k_m(n)$ is allowed to vary with time so that the lattice filter adapts to the time variations in the signal statistics. The reflection coefficients $\{k_m(n)\}$ may be optimized by employing the method of least squares, which results in the solution

$$k_m(n) = \frac{2 \sum_{l=0}^n w^{n-l} f_{m-1}(l) g_{m-1}^*(l-1)}{\sum_{l=0}^n w^{n-l} [|f_{m-1}(l)|^2 + |g_{m-1}(l-1)|^2]}, \quad m = 1, 2, \dots, M-1 \quad (13.4.103)$$

These coefficients can also be updated recursively in time. The ladder coefficients are computed recursively in time by employing an LMS-type algorithm that is obtained by applying the mean-square-error criterion. A description of this algorithm is given in the paper by Griffiths (1978) and in the book by Proakis et al. (2002).

13.4.3 Properties of Lattice-Ladder Algorithms

The lattice algorithms that we have derived in the two previous subsections have a number of desirable properties. In this subsection, we consider the properties of these algorithms and compare them with the corresponding properties of the LMS algorithm and the RLS direct-form FIR filtering algorithms.

Convergence Rate. The RLS lattice-ladder algorithms basically have the same convergence rate as the RLS direct-form FIR filter structures. This characteristic behavior is not surprising, since both filter structures are optimum in the least-squares sense. Although the gradient lattice algorithm retains some of the optimal characteristics of the RLS lattice, nevertheless the former is not optimum in the least-squares sense and, hence, its convergence rate is slower.

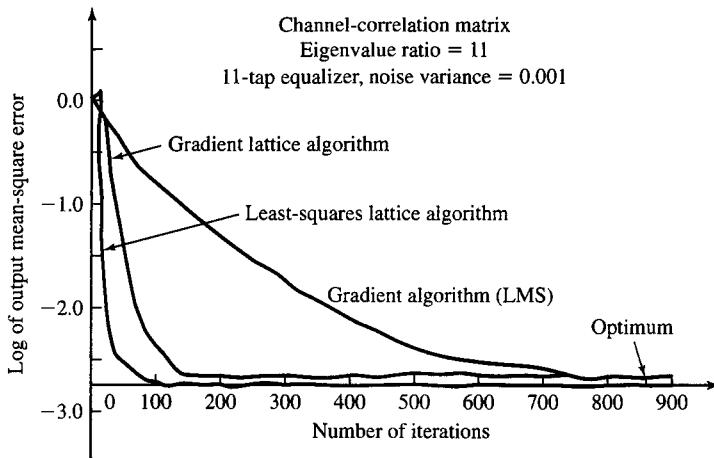


Figure 13.4.3 Learning curves for RLS lattice, gradient lattice, and LMS algorithms for adaptive equalizer of length $M = 11$. (From *Digital Communications* by John G. Proakis. ©1989 by McGraw-Hill Book Company. Reprinted with permission of the publisher.)

For comparison purposes, Figures 13.4.3 and 13.4.4 illustrate the learning curves for an adaptive equalizer of length $M = 11$, implemented as an RLS lattice-ladder filter, a gradient lattice-ladder filter, and a direct-form FIR filter using the LMS algorithm, for a channel autocorrelation matrix that has eigenvalue ratios of $\lambda_{\max}/\lambda_{\min} = 11$ and $\lambda_{\max}/\lambda_{\min} = 21$, respectively. From these learning curves, we observe that the gradient lattice algorithm takes about twice as many iterations to converge as the optimum RLS lattice algorithm. Furthermore, the gradient lattice algorithm provides significantly faster convergence than the LMS algorithm. For both lattice structures, the convergence rate does not depend on the eigenvalue spread of the correlation matrix.

Computational Requirements. The RLS lattice algorithms described in the previous subsection have a computational complexity that is proportional to M . In contrast, the computational complexity of the RLS square-root algorithms is proportional to M^2 . On the other hand, the direct-form fast algorithms, which are a derivative of the lattice algorithm, have a complexity proportional to M , and they are a little more efficient than the lattice-ladder algorithms.

In Figure 13.4.5 we illustrate the computational complexity (number of complex multiplications and divisions) of the various adaptive filtering algorithms that we have described. Clearly, the LMS algorithm requires the fewest computations. The fast RLS algorithms in Tables 13.3 and 13.9 are the most efficient of the RLS algorithms shown, closely followed by the gradient lattice algorithm, then the RLS lattice algorithms and, finally, the square-root algorithms. Note that for small values of M , there is little difference in complexity among the rapidly convergent algorithms.

Numerical Properties. In addition to providing fast convergence, the RLS and gradient lattice algorithms are numerically robust. First, these lattice algorithms are

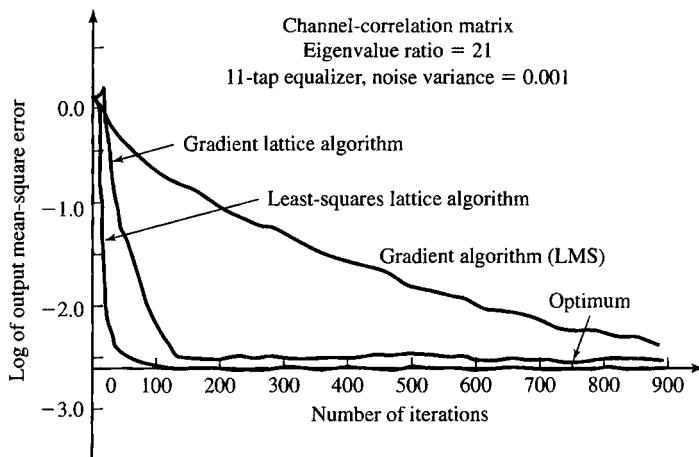


Figure 13.4.4 Learning curves for RLS lattice, gradient lattice, and LMS algorithms for adaptive equalizer of length $M = 11$. (From *Digital Communications* by John G. Proakis. ©1989 by McGraw-Hill Book Company. Reprinted with permission of the publisher.)

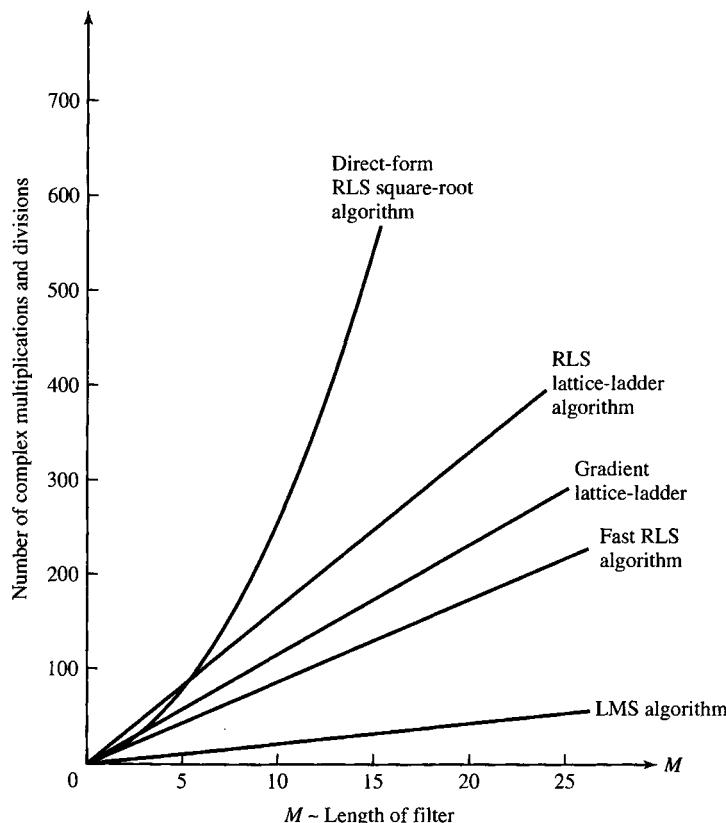


Figure 13.4.5 Computational complexity of adaptive filter algorithms.

TABLE 13.10 Numerical Accuracy, in Terms of Output MSE for Channel with $\lambda_{\max}/\lambda_{\min} = 11$ and $w = 0.975$, $\text{MSE} \times 10^{-3}$

Number of bits (including sign)	Algorithm				
	RLS square root	Fast RLS	Conventional RLS lattice	Error feedback RLS lattice	LMS
16	2.17	2.17	2.16	2.16	2.30
13	2.33	2.21	3.09	2.22	2.30
11	6.14	3.34	25.2	3.09	19.0
9	75.3	^a	365	31.6	311

^a Algorithm did not converge.

numerically stable, which means that the output estimation error from the computational procedure is bounded when a bounded error signal is introduced at the input. Second, the numerical accuracy of the optimum solution is also relatively good when compared to the LMS and the RLS direct-form FIR algorithms.

For purposes of comparison, we illustrate in Table 13.10 the steady-state average squared error or (estimated) minimum MSE obtained through computer simulation from the two RLS lattice algorithms and the direct-form FIR filter algorithms described in Section 13.2. The striking result in Table 13.10 is the superior performance obtained with the RLS lattice-ladder algorithm, in which the reflection coefficients and the ladder gain are updated directly according to (13.4.96–13.4.98). This is the error-feedback form of the RLS lattice algorithm. It is clear that the direct updating of these coefficients is significantly more robust to round-off errors than all the other adaptive algorithms, including the LMS algorithm. It is also apparent that the two-step process used in the conventional RLS lattice algorithm to estimate the reflection coefficients is not as accurate. Furthermore, the estimation errors that are generated in the coefficients at each stage propagate from stage to stage, causing additional errors.

The effect of changing the weighting factor w is illustrated in the numerical results given in Table 13.11. In this table, we give the minimum (estimated) MSE

TABLE 13.11 Numerical Accuracy, in Terms of Output MSE, of A Prior Least-Squares Lattice Algorithm with Different Values of the Weighting Factor w , $\text{MSE} \times 10^{-3}$

Number of bits with sign	Algorithm						
	$w = 0.99$		$w = 0.975$		$w = 0.95$		
	Conventional	Error feedback	Conventional	Error feedback	Conventional	Error feedback	
16	2.14	2.08	2.18	2.16	2.66	2.62	
13	7.08	2.11	3.09	2.22	3.65	2.66	
11	39.8	3.88	25.2	3.09	15.7	2.78	
9	750	44.1	365	31.6	120	15.2	

obtained with the conventional and error-feedback forms of the RLS lattice algorithm. We observe that the output MSE decreases with an increase in the weighting factor when the precision is high (13 bits and 16 bits). This reflects the improvement in performance obtained by increasing the observation interval. As the number of bits of precision is decreased, we observe that the weighting factor should also be decreased in order to maintain good performance. In effect, with low precision, the effect of a longer averaging time results in a larger round-off noise. Of course, these results were obtained with time-invariant signal statistics. If the signal statistics are time-variant, the rate of the time variations will also influence the choice of w .

In the gradient lattice algorithm, the reflection coefficients and the ladder gains are also updated directly. Consequently, the numerical accuracy of the gradient lattice algorithm is comparable to that obtained with the direct update form of the RLS lattice.

Analytical and simulation results on numerical stability and numerical accuracy in fixed-point implementation of these algorithms can be found in Ling and Proakis (1984), Ling, Manolakis, and Proakis (1985; 1986), Ljung and Ljung (1985), and Gardner (1984).

Implementation Considerations. As we have observed, the lattice filter structure is highly modular and allows for the computations to be pipelined. Because of the high degree of modularity, the RLS and gradient lattice algorithms are particularly suitable for implementation in VLSI. As a result of this advantage in implementation and the desirable properties of stability, excellent numerical accuracy, and fast convergence, we anticipate that adaptive filters will be increasingly implemented as lattice-ladder structures in the near future.

13.5 Summary and References

We have presented adaptive algorithms for direct-form FIR and lattice filter structures. The algorithms for the direct-form FIR filter consisted of the simple LMS algorithm due to Widrow and Hoff (1960) and the direct-form, time-recursive least-squares algorithms, including the conventional RLS form given by (13.3.23–13.3.27), the square-root RLS forms described by Bierman (1977), Carlson and Culmone (1979), and Hsu (1982), and the RLS fast Kalman algorithms, one form of which was described by Falconer and Ljung (1978), and other forms later derived by Carayannis, Manolakis, and Kalouptsidis (1983), Proakis (1989), and Cioffi and Kailath (1984).

Of these algorithms, the LMS algorithm is the simplest. It is used in many applications where its slow convergence is adequate. Of the direct-form RLS algorithms, the square-root algorithms have been used in applications where fast convergence is required. The algorithms have good numerical properties. The family of stabilized fast RLS algorithms is very attractive from the viewpoint of computational efficiency. Methods to avoid instability due to round-off errors have been proposed by Hsu (1982), Cioffi and Kailath (1984), Lin (1984), Eleftheriou and Falconer (1987), and Slock and Kailath (1988; 1991).

The adaptive lattice-ladder filter algorithm derived in this chapter is the optimum RLS lattice-ladder algorithms (in both conventional and error-feedback form). Only the a priori form of the lattice-ladder algorithm was derived, which is the form most

often used in applications. In addition, there is an a posteriori form of the RLS lattice-ladder algorithms (both conventional and error-feedback), as described by Ling, Manolakis, and Proakis (1986). The error-feedback form of the RLS lattice-ladder algorithm has excellent numerical properties, and is particularly suitable for implementation in fixed-point arithmetic and in VLSI.

In the direct-form and lattice RLS algorithms, we used exponential weighting into the past in order to reduce the effective memory in the adaptation process. As an alternative to exponential weighting, we may employ finite-length uniform weighting into the past. This approach leads to the class of finite-memory RLS direct-form and lattice structures described in Cioffi and Kalaith (1985) and Manolakis, Ling, and Proakis (1987).

In addition to the various algorithms that we have presented in this chapter, there has been considerable research into efficient implementation of these algorithms using systolic arrays and other parallel architectures. The reader is referred to Kung (1982), and Kung, Whitehouse, and Kailath (1985).

Problems

- 13.1** Use the least-squares criterion to determine the equations for the parameters of the FIR filter model in Figure 13.1.2, when the plant output is corrupted by additive noise, $w(n)$.
- 13.2** Determine the equations for the coefficients of an adaptive echo canceler based on the least-squares criterion. Use the configuration in Figure 13.1.8 and assume the presence of a near-end echo only.
- 13.3** If the sequences $w_1(n)$, $w_2(n)$, and $w_3(n)$ in the adaptive noise-canceling system shown in Figure 13.1.14 are mutually uncorrelated, determine the expected value of the estimated correlation sequences $r_{vv}(k)$ and $r_{yu}(k)$ contained in (13.1.26).
- 13.4** Prove the result in (13.4.34).
- 13.5** Derive the equation for the direct update of the ladder gain given by (13.4.98).
- 13.6** Derive the equation for the reflection coefficients in a gradient lattice-algorithm given in (13.4.103).
- 13.7** Derive the FAEST algorithm given in Table 13.7 by using the alternative Kalman gain vector

$$\tilde{\mathbf{K}}_M(n) = \frac{1}{w} \mathbf{P}_M(n-1) \mathbf{X}_M^*(n)$$

instead of the Kalman gain vector $\mathbf{K}_M(n)$.

- 13.8** The tap-leaky LMS algorithm proposed by Gitlin, Meadors, and Weinstein (1982) may be expressed as

$$\mathbf{h}_M(n+1) = w\mathbf{h}_M(n) + \Delta e(n) \mathbf{X}_M^*(n)$$

where $0 < w < 1$, Δ is the step size, and $\mathbf{X}_M(n)$ is the data vector at time n . Determine the condition for the convergence of the mean value of $\mathbf{h}_M(n)$.

- 13.9** The tap-leaky LMS algorithm in Problem 13.8 may be obtained by minimizing the cost function

$$\mathcal{E}(n) = |e(n)|^2 + c||\mathbf{h}_M(n)||^2$$

where c is a constant and $e(n)$ is the error between the desired filter output and the actual filter output. Show that the minimization of $\mathcal{E}(n)$ with respect to the filter coefficient vector $\mathbf{h}_M(n)$ leads to the following tap-leaky LMS algorithm.

$$\mathbf{h}_M(n+1) = (1 - \Delta c)\mathbf{h}_M(n) + \Delta e(n)\mathbf{X}_M^*(n)$$

- 13.10** For the normalized LMS algorithm given by (13.2.31), determine the range of values of step size Δ to ensure the stability of the algorithm in the mean-square-error sense.
13.11 By using the alternative Kalman gain vector given in Problem 13.8, modify the a priori fast least-squares algorithms given in Tables 13.2 and 13.3, and thus reduce the number of computations.
13.12 Consider the random process

$$x(n) = gv(n) + w(n), \quad n = 0, 1, \dots, M-1$$

where $v(n)$ is a known sequence, g is a random variable with $E[g] = 0$, and $E[g^2] = G$. The process $w(n)$ is a white noise sequence with

$$\gamma_{ww}(m) = \sigma_w^2 \delta(m)$$

Determine the coefficients of the linear estimator for g , that is,

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

that minimizes the mean-square error

$$\mathcal{E} = E[(g - \hat{g})^2]$$

- 13.13** Recall that an FIR filter can be realized in the frequency-sampling form with system function

$$\begin{aligned} H(z) &= \frac{1 - z^{-M}}{M} \sum_{k=0}^{M-1} \frac{H_k}{1 - e^{j2\pi k/M} z^{-1}} \\ &= H_1(z)H_2(z) \end{aligned}$$

where $H_1(z)$ is the comb filter and $H_2(z)$ is the parallel bank of resonators.

- (a) Suppose that this structure is implemented as an adaptive filter using the LMS algorithm to adjust the filter (DFT) parameters H_k . Give the time-update equation for these parameters. Sketch the adaptive filter structure.
(b) Suppose that this structure is used as an adaptive channel equalizer, in which the desired signal is

$$d(n) = \sum_{k=0}^{M-1} A_k \cos \omega_k n, \quad \omega_k = \frac{2\pi k}{M}$$

With this form for the desired signal, what advantages are there in the LMS adaptive algorithm for the DFT coefficients H_k over the direct-form structure with coefficients $h(n)$? (Hint: Refer to Proakis (1970).)

- 13.14** Consider the performance index

$$J = h^2 - 40h + 28$$

Suppose that we search for the minimum of J by using the steepest-descent algorithm

$$h(n+1) = h(n) - \frac{1}{2} \Delta g(n)$$

where $g(n)$ is the gradient.

- (a) Determine the range of values of Δ that provides an overdamped system for the adjustment process.

- (b) Plot the expression for J as a function of n for a value of Δ in this range.

- 13.15** Consider the noise-canceling adaptive filter shown in Figure 13.1.14. Assume that the additive noise processes are white and mutually uncorrelated, with equal variances σ_w^2 . Suppose that the linear system has a known system function

$$H(z) = \frac{1}{1 - \frac{1}{2}z^{-1}}$$

Determine the optimum weights of a three-tap noise canceler that minimizes the MSE.

- 13.16** Determine the coefficients a_1 and a_2 for the linear predictor shown in Figure P13.16, given that the autocorrelation $\gamma_{xx}(n)$ of the input signal is

$$\gamma_{xx}(m) = a^{|m|}, \quad 0 < a < 1$$

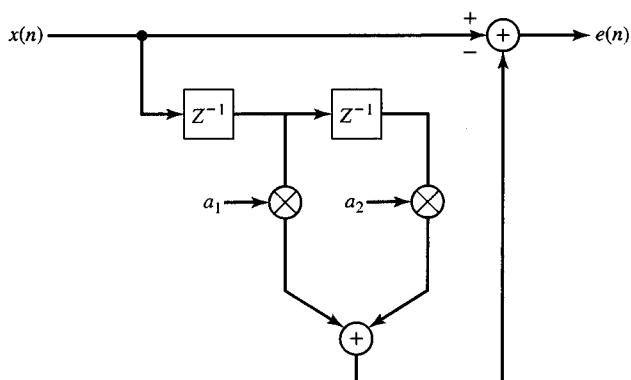


Figure P13.16

- 13.17** Determine the lattice filter and its optimum reflection coefficients corresponding to the linear predictor in Problem 13.16.
- 13.18** Consider the adaptive FIR filter shown in Figure P13.18. The system $C(z)$ is characterized by the system function

$$C(z) = \frac{1}{1 + 0.9z^{-1}}$$

Determine the optimum coefficients of the adaptive FIR filter $B(z) = b_0 + b_1 z^{-1}$ that minimize the MSE. The additive noise is white with variance $\sigma_w^2 = 0.1$.

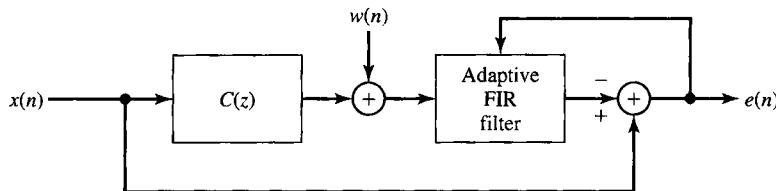


Figure P13.18

- 13.19** In the gradient lattice algorithm, the forward and backward prediction errors are given by (13.4.102).

- (a) Show that the minimization of the least-squares error

$$\epsilon_m^{\text{LS}} = \sum_{l=0}^n w^{n-l} [|f_m(l)|^2 + |g_m(l)|^2]$$

with respect to the reflection coefficients $\{k_m(n)\}$ results in the equation given by (13.4.103).

- (b) To determine an equation for the recursive computation of the reflection coefficients given by (13.4.103) define

$$\begin{aligned} u_m(n) &= w u_m(n-1) + 2 f_{m-1}(n) g_{m-1}^*(n-1) \\ v_m(n) &= w v_m(n-1) + |f_{m-1}(n)|^2 + |g_{m-1}(n-1)|^2 \end{aligned}$$

so that $k_m(n) = u_m(n)/v_m(n)$. Then, show that $k_m(n)$ can be computed recursively by the relation

$$k_m(n) = k_m(n-1) + \frac{f_m(n) g_{m-1}^*(n-1) + g_m^*(n) f_{m-1}(n)}{w v_m(n-1)}$$

- 13.20** Consider the adaptive predictor shown in Figure P13.16.

- (a) Determine the quadratic performance index and the optimum parameters for the signal

$$x(n) = \sin \frac{n\pi}{4} + w(n)$$

where $w(n)$ is white noise with variance $\sigma_w^2 = 0.1$.

- (b) Generate a sequence of 1000 samples of $x(n)$, and use the LMS algorithm to adaptively obtain the predictor coefficients. Compare the experimental results with the theoretical values obtained in part (a). Use a step size of $\Delta \leq \frac{1}{10} \Delta_{\max}$.
- (c) Repeat the experiment in part (b) for $N = 10$ trials with different noise sequences, and compute the average values of the predictor coefficients. Comment on how these results compare with the theoretical values in part (a).

13.21 An autoregressive process is described by the difference equation

$$x(n) = 1.26x(n-1) - 0.81x(n-2) + w(n)$$

- (a) Generate a sequence of $N = 1000$ samples of $x(n)$, where $w(n)$ is a white noise sequence with variance $\sigma_w^2 = 0.1$. Use the LMS algorithm to determine the parameters of a second-order ($p = 2$) linear predictor. Begin with $a_1(0) = a_2(0) = 0$. Plot the coefficients $a_1(n)$ and $a_2(n)$ as a function of the iteration number.
- (b) Repeat part (a) for 10 trials, using different noise sequences, and superimpose the 10 plots of $a_1(n)$ and $a_2(n)$.
- (c) Plot the learning curve for the average (over the 10 trials) MSE for the data in part (b).

13.22 A random process $x(n)$ is given as

$$\begin{aligned} x(n) &= s(n) + w(n) \\ &= \sin(\omega_0 n + \phi) + w(n), \quad \omega_0 = \pi/4, \quad \phi = 0 \end{aligned}$$

where $w(n)$ is an additive white noise sequence with variance $\sigma_w^2 = 0.1$.

- (a) Generate $N = 1000$ samples of $x(n)$ and simulate an adaptive line enhancer of length $L = 4$. Use the LMS algorithm to adapt the ALE.
- (b) Plot the output of the ALE.
- (c) Compute the autocorrelation $\gamma_{xx}(m)$ of the sequence $x(n)$.
- (d) Determine the theoretical values of the ALE coefficients and compare them with the experimental values.
- (e) Compute and plot the frequency response of the linear predictor (ALE).
- (f) Compute and plot the frequency response of the prediction-error filter.
- (g) Compute and plot the experimental values of the autocorrelation $r_{ee}(m)$ of the output error sequence for $0 \leq m < 10$.
- (h) Repeat the experiment for 10 trials, using different noise sequences, and superimpose the frequency response plots on the same graph.
- (i) Comment on the result in parts (a) through (h).

Power Spectrum Estimation

In this chapter we are concerned with the estimation of the spectral characteristics of signals characterized as random processes. Many of the phenomena that occur in nature are best characterized statistically in terms of averages. For example, meteorological phenomena such as the fluctuations in air temperature and pressure are best characterized statistically as random processes. Thermal noise voltages generated in resistors and electronic devices are additional examples of physical signals that are well modeled as random processes.

Due to the random fluctuations in such signals, we must adopt a statistical viewpoint, which deals with the average characteristics of random signals. In particular, the autocorrelation function of a random process is the appropriate statistical average that we will use for characterizing random signals in the time domain, and the Fourier transform of the autocorrelation function, which yields the power density spectrum, provides the transformation from the time domain to the frequency domain.

Power spectrum estimation methods have a relatively long history. For a historical perspective, the reader is referred to the paper by Robinson (1982) and the book by Marple (1987). Our treatment of this subject covers the classical power spectrum estimation methods based on the periodogram, originally introduced by Schuster (1898), and by Yule (1927), who originated the modern model-based or parametric methods. These methods were subsequently developed and applied by Walker (1931), Bartlett (1948), Parzen (1957), Blackman and Tukey (1958), Burg (1967), and others. We also describe the method of Capon (1969) and methods based on eigenanalysis of the data correlation matrix.

14.1 Estimation of Spectra from Finite-Duration Observations of Signals

The basic problem that we consider in this chapter is the estimation of the power density spectrum of a signal from observation of the signal over a finite time interval. As we will see, the finite record length of the data sequence is a major limitation on the quality of the power spectrum estimate. When dealing with signals that are statistically stationary, the longer the data record, the better the estimate that can be extracted from the data. On the other hand, if the signal statistics are nonstationary, we cannot select an arbitrarily long data record to estimate the spectrum. In such a case, the length of the data record that we select is determined by the rapidity of the time variations in the signal statistics. Ultimately, our goal is to select as short a data record as possible that still allows us to resolve the spectral characteristics of different signal components in the data record that have closely spaced spectra.

One of the problems that we encounter with classical power spectrum estimation methods based on a finite-length data record is the distortion of the spectrum that we are attempting to estimate. This problem occurs in both the computation of the spectrum for a deterministic signal and the estimation of the power spectrum of a random signal. Since it is easier to observe the effect of the finite length of the data record on a deterministic signal, we treat this case first. Thereafter, we consider only random signals and the estimation of their power spectra.

14.1.1 Computation of the Energy Density Spectrum

Let us consider the computation of the spectrum of a deterministic signal from a finite sequence of data. The sequence $x(n)$ is usually the result of sampling a continuous-time signal $x_a(t)$ at some uniform sampling rate F_s . Our objective is to obtain an estimate of the true spectrum from a finite-duration sequence $x(n)$.

Recall that if $x(t)$ is a finite-energy signal, that is,

$$E = \int_{-\infty}^{\infty} |x_a(t)|^2 dt < \infty$$

then its Fourier transform exists and is given as

$$X_a(F) = \int_{-\infty}^{\infty} x_a(t) e^{-j2\pi F t} dt$$

From Parseval's theorem we have

$$E = \int_{-\infty}^{\infty} |x_a(t)|^2 dt = \int_{-\infty}^{\infty} |X_a(F)|^2 dF \quad (14.1.1)$$

3.8

The quantity $|X_a(F)|^2$ represents the distribution of signal energy as a function of frequency, and hence it is called the energy density spectrum of the signal, that is,

$$S_{xx}(F) = |X_a(F)|^2 \quad (14.1.2)$$

as described in Chapter 4. Thus the total energy in the signal is simply the integral of $S_{xx}(F)$ over all F [i.e., the total area under $S_{xx}(F)$].

It is also interesting to note that $S_{xx}(F)$ can be viewed as the Fourier transform of another function, $R_{xx}(\tau)$, called the *autocorrelation function* of the finite-energy signal $x_a(t)$, defined as

$$R_{xx}(\tau) = \int_{-\infty}^{\infty} x_a^*(t)x_a(t + \tau) dt \quad (14.1.3)$$

Indeed, it easily follows that

$$\int_{-\infty}^{\infty} R_{xx}(\tau)e^{-j2\pi F\tau} d\tau = S_{xx}(F) = |X_a(F)|^2 \quad (14.1.4)$$

so that $R_{xx}(\tau)$ and $S_{xx}(F)$ are a Fourier transform pair.

Now suppose that we compute the energy density spectrum of the signal $x_a(t)$ from its samples taken at the rate F_s samples per second. To ensure that there is no spectral aliasing resulting from the sampling process, the signal is assumed to be prefiltered, so that, for practical purposes, its bandwidth is limited to B hertz. Then the sampling frequency F_s is selected such that $F_s > 2B$.

The sampled version of $x_a(t)$ is a sequence $x(n)$, $-\infty < n < \infty$, which has a Fourier transform (voltage spectrum)

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

or, equivalently,

$$X(f) = \sum_{n=-\infty}^{\infty} x(n)e^{-j2\pi f n} \quad (14.1.5)$$

Recall that $X(f)$ can be expressed in terms of the voltage spectrum of the analog signal $x_a(t)$ as

$$X(F) = F_s \sum_{k=-\infty}^{\infty} X_a(F - kF_s) \quad (14.1.6)$$

where $f = F/F_s$ is the normalized frequency variable.

In the absence of aliasing, within the fundamental range $|F| \leq F_s/2$, we have

$$X(F) = F_s X_a(F), |F| \leq F_s/2 \quad (14.1.7)$$

Hence the voltage spectrum of the sampled signal is identical to the voltage spectrum of the analog signal. As a consequence, the energy density spectrum of the sampled signal is

$$S_{xx}(F) = |X(F)|^2 = F_s^2 |X_a(F)|^2 \quad (14.1.8)$$

We can proceed further by noting that the autocorrelation of the sampled signal, which is defined as

$$r_{xx}(k) = \sum_{n=-\infty}^{\infty} x^*(n)x(n+k) \quad (14.1.9)$$

has the Fourier transform (Wiener–Khintchine theorem)

$$S_{xx}(f) = \sum_{k=-\infty}^{\infty} r_{xx}(k)e^{-j2\pi kf} \quad (14.1.10)$$

Hence the energy density spectrum can be obtained by the Fourier transform of the autocorrelation of the sequence $\{x(n)\}$.

The relations above lead us to distinguish between two distinct methods for computing the energy density spectrum of a signal $x_a(t)$ from its samples $x(n)$. One is the *direct method*, which involves computing the Fourier transform of $\{x(n)\}$, and then

$$\begin{aligned} S_{xx}(f) &= |X(f)|^2 \\ &= \left| \sum_{n=-\infty}^{\infty} x(n)e^{-j2\pi fn} \right|^2 \end{aligned} \quad (14.1.11)$$

The second approach is called the *indirect method* because it requires two steps. First, the autocorrelation $r_{xx}(k)$ is computed from $x(n)$ and then the Fourier transform of the autocorrelation is computed as in (14.1.10) to obtain the energy density spectrum.

In practice, however, only the finite-duration sequence $x(n)$, $0 \leq n \leq N - 1$, is available for computing the spectrum of the signal. In effect, limiting the duration of the sequence $x(n)$ to N points is equivalent to multiplying $x(n)$ by a rectangular window. Thus we have

$$\tilde{x}(n) = x(n)w(n) = \begin{cases} x(n), & 0 \leq n \leq N - 1 \\ 0, & \text{otherwise} \end{cases} \quad (14.1.12)$$

From our discussion of FIR filter design based on the use of windows to limit the duration of the impulse response, we recall that multiplication of two sequences is equivalent to convolution of their voltage spectra. Consequently, the frequency-domain relation corresponding to (14.1.12) is

$$\begin{aligned} \tilde{X}(f) &= X(f) * W(f) \\ &= \int_{-1/2}^{1/2} X(\alpha)W(f - \alpha)d\alpha \end{aligned} \quad (14.1.13)$$

Recall from our discussion in Section 10.2.1 that convolution of the window function $W(f)$ with $X(f)$ smooths the spectrum $X(f)$, provided that the spectrum $W(f)$ is relatively narrow compared to $X(f)$. But this condition implies that the window $w(n)$ be sufficiently long (i.e., N must be sufficiently large) such that $W(f)$

is narrow compared to $X(f)$. Even if $W(f)$ is narrow compared to $X(f)$, the convolution of $X(f)$ with the sidelobes of $W(f)$ results in sidelobe energy in $\tilde{X}(f)$, in frequency bands where the true signal spectrum $X(f) = 0$. This sidelobe energy is called *leakage*. The following example illustrates the leakage problem.

EXAMPLE 14.1.1

A signal with (voltage) spectrum

$$X(f) = \begin{cases} 1, & |f| \leq 0.1 \\ 0, & \text{otherwise} \end{cases}$$

is convolved with the rectangular window of length $N = 61$. Determine the spectrum of $\tilde{X}(f)$ given by (14.1.13).

Solution. The spectral characteristic $W(f)$ for the length $N = 61$ rectangular window is illustrated in Fig. 10.2.2. Note that the width of the main lobe of the window function is $\Delta\omega = 4\pi/61$ or $\Delta f = 2/61$, which is narrow compared to $X(f)$.

The convolution of $X(f)$ with $W(f)$ is illustrated in Fig. 14.1.1. We note that energy has leaked into the frequency band $0.1 < |f| \leq 0.5$, where $X(f) = 0$. A part of this is due to the width of the main lobe in $W(f)$, which causes a broadening or smearing of $X(f)$ outside the range $|f| \leq 0.1$. However, the sidelobe energy in $\tilde{X}(f)$ is due to the presence of the sidelobes of $W(f)$, which are convolved with $X(f)$. The smearing of $X(f)$ for $|f| > 0.1$ and the sidelobes in the range $0.1 \leq |f| \leq 0.5$ constitute the leakage.

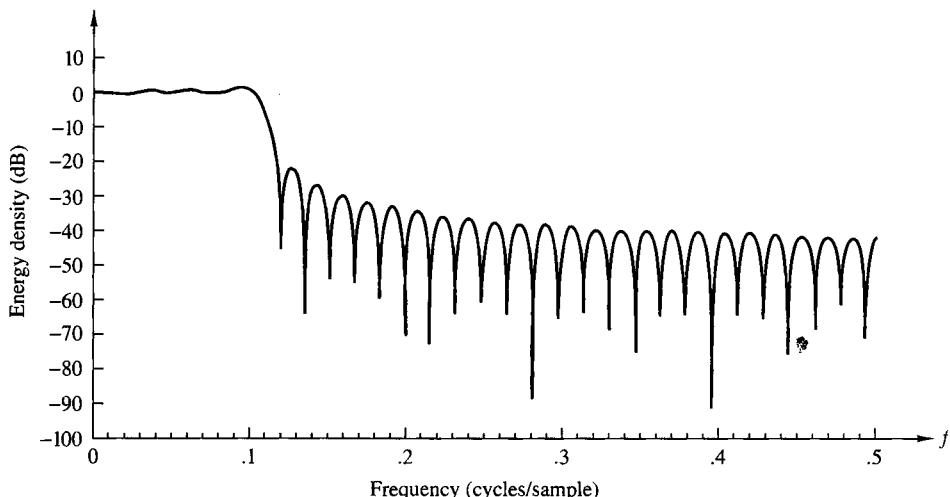


Figure 14.1.1 Spectrum obtained by convolving an $M = 61$ rectangular window with the ideal lowpass spectrum in Example 14.1.1.

Just as in the case of FIR filter design, we can reduce sidelobe leakage by selecting windows that have low sidelobes. This implies that the windows have a smooth time-domain cutoff instead of the abrupt cutoff in the rectangular window. Although such window functions reduce sidelobe leakage, they result in an increase in smoothing or

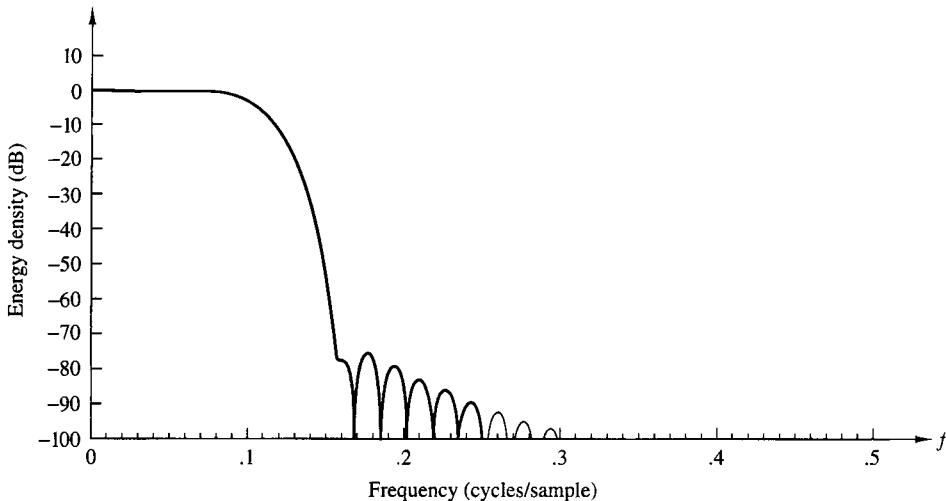


Figure 14.1.2 Spectrum obtained by convolving an $M = 61$ Blackman window with the ideal lowpass spectrum in Example 14.1.1.

broadening of the spectral characteristic $X(f)$. For example, the use of a Blackman window of length $N = 61$ in Example 14.1.1 results in the spectral characteristic $\tilde{X}(f)$ shown in Fig. 14.1.2. The sidelobe leakage has certainly been reduced, but the spectral width has been increased by about 50%.

The broadening of the spectrum being estimated due to windowing is particularly a problem when we wish to resolve signals with closely spaced frequency components. For example, the signal with spectral characteristic $X(f) = X_1(f) + X_2(f)$, as shown in Fig. 14.1.3, cannot be resolved as two separate signals unless the width of the window function is significantly narrower than the frequency separation Δf . Thus we observe that using smooth time-domain windows reduces leakage at the expense of a decrease in frequency resolution.

It is clear from this discussion that the energy density spectrum of the windowed sequence $\{\tilde{x}(n)\}$ is an approximation of the desired spectrum of the sequence $\{x(n)\}$.

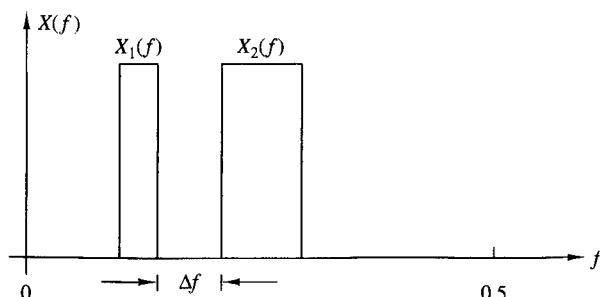


Figure 14.1.3
Two narrowband signal spectra.

The spectral density obtained from $\{\tilde{x}(n)\}$ is

$$S_{\tilde{x}\tilde{x}}(f) = |\tilde{X}(f)|^2 = \left| \sum_{n=0}^{N-1} \tilde{x}(n) e^{-j2\pi f n} \right|^2 \quad (14.1.14)$$

The spectrum given by (14.1.14) can be computed numerically at a set of N frequency points by means of the DFT. Thus

$$\tilde{X}(k) = \sum_{n=0}^{N-1} \tilde{x}(n) e^{-j2\pi k n / N} \quad (14.1.15)$$

Then

$$|\tilde{X}(k)|^2 = S_{\tilde{x}\tilde{x}}(f)|_{f=k/N} = S_{\tilde{x}\tilde{x}}\left(\frac{k}{N}\right) \quad (14.1.16)$$

and hence

$$S_{\tilde{x}\tilde{x}}\left(\frac{k}{N}\right) = \left| \sum_{n=0}^{N-1} \tilde{x}(n) e^{-j2\pi k n / N} \right|^2 \quad (14.1.17)$$

which is a distorted version of the true spectrum $S_{xx}(k/N)$.

14.1.2 Estimation of the Autocorrelation and Power Spectrum of Random Signals: The Periodogram

The finite-energy signals considered in the preceding section possess a Fourier transform and are characterized in the spectral domain by their energy density spectrum. On the other hand, the important class of signals characterized as stationary random processes do not have finite energy and hence do not possess a Fourier transform. Such signals have finite average power and hence are characterized by a *power density spectrum*. If $x(t)$ is a stationary random process, its autocorrelation function is

$$\gamma_{xx}(\tau) = E[x^*(t)x(t + \tau)] \quad (14.1.18)$$

where $E[\cdot]$ denotes the statistical average. Then, via the Wiener–Khintchine theorem, the power density spectrum of the stationary random process is the Fourier transform of the autocorrelation function, that is,

$$\Gamma_{xx}(F) = \int_{-\infty}^{\infty} \gamma_{xx}(\tau) e^{-j2\pi F \tau} dt \quad (14.1.19)$$

In practice, we deal with a single realization of the random process from which we estimate the power spectrum of the process. We do not know the true autocorrelation function $\gamma_{xx}(\tau)$ and as a consequence, we cannot compute the Fourier transform in

(14.1.19) to obtain $\Gamma_{xx}(F)$. On the other hand, from a single realization of the random process we can compute the time-average autocorrelation function

$$R_{xx}(\tau) = \frac{1}{2T_0} \int_{-T_0}^{T_0} x^*(t)x(t + \tau) dt \quad (14.1.20)$$

where $2T_0$ is the observation interval. If the stationary random process is *ergodic* in the first and second moments (mean and autocorrelation function), then

$$\begin{aligned} \gamma_{xx}(\tau) &= \lim_{T_0 \rightarrow \infty} R_{xx}(\tau) \\ &= \lim_{T_0 \rightarrow \infty} \frac{1}{2T_0} \int_{-T_0}^{T_0} x^*(t)x(t + \tau) dt \end{aligned} \quad (14.1.21)$$

This relation justifies the use of the time-average autocorrelation $R_{xx}(\tau)$ as an estimate of the statistical autocorrelation function $\gamma_{xx}(\tau)$. Furthermore, the Fourier transform of $R_{xx}(\tau)$ provides an estimate $P_{xx}(F)$ of the power density spectrum, that is,

$$\begin{aligned} P_{xx}(F) &= \int_{-T_0}^{T_0} R_{xx}(\tau) e^{-j2\pi F\tau} d\tau \\ &= \frac{1}{2T_0} \int_{-T_0}^{T_0} \left[\int_{-T_0}^{T_0} x^*(t)x(t + \tau) dt \right] e^{-j2\pi F\tau} d\tau \\ &= \frac{1}{2T_0} \left| \int_{-T_0}^{T_0} x(t) e^{-j2\pi Ft} dt \right|^2 \end{aligned} \quad (14.1.22)$$

The actual power density spectrum is the expected value of $P_{xx}(F)$ in the limit as $T_0 \rightarrow \infty$,

$$\begin{aligned} \Gamma_{xx}(F) &= \lim_{T_0 \rightarrow \infty} E[P_{xx}(F)] \\ &= \lim_{T_0 \rightarrow \infty} E \left[\frac{1}{2T_0} \left| \int_{-T_0}^{T_0} x(t) e^{-j2\pi Ft} dt \right|^2 \right] \end{aligned} \quad (14.1.23)$$

From (14.1.20) and (14.1.22) we again note the two possible approaches to computing $P_{xx}(F)$, the direct method as given by (14.1.22) or the indirect method, in which we obtain $R_{xx}(\tau)$ first and then compute the Fourier transform.

We shall consider the estimation of the power density spectrum from samples of a single realization of the random process. In particular, we assume that $x_a(t)$ is sampled at a rate $F_s > 2B$, where B is the highest frequency contained in the power density spectrum of the random process. Thus we obtain a finite-duration

sequence $x(n)$, $0 \leq n \leq N - 1$, by sampling $x_a(t)$. From these samples we compute the time-average autocorrelation sequence

$$r'_{xx}(m) = \frac{1}{N-m} \sum_{n=0}^{N-m-1} x^*(n)x(n+m), \quad m = 0, 1, \dots, N-1 \quad (14.1.24)$$

and for negative values of m we have $r'_{xx}(m) = [r'_{xx}(-m)]^*$. Then, we compute the Fourier transform

$$P'_{xx}(f) = \sum_{m=-N+1}^{N-1} r'_{xx}(m)e^{-j2\pi fm} \quad (14.1.25)$$

The normalization factor $N - m$ in (14.1.24) results in an estimate with mean value

$$\begin{aligned} E[r'_{xx}(m)] &= \frac{1}{N-m} \sum_{n=0}^{N-m-1} E[x^*(n)x(n+m)] \\ &= \gamma_{xx}(m) \end{aligned} \quad (14.1.26)$$

where $\gamma_{xx}(m)$ is the true (statistical) autocorrelation sequence of $x(n)$. Hence $r'_{xx}(m)$ is an unbiased estimate of the autocorrelation function $\gamma_{xx}(m)$. The variance of the estimate $r'_{xx}(m)$ is approximately

$$\text{var}[r'_{xx}(m)] \approx \frac{N}{[N-m]^2} \sum_{n=-\infty}^{\infty} \left[|\gamma_{xx}(n)|^2 + \gamma_{xx}^*(n-m)\gamma_{xx}(n+m) \right] \quad (14.1.27)$$

which is a result given by Jenkins and Watts (1968). Clearly,

$$\lim_{N \rightarrow \infty} \text{var}[r'_{xx}(m)] = 0 \quad (14.1.28)$$

provided that

$$\sum_{n=-\infty}^{\infty} |\gamma_{xx}(n)|^2 < \infty$$

Since $E[r'_{xx}(m)] = \gamma_{xx}(m)$ and the variance of the estimate converges to zero as $N \rightarrow \infty$, the estimate $r'_{xx}(m)$ is said to be *consistent*.

For large values of the lag parameter m , the estimate $r'_{xx}(m)$ given by (14.1.24) has a large variance, especially as m approaches N . This is due to the fact that fewer data points enter into the estimate for large lags. As an alternative to (14.1.24) we can use the estimate

$$r_{xx}(m) = \frac{1}{N} \sum_{n=0}^{N-m-1} x^*(n)x(n+m), \quad 0 \leq m \leq N-1 \quad (14.1.29)$$

$$r_{xx}(m) = \frac{1}{N} \sum_{n=|m|}^{N-1} x^*(n)x(n+m), \quad m = -1, -2, \dots, 1-N$$

which has a bias of $|m|\gamma_{xx}(m)/N$, since its mean value is

$$\begin{aligned} E[r_{xx}(m)] &= \frac{1}{N} \sum_{n=0}^{N-m-1} E[x^*(n)x(n+m)] \\ &= \frac{N-|m|}{N} \gamma_{xx}(m) = \left(1 - \frac{|m|}{N}\right) \gamma_{xx}(m) \end{aligned} \quad (14.1.30)$$

However, this estimate has a smaller variance, given approximately as

$$\text{var}[r_{xx}(m)] \approx \frac{1}{N} \sum_{n=-\infty}^{\infty} [\gamma_{xx}(n)]^2 + \gamma_{xx}^*(n-m)\gamma_{xx}(n+m) \quad (14.1.31)$$

We observe that $r_{xx}(m)$ is *asymptotically unbiased*, that is,

$$\lim_{N \rightarrow \infty} E[r_{xx}(m)] = \gamma_{xx}(m) \quad (14.1.32)$$

and its variance converges to zero as $N \rightarrow \infty$. Therefore, the estimate $r_{xx}(m)$ is also a *consistent estimate* of $\gamma_{xx}(m)$.

We shall use the estimate $r_{xx}(m)$ given by (14.1.29) in our treatment of power spectrum estimation. The corresponding estimate of the power density spectrum is

$$P_{xx}(f) = \sum_{m=-(N-1)}^{N-1} r_{xx}(m) e^{-j2\pi fm} \quad (14.1.33)$$

If we substitute for $r_{xx}(m)$ from (14.1.29) into (14.1.33), the estimate $P_{xx}(f)$ can also be expressed as

$$P_{xx}(f) = \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n) e^{-j2\pi fn} \right|^2 = \frac{1}{N} |X(f)|^2 \quad (14.1.34)$$

where $X(f)$ is the Fourier transform of the sample sequence $x(n)$. This well-known form of the power density spectrum estimate is called the *periodogram*. It was originally introduced by Schuster (1898) to detect and measure “hidden periodicities” in data.

From (14.1.33), the average value of the periodogram estimate $P_{xx}(f)$ is

$$E[P_{xx}(f)] = E \left[\sum_{m=-(N-1)}^{N-1} r_{xx}(m) e^{-j2\pi fm} \right] = \sum_{m=-(N-1)}^{N-1} E[r_{xx}(m)] e^{-j2\pi fm} \quad (14.1.35)$$

$$E[P_{xx}(f)] = \sum_{m=-(N-1)}^{N-1} \left(1 - \frac{|m|}{N}\right) \gamma_{xx}(m) e^{-j2\pi fm}$$

The interpretation that we give to (14.1.35) is that the mean of the estimated spectrum is the Fourier transform of the windowed autocorrelation function

$$\tilde{\gamma}_{xx}(m) = \left(1 - \frac{|m|}{N}\right) \gamma_{xx}(m) \quad (14.1.36)$$

where the window function is the (triangular) Bartlett window. Hence the mean of the estimated spectrum is

$$\begin{aligned} E[P_{xx}(f)] &= \sum_{m=-\infty}^{\infty} \tilde{\gamma}_{xx}(m) e^{-j2\pi f m} \\ &= \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha) W_B(f - \alpha) d\alpha \end{aligned} \quad (14.1.37)$$

where $W_B(f)$ is the spectral characteristic of the Bartlett window. The relation (14.1.37) illustrates that the mean of the estimated spectrum is the convolution of the true power density spectrum $\Gamma_{xx}(f)$ with the Fourier transform $W_B(f)$ of the Bartlett window. Consequently, the mean of the estimated spectrum is a smoothed version of the true spectrum and suffers from the same spectral leakage problems, which are due to the finite number of data points.

We observe that the estimated spectrum is asymptotically unbiased, that is,

$$\lim_{N \rightarrow \infty} E \left[\sum_{m=-(N-1)}^{N-1} r_{xx}(m) e^{-j2\pi f m} \right] = \sum_{m=-\infty}^{\infty} \gamma_{xx}(m) e^{-j2\pi f m} = \Gamma_{xx}(f)$$

However, in general, the variance of the estimate $P_{xx}(f)$ does not decay to zero as $N \rightarrow \infty$. For example, when the data sequence is a Gaussian random process, the variance is easily shown to be (see Problem 14.4)

$$\text{var}[P_{xx}(f)] = \Gamma_{xx}^2(f) \left[1 + \left(\frac{\sin 2\pi f N}{N \sin 2\pi f} \right)^2 \right] \quad (14.1.38)$$

which, in the limit as $N \rightarrow \infty$, becomes

$$\lim_{N \rightarrow \infty} \text{var}[P_{xx}(f)] = \Gamma_{xx}^2(f) \quad (14.1.39)$$

Hence we conclude that the *periodogram is not a consistent estimate of the true power density spectrum* (i.e., it does not converge to the true power density spectrum).

In summary, the estimated autocorrelation $r_{xx}(m)$ is a consistent estimate of the true autocorrelation function $\gamma_{xx}(m)$. However, its Fourier transform $P_{xx}(f)$, the periodogram, is not a consistent estimate of the true power density spectrum. We observed that $P_{xx}(f)$ is an asymptotically unbiased estimate of $\Gamma_{xx}(f)$, but for a finite-duration sequence, the mean value of $P_{xx}(f)$ contains a bias, which from (14.1.37) is evident as a distortion of the true power density spectrum. Thus the

estimated spectrum suffers from the smoothing effects and the leakage embodied in the Bartlett window. The smoothing and leakage ultimately limit our ability to resolve closely spaced spectra.

The problems of leakage and frequency resolution that we have just described, as well as the problem that the periodogram is not a consistent estimate of the power spectrum, provide the motivation for the power spectrum estimation methods described in Sections 14.2, 14.3, and 14.4. The methods described in Section 14.2 are classical nonparametric methods, which make no assumptions about the data sequence. The emphasis of the classical methods is on obtaining a consistent estimate of the power spectrum through some averaging or smoothing operations performed directly on the periodogram or on the autocorrelation. As we will see, the effect of these operations is to reduce the frequency resolution further, while the variance of the estimate is decreased.

The spectrum estimation methods described in Section 14.3 are based on some model of how the data were generated. In general, the model-based methods that have been developed over the past few decades provide significantly higher resolution than do the classical methods.

Additional methods are described in Sections 14.4 and 14.5. In Section 14.4, we consider filter bank methods for the estimation of the power spectrum. The methods described in Section 14.5 are based on an eigenvalue/eigenvector decomposition of the data correlation matrix.

14.1.3 The Use of the DFT in Power Spectrum Estimation

As given by (14.1.14) and (14.1.34), the estimated energy density spectrum $S_{xx}(f)$ and the periodogram $P_{xx}(f)$, respectively, can be computed by use of the DFT, which in turn is efficiently computed by an FFT algorithm. If we have N data points, we compute as a minimum the N -point DFT. For example, the computation yields samples of the periodogram

$$P_{xx}\left(\frac{k}{N}\right) = \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N} \right|^2, \quad k = 0, 1, \dots, N-1 \quad (14.1.40)$$

at the frequencies $f_k = k/N$.

In practice, however, such a sparse sampling of the spectrum does not provide a very good representation or a good picture of the continuous spectrum estimate $P_{xx}(f)$. This is easily remedied by evaluating $P_{xx}(f)$ at additional frequencies. Equivalently, we can effectively increase the length of the sequence by means of zero padding and then evaluate $P_{xx}(f)$ at a more dense set of frequencies. Thus if we increase the data sequence length to L points by means of zero padding and evaluate the L -point DFT, we have

$$P_{xx}\left(\frac{k}{L}\right) = \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/L} \right|^2, \quad k = 0, 1, \dots, L-1 \quad (14.1.41)$$

We emphasize that zero padding and evaluating the DFT at $L > N$ points does not improve the frequency resolution in the spectral estimate. It simply provides us with a method for interpolating the values of the measured spectrum at more frequencies. The frequency resolution in the spectral estimate $P_{xx}(f)$ is determined by the length N of the data record.

EXAMPLE 14.1.2

A sequence of $N = 16$ samples is obtained by sampling an analog signal consisting of two frequency components. The resulting discrete-time sequence is

$$x(n) = \sin 2\pi(0.135)n + \cos 2\pi(0.135 + \Delta f)n, \quad n = 0, 1, \dots, 15$$

where Δf is the frequency separation. Evaluate the power spectrum $P(f) = (1/N)|X(f)|^2$ at the frequencies $f_k = k/L$, $k = 0, 1, \dots, L - 1$, for $L = 8, 16, 32$, and 128 for values of $\Delta f = 0.06$ and $\Delta f = 0.01$.

Solution. By zero padding, we increase the data sequence to obtain the power spectrum estimate $P_{xx}(k/L)$. The results for $\Delta f = 0.06$ are plotted in Fig. 14.1.4. Note that zero padding

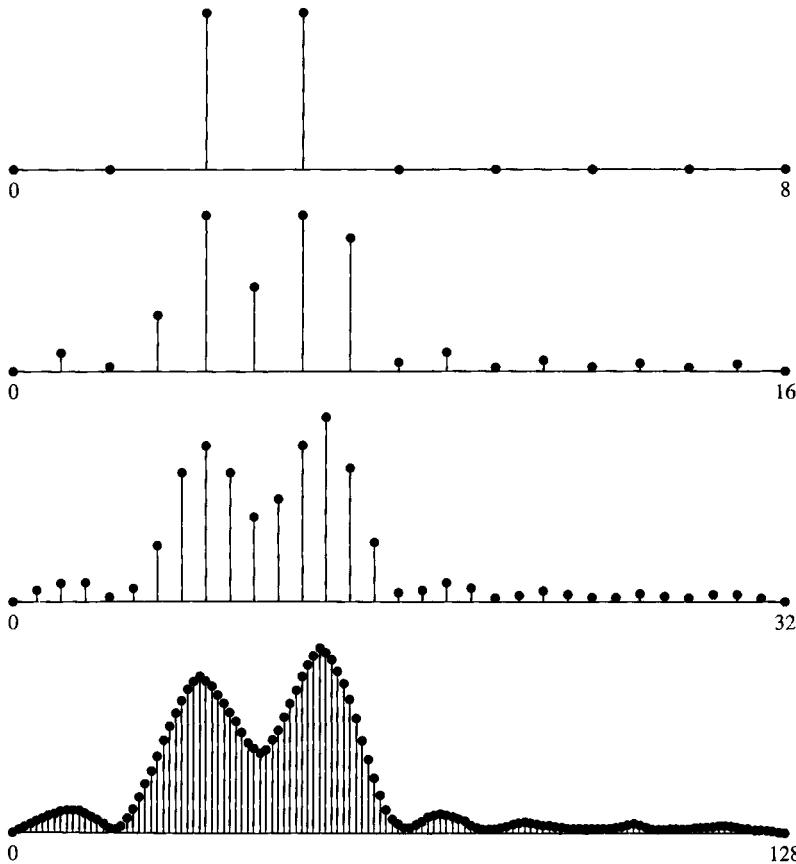


Figure 14.1.4 Spectra of two sinusoids with frequency separation $\Delta f = 0.06$.

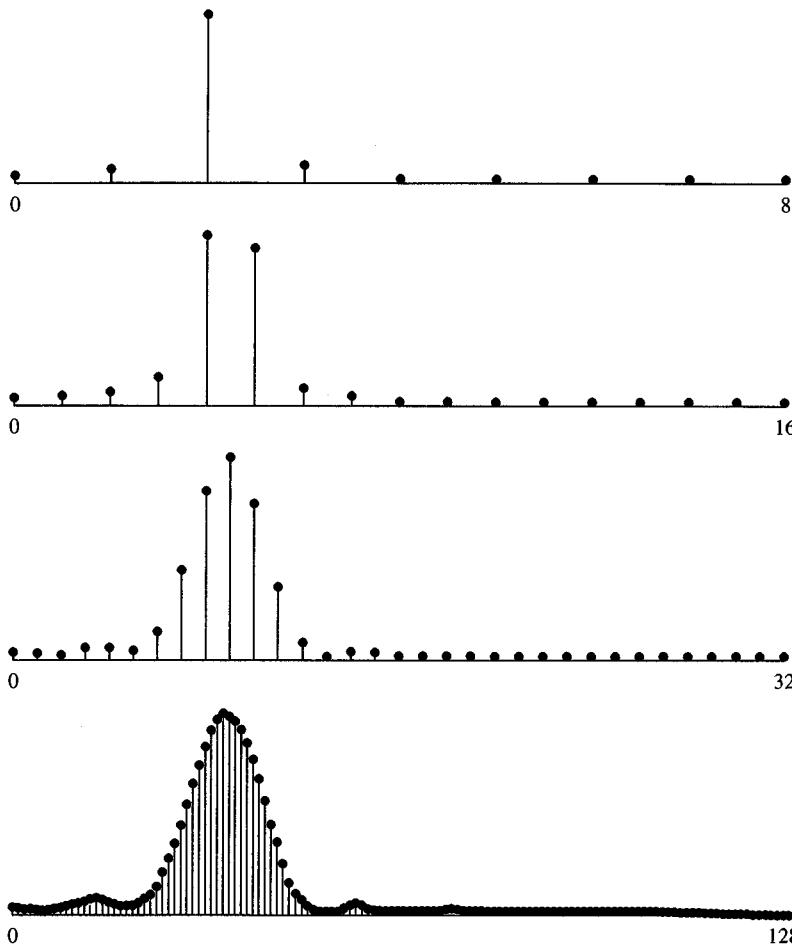


Figure 14.1.5 Spectra of two sinusoids with frequency separation $\Delta f = 0.01$.

does not change the resolution, but it does have the effect of interpolating the spectrum $P_{xx}(f)$. In this case the frequency separation Δf is sufficiently large so that the two frequency components are resolvable.

The spectral estimates for $\Delta f = 0.01$ are shown in Fig. 14.1.5. In this case the two spectral components are not resolvable. Again, the effect of zero padding is to provide more interpolation, thus giving us a better picture of the estimated spectrum. It does not improve the frequency resolution.

When only a few points of the periodogram are needed, the Goertzel algorithm described in Chapter 8 may provide a more efficient computation. Since the Goertzel algorithm has been interpreted as a linear filtering approach to computing the DFT, it is clear that the periodogram estimate can be obtained by passing the signal through a bank of parallel tuned filters and squaring their outputs (see Problem 14.5).

14.2 Nonparametric Methods for Power Spectrum Estimation

The power spectrum estimation methods described in this section are the classical methods developed by Bartlett (1948), Blackman and Tukey (1958), and Welch (1967). These methods make no assumption about how the data were generated and hence are called *nonparametric*.

Since the estimates are based entirely on a finite record of data, the frequency resolution of these methods is, at best, equal to the spectral width of the rectangular window of length N , which is approximately $1/N$ at the -3 -dB points. We shall be more precise in specifying the frequency resolution of the specific methods. All the estimation techniques described in this section decrease the frequency resolution in order to reduce the variance in the spectral estimate.

First, we describe the estimates and derive the mean and variance of each. A comparison of the three methods is given in Section 14.2.4. Although the spectral estimates are expressed as a function of the continuous frequency variable f , in practice, the estimates are computed at discrete frequencies via the FFT algorithm. The FFT-based computational requirements are considered in Section 14.2.5.

14.2.1 The Bartlett Method: Averaging Periodograms

Bartlett's method for reducing the variance in the periodogram involves three steps. First, the N -point sequence is subdivided into K nonoverlapping segments, where each segment has length M . This results in the K data segments

$$x_i(n) = x(n + iM), \quad \begin{matrix} i = 0, 1, \dots, K-1 \\ n = 0, 1, \dots, M-1 \end{matrix} \quad (14.2.1)$$

For each segment, we compute the periodogram

$$P_{xx}^{(i)}(f) = \frac{1}{M} \left| \sum_{n=0}^{M-1} x_i(n) e^{-j2\pi f n} \right|^2, \quad i = 0, 1, \dots, K-1 \quad (14.2.2)$$

Finally, we average the periodograms for the K segments to obtain the Bartlett power spectrum estimate [Bartlett (1948)]

$$P_{xx}^B(f) = \frac{1}{K} \sum_{i=0}^{K-1} P_{xx}^{(i)}(f) \quad (14.2.3)$$

The statistical properties of this estimate are easily obtained. First, the mean value is

$$\begin{aligned} E[P_{xx}^B(f)] &= \frac{1}{K} \sum_{i=0}^{K-1} E[P_{xx}^{(i)}(f)] \\ &= E[P_{xx}^{(i)}(f)] \end{aligned} \quad (14.2.4)$$

From (14.1.35) and (14.1.37) we have the expected value for the single periodogram as

$$\begin{aligned} E[P_{xx}^{(i)}(f)] &= \sum_{m=-(M-1)}^{M-1} \left(1 - \frac{|m|}{M}\right) \gamma_{xx}(m) e^{-j2\pi f m} \\ &= \frac{1}{M} \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha) \left(\frac{\sin \pi(f-\alpha)M}{\sin \pi(f-\alpha)}\right)^2 d\alpha \end{aligned} \quad (14.2.5)$$

where

$$W_B(f) = \frac{1}{M} \left(\frac{\sin \pi f M}{\sin \pi f}\right)^2 \quad (14.2.6)$$

is the frequency characteristic of the Bartlett window

$$w_B(n) = \begin{cases} 1 - \frac{|n|}{M}, & |n| \leq M-1 \\ 0, & \text{otherwise} \end{cases} \quad (14.2.7)$$

From (14.2.5) we observe that the true spectrum is now convolved with the frequency characteristic $W_B(f)$ of the Bartlett window. The effect of reducing the length of the data from N points to $M = N/K$ results in a window whose spectral width has been increased by a factor of K . Consequently, the frequency resolution has been reduced by a factor K .

In return for this reduction in resolution, we have reduced the variance. The variance of the Bartlett estimate is

$$\begin{aligned} \text{var}[P_{xx}^B(f)] &= \frac{1}{K^2} \sum_{i=0}^{K-1} \text{var}[P_{xx}^{(i)}(f)] \\ &= \frac{1}{K} \text{var}[P_{xx}^{(i)}(f)] \end{aligned} \quad (14.2.8)$$

If we make use of (14.1.38) in (14.2.8), we obtain

$$\text{var}[P_{xx}^B(f)] = \frac{1}{K} \Gamma_{xx}^2(f) \left[1 + \left(\frac{\sin 2\pi f M}{M \sin 2\pi f} \right)^2 \right] \quad \text{for } M \rightarrow \infty \quad (14.2.9)$$

Therefore, the variance of the Bartlett power spectrum estimate has been reduced by the factor K .

14.2.2 The Welch Method: Averaging Modified Periodograms

Welch (1967) made two basic modifications to the Bartlett method. First, he allowed the data segments to overlap. Thus the data segments can be represented as

$$x_i(n) = x(n + iD), \quad \begin{matrix} n = 0, 1, \dots, M-1 \\ i = 0, 1, \dots, L-1 \end{matrix} \quad (14.2.10)$$

where iD is the starting point for the i th sequence. Observe that if $D = M$, the segments do not overlap and the number L of data segments is identical to the number K in the Bartlett method. However, if $D = M/2$, there is 50% overlap between successive data segments and $L = 2K$ segments are obtained. Alternatively, we can form K data segments each of length $2M$.

The second modification made by Welch to the Bartlett method is to window the data segments prior to computing the periodogram. The result is a “modified” periodogram

$$\tilde{P}_{xx}^{(i)}(f) = \frac{1}{MU} \left| \sum_{n=0}^{M-1} x_i(n)w(n)e^{-j2\pi fn} \right|^2, \quad i = 0, 1, \dots, L-1 \quad (14.2.11)$$

where U is a normalization factor for the power in the window function and is selected as

$$U = \frac{1}{M} \sum_{n=0}^{M-1} w^2(n) \quad (14.2.12)$$

The Welch power spectrum estimate is the average of these modified periodograms, that is,

$$P_{xx}^W(f) = \frac{1}{L} \sum_{i=0}^{L-1} \tilde{P}_{xx}^{(i)}(f) \quad (14.2.13)$$

The mean value of the Welch estimate is

$$\begin{aligned} E[P_{xx}^W(f)] &= \frac{1}{L} \sum_{i=0}^{L-1} E[\tilde{P}_{xx}^{(i)}(f)] \\ &= E[\tilde{P}_{xx}^{(i)}(f)] \end{aligned} \quad (14.2.14)$$

But the expected value of the modified periodogram is

$$\begin{aligned} E[\tilde{P}_{xx}^{(i)}(f)] &= \frac{1}{MU} \sum_{n=0}^{M-1} \sum_{m=0}^{M-1} w(n)w(m) E[x_i(n)x_i^*(m)] e^{-j2\pi f(n-m)} \\ &= \frac{1}{MU} \sum_{n=0}^{M-1} \sum_{m=0}^{M-1} w(n)w(m) \gamma_{xx}(n-m) e^{-j2\pi f(n-m)} \end{aligned} \quad (14.2.15)$$

Since

$$\gamma_{xx}(n) = \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha) e^{j2\pi\alpha n} d\alpha \quad (14.2.16)$$

substitution for $\gamma_{xx}(n)$ from (14.2.16) into (14.2.15) yields

$$\begin{aligned} E[\tilde{P}_{xx}^{(i)}(f)] &= \frac{1}{MU} \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha) \left[\sum_{n=0}^{M-1} \sum_{m=0}^{M-1} w(n)w(m) e^{-j2\pi(n-m)(f-\alpha)} \right] d\alpha \\ &= \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha) W(f - \alpha) d\alpha \end{aligned} \quad (14.2.17)$$

where, by definition,

$$W(f) = \frac{1}{MU} \left| \sum_{n=0}^{M-1} w(n) e^{-j2\pi f n} \right|^2 \quad (14.2.18)$$

The normalization factor U ensures that

$$\int_{-1/2}^{1/2} W(f) df = 1 \quad (14.2.19)$$

The variance of the Welch estimate is

$$\text{var}[P_{xx}^W(f)] = \frac{1}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} E[\tilde{P}_{xx}^{(i)}(f) \tilde{P}_{xx}^{(j)}(f)] - \{E[P_{xx}^W(f)]\}^2 \quad (14.2.20)$$

In the case of no overlap between successive data segments ($L = K$), Welch has shown that

$$\begin{aligned} \text{var}[P_{xx}^W(f)] &= \frac{1}{L} \text{var}[\tilde{P}_{xx}^{(i)}(f)] \\ &\approx \frac{1}{L} \Gamma_{xx}^2(f) \end{aligned} \quad (14.2.21)$$

In the case of 50% overlap between successive data segments ($L = 2K$), the variance of the Welch power spectrum estimate with the Bartlett (triangular) window, also derived in the paper by Welch, is

$$\text{var}[P_{xx}^W(f)] \approx \frac{9}{8L} \Gamma_{xx}^2(f) \quad (14.2.22)$$

Although we considered only the triangular window in the computation of the variance, other window functions may be used. In general, they will yield a different variance. In addition, one may vary the data segment overlapping by either more or less than the 50% considered in this section in an attempt to improve the relevant characteristics of the estimate.

14.2.3 The Blackman and Tukey Method: Smoothing the Periodogram

Blackman and Tukey (1958) proposed and analyzed the method in which the sample autocorrelation sequence is windowed first and then Fourier transformed to yield the estimate of the power spectrum. The rationale for windowing the estimated autocorrelation sequence $r_{xx}(m)$ is that, for large lags, the estimates are less reliable because a smaller number ($N - m$) of data points enter into the estimate. For values of m approaching N , the variance of these estimates is very high, and hence these estimates should be given a smaller weight in the formation of the estimated power spectrum. Thus the Blackman–Tukey estimate is

$$P_{xx}^{\text{BT}}(f) = \sum_{m=-(M-1)}^{M-1} r_{xx}(m)w(m)e^{-j2\pi fm} \quad (14.2.23)$$

where the window function $w(n)$ has length $2M - 1$ and is zero for $|m| \geq M$. With this definition for $w(n)$, the limits on the sum in (14.2.23) can be extended to $(-\infty, \infty)$. Hence the frequency-domain equivalent expression for (14.2.23) is the convolution integral

$$P_{xx}^{\text{BT}}(f) = \int_{-1/2}^{1/2} P_{xx}(\alpha)W(f - \alpha)d\alpha \quad (14.2.24)$$

where $P_{xx}(f)$ is the periodogram. It is clear from (14.2.24) that the effect of windowing the autocorrelation is to smooth the periodogram estimate, thus decreasing the variance in the estimate at the expense of reducing the resolution.

The window sequence $w(n)$ should be symmetric (even) about $m = 0$ to ensure that the estimate of the power spectrum is real. Furthermore, it is desirable to select the window spectrum to be nonnegative, that is,

$$W(f) \geq 0, \quad |f| \leq 1/2 \quad (14.2.25)$$

This condition ensures that $P_{xx}^{\text{BT}}(f) \geq 0$ for $|f| \leq 1/2$, which is a desirable property for any power spectrum estimate. We should indicate, however, that some of the window functions we have introduced do not satisfy this condition. For example, in spite of their low sidelobe levels, the Hamming and Hann (or Hanning) windows do not satisfy the property in (14.2.25) and, consequently, may result in negative spectrum estimates in some parts of the frequency range.

The expected value of the Blackman–Tukey power spectrum estimate is

$$E[P_{xx}^{\text{BT}}(f)] = \int_{-1/2}^{1/2} E[P_{xx}(\alpha)]W(f - \alpha)d\alpha \quad (14.2.26)$$

where from (14.1.37) we have

$$E[P_{xx}(\alpha)] = \int_{-1/2}^{1/2} \Gamma_{xx}(\theta)W_B(\alpha - \theta)d\theta \quad (14.2.27)$$

and $W_B(f)$ is the Fourier transform of the Bartlett window. Substitution of (14.2.27) into (14.2.26) yields the double convolution integral

$$E[P_{xx}^{\text{BT}}(f)] = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} \Gamma_{xx}(\theta) W_B(\alpha - \theta) W(f - \alpha) d\alpha d\theta \quad (14.2.28)$$

Equivalently, by working in the time domain, the expected value of the Blackman–Tukey power spectrum estimate is

$$\begin{aligned} E[P_{xx}^{\text{BT}}(f)] &= \sum_{m=-(M-1)}^{M-1} E[r_{xx}(m)] w(m) e^{-j2\pi f m} \\ &= \sum_{m=-(M-1)}^{M-1} \gamma_{xx}(m) w_B(m) w(m) e^{-j2\pi f m} \end{aligned} \quad (14.2.29)$$

where the Bartlett window is

$$w_B(m) = \begin{cases} 1 - \frac{|m|}{N}, & |m| < N \\ 0, & \text{otherwise} \end{cases} \quad (14.2.30)$$

Clearly, we should select the window length for $w(n)$ such that $M \ll N$, that is, $w(n)$ should be narrower than $w_B(m)$ to provide additional smoothing of the periodogram. Under this condition, (14.2.28) becomes

$$E[P_{xx}^{\text{BT}}(f)] \approx \int_{-1/2}^{1/2} \Gamma_{xx}(\theta) W(f - \theta) d\theta \quad (14.2.31)$$

since

$$\begin{aligned} \int_{-1/2}^{1/2} W_B(\alpha - \theta) W(f - \alpha) d\alpha &= \int_{-1/2}^{1/2} W_B(\alpha) W(f - \theta - \alpha) d\alpha \\ &\approx W(f - \theta) \end{aligned} \quad (14.2.32)$$

The variance of the Blackman–Tukey power spectrum estimate is

$$\text{var}[P_{xx}^{\text{BT}}(f)] = E\{[P_{xx}^{\text{BT}}(f)]^2\} - \{E[P_{xx}^{\text{BT}}(f)]\}^2 \quad (14.2.33)$$

where the mean can be approximated as in (14.2.31). The second moment in (14.2.33) is

$$E\{[P_{xx}^{\text{BT}}(f)]^2\} = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} E[P_{xx}(\alpha) P_{xx}(\theta)] W(f - \alpha) W(f - \theta) d\alpha d\theta \quad (14.2.34)$$

On the assumption that the random process is Gaussian (see Problem 14.5), we find that

$$E[P_{xx}(\alpha)P_{xx}(\theta)] = \Gamma_{xx}(\alpha)\Gamma_{xx}(\theta) \left\{ 1 + \left[\frac{\sin \pi(\theta + \alpha)N}{N \sin \pi(\theta + \alpha)} \right]^2 + \left[\frac{\sin \pi(\theta - \alpha)N}{N \sin \pi(\theta - \alpha)} \right]^2 \right\} \quad (14.2.35)$$

Substitution of (14.2.35) into (14.2.34) yields

$$\begin{aligned} E\{[P_{xx}^{\text{BT}}(f)]^2\} &= \left[\int_{-1/2}^{1/2} \Gamma_{xx}(\theta)W(f - \theta)d\theta \right]^2 \\ &+ \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha)\Gamma_{xx}(\theta)W(f - \alpha)W(f - \theta) \\ &\times \left\{ \left[\frac{\sin \pi(\theta + \alpha)N}{N \sin \pi(\theta + \alpha)} \right]^2 + \left[\frac{\sin \pi(\theta - \alpha)N}{N \sin \pi(\theta - \alpha)} \right]^2 \right\} d\alpha d\theta \quad (14.2.36) \end{aligned}$$

The first term in (14.2.36) is simply the square of the mean of $P_{xx}^{\text{BT}}(f)$, which is to be subtracted out according to (14.2.33). This leaves the second term in (14.2.36), which constitutes the variance. For the case in which $N \gg M$, the functions $\sin \pi(\theta + \alpha)N/N \sin \pi(\theta + \alpha)$ and $\sin \pi(\theta - \alpha)N/N \sin \pi(\theta - \alpha)$ are relatively narrow compared to $W(f)$ in the vicinity of $\theta = -\alpha$ and $\theta = \alpha$, respectively. Therefore,

$$\begin{aligned} \int_{-1/2}^{1/2} \Gamma_{xx}(\theta)W(f - \theta) &\left\{ \left[\frac{\sin \pi(\theta + \alpha)N}{N \sin \pi(\theta + \alpha)} \right]^2 + \left[\frac{\sin \pi(\theta - \alpha)N}{N \sin \pi(\theta - \alpha)} \right]^2 \right\} d\theta \\ &\approx \frac{\Gamma_{xx}(-\alpha)W(f + \alpha) + \Gamma_{xx}(\alpha)W(f - \alpha)}{N} \quad (14.2.37) \end{aligned}$$

With this approximation, the variance of $P_{xx}^{\text{BT}}(f)$ becomes

$$\begin{aligned} \text{var}[P_{xx}^{\text{BT}}(f)] &\approx \frac{1}{N} \int_{-1/2}^{1/2} \Gamma_{xx}(\alpha)W(f - \alpha)[\Gamma_{xx}(-\alpha)W(f + \alpha) + \Gamma_{xx}(\alpha)W(f - \alpha)]d\alpha \\ &\approx \frac{1}{N} \int_{-1/2}^{1/2} \Gamma_{xx}^2(\alpha)W^2(f - \alpha)d\alpha \quad (14.2.38) \end{aligned}$$

where in the last step we made the approximation

$$\int_{-1/2}^{1/2} \Gamma_{xx}(\alpha)\Gamma_{xx}(-\alpha)W(f - \alpha)W(f + \alpha)d\alpha \approx 0 \quad (14.2.39)$$

We shall make one additional approximation in (14.2.38). When $W(f)$ is narrow compared to the true power spectrum $\Gamma_{xx}(f)$, (14.2.38) is further approximated as

$$\begin{aligned}\text{var}[P_{xx}^{\text{BT}}(f)] &\approx \Gamma_{xx}^2(f) \left[\frac{1}{N} \int_{-1/2}^{1/2} W^2(\theta) d\theta \right] \\ &\approx \Gamma_{xx}^2(f) \left[\frac{1}{N} \sum_{m=-(M-1)}^{M-1} w^2(m) \right]\end{aligned}\quad (14.2.40)$$

14.2.4 Performance Characteristics of Nonparametric Power Spectrum Estimators

In this section we compare the quality of the Bartlett, Welch, and Blackman and Tukey power spectrum estimates. As a measure of quality, we use the ratio of its variance to the square of the mean of the power spectrum estimate, that is,

$$Q_A = \frac{\{E[P_{xx}^A(f)]\}^2}{\text{var}[P_{xx}^A(f)]} \quad (14.2.41)$$

where $A = B$, W , or BT for the three power spectrum estimates. The reciprocal of this quantity, called the *variability*, can also be used as a measure of performance.

For reference, the periodogram has a mean and variance

$$E[P_{xx}(f)] = \int_{-1/2}^{1/2} \Gamma_{xx}(\theta) W_B(f - \theta) d\theta \quad (14.2.42)$$

$$\text{var}[P_{xx}(f)] = \Gamma_{xx}^2(f) \left[1 + \left(\frac{\sin 2\pi f N}{N \sin 2\pi f} \right)^2 \right] \quad (14.2.43)$$

where

$$W_B(f) = \frac{1}{N} \left(\frac{\sin \pi f N}{\sin \pi f} \right)^2 \quad (14.2.44)$$

For large N (i.e., $N \rightarrow \infty$),

$$\begin{aligned}E[P_{xx}(f)] &\rightarrow \Gamma_{xx}(f), \quad \int_{-1/2}^{1/2} W_B(\theta) d\theta = w_B(0) \Gamma_{xx}(f) = \Gamma_{xx}(f) \\ \text{var}[P_{xx}(f)] &\rightarrow \Gamma_{xx}^2(f)\end{aligned}\quad (14.2.45)$$

Hence, as indicated previously, the periodogram is an asymptotically unbiased estimate of the power spectrum, but it is not consistent because its variance does not approach zero as N increases toward infinity.

Asymptotically, the periodogram is characterized by the quality factor

$$Q_P = \frac{\Gamma_{xx}^2(f)}{\Gamma_{xx}^2(f)} = 1 \quad (14.2.46)$$

The fact that Q_P is fixed and independent of the data length N is another indication of the poor quality of this estimate.

Bartlett power spectrum estimate. The mean and variance of the Bartlett power spectrum estimate are

$$E[P_{xx}^B(f)] = \int_{-1/2}^{1/2} \Gamma_{xx}(\theta) W_B(f - \theta) d\theta \quad (14.2.47)$$

$$\text{var}[P_{xx}^B(f)] = \frac{1}{K} \Gamma_{xx}^2(f) \left[1 + \left(\frac{\sin 2\pi f M}{M \sin 2\pi f} \right)^2 \right] \quad (14.2.48)$$

and

$$W_B(f) = \frac{1}{M} \left(\frac{\sin \pi f M}{\sin \pi f} \right)^2 \quad (14.2.49)$$

As $N \rightarrow \infty$ and $M \rightarrow \infty$, while $K = N/M$ remains fixed, we find that

$$\begin{aligned} E[P_{xx}^B(f)] &\rightarrow \Gamma_{xx}(f), \quad \int_{-1/2}^{1/2} W_B(f) df = \Gamma_{xx}(f) w_B(0) = \Gamma_{xx}(f) \\ \text{var}[P_{xx}^B(f)] &\rightarrow \frac{1}{K} \Gamma_{xx}^2(f) \end{aligned} \quad (14.2.50)$$

We observe that the Bartlett power spectrum estimate is asymptotically unbiased and if K is allowed to increase with an increase in N , the estimate is also consistent. Hence, asymptotically, this estimate is characterized by the quality factor

$$Q_B = K = \frac{N}{M} \quad (14.2.51)$$

The frequency resolution of the Bartlett estimate, measured by taking the 3-dB width of the main lobe of the rectangular window, is

$$\Delta f = \frac{0.9}{M} \quad (14.2.52)$$

Hence, $M = 0.9/\Delta f$ and the quality factor becomes

$$Q_B = \frac{N}{0.9/\Delta f} = 1.1N\Delta f \quad (14.2.53)$$

Welch power spectrum estimate. The mean and variance of the Welch power spectrum estimate are

$$E[P_{xx}^W(f)] = \int_{-1/2}^{1/2} \Gamma_{xx}(\theta) W(f - \theta) d\theta \quad (14.2.54)$$

where

$$W(f) = \frac{1}{MU} \left| \sum_{n=0}^{M-1} w(n) e^{-j2\pi f n} \right|^2 \quad (14.2.55)$$

and

$$\text{var}[P_{xx}^W(f)] = \begin{cases} \frac{1}{L} \Gamma_{xx}^2(f), & \text{for no overlap} \\ \frac{9}{8L} \Gamma_{xx}^2(f), & \text{for 50% overlap and} \\ & \text{triangular window} \end{cases} \quad (14.2.56)$$

As $N \rightarrow \infty$ and $M \rightarrow \infty$, the mean converges to

$$E[P_{xx}^W(f)] \rightarrow \Gamma_{xx}(f) \quad (14.2.57)$$

and the variance converges to zero, so that the estimate is consistent.

Under the two conditions given by (14.2.56) the quality factor is

$$Q_W = \begin{cases} L = \frac{N}{M}, & \text{for no overlap} \\ \frac{8L}{9} = \frac{16N}{9M}, & \text{for 50% overlap and} \\ & \text{triangular window} \end{cases} \quad (14.2.58)$$

On the other hand, the spectral width of the triangular window at the 3-dB points is

$$\Delta f = \frac{1.28}{M} \quad (14.2.59)$$

Consequently, the quality factor expressed in terms of Δf and N is

$$Q_W = \begin{cases} 0.78N\Delta f, & \text{for no overlap} \\ 1.39N\Delta f, & \text{for 50% overlap and} \\ & \text{triangular window} \end{cases} \quad (14.2.60)$$

Blackman–Tukey power spectrum estimate. The mean and variance of this estimate are approximated as

$$\begin{aligned} E[P_{xx}^{\text{BT}}(f)] &\approx \int_{-1/2}^{1/2} \Gamma_{xx}(\theta) W(f - \theta) d\theta \\ \text{var}[P_{xx}^{\text{BT}}(f)] &\approx \Gamma_{xx}^2(f) \left[\frac{1}{N} \sum_{m=-(M-1)}^{M-1} w^2(m) \right] \end{aligned} \quad (14.2.61)$$

where $w(m)$ is the window sequence used to taper the estimated autocorrelation sequence. For the rectangular and Bartlett (triangular) windows we have

$$\frac{1}{N} \sum_{n=-(M-1)}^{M-1} w^2(n) = \begin{cases} 2M/N, & \text{rectangular window} \\ 2M/3N, & \text{triangular window} \end{cases} \quad (14.2.62)$$

It is clear from (14.2.61) that the mean value of the estimate is asymptotically unbiased. Its quality factor for the triangular window is

$$Q_{\text{BT}} = 1.5 \frac{N}{M} \quad (14.2.63)$$

TABLE 14.1 Quality of Power Spectrum Estimates

Estimate	Quality Factor
Bartlett	$1.11N\Delta f$
Welch (50% overlap)	$1.39N\Delta f$
Blackman-Tukey	$2.34N\Delta f$

Since the window length is $2M - 1$, the frequency resolution measured at the 3-dB points is

$$\Delta f = \frac{1.28}{2M} = \frac{0.64}{M} \quad (14.2.64)$$

and hence

$$Q_{BT} = \frac{1.5}{0.64} N \Delta f = 2.34 N \Delta f \quad (14.2.65)$$

These results are summarized in Table 14.1. It is apparent from the results we have obtained that the Welch and Blackman-Tukey power spectrum estimates are somewhat better than the Bartlett estimate. However, the differences in performance are relatively small. The main point is that the quality factor increases with an increase in the length N of the data. This characteristic behavior is not shared by the periodogram estimate. Furthermore, the quality factor depends on the product of the data length N and the frequency resolution Δf . For a desired level of quality, Δf can be decreased (frequency resolution increased) by increasing the length N of the data, and vice versa.

14.2.5 Computational Requirements of Nonparametric Power Spectrum Estimates

The other important aspect of the nonparametric power spectrum estimates is their computational requirements. For this comparison we assume the estimates are based on a fixed amount of data N and a specified resolution Δf . The radix-2 FFT algorithm is assumed in all the computations. We shall count only the number of complex multiplications required to compute the power spectrum estimate.

Bartlett power spectrum estimate.

$$\text{FFT length} = M = 0.9/\Delta f$$

$$\text{Number of FFTs} = \frac{N}{M} = 1.11N\Delta f$$

$$\text{Number of computations} = \frac{N}{M} \left(\frac{M}{2} \log_2 M \right) = \frac{N}{2} \log_2 \frac{0.9}{\Delta f}$$

Welch power spectrum estimate (50% overlap).

$$\text{FFT length} = M = 1.28/\Delta f$$

$$\text{Number of FFTs} = \frac{2N}{M} = 1.56N\Delta f$$

$$\text{Number of computations} = \frac{2N}{M} \left(\frac{M}{2} \log_2 M \right) = N \log_2 \frac{1.28}{\Delta f}$$

In addition to the $2N/M$ FFTs, there are additional multiplications required for windowing the data. Each data record requires M multiplications. Therefore, the total number of computations is

$$\text{Total computations} = 2N + N \log_2 \frac{1.28}{\Delta f} = N \log_2 \frac{5.12}{\Delta f}$$

Blackman–Tukey power spectrum estimate. In the Blackman–Tukey method, the autocorrelation $r_{xx}(m)$ can be computed efficiently via the FFT algorithm. However, if the number of data points is large, it may not be possible to compute one N -point DFT. For example, we may have $N = 10^5$ data points but only the capacity to perform 1024-point DFTs. Since the autocorrelation sequence is windowed to $2M - 1$ points where $M \ll N$, it is possible to compute the desired $2M - 1$ points of $r_{xx}(m)$ by segmenting the data into $K = N/2M$ records and then computing $2M$ -point DFTs and one $2M$ -point IDFT via the FFT algorithm. Rader (1970) has described a method for performing this computation (see Problem 14.7).

If we base the computational complexity of the Blackman–Tukey method on this approach, we obtain the following computational requirements.

$$\text{FFT length} = 2M = 1.28/\Delta f$$

$$\text{Number of FFTs} = 2K + 1 = 2 \left(\frac{N}{2M} \right) + 1 \approx \frac{N}{M}$$

$$\text{Number of computations} = \frac{N}{M} (M \log_2 2M) = N \log_2 \frac{1.28}{\Delta f}$$

We can neglect the additional M multiplications required to window the autocorrelation sequence $r_{xx}(m)$, since it is a relatively small number. Finally, there is the additional computation required to perform the Fourier transform of the windowed autocorrelation sequence. The FFT algorithm can be used for this computation with some zero padding for purposes of interpolating the spectral estimate. As a result of these additional computations, the number of computations is increased by a small amount.

From these results we conclude that the Welch method requires a little more computational power than do the other two methods. The Bartlett method apparently requires the smallest number of computations. However, the differences in the computational requirements of the three methods are relatively small.

14.3 Parametric Methods for Power Spectrum Estimation

The nonparametric power spectrum estimation methods described in the preceding section are relatively simple, well understood, and easy to compute using the FFT algorithm. However, these methods require the availability of long data records in order to obtain the necessary frequency resolution required in many applications. Furthermore, these methods suffer from spectral leakage effects, due to windowing, that are inherent in finite-length data records. Often, the spectral leakage masks weak signals that are present in the data.

From one point of view, the basic limitation of the nonparametric methods is the inherent assumption that the autocorrelation estimate $r_{xx}(m)$ is zero for $m \geq N$, as implied by (14.1.33). This assumption severely limits the frequency resolution and the quality of the power spectrum estimate that is achieved. From another viewpoint, the inherent assumption in the periodogram estimate is that the data are periodic with period N . Neither one of these assumptions is realistic.

In this section we describe power spectrum estimation methods that do not require such assumptions. In fact, these methods *extrapolate* the values of the autocorrelation for lags $m \geq N$. Extrapolation is possible if we have some *a priori* information on how the data were generated. In such a case a model for the signal generation can be constructed with a number of parameters that can be estimated from the observed data. From the model and the estimated parameters, we can compute the power density spectrum implied by the model.

In effect, the modeling approach eliminates the need for window functions and the assumption that the autocorrelation sequence is zero for $|m| \geq N$. As a consequence, *parametric* (model-based) power spectrum estimation methods avoid the problem of leakage and provide better frequency resolution than do the FFT-based, nonparametric methods described in the preceding section. This is especially true in applications where short data records are available due to time-variant or transient phenomena.

The parametric methods considered in this section are based on modeling the data sequence $x(n)$ as the output of a linear system characterized by a rational system function of the form

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^q b_k z^{-k}}{1 + \sum_{k=1}^p a_k z^{-k}} \quad (14.3.1)$$

The corresponding difference equation is

$$x(n) = -\sum_{k=1}^p a_k x(n-k) + \sum_{k=0}^q b_k w(n-k) \quad (14.3.2)$$

where $w(n)$ is the input sequence to the system and the observed data, $x(n)$, represent the output sequence.

In power spectrum estimation, the input sequence is not observable. However, if the observed data are characterized as a stationary random process, then the input sequence is also assumed to be a stationary random process. In such a case the power density spectrum of the data is

$$\Gamma_{xx}(f) = |H(f)|^2 \Gamma_{ww}(f)$$

where $\Gamma_{ww}(f)$ is the power density spectrum of the input sequence and $H(f)$ is the frequency response of the model.

Since our objective is to estimate the power density spectrum $\Gamma_{xx}(f)$, it is convenient to assume that the input sequence $w(n)$ is a zero-mean white noise sequence with autocorrelation

$$\gamma_{ww}(m) = \sigma_w^2 \delta(m)$$

where σ_w^2 is the variance (i.e., $\sigma_w^2 = E[|w(n)|^2]$). Then the power density spectrum of the observed data is simply

$$\Gamma_{xx}(f) = \sigma_w^2 |H(f)|^2 = \sigma_w^2 \frac{|B(f)|^2}{|A(f)|^2} \quad (14.3.3)$$

In Section 12.2 we described the representation of a stationary random process as given by (14.3.3).

In the model-based approach, the spectrum estimation procedure consists of two steps. Given the data sequence $x(n)$, $0 \leq n \leq N - 1$, we estimate the parameters $\{a_k\}$ and $\{b_k\}$ of the model. Then from these estimates, we compute the power spectrum estimate according to (14.3.3).

Recall that the random process $x(n)$ generated by the pole-zero model in (14.3.1) or (14.3.2) is called an *autoregressive moving average* (ARMA) process of order (p, q) and it is usually denoted as ARMA (p, q) . If $q = 0$ and $b_0 = 1$, the resulting system model has a system function $H(z) = 1/A(z)$ and its output $x(n)$ is called an *autoregressive* (AR) process of order p . This is denoted as AR(p). The third possible model is obtained by setting $A(z) = 1$, so that $H(z) = B(z)$. Its output $x(n)$ is called a *moving average* (MA) process of order q and denoted as MA(q).

Of these three linear models the AR model is by far the most widely used. The reasons are twofold. First, the AR model is suitable for representing spectra with narrow peaks (resonances). Second, the AR model results in very simple linear equations for the AR parameters. On the other hand, the MA model, as a general rule, requires many more coefficients to represent a narrow spectrum. Consequently, it is rarely used by itself as a model for spectrum estimation. By combining poles and zeros, the ARMA model provides a more efficient representation, from the viewpoint of the number of model parameters, of the spectrum of a random process.

The decomposition theorem due to Wold (1938) asserts that any ARMA or MA process can be represented uniquely by an AR model of possibly infinite order, and any ARMA or AR process can be represented by an MA model of possibly infinite order. In view of this theorem, the issue of model selection reduces to selecting the model that requires the smallest number of parameters that are also easy to compute.

Usually, the choice in practice is the AR model. The ARMA model is used to a lesser extent.

Before describing methods for estimating the parameters in $\text{AR}(p)$, $\text{MA}(q)$, and $\text{ARMA}(p, q)$ models, it is useful to establish the basic relationships between the model parameters and the autocorrelation sequence $\gamma_{xx}(m)$. In addition, we relate the AR model parameters to the coefficients in a linear predictor for the process $x(n)$.

14.3.1 Relationships Between the Autocorrelation and the Model Parameters

In Section 12.2.2 we established the basic relationships between the autocorrelation $\{\gamma_{xx}(m)\}$ and the model parameters $\{a_k\}$ and $\{b_k\}$. For the $\text{ARMA}(p, q)$ process, the relationship given by (12.2.18) is

$$\gamma_{xx}(m) = \begin{cases} - \sum_{k=1}^p a_k \gamma_{xx}(m-k), & m > q \\ - \sum_{k=1}^p a_k \gamma_{xx}(m-k) + \sigma_w^2 \sum_{k=0}^{q-m} h(k) b_{k+m}, & 0 \leq m \leq q \\ \gamma_{xx}^*(-m), & m < 0 \end{cases} \quad (14.3.4)$$

The relationships in (14.3.4) provide a formula for determining the model parameters $\{a_k\}$ by restricting our attention to the case $m > q$. Thus the set of linear equations

$$\begin{bmatrix} \gamma_{xx}(q) & \gamma_{xx}(q-1) & \cdots & \gamma_{xx}(q-p+1) \\ \gamma_{xx}(q+1) & \gamma_{xx}(q) & \cdots & \gamma_{xx}(q+p+2) \\ \vdots & \vdots & & \\ \gamma_{xx}(q+p-1) & \gamma_{xx}(q+p-2) & \cdots & \gamma_{xx}(q) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = - \begin{bmatrix} \gamma_{xx}(q+1) \\ \gamma_{xx}(q+2) \\ \vdots \\ \gamma_{xx}(q+p) \end{bmatrix} \quad (14.3.5)$$

can be used to solve for the model parameters $\{a_k\}$ by using estimates of the autocorrelation sequence in place of $\gamma_{xx}(m)$ for $m \geq q$. This problem is discussed in Section 14.3.8.

Another interpretation of the relationship in (14.3.5) is that the values of the autocorrelation $\gamma_{xx}(m)$ for $m > q$ are uniquely determined from the pole parameters $\{a_k\}$ and the values of $\gamma_{xx}(m)$ for $0 \leq m \leq p$. Consequently, the linear system model automatically extends the values of the autocorrelation sequence $\gamma_{xx}(m)$ for $m > p$.

If the pole parameters $\{a_k\}$ are obtained from (14.3.5), the result does not help us in determining the MA parameters $\{b_k\}$, because the equation

$$\sigma_w^2 \sum_{k=0}^{q-m} h(k) b_{k+m} = \gamma_{xx}(m) + \sum_{k=1}^p a_k \gamma_{xx}(m-k), \quad 0 \leq m \leq q$$

depends on the impulse response $h(n)$. Although the impulse response can be expressed in terms of the parameters $\{b_k\}$ by long division of $B(z)$ with the known $A(z)$, this approach results in a set of nonlinear equations for the MA parameters.

If we adopt an AR(p) model for the observed data, the relationship between the AR parameters and the autocorrelation sequence is obtained by setting $q = 0$ in (14.3.4). Thus we obtain

$$\gamma_{xx}(m) = \begin{cases} -\sum_{k=1}^p a_k \gamma_{xx}(m-k), & m > 0 \\ -\sum_{k=1}^p a_k \gamma_{xx}(m-k) + \sigma_w^2, & m = 0 \\ \gamma_{xx}^*(-m), & m < 0 \end{cases} \quad (14.3.6)$$

In this case, the AR parameters $\{a_k\}$ are obtained from the solution of the Yule–Walker or normal equations

$$\begin{bmatrix} \gamma_{xx}(0) & \gamma_{xx}(-1) & \vdots & \gamma_{xx}(-p+1) \\ \gamma_{xx}(1) & \gamma_{xx}(0) & \vdots & \gamma_{xx}(-p+2) \\ \dots & \dots & \vdots & \vdots \\ \gamma_{xx}(p-1) & \gamma_{xx}(p-2) & \vdots & \gamma_{xx}(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = - \begin{bmatrix} \gamma_{xx}(1) \\ \gamma_{xx}(2) \\ \vdots \\ \gamma_{xx}(p) \end{bmatrix} \quad (14.3.7)$$

and the variance σ_w^2 can be obtained from the equation

$$\sigma_w^2 = \gamma_{xx}(0) + \sum_{k=1}^p a_k \gamma_{xx}(-k) \quad (14.3.8)$$

The equations in (14.3.7) and (14.3.8) are usually combined into a single matrix equation of the form

$$\begin{bmatrix} \gamma_{xx}(0) & \gamma_{xx}(-1) & \cdots & \gamma_{xx}(-p) \\ \gamma_{xx}(1) & \gamma_{xx}(0) & \cdots & \gamma_{xx}(-p+1) \\ \vdots & \vdots & & \vdots \\ \gamma_{xx}(p) & \gamma_{xx}(p-1) & \cdots & \gamma_{xx}(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma_w^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (14.3.9)$$

Since the correlation matrix in (14.3.7), or in (14.3.9), is Toeplitz, it can be efficiently inverted by use of the Levinson–Durbin algorithm.

Thus all the system parameters in the AR(p) model are easily determined from knowledge of the autocorrelation sequence $\gamma_{xx}(m)$ for $0 \leq m \leq p$. Furthermore, (14.3.6) can be used to extend the autocorrelation sequence for $m > p$, once the $\{a_k\}$ are determined.

Finally, for completeness, we indicate that in an MA(q) model for the observed data, the autocorrelation sequence $\gamma_{xx}(m)$ is related to the MA parameters $\{b_k\}$ by the equation

$$\gamma_{xx}(m) = \begin{cases} \sigma_w^2 \sum_{k=0}^q b_k b_{k+m}, & 0 \leq m \leq q \\ 0, & m > q \\ \gamma_{xx}^*(-m), & m < 0 \end{cases} \quad (14.3.10)$$

which was established in Section 12.2.

With this background established, we now describe the power spectrum estimation methods for the AR(p), ARMA(p, q), and MA(q) models.

14.3.2 The Yule–Walker Method for the AR Model Parameters

In the Yule–Walker method we simply estimate the autocorrelation from the data and use the estimates in (14.3.7) to solve for the AR model parameters. In this method it is desirable to use the biased form of the autocorrelation estimate,

$$r_{xx}(m) = \frac{1}{N} \sum_{n=0}^{N-m-1} x^*(n)x(n+m), \quad m \geq 0 \quad (14.3.11)$$

to ensure that the autocorrelation matrix is positive semidefinite. The result is a stable AR model. Although stability is not a critical issue in power spectrum estimation, it is conjectured that a stable AR model best represents the data.

The Levinson–Durbin algorithm described in Chapter 12 with $r_{xx}(m)$ substituted for $\gamma_{xx}(m)$ yields the AR parameters. The corresponding power spectrum estimate is

$$P_{xx}^{\text{YW}}(f) = \frac{\hat{\sigma}_{wp}^2}{|1 + \sum_{k=1}^p \hat{a}_p(k)e^{-j2\pi fk}|^2} \quad (14.3.12)$$

where $\hat{a}_p(k)$ are estimates of the AR parameters obtained from the Levinson–Durbin recursions and

$$\hat{\sigma}_{wp}^2 = \hat{E}_p^f = r_{xx}(0) \prod_{k=1}^p [1 - |\hat{a}_k(k)|^2] \quad (14.3.13)$$

is the estimated minimum mean-square value for the p th-order predictor. An example illustrating the frequency resolution capabilities of this estimator is given in Section 14.3.9.

In estimating the power spectrum of sinusoidal signals via AR models, Lacoss (1971) showed that spectral peaks in an AR spectrum estimate are proportional to the square of the power of the sinusoidal signal. On the other hand, the area under the peak in the power density spectrum is linearly proportional to the power of the sinusoid. This characteristic behavior holds for all AR model-based estimation methods.

14.3.3 The Burg Method for the AR Model Parameters

The method devised by Burg (1968) for estimating the AR parameters can be viewed as an order-recursive least-squares lattice method, based on the minimization of the forward and backward errors in linear predictors, with the constraint that the AR parameters satisfy the Levinson–Durbin recursion.

To derive the estimator, suppose that we are given the data $x(n)$, $n = 0, 1, \dots, N - 1$, and let us consider the forward and backward linear prediction estimates of order m , given as

$$\begin{aligned}\hat{x}(n) &= -\sum_{k=1}^m a_m(k)x(n-k) \\ \hat{x}(n-m) &= -\sum_{k=1}^m a_m^*(k)x(n+k-m)\end{aligned}\tag{14.3.14}$$

and the corresponding forward and backward errors $f_m(n)$ and $g_m(n)$ defined as $f_m(n) = x(n) - \hat{x}(n)$ and $g_m(n) = x(n-m) - \hat{x}(n-m)$ where $a_m(k)$, $0 \leq k \leq m-1$, $m = 1, 2, \dots, p$, are the prediction coefficients. The least-squares error is

$$\mathcal{E}_m = \sum_{n=m}^{N-1} [|f_m(n)|^2 + |g_m(n)|^2]\tag{14.3.15}$$

This error is to be minimized by selecting the prediction coefficients, subject to the constraint that they satisfy the Levinson–Durbin recursion given by

$$a_m(k) = a_{m-1}(k) + K_m a_{m-1}^*(m-k), \quad \begin{matrix} 1 \leq k \leq m-1 \\ 1 \leq m \leq p \end{matrix}\tag{14.3.16}$$

where $K_m = a_m(m)$ is the m th reflection coefficient in the lattice filter realization of the predictor. When (14.3.16) is substituted into the expressions for $f_m(n)$ and $g_m(n)$, the result is the pair of order-recursive equations for the forward and backward prediction errors given by (12.3.4).

Now, if we substitute from (12.3.4) into (14.3.16) and perform the minimization of \mathcal{E}_m with respect to the complex-valued reflection coefficient K_m , we obtain the result

$$\hat{K}_m = \frac{-\sum_{n=m}^{N-1} f_{m-1}(n) g_{m-1}^*(n-1)}{\frac{1}{2} \sum_{n=m}^{N-1} [|f_{m-1}(n)|^2 + |g_{m-1}(n-1)|^2]}, \quad m = 1, 2, \dots, p\tag{14.3.17}$$

The term in the numerator of (14.3.17) is an estimate of the crosscorrelation between the forward and backward prediction errors. With the normalization factors in the denominator of (14.3.17), it is apparent that $|\hat{K}_m| < 1$, so that the all-pole model

obtained from the data is stable. The reader should note the similarity of (14.3.17) to its statistical counterparts given by (12.3.28).

We note that the denominator in (14.3.17) is simply the least-squares estimate of the forward and backward errors, E_{m-1}^f and E_{m-1}^b , respectively. Hence (14.3.17) can be expressed as

$$\hat{K}_m = \frac{-\sum_{n=m}^{N-1} f_{m-1}(n)g_{m-1}^*(n-1)}{\frac{1}{2}[\hat{E}_{m-1}^f + \hat{E}_{m-1}^b]}, \quad m = 1, 2, \dots, p \quad (14.3.18)$$

where $\hat{E}_{m-1}^f + \hat{E}_{m-1}^b$ is an estimate of the total squared error E_m . We leave it as an exercise for the reader to verify that the denominator term in (14.3.18) can be computed in an order-recursive fashion according to the relation

$$\hat{E}_m = (1 - |\hat{K}_m|^2)\hat{E}_{m-1} - |f_{m-1}(m-1)|^2 - |g_{m-1}(m-2)|^2 \quad (14.3.19)$$

where $\hat{E}_m \equiv \hat{E}_m^f + \hat{E}_m^b$ is the total least-squares error. This result is due to Anderson (1978).

To summarize, the Burg algorithm computes the reflection coefficients in the equivalent lattice structure as specified by (14.3.18) and (14.3.19), and the Levinson-Durbin algorithm is used to obtain the AR model parameters. From the estimates of the AR parameters, we form the power spectrum estimate

$$P_{xx}^{\text{BU}}(f) = \frac{\hat{E}_p}{\left| 1 + \sum_{k=1}^p \hat{a}_p(k)e^{-j2\pi fk} \right|^2} \quad (14.3.20)$$

The major advantages of the Burg method for estimating the parameters of the AR model are (1) it results in high frequency resolution, (2) it yields a stable AR model, and (3) it is computationally efficient.

The Burg method is known to have several disadvantages, however. First, it exhibits spectral line splitting at high signal-to-noise ratios [see the paper by Fougere et al. (1976)]. By line splitting, we mean that the spectrum of $x(n)$ may have a single sharp peak, but the Burg method may result in two or more closely spaced peaks. For high-order models, the method also introduces spurious peaks. Furthermore, for sinusoidal signals in noise, the Burg method exhibits a sensitivity to the initial phase of a sinusoid, especially in short data records. This sensitivity is manifest as a frequency shift from the true frequency, resulting in a phase-dependent frequency bias. For more details on some of these limitations the reader is referred to the papers of Chen and Stegen (1974), Ulrych and Clayton (1976), Fougere et al. (1976), Kay and Marple (1979), Swingler (1979a, 1980), Herring (1980), and Thorvaldsen (1981).

Several modifications have been proposed to overcome some of the more important limitations of the Burg method: namely, the line splitting, spurious peaks,

and frequency bias. Basically, the modifications involve the introduction of a weighting (window) sequence on the squared forward and backward errors. That is, the least-squares optimization is performed on the weighted squared errors

$$\mathcal{E}_m^{\text{WB}} = \sum_{n=m}^{N-1} w_m(n)[|f_m(n)|^2 + |g_m(n)|^2] \quad (14.3.21)$$

which, when minimized, results in the reflection coefficient estimates

$$\hat{K}_m = \frac{-\sum_{n=m}^{N-1} w_{m-1}(n) f_{m-1}(n) g_{m-1}^*(n-1)}{\frac{1}{2} \sum_{n=m}^{N-1} w_{m-1}(n)[|f_{m-1}(n)|^2 + |g_{m-1}(n-1)|^2]} \quad (14.3.22)$$

In particular, we mention the use of a Hamming window used by Swingler (1979b), a quadratic or parabolic window used by Kaveh and Lippert (1983), the energy weighting method used by Nikias and Scott (1982), and the data-adaptive energy weighting used by Helme and Nikias (1985).

These windowing and energy weighting methods have proved effective in reducing the occurrence of line splitting and spurious peaks, and are also effective in reducing frequency bias.

The Burg method for power spectrum estimation is usually associated with *maximum entropy spectrum estimation*, a criterion used by Burg (1967, 1975) as a basis for AR modeling in parametric spectrum estimation. The problem considered by Burg was how best to extrapolate from the given values of the autocorrelation sequence $\gamma_{xx}(m)$, $0 \leq m \leq p$, the values for $m > p$, such that the entire autocorrelation sequence is positive semidefinite. Since an infinite number of extrapolations are possible, Burg postulated that the extrapolations be made on the basis of maximizing uncertainty (entropy) or randomness, in the sense that the spectrum $\Gamma_{xx}(f)$ of the process is the flattest of all spectra which have the given autocorrelation values $\gamma_{xx}(m)$, $0 \leq m \leq p$. In particular the entropy per sample is proportional to the integral [see Burg (1975)]

$$\int_{-1/2}^{1/2} \ln \Gamma_{xx}(f) df \quad (14.3.23)$$

Burg found that the maximum of this integral subject to the $(p+1)$ constraints

$$\int_{-1/2}^{1/2} \Gamma_{xx}(f) e^{j2\pi f m} df = \gamma_{xx}(m), \quad 0 \leq m \leq p \quad (14.3.24)$$

is the AR(p) process for which the given autocorrelation sequence $\gamma_{xx}(m)$, $0 \leq m \leq p$ is related to the AR parameters by the equation (14.3.6). This solution provides an additional justification for the use of the AR model in power spectrum estimation.

In view of Burg's basic work in maximum entropy spectral estimation, the Burg power spectrum estimation procedure is often called the *maximum entropy method*

(MEM). We should emphasize, however, that the maximum entropy spectrum is identical to the AR-model spectrum only when the exact autocorrelation $\gamma_{xx}(m)$ is known. When only an estimate of $\gamma_{xx}(m)$ is available for $0 \leq m \leq p$, the AR-model estimates of Yule-Walker and Burg are not maximum entropy spectral estimates. The general formulation for the maximum entropy spectrum based on estimates of the autocorrelation sequence results in a set of nonlinear equations. Solutions for the maximum entropy spectrum with measurement errors in the correlation sequence have been obtained by Newman (1981) and Schott and McClellan (1984).

14.3.4 Unconstrained Least-Squares Method for the AR Model Parameters

As described in the preceding section, the Burg method for determining the parameters of the AR model is basically a least-squares lattice algorithm with the added constraint that the predictor coefficients satisfy the Levinson recursion. As a result of this constraint, an increase in the order of the AR model requires only a single parameter optimization at each stage. In contrast to this approach, we may use an unconstrained least-squares algorithm to determine the AR parameters.

To elaborate, we form the forward and backward linear prediction estimates and their corresponding forward and backward errors as indicated in (14.3.14) and (14.3.15). Then we minimize the sum of squares of both errors, that is,

$$\begin{aligned}\mathcal{E}_p &= \sum_{n=p}^{N-1} [|f_p(n)|^2 + |g_p(n)|^2] \\ &= \sum_{n=p}^{N-1} \left[\left| x(n) + \sum_{k=1}^p a_p(k)x(n-k) \right|^2 \right. \\ &\quad \left. + \left| x(n-p) + \sum_{k=1}^p a_p^*(k)x(n+k-p) \right|^2 \right] \end{aligned} \quad (14.3.25)$$

which is the same performance index as in the Burg method. However, we do not impose the Levinson-Durbin recursion in (14.3.25) for the AR parameters. The unconstrained minimization of \mathcal{E}_p with respect to the prediction coefficients yields the set of linear equations

$$\sum_{k=1}^p a_p(k)r_{xx}(l, k) = -r_{xx}(l, 0), \quad l = 1, 2, \dots, p \quad (14.3.26)$$

where, by definition, the autocorrelation $r_{xx}(l, k)$ is

$$r_{xx}(l, k) = \sum_{n=p}^{N-1} [x(n-k)x^*(n-l) + x(n-p+l)x^*(n-p+k)] \quad (14.3.27)$$

The resulting residual least-squares error is

$$\epsilon_p^{\text{LS}} = r_{xx}(0, 0) + \sum_{k=1}^p \hat{a}_p(k) r_{xx}(0, k) \quad (14.3.28)$$

Hence the unconstrained least-squares power spectrum estimate is

$$P_{xx}^{\text{LS}}(f) = \frac{\epsilon_p^{\text{LS}}}{\left| 1 + \sum_{k=1}^p \hat{a}_p(k) e^{-j2\pi fk} \right|^2} \quad (14.3.29)$$

The correlation matrix in (14.3.27), with elements $r_{xx}(l, k)$, is not Toeplitz, so that the Levinson–Durbin algorithm cannot be applied. However, the correlation matrix has sufficient structure to make it possible to devise computationally efficient algorithms with computational complexity proportional to p^2 . Marple (1980) devised such an algorithm, which has a lattice structure and employs Levinson–Durbin-type order recursions and additional time recursions.

This form of the unconstrained least-squares method described has also been called the *unwindowed data* least-squares method. It has been proposed for spectrum estimation in several papers, including the papers by Burg (1967), Nuttall (1976), and Ulrych and Clayton (1976). Its performance characteristics have been found to be superior to the Burg method, in the sense that the unconstrained least-squares method does not exhibit the same sensitivity to such problems as line splitting, frequency bias, and spurious peaks. In view of the computational efficiency of Marple's algorithm, which is comparable to the efficiency of the Levinson–Durbin algorithm, the unconstrained least-squares method is very attractive. With this method there is no guarantee that the estimated AR parameters yield a stable AR model. However, in spectrum estimation, this is not considered to be a problem.

14.3.5 Sequential Estimation Methods for the AR Model Parameters

The three power spectrum estimation methods described in the preceding sections for the AR model can be classified as block processing methods. These methods obtain estimates of the AR parameters from a block of data, say $x(n)$, $n = 0, 1, \dots, N - 1$. The AR parameters, based on the block of N data points, are then used to obtain the power spectrum estimate.

In situations where data are available on a continuous basis, we can still segment the data into blocks of N points and perform spectrum estimation on a block-by-block basis. This is often done in practice, for both real-time and non-real-time applications. However, in such applications, there is an alternative approach based on sequential (in time) estimation of the AR model parameters as each new data point becomes available. By introducing a weighting function into past data samples, it is possible to deemphasize the effect of older data samples as new data are received.

Sequential lattice methods based on recursive least squares can be employed to optimally estimate the prediction and reflection coefficients in the lattice realization

of the forward and backward linear predictors. The recursive equations for the prediction coefficients relate directly to the AR model parameters. In addition to the order-recursive nature of these equations, as implied by the lattice structure, we can also obtain time-recursive equations for the reflection coefficients in the lattice and for the forward and backward prediction coefficients.

Sequential recursive least-squares algorithms are equivalent to the unconstrained least-squares, block processing method described in the preceding section. Hence the power spectrum estimates obtained by the sequential recursive least-squares method retain the desirable properties of the block processing algorithm described in Section 14.3.4. Since the AR parameters are being continuously estimated in a sequential estimation algorithm, power spectrum estimates can be obtained as often as desired, from once per sample to once every N samples. By properly weighting past data samples, the sequential estimation methods are particularly suitable for estimating and tracking time-variant power spectra resulting from nonstationary signal statistics.

The computational complexity of sequential estimation methods is generally proportional to p , the order of the AR process. As a consequence, sequential estimation algorithms are computationally efficient and, from this viewpoint, may offer some advantage over the block processing methods.

There are numerous references on sequential estimation methods. The papers by Griffiths (1975), Friedlander (1982a, b), and Kalouptsidis and Theodoridis (1987) are particularly relevant to the spectrum estimation problem.

14.3.6 Selection of AR Model Order

One of the most important aspects of the use of the AR model is the selection of the order p . As a general rule, if we select a model with too low an order, we obtain a highly smoothed spectrum. On the other hand, if p is selected too high, we run the risk of introducing spurious low-level peaks in the spectrum. We mentioned previously that one indication of the performance of the AR model is the mean-square value of the residual error, which, in general, is different for each of the estimators described above. The characteristic of this residual error is that it decreases as the order of the AR model is increased. We can monitor the rate of decrease and decide to terminate the process when the rate of decrease becomes relatively slow. It is apparent, however, that this approach may be imprecise and ill defined, and other methods should be investigated.

Much work has been done by various researchers on this problem and many experimental results have been given in the literature [e.g., the papers by Gersch and Sharpe (1973), Ulrych and Bishop (1975), Tong (1975, 1977), Jones (1976), Nuttall (1976), Berryman (1978), Kaveh and Bruzzone (1979), and Kashyap (1980)].

Two of the better-known criteria for selecting the model order have been proposed by Akaike (1969, 1974). With the first, called the *final prediction error (FPE) criterion*, the order is selected to minimize the performance index

$$\text{FPE}(p) = \hat{\sigma}_{wp}^2 \left(\frac{N + p + 1}{N - p - 1} \right) \quad (14.3.30)$$

where $\hat{\sigma}_{wp}^2$ is the estimated variance of the linear prediction error. This performance index is based on minimizing the mean-square error for a one-step predictor.

The second criterion proposed by Akaike (1974), called the *Akaike information criterion* (AIC), is based on selecting the order that minimizes

$$\text{AIC}(p) = \ln \hat{\sigma}_{wp}^2 + 2p/N \quad (14.3.31)$$

Note that the term $\hat{\sigma}_{wp}^2$ decreases and therefore $\ln \hat{\sigma}_{wp}^2$ also decreases as the order of the AR model is increased. However, $2p/N$ increases with an increase in p . Hence a minimum value is obtained for some p .

An alternative information criterion, proposed by Rissanen (1983), is based on selecting the order that *minimizes the description length* (MDL), where MDL is defined as

$$\text{MDL}(p) = N \ln \hat{\sigma}_{wp}^2 + p \ln N \quad (14.3.32)$$

A fourth criterion has been proposed by Parzen (1974). This is called the *criterion autoregressive transfer* (CAT) function and is defined as

$$\text{CAT}(p) = \left(\frac{1}{N} \sum_{k=1}^p \frac{1}{\bar{\sigma}_{wk}^2} \right) - \frac{1}{\hat{\sigma}_{wp}^2} \quad (14.3.33)$$

where

$$\bar{\sigma}_{wk}^2 = \frac{N}{N-k} \hat{\sigma}_{wk}^2 \quad (14.3.34)$$

The order p is selected to minimize $\text{CAT}(p)$.

In applying this criterion, the mean should be removed from the data. Since $\hat{\sigma}_{wk}^2$ depends on the type of spectrum estimate we obtain, the model order is also a function of the criterion.

The experimental results given in the references just cited indicate that the model-order selection criteria do not yield definitive results. For example, Ulrych and Bishop (1975), Jones (1976), and Berryman (1978) found that the FPE(p) criterion tends to underestimate the model order. Kashyap (1980) showed that the AIC criterion is statistically inconsistent as $N \rightarrow \infty$. On the other hand, the MDL information criterion proposed by Rissanen is statistically consistent. Other experimental results indicate that for small data lengths, the order of the AR model should be selected to be in the range $N/3$ to $N/2$ for good results. It is apparent that in the absence of any prior information regarding the physical process that resulted in the data, one should try different model orders and different criteria and, finally, consider the different results.

14.3.7 MA Model for Power Spectrum Estimation

As shown in Section 14.3.1, the parameters in an MA(q) model are related to the statistical autocorrelation $\gamma_{xx}(m)$ by (14.3.10). However,

$$B(z)B(z^{-1}) = D(z) = \sum_{m=-q}^q d_m z^{-m} \quad (14.3.35)$$

where the coefficients $\{d_m\}$ are related to the MA parameters by the expression

$$d_m = \sum_{k=0}^{q-|m|} b_k b_{k+m}, \quad |m| \leq q \quad (14.3.36)$$

Clearly, then,

$$\gamma_{xx}(m) = \begin{cases} \sigma_w^2 d_m, & |m| \leq q \\ 0, & |m| > q \end{cases} \quad (14.3.37)$$

and the power spectrum for the MA(q) process is

$$\Gamma_{xx}^{\text{MA}}(f) = \sum_{m=-q}^q \gamma_{xx}(m) e^{-j2\pi f m} \quad (14.3.38)$$

It is apparent from these expressions that we do not have to solve for the MA parameters $\{b_k\}$ to estimate the power spectrum. The estimates of the autocorrelation $\gamma_{xx}(m)$ for $|m| \leq q$ suffice. From such estimates we compute the estimated MA power spectrum, given as

$$P_{xx}^{\text{MA}}(f) = \sum_{m=-q}^q r_{xx}(m) e^{-j2\pi f m} \quad (14.3.39)$$

which is identical to the classical (nonparametric) power spectrum estimate described in Section 14.1.

There is an alternative method for determining $\{b_k\}$ based on a high-order AR approximation to the MA process. To be specific, let the MA(q) process be modeled by an AR(p) model, where $p >> q$. Then $B(z) = 1/A(z)$, or equivalently, $B(z)A(z) = 1$. Thus the parameters $\{b_k\}$ and $\{a_k\}$ are related by a convolution sum, which can be expressed as

$$\hat{a}_n + \sum_{k=1}^q b_k \hat{a}_{n-k} = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} \quad (14.3.40)$$

where $\{\hat{a}_n\}$ are the parameters obtained by fitting the data to an AR(p) model.

Although this set of equations can be easily solved for the $\{b_k\}$, a better fit is obtained by using a least-squares error criterion. That is, we form the squared error

$$\mathcal{E} = \sum_{n=0}^p \left[\hat{a}_n + \sum_{k=1}^q b_k \hat{a}_{n-k} \right]^2 - 1, \quad \hat{a}_0 = 1, \quad \hat{a}_k = 0, \quad k < 0 \quad (14.3.41)$$

which is minimized by selecting the MA(q) parameters $\{b_k\}$. The result of this minimization is

$$\hat{\mathbf{b}} = -\mathbf{R}_{aa}^{-1} \mathbf{r}_{aa} \quad (14.3.42)$$

where the elements of \mathbf{R}_{aa} and \mathbf{r}_{aa} are given as

$$\begin{aligned} R_{aa}(|i - j|) &= \sum_{n=0}^{p-|i-j|} \hat{a}_n \hat{a}_{n+|i-j|}, \quad i, j = 1, 2, \dots, q \\ r_{aa}(i) &= \sum_{n=0}^{p-i} \hat{a}_n \hat{a}_{n+i}, \quad i = 1, 2, \dots, q \end{aligned} \quad (14.3.43)$$

This least-squares method for determining the parameters of the MA(q) model is attributed to Durbin (1959). It has been shown by Kay (1988) that this estimation method is approximately maximum likelihood under the assumption that the observed process is Gaussian.

The order q of the MA model may be determined empirically by several methods. For example, the AIC for MA models has the same form as for AR models,

$$\text{AIC}(q) = \ln \sigma_{wq}^2 + \frac{2q}{N} \quad (14.3.44)$$

where σ_{wq}^2 is an estimate of the variance of the white noise. Another approach, proposed by Chow (1972b), is to filter the data with the inverse MA(q) filter and test the filtered output for whiteness.

14.3.8 ARMA Model for Power Spectrum Estimation

The Burg algorithm, its variations, and the least-squares method described in the previous sections provide reliable high-resolution spectrum estimates based on the AR model. An ARMA model provides us with an opportunity to improve on the AR spectrum estimate, perhaps, by using fewer model parameters.

The ARMA model is particularly appropriate when the signal has been corrupted by noise. For example, suppose that the data $x(n)$ are generated by an AR system, where the system output is corrupted by additive white noise. The z -transform of the autocorrelation of the resultant signal can be expressed as

$$\begin{aligned} \Gamma_{xx}(z) &= \frac{\sigma_w^2}{A(z)A(z^{-1})} + \sigma_n^2 \\ &= \frac{\sigma_w^2 + \sigma_n^2 A(z)A(z^{-1})}{A(z)A(z^{-1})} \end{aligned} \quad (14.3.45)$$

where σ_n^2 is the variance of the additive noise. Therefore, the process $x(n)$ is ARMA(p, p), where p is the order of the autocorrelation process. This relationship provides some motivation for investigating ARMA models for power spectrum estimation.

As we have demonstrated in Section 14.3.1, the parameters of the ARMA model are related to the autocorrelation by the equation in (14.3.4). For lags $|m| > q$, the equation involves only the AR parameters $\{a_k\}$. With estimates substituted in place

of $\gamma_{xx}(m)$, we can solve the p equations in (14.3.5) to obtain \hat{a}_k . For high-order models, however, this approach is likely to yield poor estimates of the AR parameters due to the poor estimates of the autocorrelation for large lags. Consequently, this approach is not recommended.

A more reliable method is to construct an overdetermined set of linear equations for $m > q$, and to use the method of least squares on the set of overdetermined equations, as proposed by Cadzow (1979). To elaborate, suppose that the autocorrelation sequence can be accurately estimated up to lag M , where $M > p + q$. Then we can write the following set of linear equations:

$$\begin{bmatrix} r_{xx}(q) & r_{xx}(q-1) & \cdots & r_{xx}(q-p+1) \\ r_{xx}(q+1) & r_{xx}(q) & \cdots & r_{xx}(q-p+2) \\ \vdots & \vdots & & \\ r_{xx}(M-1) & r_{xx}(M-2) & & r_{xx}(M-p) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = - \begin{bmatrix} r_{xx}(q+1) \\ r_{xx}(q+2) \\ \vdots \\ r_{xx}(M) \end{bmatrix} \quad (14.3.46)$$

or equivalently,

$$\mathbf{R}_{xx}\mathbf{a} = -\mathbf{r}_{xx} \quad (14.3.47)$$

Since \mathbf{R}_{xx} is of dimension $(M-q) \times p$, and $M-q > p$ we can use the least-squares criterion to solve for the parameter vector \mathbf{a} . The result of this minimization is

$$\hat{\mathbf{a}} = -(\mathbf{R}_{xx}^t \mathbf{R}_{xx})^{-1} \mathbf{R}_{xx}^t \mathbf{r}_{xx} \quad (14.3.48)$$

This procedure is called the *least-squares modified Yule–Walker method*. A weighting factor can also be applied to the autocorrelation sequence to deemphasize the less reliable estimates for large lags.

Once the parameters for the AR part of the model have been estimated as indicated above, we have the system

$$\hat{A}(z) = 1 + \sum_{k=1}^p \hat{a}_k z^{-k} \quad (14.3.49)$$

The sequence $x(n)$ can now be filtered by the FIR filter $\hat{A}(z)$ to yield the sequence

$$v(n) = x(n) + \sum_{k=1}^p \hat{a}_k x(n-k), \quad n = 0, 1, \dots, N-1 \quad (14.3.50)$$

The cascade of the ARMA(p, q) model with $\hat{A}(z)$ is approximately the MA(q) process generated by the model $B(z)$. Hence we can apply the MA estimate given in the preceding section to obtain the MA spectrum. To be specific, the filtered sequence

$v(n)$ for $p \leq n \leq N - 1$ is used to form the estimated correlation sequences $r_{vv}(m)$, from which we obtain the MA spectrum

$$P_{vv}^{\text{MA}}(f) = \sum_{m=-q}^q r_{vv}(m)e^{-j2\pi fm} \quad (14.3.51)$$

First, we observe that the parameters $\{b_k\}$ are not required to determine the power spectrum. Second, we observe that $r_{vv}(m)$ is an estimate of the autocorrelation for the MA model given by (14.3.10). In forming the estimate $r_{vv}(m)$, weighting (e.g., with the Bartlett window) may be used to deemphasize correlation estimates for large lags. In addition, the data may be filtered by a backward filter, thus creating another sequence, say $v^b(n)$, so that both $v(n)$ and $v^b(n)$ can be used in forming the estimate of the autocorrelation $r_{vv}(m)$, as proposed by Kay (1980). Finally, the estimated ARMA power spectrum is

$$\hat{P}_{xx}^{\text{ARMA}}(f) = \frac{P_{vv}^{\text{MA}}(f)}{\left|1 + \sum_{k=1}^p \hat{a}_k e^{-j2\pi fk}\right|^2} \quad (14.3.52)$$

The problem of order selection for the ARMA (p, q) model has been investigated by Chow (1972a, b) and Bruzzone and Kaveh (1980). For this purpose the minimum of the AIC index

$$\text{AIC}(p, q) = \ln \hat{\sigma}_{wpq}^2 + \frac{2(p+q)}{N} \quad (14.3.53)$$

can be used, where $\hat{\sigma}_{wpq}^2$ is an estimate of the variance of the input error. An additional test on the adequacy of a particular ARMA (p, q) model is to filter the data through the model and test for whiteness of the output data. This requires that the parameters of the MA model be computed from the estimated autocorrelation, using spectral factorization to determine $B(z)$ from $D(z) = B(z)B(z^{-1})$.

For additional reading on ARMA power spectrum estimation, the reader is referred to the papers by Graupe et al. (1975), Cadzow (1981, 1982), Kay (1980), and Friedlander (1982b).

14.3.9 Some Experimental Results

In this section we present some experimental results on the performance of AR and ARMA power spectrum estimates obtained by using artificially generated data. Our objective is to compare the spectral estimation methods on the basis of their frequency resolution, bias, and their robustness in the presence of additive noise.

The data consist of either one or two sinusoids and additive Gaussian noise. The two sinusoids are spaced Δf apart. Clearly, the underlying process is ARMA $(4, 4)$. The results that are shown employ an AR (p) model for these data. For high signal-to-noise ratios (SNRs) we expect the AR(4) to be adequate. However, for low SNRs, a higher-order AR model is needed to approximate the ARMA $(4, 4)$ process.

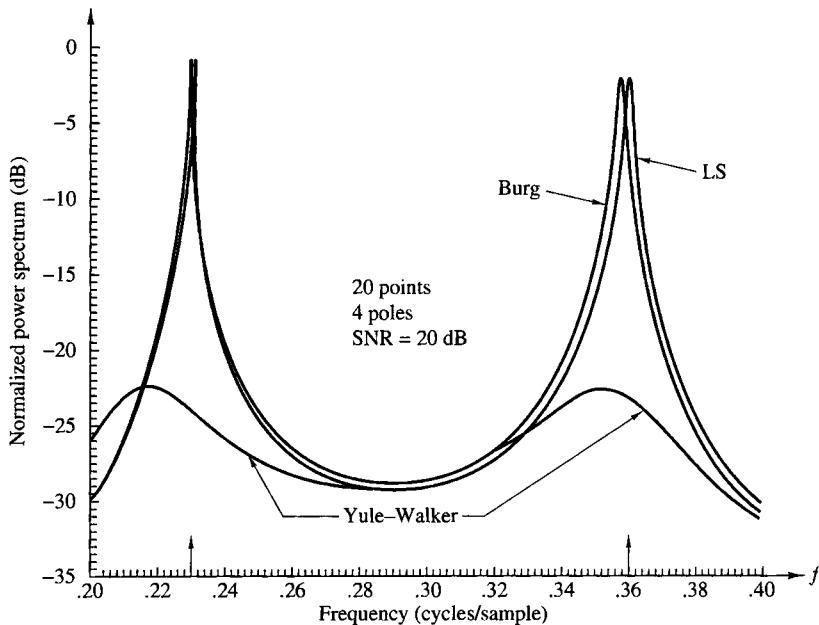


Figure 14.3.1 Comparison of AR spectrum estimation methods.

The results given below are consistent with this statement. The SNR is defined as $10 \log_{10} A^2 / 2\sigma^2$, where σ^2 is variance of the additive noise and A is the amplitude of the sinusoid.

In Fig. 14.3.1 we illustrate the results for $N = 20$ data points based on an AR(4) model with an SNR = 20 dB and $\Delta f = 0.13$. Note that the Yule–Walker method gives an extremely smooth (broad) spectral estimate with small peaks. If Δf is decreased to $\Delta f = 0.07$, the Yule–Walker method no longer resolves the peaks as illustrated in Fig. 14.3.2. Some bias is also evident in the Burg method. Of course, by increasing the number of data points the Yule–Walker method eventually is able to resolve the peaks. However, the Burg and least-squares methods are clearly superior for short data records.

The effect of additive noise on the estimate is illustrated in Fig. 14.3.3 for the least-squares method. The effect of filter order on the Burg and least-squares methods is illustrated in Figs. 14.3.4 and 14.3.5, respectively. Both methods exhibit spurious peaks as the order is increased to $p = 12$.

The effect of initial phase is illustrated in Figs. 14.3.6 and 14.3.7 for the Burg and least-squares methods. It is clear that the least-squares method exhibits less sensitivity to initial phase than the Burg algorithm.

An example of line splitting for the Burg method is shown in Fig. 14.3.8 with $p = 12$. It does not occur for the AR(8) model. The least-squares method did not exhibit line splitting under the same conditions. On the other hand, the line splitting on the Burg method disappeared with an increase in the number of data points N .

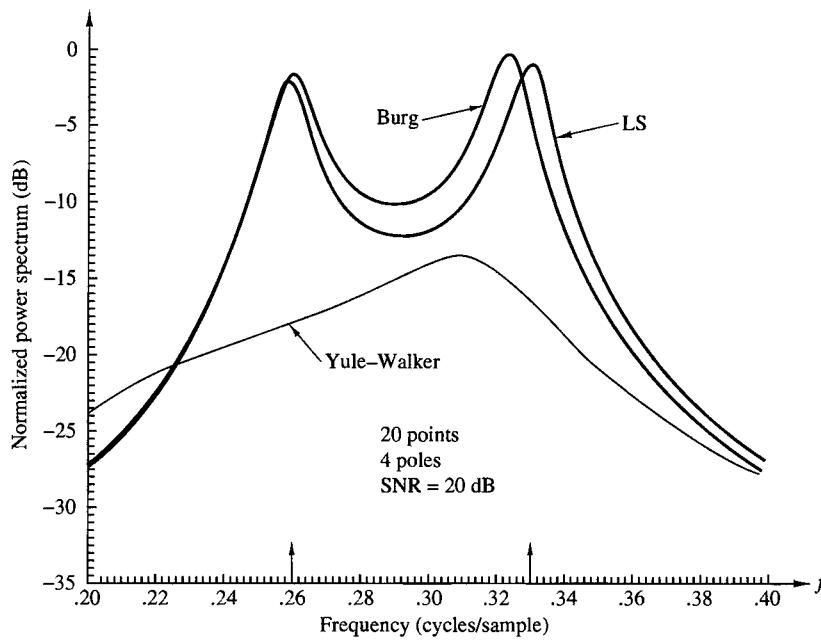


Figure 14.3.2 Comparison of AR spectrum estimation methods.

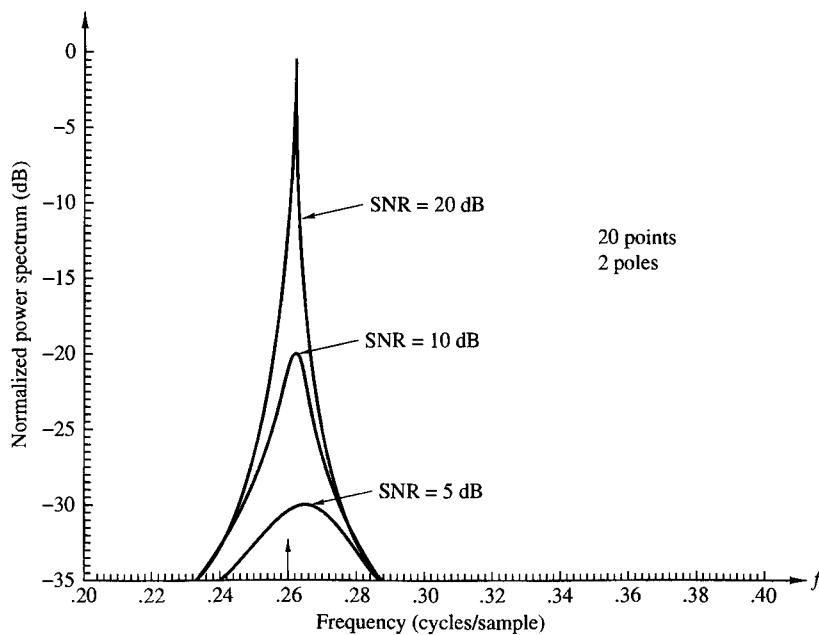


Figure 14.3.3 Effect of additive noise on LS method.

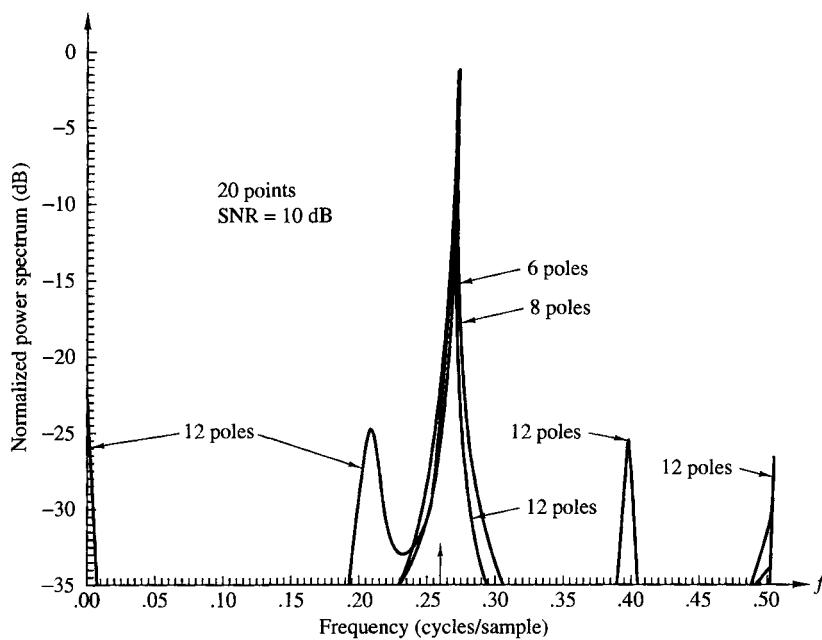


Figure 14.3.4 Effect of filter order of Burg method.

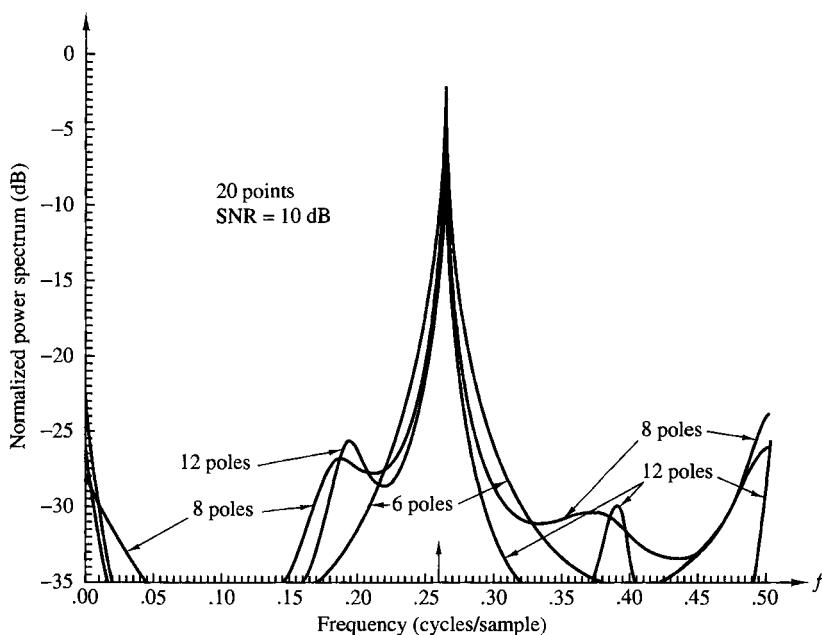


Figure 14.3.5 Effect of filter order on LS method.

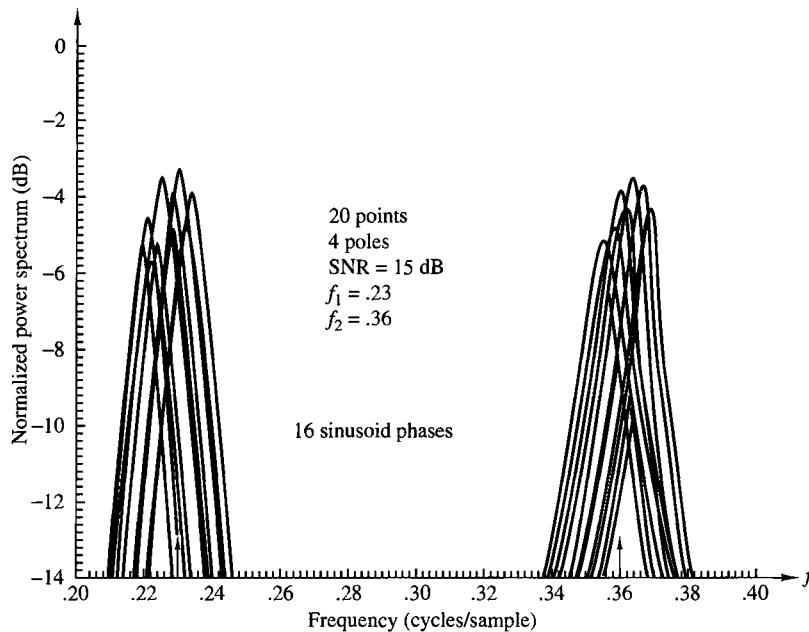


Figure 14.3.6 Effect of initial phase on Burg method.

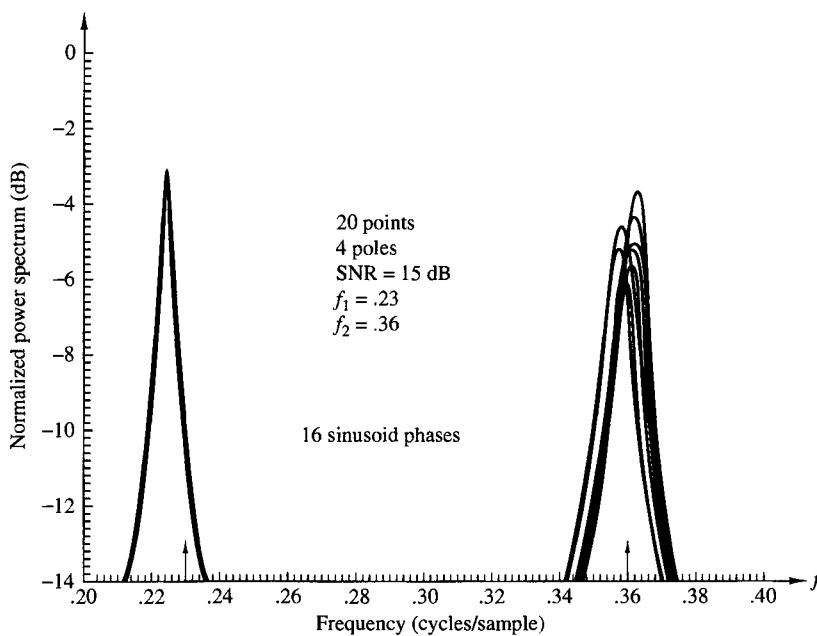


Figure 14.3.7 Effect of initial phase on LS method.

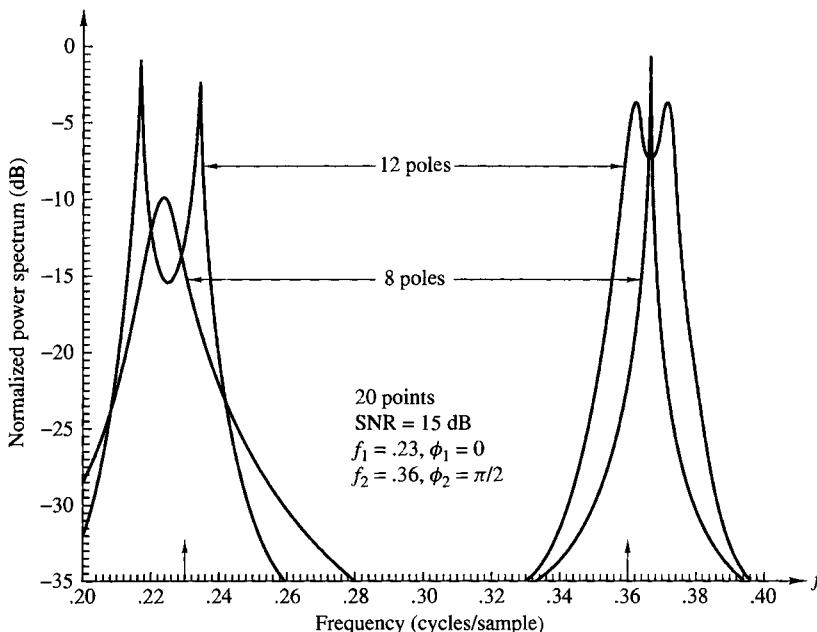


Figure 14.3.8 Line splitting in Burg method.

Figures 14.3.9 and 14.3.10 illustrate the resolution properties of the Burg and least-squares methods for $\Delta f = 0.07$ and $N = 20$ points at low SNR (3 dB). Since the additive noise process is ARMA, a higher-order AR model is required to provide a good approximation at low SNR. Hence the frequency resolution improves as the order is increased.

The FPE for the Burg method is illustrated in Fig. 14.3.11 for an SNR = 3 dB. For this SNR the optimum value is $p = 12$ according to the FPE criterion.

The Burg and least-squares methods were also tested with data from a narrowband process, obtained by exciting a four-pole (two pairs of complex-conjugate poles) narrowband filter and selecting a portion of the output sequence for the data record. Figure 14.3.12 illustrates the superposition of 20 data records of 20 points each. We observe a relatively small variability. In contrast, the Burg method exhibited a much larger variability, approximately a factor of 2 compared to the least-squares method. The results shown in Figs. 14.3.1 through 14.3.12 are taken from Poole (1981).

Finally, we show in Fig. 14.3.13 the ARMA(10, 10) spectral estimates obtained by Kay (1980) for two sinusoids in noise using the least-squares ARMA method described in Section 14.3.8, as an illustration of the quality of power spectrum estimation obtained with the ARMA model.

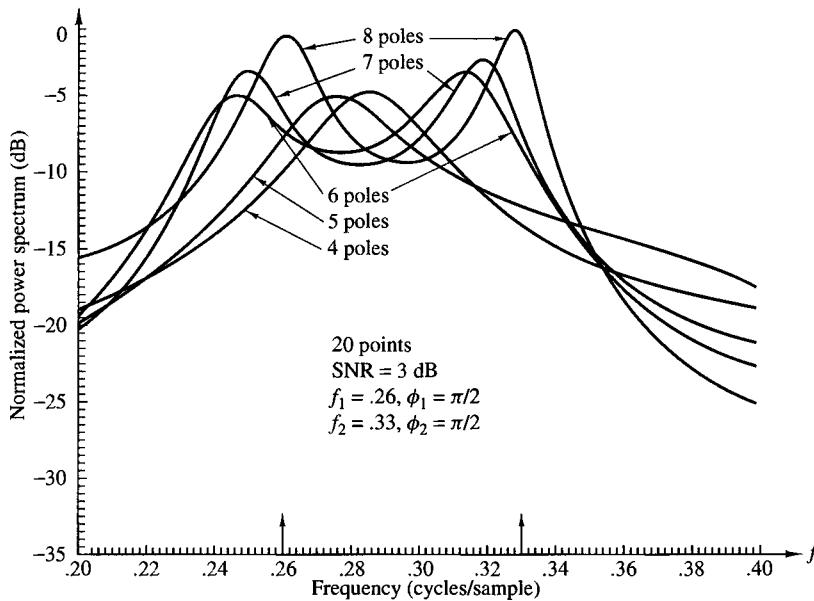


Figure 14.3.9 Frequency resolution of Burg method with $N = 20$ points.

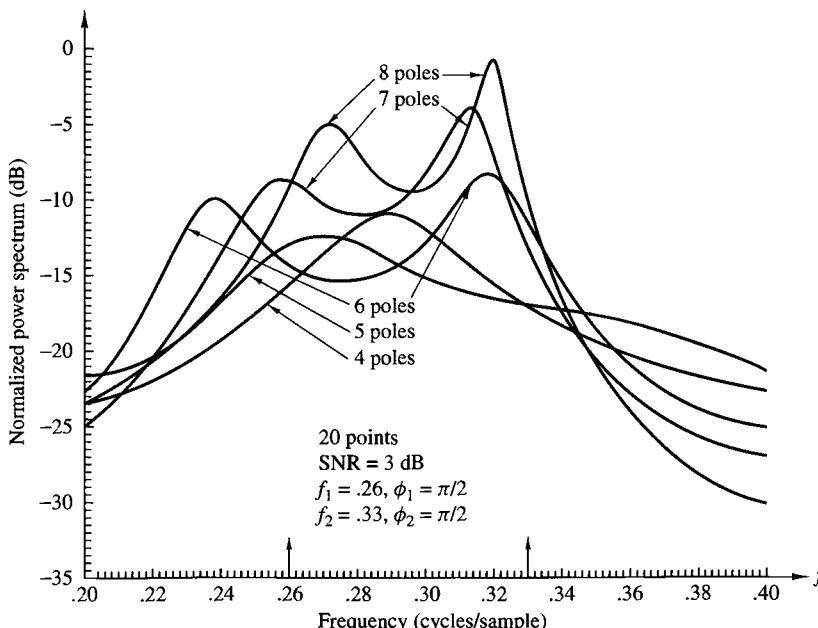


Figure 14.3.10 Frequency resolution of LS method with $N = 20$ points.

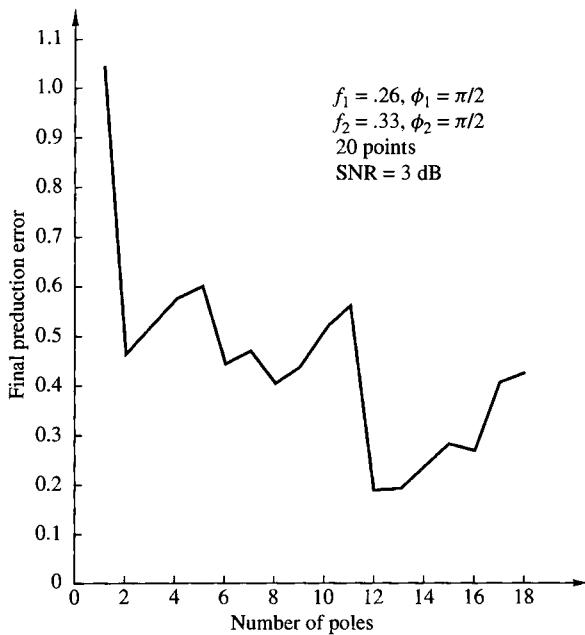


Figure 14.3.11 Final prediction error for Burg estimate.

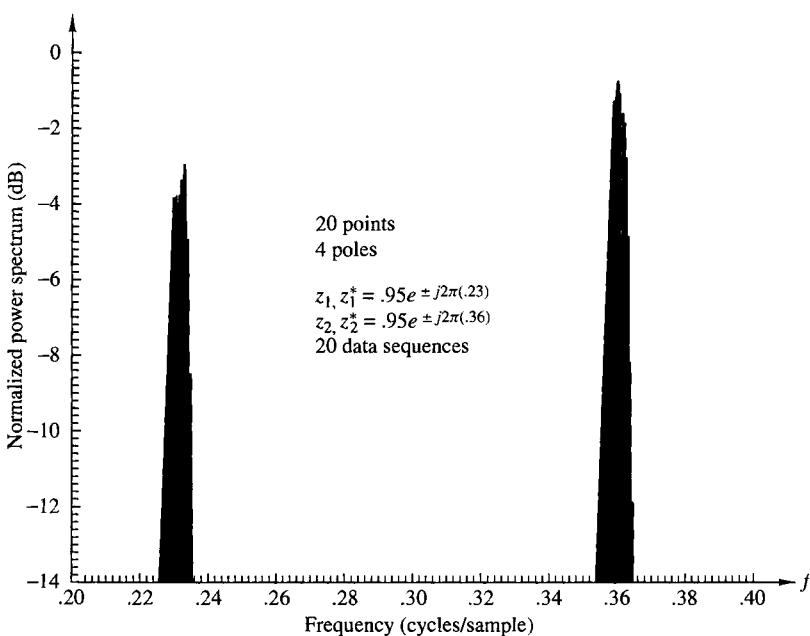


Figure 14.3.12 Effect of starting point in sequence on LS method.

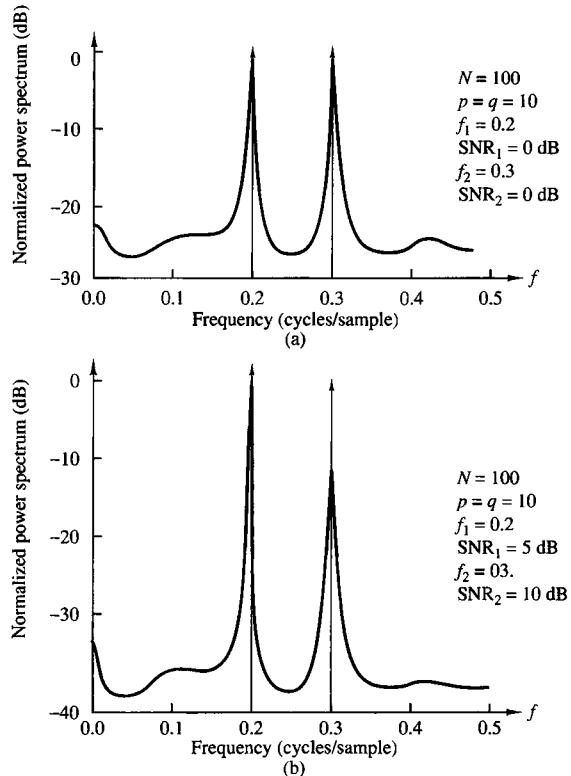


Figure 14.3.13
ARMA (10, 10) power spectrum estimates from paper by Kay (1980). Reprinted with permission from the IEEE.

14.4 Filter Bank Methods

In this section, we consider the estimation of the power spectrum of a signal sequence $\{x(n)\}$ by filtering $\{x(n)\}$ with a parallel bank of FIR filters tuned to the desired frequencies and rectifying the filter outputs. Thus, an estimate of the power spectrum at a particular frequency is obtained as illustrated in the system shown in Figure 14.4.1.

We begin by showing that the conventional periodogram can be computed by using a filter bank, where the FIR filters basically function as rectangular windows on the signal sequence $\{x(n)\}$. Then, we describe a method for designing the filters that exploit the characteristics of the data. This approach leads to a high resolution spectral estimation method that is data adaptive.

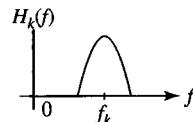
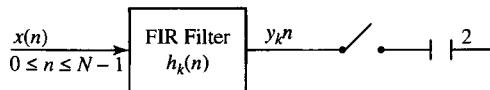


Figure 14.4.1
Measurement of signal power at a frequency in the vicinity of f_k .

14.4.1 Filter Bank Realization of the Periodogram

The power spectrum estimate at a given frequency $f_k = k/N$ that is obtained from the periodogram is

$$\begin{aligned} P_{xx}(f_k) &= \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n) e^{-j2\pi f_k n} \right|^2 \\ &= \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n) e^{j2\pi f_k (N-n)} \right|^2 \end{aligned} \quad (14.4.1)$$

where $\{x(n), 0 \leq n \leq N-1\}$ is the signal sequence. Following an approach similar to the development of the Goertzel algorithm (Section 8.3.1), we define a linear filter with impulse response

$$h_k(n) = \begin{cases} \frac{1}{\sqrt{N}} e^{j2\pi f_k n}, & n \geq 0 \\ 0, & n < 0 \end{cases} \quad (14.4.2)$$

Then, after substituting $h_k(n)$ in equation (14.4.1), we obtain the power spectral estimate at f_k as

$$\begin{aligned} P_{xx}(f) &= \left| \sum_{n=0}^{N-1} x(n) h_k(N-n) \right|^2 \\ &= \left| \sum_{n=0}^{N-1} x(n) h_k(l-n) \right|_{l=N}^2 \end{aligned} \quad (14.4.3)$$

Thus, the power spectral estimate at $f = f_k = k/N$ may be obtained by passing the signal sequence $\{x(n), 0 \leq n \leq N-1\}$ through a linear FIR filter with impulse response $h_k(n)$, as given by equation (14.4.2), evaluating the filter output at $l = N$, and computing the square magnitude of the filter output.

The system function of the filter is

$$\begin{aligned} H_k(z) &= \sum_{n=0}^{\infty} h_k(n) z^{-n} \\ &= \frac{1/\sqrt{N}}{1 - e^{j2\pi f_k} z^{-1}} \end{aligned} \quad (14.4.4)$$

This filter is a single pole filter with the pole on the unit circle, corresponding to the frequency f_k .

The above development suggests that the periodogram can be computed by linearly filtering the signal sequence $\{x(n), 0 \leq n \leq N-1\}$ by a bank of parallel filters with impulse responses $\{h_k(n), 0 \leq n \leq N-1\}$, as illustrated in Figure 14.4.2. The 3-dB bandwidth of each of these filters is approximately $1/N$ cycles per sample interval.

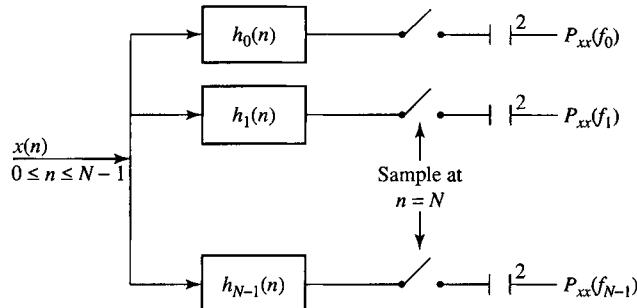


Figure 14.4.2
Filter bank implementation
of periodogram estimator.

Another equivalent filter bank implementation that produces the periodogram is shown in Figure 14.4.3. In this filter bank, the signal sequence $\{x(n), 0 \leq n \leq N - 1\}$ is multiplied by the exponential factors $\{e^{-j2\pi kn/N}, 0 \leq k \leq N - 1\}$ and each of the resultant products is passed through a rectangular lowpass filter with impulse response

$$h_0(n) = \begin{cases} 1/\sqrt{N}, & 0 \leq n \leq N - 1 \\ 0, & \text{otherwise} \end{cases} \quad (14.4.5)$$

whose complex-valued output at $n = N - 1$ is magnitude squared to yield the spectral estimate at the corresponding frequency $f = f_k$. We may call this spectral estimate the *unwindowed periodogram*.

We may replace the rectangular window in Figure 14.4.3 with another temporal window $h_0(n)$, say a Bartlett or a Hamming or a Kaiser window spanning the same time duration $0 \leq n \leq N - 1$. This would generate a *windowed periodogram*, which may be expressed as

$$P_{xx}(f) = \frac{1}{N} \left| \sum_{n=1}^{N-1} x(n) e^{j2\pi f n} h_0(N-n) \right|^2 \quad (14.4.6)$$

Applying a window function to the signal sequence is similar to the Welch method. However, in the latter, the window function is applied to each of several

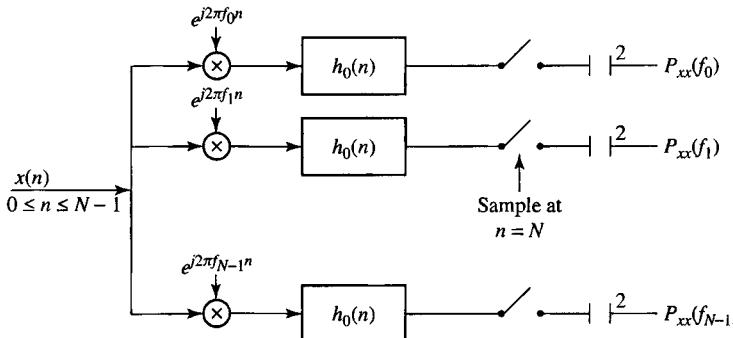


Figure 14.4.3 Alternative filter bank implementation of periodogram estimator.

segments of the data sequence and the windowed periodograms of the individual segments are averaged. Consequently, the variability of the spectral estimate obtained by the Welch method is significantly less than the variability of the spectral estimate obtained by the windowed periodogram in equation (14.4.6). On the other hand, the resolution resulting from the windowed periodogram is higher than the resolution obtained with the Welch method. As we observed in our treatment of nonparametric methods, there is a trade-off between frequency resolution and the variance of the spectral estimates. By segmenting the signal sequence into several smaller sequences, computing the windowed (modified) periodogram for each of the subsequences and, finally, averaging the windowed periodograms, we were able to reduce the variance of the estimate at the cost of decreasing the frequency resolution.

The filter bank interpretation of the periodogram leads us to consider other possible filter characteristics that may result in high resolution spectral estimates. We note that the filters used in the filter bank implementations shown in Figures 14.4.2 and 14.4.3 were not optimized in any manner. Thus, we may say that the filters were selected without regard to the characteristics of the data (i.e., the filters used are data independent). In the following section, we describe a filter bank implementation of the spectral estimator due to Capon (1969), in which the filters are designed based on the statistical characteristics of the data (i.e., the filter impulse responses are adapted to the characteristics of the data).

14.4.2 Minimum Variance Spectral Estimates

The spectral estimator proposed by Capon (1969) was intended for use in large seismic arrays for frequency-wave number estimation. It was later adapted to single-time-series spectrum estimation by Lacoss (1971), who demonstrated that the method provides a minimum-variance unbiased estimate of the spectral components in the signal. This estimator has also been called a “maximum-likelihood spectral estimator” because the filter design method results in the minimum-variance unbiased spectral estimate when the signal is corrupted by additive Gaussian noise.

Following the development of Lacoss, let us consider the design of an FIR filter with coefficients $h(k)$, $0 \leq k \leq p$, to be determined. Then, if the observed data $x(n)$, $0 \leq n \leq N - 1$, are passed through the filter, the response is

$$y(n) = \sum_{k=0}^p h(k)x(n-k) \equiv \mathbf{X}'(n)\mathbf{h} \quad (14.4.7)$$

where $\mathbf{X}'(n) = [x(n) \ x(n-1) \ \dots \ x(n-p)]$ is the data vector and \mathbf{h} is the filter coefficient vector. If we assume that $E[x(n)] = 0$, the variance of the output sequence is

$$\begin{aligned} \sigma_y^2 &= E[|y(n)|^2] = E[\mathbf{h}^H \mathbf{X}'(n) \mathbf{X}'(n) \mathbf{h}] \\ &= \mathbf{h}^H \boldsymbol{\Gamma}_{xx} \mathbf{h} \end{aligned} \quad (14.4.8)$$

where $\boldsymbol{\Gamma}_{xx}$ is the autocorrelation matrix of the sequence $x(n)$, with elements $\gamma_{xx}(m)$.

The filter coefficients are selected so that at the frequency f_l , the frequency response of the FIR filter is normalized to unity, that is,

$$\sum_{k=0}^p h(k) e^{-j2\pi kf_l} = 1$$

This constraint can also be written in matrix form as

$$\mathbf{E}^H(f_l) \mathbf{h} = 1 \quad (14.4.9)$$

where

$$\mathbf{E}'(f_l) = [1 \quad e^{j2\pi f_l} \quad \dots \quad e^{j2\pi p f_l}]$$

By minimizing the variance σ_y^2 subject to the constraint (14.4.9), we obtain an FIR filter that passes the frequency component f_l undistorted, while components distant from f_l are severely attenuated. The result of this minimization leads to the coefficient vector

$$\hat{\mathbf{h}}_{\text{opt}} = \boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f_l) / \mathbf{E}^H(f_l) \boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f_l) \quad (14.4.10)$$

If $\hat{\mathbf{h}}$ is substituted into (14.4.8), we obtain the minimum variance

$$\sigma_{\min}^2 = \frac{1}{\mathbf{E}^H(f_l) \boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f_l)} \quad (14.4.11)$$

The expression in (14.4.11) is the minimum-variance power spectrum estimate at the frequency f_l . By changing f_l over the range $0 \leq f_l \leq 0.5$, we can obtain the power spectrum estimate. Thus, the minimum-variance method is basically a filter bank implementation for the spectrum estimator. It differs basically from the filter bank interpretation of the periodogram in that the filter coefficients in the Capon method are optimized. It should be noted that although $\mathbf{E}(f)$ changes with the choice of frequency, $\boldsymbol{\Gamma}_{xx}^{-1}$ is computed only once. As demonstrated by Lacoss (1971), the computation of the quadratic form $\mathbf{E}^H(f) \boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f)$ can be done with a single DFT.

With an estimate \mathbf{R}_{xx} of the autocorrelation matrix substituted in place of $\boldsymbol{\Gamma}_{xx}$, we obtain the minimum variance power spectrum estimate of Capon as

$$P_{xx}^{\text{MV}}(f) = \frac{1}{\mathbf{E}^H(f) \mathbf{R}_{xx}^{-1} \mathbf{E}(f)} \quad (14.4.12)$$

It has been shown by Lacoss (1971) that this power spectrum estimator yields estimates of the spectral peaks proportional to the power at that frequency. In contrast, the AR methods described in Section 14.3 result in estimates of the spectral peaks proportional to the square of the power at that frequency.

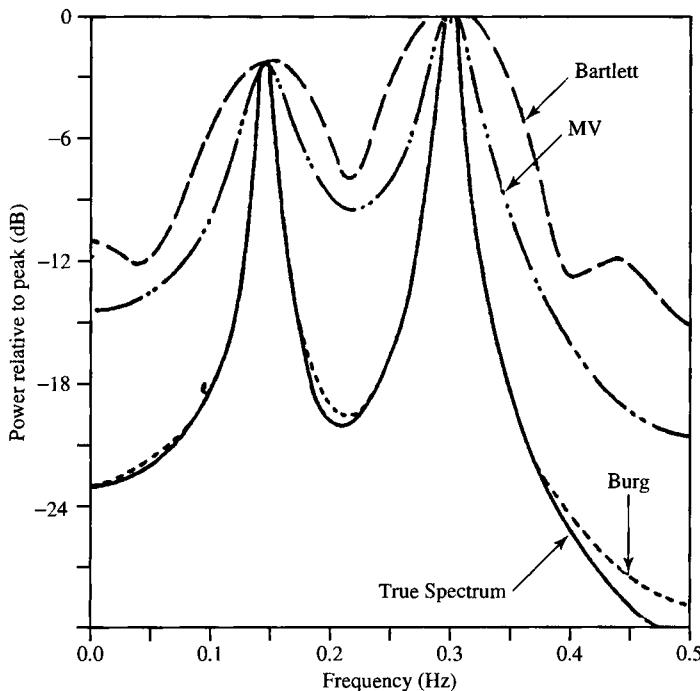


Figure 14.4.4 A comparison of three spectral estimates (Bartlett, minimum variance (MV) and Burg) for a signal with two narrowband peaks at frequencies 0.15 and 0.30. (Taken from R.I. Lacoss, “Data Adaptive Spectral Analysis Methods,” *Geophysics*, Vol. 36, pp. 661–675, August 1971. Reproduced by permission.)

Experiments on the performance of this method compared with the performance of the Burg method have been done by Lacoss (1971) and others. For example, Figure 14.4.4 taken from the paper by Lacoss (1971) illustrates a comparison of three spectral estimates, the Bartlett estimate, the minimum variance estimate, and the Burg estimate for a signal containing two narrowband peaks at frequencies 0.15 and 0.30. This figure illustrates that the minimum variance spectral estimator yields a better spectral estimate than the Bartlett estimate, but a poorer estimate compared to the Burg estimate. The true power spectrum of the signal is also shown in this figure. In general, the minimum-variance estimate in (14.4.12) outperforms the non-parametric spectral estimators in frequency resolution, but it does not provide the high frequency resolution obtained from the AR methods of Burg and the unconstrained least squares. Furthermore, Burg (1972) demonstrated that for a known correlation sequence, the minimum variance spectrum is related to the AR model spectrum through the equation

$$\frac{1}{\Gamma_{xx}^{\text{MV}}(f)} = \frac{1}{p} \sum_{k=0}^p \frac{1}{\Gamma_{xx}^{\text{AR}}(f, k)} \quad (14.4.13)$$

where $\Gamma_{xx}^{\text{AR}}(f, k)$ is the AR power spectrum obtained with an AR(k) model. Thus

the reciprocal of the minimum variance estimate is equal to the average of the reciprocals of all spectra obtained with AR(k) models for $1 \leq k \leq p$. Since low-order AR models, in general, do not provide good resolution, the averaging operation in (14.4.13) reduces the frequency resolution in the spectral estimate. Hence we conclude that the AR power spectrum estimate of order p is superior to the minimum variance estimate of order $p + 1$.

The relationship given by (14.4.13) represents a frequency-domain relationship between the Capon minimum variance estimate and the Burg AR estimate. A time-domain relationship between these two estimates also can be established as shown by Musicus (1985). This has led to a computationally efficient algorithm for the minimum variance estimate.

Additional references to the method of Capon and comparisons with other estimators can be found in the literature. We cite the papers of Capon and Goodman (1971), Marzetta (1983), Marzetta and Lang (1983, 1984), Capon (1983), and McDonough (1983).

14.5 Eigenanalysis Algorithms for Spectrum Estimation

In Section 14.3.8 we demonstrated that an AR(p) process corrupted by additive (white) noise is equivalent to an ARMA(p, p) process. In this section we consider the special case in which the signal components are sinusoids corrupted by additive white noise. The algorithms are based on an eigen-decomposition of the correlation matrix of the noise-corrupted signal.

From our previous discussion on the generation of sinusoids in Chapter 5, we recall that a real sinusoidal signal can be generated via the difference equation,

$$x(n) = -a_1 x(n-1) - a_2 x(n-2) \quad (14.5.1)$$

where $a_1 = 2 \cos 2\pi f_k$, $a_2 = 1$, and initially, $x(-1) = -1$, $x(-2) = 0$. This system has a pair of complex-conjugate poles (at $f = f_k$ and $f = -f_k$) and therefore generates the sinusoid $x(n) = \cos 2\pi f_k n$, for $n \geq 0$.

In general, a signal consisting of p sinusoidal components satisfies the difference equation

$$x(n) = - \sum_{m=1}^{2p} a_m x(n-m) \quad (14.5.2)$$

and corresponds to the system with system function

$$H(z) = \frac{1}{1 + \sum_{m=1}^{2p} a_m z^{-m}} \quad (14.5.3)$$

The polynomial

$$A(z) = 1 + \sum_{m=1}^{2p} a_m z^{-m} \quad (14.5.4)$$

has $2p$ roots on the unit circle which correspond to the frequencies of the sinusoids.

Now, suppose that the sinusoids are corrupted by a white noise sequence $w(n)$ with $E[|w(n)|^2] = \sigma_w^2$. Then we observe that

$$y(n) = x(n) + w(n) \quad (14.5.5)$$

If we substitute $x(n) = y(n) - w(n)$ in (14.5.2), we obtain

$$y(n) - w(n) = - \sum_{m=1}^{2p} [y(n-m) - w(n-m)] a_m$$

or, equivalently,

$$\sum_{m=0}^{2p} a_m y(n-m) = \sum_{m=0}^{2p} a_m w(n-m) \quad (14.5.6)$$

where, by definition, $a_0 = 1$.

We observe that (14.5.6) is the difference equation for an ARMA(2p, 2p) process in which both the AR and MA parameters are identical. This symmetry is a characteristic of the sinusoidal signals in white noise. The difference equation in (14.5.6) may be expressed in matrix form as

$$\mathbf{Y}^t \mathbf{a} = \mathbf{W}^t \mathbf{a} \quad (14.5.7)$$

where $\mathbf{Y}^t = [y(n) \ y(n-1) \ \cdots \ y(n-2p)]$ is the observed data vector of dimension $(2p+1)$, $\mathbf{W}^t = [w(n) \ w(n-1) \ \cdots \ w(n-2p)]$ is the noise vector, and $\mathbf{a} = [1 \ a_1 \ \cdots \ a_{2p}]$ is the coefficient vector.

If we premultiply (14.5.7) by \mathbf{Y} and take the expected value, we obtain

$$\begin{aligned} E(\mathbf{Y}\mathbf{Y}^t)\mathbf{a} &= E(\mathbf{Y}\mathbf{W}^t)\mathbf{a} = E[(\mathbf{X} + \mathbf{W})\mathbf{W}^t]\mathbf{a} \\ \Gamma_{yy}\mathbf{a} &= \sigma_w^2 \mathbf{a} \end{aligned} \quad (14.5.8)$$

where we have used the assumption that the sequence $w(n)$ is zero mean and white, and \mathbf{X} is a deterministic signal.

The equation in (14.5.8) is in the form of an eigenequation, that is,

$$(\Gamma_{yy} - \sigma_w^2 \mathbf{I})\mathbf{a} = \mathbf{0} \quad (14.5.9)$$

where σ_w^2 is an eigenvalue of the autocorrelation matrix Γ_{yy} . Then the parameter vector \mathbf{a} is an eigenvector associated with the eigenvalue σ_w^2 . The eigenequation in (14.5.9) forms the basis for the Pisarenko harmonic decomposition method.

14.5.1 Pisarenko Harmonic Decomposition Method

For p randomly phased sinusoids in additive white noise, the autocorrelation values are

$$\begin{aligned}\gamma_{yy}(0) &= \sigma_w^2 + \sum_{i=1}^p P_i \\ \gamma_{yy}(k) &= \sum_{i=1}^p P_i \cos 2\pi f_i k, \quad k \neq 0\end{aligned}\tag{14.5.10}$$

where $P_i = A_i^2/2$ is the average power in the i th sinusoid and A_i is the corresponding amplitude. Hence we may write

$$\begin{bmatrix} \cos 2\pi f_1 & \cos 2\pi f_2 & \cdots & \cos 2\pi f_p \\ \cos 4\pi f_1 & \cos 4\pi f_2 & \cdots & \cos 4\pi f_p \\ \vdots & \vdots & & \vdots \\ \cos 2\pi p f_1 & \cos 2\pi p f_2 & \cdots & \cos 2\pi p f_p \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_p \end{bmatrix} = \begin{bmatrix} \gamma_{yy}(1) \\ \gamma_{yy}(2) \\ \vdots \\ \gamma_{yy}(p) \end{bmatrix}\tag{14.5.11}$$

If we know the frequencies f_i , $1 \leq i \leq p$, we can use this equation to determine the powers of the sinusoids. In place of $\gamma_{xx}(m)$, we use the estimates $r_{xx}(m)$. Once the powers are known, the noise variance can be obtained from (14.5.10) as

$$\sigma_w^2 = r_{yy}(0) - \sum_{i=1}^p P_i\tag{14.5.12}$$

The problem that remains is to determine the p frequencies f_i , $1 \leq i \leq p$, which, in turn, require knowledge of the eigenvector \mathbf{a} corresponding to the eigenvalue σ_w^2 . Pisarenko (1973) observed [see also Papoulis (1984) and Grenander and Szegö (1958)] that for an ARMA process consisting of p sinusoids in additive white noise, the variance σ_w^2 corresponds to the minimum eigenvalue of Γ_{yy} when the dimension of the autocorrelation matrix equals or exceeds $(2p+1) \times (2p+1)$. The desired ARMA coefficient vector corresponds to the eigenvector associated with the minimum eigenvalue. Therefore, the frequencies f_i , $1 \leq i \leq p$ are obtained from the roots of the polynomial in (14.5.4), where the coefficients are the elements of the eigenvector \mathbf{a} corresponding to the minimum eigenvalue σ_w^2 .

In summary, the Pisarenko harmonic decomposition method proceeds as follows. First we estimate Γ_{yy} from the data (i.e., we form the autocorrelation matrix \mathbf{R}_{yy}). Then we find the minimum eigenvalue and the corresponding minimum eigenvector. The minimum eigenvector yields the parameters of the ARMA(2p, 2p) model. From (14.5.4) we can compute the roots that constitute the frequencies {f_i}. By using these frequencies, we can solve (14.5.11) for the signal powers {P_i} by substituting the estimates $r_{yy}(m)$ for $\gamma_{yy}(m)$.

As will be seen in the following example, the Pisarenko method is based on the use of a noise subspace eigenvector to estimate the frequencies of the sinusoids.

EXAMPLE 14.5.1

Suppose that we are given the autocorrelation values $\gamma_{yy}(0) = 3$, $\gamma_{yy}(1) = 1$, and $\gamma_{yy}(2) = 0$ for a process consisting of a single sinusoid in additive white noise. Determine the frequency, its power, and the variance of the additive noise.

Solution. The correlation matrix is

$$\mathbf{\Gamma}_{yy} = \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

The minimum eigenvalue is the smallest root of the characteristic polynomial

$$g(\lambda) = \begin{vmatrix} 3 - \lambda & 1 & 0 \\ 1 & 3 - \lambda & 1 \\ 0 & 1 & 3 - \lambda \end{vmatrix} = (3 - \lambda)(\lambda^2 - 6\lambda + 7) = 0$$

Therefore, the eigenvalues are $\lambda_1 = 3$, $\lambda_2 = 3 + \sqrt{2}$, $\lambda_3 = 3 - \sqrt{2}$.

The variance of the noise is

$$\sigma_w^2 = \lambda_{\min} = 3 - \sqrt{2}$$

The corresponding eigenvalue is the vector that satisfies (14.5.9), that is,

$$\begin{bmatrix} \sqrt{2} & 1 & 0 \\ 1 & \sqrt{2} & 1 \\ 0 & 1 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

The solution is $a_1 = -\sqrt{2}$ and $a_2 = 1$.

The next step is to use the value a_1 and a_2 to determine the roots of the polynomial in (14.5.4). We have

$$z^2 - \sqrt{2}z + 1 = 0$$

Thus

$$z_1, z_2 = \frac{1}{\sqrt{2}} \pm j \frac{1}{\sqrt{2}}$$

Note that $|z_1| = |z_2| = 1$, so that the roots are on the unit circle. The corresponding frequency is obtained from

$$z_i = e^{j2\pi f_i} = \frac{1}{\sqrt{2}} + j \frac{1}{\sqrt{2}}$$

which yields $f_1 = \frac{1}{8}$. Finally, the power of the sinusoid is

$$P_1 \cos 2\pi f_1 = \gamma_{yy}(1) = 1$$

$$P_1 = \sqrt{2}$$

and its amplitude is $A = \sqrt{2P_1} = \sqrt{2\sqrt{2}}$.

As a check on our computations, we have

$$\begin{aligned} \sigma_w^2 &= \gamma_{yy}(0) - P_1 \\ &= 3 - \sqrt{2} \end{aligned}$$

which agrees with λ_{\min} .

14.5.2 Eigen-decomposition of the Autocorrelation Matrix for Sinusoids in White Noise

In the previous discussion we assumed that the sinusoidal signal consists of p real sinusoids. For mathematical convenience we shall now assume that the signal consists of p complex sinusoids of the form

$$x(n) = \sum_{i=1}^p A_i e^{j(2\pi f_i n + \phi_i)} \quad (14.5.13)$$

where the amplitudes $\{A_i\}$ and the frequencies $\{f_i\}$ are unknown and the phases $\{\phi_i\}$ are statistically independent random variables uniformly distributed on $(0, 2\pi)$. Then the random process $x(n)$ is wide-sense stationary with autocorrelation function

$$\gamma_{xx}(m) = \sum_{i=1}^p P_i e^{j2\pi f_i m} \quad (14.5.14)$$

where, for complex sinusoids, $P_i = A_i^2$ is the power of the i th sinusoid.

Since the sequence observed is $y(n) = x(n) + w(n)$, where $w(n)$ is a white noise sequence with spectral density σ_w^2 , the autocorrelation function for $y(n)$ is

$$\gamma_{yy}(m) = \gamma_{xx}(m) + \sigma_w^2 \delta(m), \quad m = 0, \pm 1, \dots, \pm(M-1) \quad (14.5.15)$$

Hence the $M \times M$ autocorrelation matrix for $y(n)$ can be expressed as

$$\Gamma_{yy} = \Gamma_{xx} + \sigma_w^2 \mathbf{I} \quad (14.5.16)$$

where Γ_{xx} is the autocorrelation matrix for the signal $x(n)$ and $\sigma_w^2 \mathbf{I}$ is the autocorrelation matrix for the noise. Note that if we select $M > p$, Γ_{xx} which is of dimension $M \times M$ is not of full rank, because its rank is p . However, Γ_{yy} is full rank because $\sigma_w^2 \mathbf{I}$ is of rank M .

In fact, the signal matrix Γ_{xx} can be represented as

$$\Gamma_{xx} = \sum_{i=1}^p P_i \mathbf{s}_i \mathbf{s}_i^H \quad (14.5.17)$$

where H denotes the conjugate transpose and \mathbf{s}_i is a signal vector of dimension M defined as

$$\mathbf{s}_i = [1, e^{j2\pi f_i}, e^{j4\pi f_i}, \dots, e^{j2\pi(M-1)f_i}] \quad (14.5.18)$$

Since each vector (outer product) $\mathbf{s}_i \mathbf{s}_i^H$ is a matrix of rank 1 and since there are p vector products, the matrix Γ_{xx} is of rank p . Note that if the sinusoids were real, the correlation matrix Γ_{xx} would have rank $2p$.

Now, let us perform an eigen-decomposition of the matrix Γ_{yy} . Let the eigenvalues $\{\lambda_i\}$ be ordered in decreasing value with $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \dots \geq \lambda_M$ and let the corresponding eigenvectors be denoted as $\{\mathbf{v}_i, i = 1, \dots, M\}$. We assume that

the eigenvectors are normalized so that $\mathbf{v}_i^H \cdot \mathbf{v}_j = \delta_{ij}$. In the absence of noise the eigenvalues λ_i , $i = 1, 2, \dots, p$, are nonzero while $\lambda_{p+1} = \lambda_{p+2} = \dots = \lambda_M = 0$. Furthermore, it follows that the signal correlation matrix can be expressed as

$$\boldsymbol{\Gamma}_{xx} = \sum_{i=1}^p \lambda_i \mathbf{v}_i \mathbf{v}_i^H \quad (14.5.19)$$

Thus, the eigenvectors \mathbf{v}_i , $i = 1, 2, \dots, p$ span the signal subspace as do the signal vectors \mathbf{s}_i , $i = 1, 2, \dots, p$. These p eigenvectors for the signal subspace are called the *principal eigenvectors* and the corresponding eigenvalues are called the *principal eigenvalues*.

In the presence of noise, the noise autocorrelation matrix in (14.5.16) can be represented as

$$\sigma_w^2 \mathbf{I} = \sigma_w^2 \sum_{i=1}^M \mathbf{v}_i \mathbf{v}_i^H \quad (14.5.20)$$

By substituting (14.5.19) and (14.5.20) into (14.5.16), we obtain

$$\begin{aligned} \boldsymbol{\Gamma}_{yy} &= \sum_{i=1}^p \lambda_i \mathbf{v}_i \mathbf{v}_i^H + \sum_{i=1}^M \sigma_w^2 \mathbf{v}_i \mathbf{v}_i^H \\ &= \sum_{i=1}^p (\lambda_i + \sigma_w^2) \mathbf{v}_i \mathbf{v}_i^H + \sum_{i=p+1}^M \sigma_w^2 \mathbf{v}_i \mathbf{v}_i^H \end{aligned} \quad (14.5.21)$$

This eigen-decomposition separates the eigenvectors into two sets. The set $\{\mathbf{v}_i, i = 1, 2, \dots, p\}$, which are the principal eigenvectors, span the signal subspace, while the set $\{\mathbf{v}_i, i = p+1, \dots, M\}$, which are orthogonal to the principal eigenvectors, are said to belong to the noise subspace. Since the signal vectors $\{\mathbf{s}_i, i = 1, 2, \dots, p\}$ are in the signal subspace, it follows that the $\{\mathbf{s}_i\}$ are simply linear combinations of the principal eigenvectors and are also orthogonal to the vectors in the noise subspace.

In this context we see that the Pisarenko method is based on an estimation of the frequencies by using the orthogonality property between the signal vectors and the vectors in the noise subspace. For complex sinusoids, if we select $M = p+1$ (for real sinusoids we select $M = 2p+1$), there is only a single eigenvector in the noise subspace (corresponding to the minimum eigenvalue) which must be orthogonal to the signal vectors. Thus we have

$$\mathbf{s}_i^H \mathbf{v}_{p+1} = \sum_{k=0}^p v_{p+1}(k+1) e^{-j2\pi f_i k} = 0, \quad i = 1, 2, \dots, p \quad (14.5.22)$$

But (14.5.22) implies that the frequencies $\{f_i\}$ can be determined by solving for the zeros of the polynomial

$$V(z) = \sum_{k=0}^p v_{p+1}(k+1) z^{-k} \quad (14.5.23)$$

all of which lie on the unit circle. The angles of these roots are $2\pi f_i$, $i = 1, 2, \dots, p$.

When the number of sinusoids is unknown, the determination of p may prove to be difficult, especially if the signal level is not much higher than the noise level. In theory, if $M > p + 1$, there is a multiplicity $(M - p)$ of the minimum eigenvalue. However, in practice the $(M - p)$ small eigenvalues of \mathbf{R}_{yy} will probably be different. By computing all the eigenvalues it may be possible to determine p by grouping the $M - p$ small (noise) eigenvalues into a set and averaging them to obtain an estimate of σ_w^2 . Then, the average value can be used in (14.5.9) along with \mathbf{R}_{yy} to determine the corresponding eigenvector.

14.5.3 MUSIC Algorithm

The multiple signal classification (MUSIC) method is also a noise subspace frequency estimator. To develop the method, let us first consider the “weighted” spectral estimate

$$P(f) = \sum_{k=p+1}^M w_k |\mathbf{s}^H(f)\mathbf{v}_k|^2 \quad (14.5.24)$$

where $\{\mathbf{v}_k, k = p + 1, \dots, M\}$ are the eigenvectors in the noise subspace, $\{w_k\}$ are a set of positive weights, and $\mathbf{s}(f)$ is the complex sinusoidal vector

$$\mathbf{s}(f) = [1, e^{j2\pi f}, e^{j4\pi f}, \dots, e^{j2\pi(M-1)f}] \quad (14.5.25)$$

Note that at $f = f_i$, $\mathbf{s}(f_i) \equiv \mathbf{s}_i$, so that at any one of the p sinusoidal frequency components of the signal, we have

$$P(f_i) = 0, \quad i = 1, 2, \dots, p \quad (14.5.26)$$

Hence, the reciprocal of $P(f)$ is a sharply peaked function of frequency and provides a method for estimating the frequencies of the sinusoidal components. Thus

$$\frac{1}{P(f)} = \frac{1}{\sum_{k=p+1}^M w_k |\mathbf{s}^H(f)\mathbf{v}_k|^2} \quad (14.5.27)$$

Although theoretically $1/P(f)$ is infinite at $f = f_i$, in practice the estimation errors result in finite values for $1/P(f)$ at all frequencies.

The MUSIC sinusoidal frequency estimator proposed by Schmidt (1981, 1986) is a special case of (14.5.27) in which the weights $w_k = 1$ for all k . Hence

$$P_{\text{MUSIC}}(f) = \frac{1}{\sum_{k=p+1}^M |\mathbf{s}^H(f)\mathbf{v}_k|^2} \quad (14.5.28)$$

The estimates of the sinusoidal frequencies are the peaks of $P_{\text{MUSIC}}(f)$. Once the sinusoidal frequencies are estimated, the power of each of the sinusoids can be obtained by solving (14.5.11).

EXAMPLE 14.5.2

The autocorrelation matrix for a signal consisting of a complex exponential in white noise is given as

$$\boldsymbol{\Gamma}_{yy} = \begin{bmatrix} 3 & -2j & -2 \\ 2j & 3 & -2j \\ -2 & 2j & 3 \end{bmatrix}$$

Use the MUSIC method to determine the frequencies and the powers of the complex exponential and the variance of the additive noise.

Solution. By solving for the roots of the polynomial

$$g(\lambda) = \begin{vmatrix} 3 - \lambda & -2j & -2 \\ 2j & 3 - \lambda & -2j \\ -2 & 2j & 3 - \lambda \end{vmatrix} = \lambda^3 - 9\lambda^2 + 15\lambda - 7 = 0$$

we find that the eigenvalues are $\lambda_1 = 7$, $\lambda_2 = 1$, and $\lambda_3 = 1$. Hence, we conclude that there is a single complex exponential, corresponding to the eigenvalue $\lambda = 7$. The eigenvectors corresponding to the signal and the noise subspaces are, respectively,

$$\mathbf{v}_1 = \begin{bmatrix} 1/\sqrt{3} \\ j/\sqrt{3} \\ -1/3 \end{bmatrix}, \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ j/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}, \quad \mathbf{v}_3 = \begin{bmatrix} 2j/\sqrt{6} \\ 1/\sqrt{6} \\ j/\sqrt{6} \end{bmatrix}$$

By computing the denominator of (14.5.28) we obtain

$$\sum_{k=2}^3 |\mathbf{s}^H(f)\mathbf{v}_k|^2 = 2 + \frac{5}{3} \cos\left(2\pi f + \frac{\pi}{2}\right) + \frac{2}{3} \cos 4\pi f + \frac{1}{3} \cos\left(2\pi f - \frac{\pi}{2}\right)$$

It is easily verified that this term is zero at $f = 1/4$. Furthermore, from the relation (14.5.15) and our knowledge that $\sigma_w^2 = 1$, we conclude that the power of the complex exponential signal is $P = 2$.

In comparing the Pisarenko method to the MUSIC algorithm, we note that the former selects $M = p + 1$ and projects the signal vectors onto a single noise eigenvector. Hence, the Pisarenko method assumes precise knowledge of the number of sinusoidal components in the signal. In contrast, the MUSIC method selects $M > p + 1$ and, after performing the eigenanalysis, subdivides the eigenvalues into two groups, those (p) corresponding to the signal subspace and those ($M - p$) corresponding to the noise subspace. Then, the signal vectors are projected onto the $M - p$ eigenvectors in the noise subspace. Hence precise knowledge of p is not necessary. The order selection methods described in Section 14.5.5 may be used to obtain an estimate of p and to select M such that $M > p + 1$.

14.5.4 ESPRIT Algorithm

ESPRIT (estimation of signal parameters via rotational invariance techniques) is yet another method for estimating frequencies of a sum of sinusoids by use of an eigen-decomposition approach. As we observe from the development that follows, which

is due to Roy et al. (1986), ESPRIT exploits an underlying rotational invariance of signal subspaces spanned by two temporally displaced data vectors.

We again consider the estimation of p complex-valued sinusoids in additive white noise. The received sequence is given by the vector

$$\begin{aligned}\mathbf{y}(n) &= [y(n), y(n+1), \dots, y(n+M-1)]^T \\ &= \mathbf{x}(n) + \mathbf{w}(n)\end{aligned}\quad (14.5.29)$$

where $\mathbf{x}(n)$ is the signal vector and $\mathbf{w}(n)$ is the noise vector. To exploit the deterministic character of the sinusoids, we define the time-displaced vector $\mathbf{z}(n) = \mathbf{y}(n+1)$. Thus

$$\begin{aligned}\mathbf{z}(n) &= [z(n), z(n+1), \dots, z(n+M-1)]^T \\ &= [y(n+1), y(n+2), \dots, y(n+M)]^T\end{aligned}\quad (14.5.30)$$

With these definitions we can express the vectors $\mathbf{y}(n)$ and $\mathbf{z}(n)$ as

$$\begin{aligned}\mathbf{y}(n) &= \mathbf{S}\mathbf{a} + \mathbf{w}(n) \\ \mathbf{z}(n) &= \mathbf{S}\Phi\mathbf{a} + \mathbf{w}(n)\end{aligned}\quad (14.5.31)$$

where $\mathbf{a} = [a_1, a_2, \dots, a_p]^T$, $a_i = A_i e^{j\phi_i}$, and Φ is a diagonal $p \times p$ matrix consisting of the relative phase between adjacent time samples of each of the complex sinusoids,

$$\Phi = \text{diag}[e^{j2\pi f_1}, e^{j2\pi f_2}, \dots, e^{j2\pi f_p}] \quad (14.5.32)$$

Note that the matrix Φ relates the time-displaced vectors $\mathbf{y}(n)$ and $\mathbf{z}(n)$ and can be called a rotation operator. We also note that Φ is unitary. The matrix \mathbf{S} is the $M \times p$ Vandermonde matrix specified by the column vectors

$$\mathbf{s}_i = [1, e^{j2\pi f_i}, e^{j4\pi f_i}, \dots, e^{j2\pi(M-1)f_i}], \quad i = 1, 2, \dots, p \quad (14.5.33)$$

Now the autocovariance matrix for the data vector $\mathbf{y}(n)$ is

$$\begin{aligned}\boldsymbol{\Gamma}_{yy} &= E[\mathbf{y}(n)\mathbf{y}^H(n)] \\ &= \mathbf{SPS}^H + \sigma_w^2 \mathbf{I}\end{aligned}\quad (14.5.34)$$

where \mathbf{P} is the $p \times p$ diagonal matrix consisting of the powers of the complex sinusoids,

$$\begin{aligned}\mathbf{P} &= \text{diag}[|a_1|^2, |a_2|^2, \dots, |a_p|^2] \\ &= \text{diag}[P_1, P_2, \dots, P_p]\end{aligned}\quad (14.5.35)$$

We observe that \mathbf{P} is a diagonal matrix since complex sinusoids of different frequencies are orthogonal over the infinite interval. However, we should emphasize that the ESPRIT algorithm does not require \mathbf{P} to be diagonal. Hence the algorithm

is applicable to the case in which the covariance matrix is estimated from finite data records.

The crosscovariance matrix of the signal vectors $\mathbf{y}(n)$ and $\mathbf{z}(n)$ is

$$\boldsymbol{\Gamma}_{yz} = E[\mathbf{y}(n)\mathbf{z}^H(n)] = \mathbf{SP}\boldsymbol{\Phi}^H\mathbf{S}^H + \boldsymbol{\Gamma}_w \quad (14.5.36)$$

where

$$\begin{aligned} \boldsymbol{\Gamma}_w &= E[\mathbf{w}(n)\mathbf{w}^H(n+1)] \\ &= \sigma_w^2 \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \equiv \sigma_w^2 \mathbf{Q} \end{aligned} \quad (14.5.37)$$

The auto and crosscovariance matrices $\boldsymbol{\Gamma}_{yy}$ and $\boldsymbol{\Gamma}_{yz}$ are given as

$$\boldsymbol{\Gamma}_{yy} = \begin{bmatrix} \gamma_{yy}(0) & \gamma_{yy}(1) & \cdots & \gamma_{yy}(M-1) \\ \gamma_{yy}^*(1) & \gamma_{yy}(0) & \cdots & \gamma_{yy}(M-2) \\ \vdots & \vdots & & \vdots \\ \gamma_{yy}^*(M-1) & \gamma_{yy}(M-2) & \cdots & \gamma_{yy}(0) \end{bmatrix} \quad (14.5.38)$$

$$\boldsymbol{\Gamma}_{yz} = \begin{bmatrix} \gamma_{yy}(1) & \gamma_{yy}(2) & \cdots & \gamma_{yy}(M) \\ \gamma_{yy}(0) & \gamma_{yy}(1) & \cdots & \gamma_{yy}(M-1) \\ \vdots & \vdots & & \vdots \\ \gamma_{yy}^*(M-2) & \gamma_{yy}^*(M-3) & \cdots & \gamma_{yy}(1) \end{bmatrix} \quad (14.5.39)$$

where $\gamma_{yy}(m) = E[y^*(n)y(n+m)]$. Note that both $\boldsymbol{\Gamma}_{yy}$ and $\boldsymbol{\Gamma}_{yz}$ are Toeplitz matrices.

Based on this formulation, the problem is to determine the frequencies $\{f_i\}$ and their powers $\{P_i\}$ from the autocorrelation sequence $\{\gamma_{yy}(m)\}$.

From the underlying model, it is clear that the matrix \mathbf{SPS}^H has rank p . Consequently, $\boldsymbol{\Gamma}_{yy}$ given by (14.5.34) has $(M-p)$ identical eigenvalues equal to σ_w^2 . Hence

$$\boldsymbol{\Gamma}_{yy} - \sigma_w^2 \mathbf{I} = \mathbf{SPS}^H \equiv \mathbf{C}_{yy} \quad (14.5.40)$$

From (14.5.36) we also have

$$\boldsymbol{\Gamma}_{yz} - \sigma_w^2 \boldsymbol{\Gamma}_w = \mathbf{SP}\boldsymbol{\Phi}^H\mathbf{S}^H \equiv \mathbf{C}_{yz} \quad (14.5.41)$$

Now, let us consider the matrix $\mathbf{C}_{yy} - \lambda \mathbf{C}_{yz}$, which can be written as

$$\mathbf{C}_{yy} - \lambda \mathbf{C}_{yz} = \mathbf{SP}(\mathbf{I} - \lambda \boldsymbol{\Phi}^H)\mathbf{S}^H \quad (14.5.42)$$

Clearly, the column space of \mathbf{SPS}^H is identical to the column space of $\mathbf{SP}\boldsymbol{\Phi}^H\mathbf{S}^H$. Consequently, the rank of $\mathbf{C}_{yy} - \lambda \mathbf{C}_{yz}$ is equal to p . However, we note that if $\lambda = \exp(j2\pi f_i)$, the i th row of $(\mathbf{I} - \lambda \boldsymbol{\Phi}^H)$ is zero and hence the rank of $[\mathbf{I} - \boldsymbol{\Phi}^H \exp(j2\pi f_i)]$ is $p-1$. But $\lambda_i = \exp(j2\pi f_i)$, $i = 1, 2, \dots, p$, are the generalized eigenvalues of the

matrix pair $(\mathbf{C}_{yy}, \mathbf{C}_{yz})$. Thus the p generalized eigenvalues $\{\lambda_i\}$ that lie on the unit circle correspond to the elements of the rotation operator Φ . The remaining $M - p$ generalized eigenvalues of the pair $\{\mathbf{C}_{yy}, \mathbf{C}_{yz}\}$, which correspond to the common null space of these matrices, are zero [i.e., the $(M - p)$ eigenvalues are at the origin in the complex plane].

Based on these mathematical relationships we can formulate an algorithm (ES-PRIT) for estimating the frequencies $\{f_i\}$. The procedure is as follows:

1. From the data, compute the autocorrelation values $r_{yy}(m)$, $m = 1, 2, \dots, M$, and form the matrices \mathbf{R}_{yy} and \mathbf{R}_{yz} corresponding to estimates of Γ_{yy} and Γ_{yz} .
2. Compute the eigenvalues of \mathbf{R}_{yy} . For $M > p$, the minimum eigenvalue is an estimate of σ_w^2 .
3. Compute $\hat{\mathbf{C}}_{yy} = \mathbf{R}_{yy} - \hat{\sigma}_w^2 \mathbf{I}$ and $\hat{\mathbf{C}}_{yz} = \mathbf{R}_{yz} - \hat{\sigma}_w^2 \mathbf{Q}$, where \mathbf{Q} is defined in (14.5.37).
4. Compute the generalized eigenvalues of the matrix pair $\{\hat{\mathbf{C}}_{yy}, \hat{\mathbf{C}}_{yz}\}$. The p generalized eigenvalues of these matrices that lie on (or near) the unit circle determine the (estimate) elements of Φ and hence the sinusoidal frequencies. The remaining $M - p$ eigenvalues will lie at (or near) the origin.

One method for determining the power in the sinusoidal components is to solve the equation in (14.5.11) with $r_{yy}(m)$ substituted for $\gamma_{yy}(m)$.

Another method is based on the computation of the generalized eigenvectors $\{\mathbf{v}_i\}$ corresponding to the generalized eigenvalues $\{\lambda_i\}$. We have

$$(\mathbf{C}_{yy} - \lambda_i \mathbf{C}_{yz}) \mathbf{v}_i = \mathbf{S} \mathbf{P} (\mathbf{I} - \lambda_i \boldsymbol{\Phi}^H) \mathbf{S}^H \mathbf{v}_i = 0 \quad (14.5.43)$$

Since the column space of $(\mathbf{C}_{yy} - \lambda_i \cdot \mathbf{C}_{yz})$ is identical to the column space spanned by the vectors $\{\mathbf{s}_j, j \neq i\}$ given by (14.5.33), it follows that the generalized eigenvector \mathbf{v}_i is orthogonal to \mathbf{s}_j , $j \neq i$. Since \mathbf{P} is diagonal, it follows from (14.5.43) that the signal powers are

$$P_i = \frac{\mathbf{v}_i^H \mathbf{C}_{yy} \mathbf{v}_i}{|\mathbf{v}_i^H \mathbf{s}_i|^2}, \quad i = 1, 2, \dots, p \quad (14.5.44)$$

14.5.5 Order Selection Criteria

The eigenanalysis methods described in this section for estimating the frequencies and the powers of the sinusoids also provide information about the number of sinusoidal components. If there are p sinusoids, the eigenvalues associated with the signal subspace are $\{\lambda_i + \sigma_w^2, i = 1, 2, \dots, p\}$ while the remaining $(M - p)$ eigenvalues are all equal to σ_w^2 . Based on this eigenvalue decomposition, a test can be designed that compares the eigenvalues with a specified threshold. An alternative method also uses the eigenvector decomposition of the estimated autocorrelation matrix of the observed signal and is based on matrix perturbation analysis. This method is described in the paper by Fuchs (1988).

Another approach based on an extension and modification of the AIC criterion to the eigen-decomposition method has been proposed by Wax and Kailath (1985).

If the eigenvalues of the sample autocorrelation matrix are ranked so that $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_M$, where $M > p$, the number of sinusoids in the signal subspace is estimated by selecting the minimum value of $\text{MDL}(p)$, given as

$$\text{MDL}(p) = -\log \left[\frac{G(p)}{A(p)} \right]^N + E(p) \quad (14.5.45)$$

where

$$\begin{aligned} G(p) &= \prod_{i=p+1}^M \lambda_i, \quad p = 0, 1, \dots, M-1 \\ A(p) &= \left[\frac{1}{M-p} \sum_{i=p+1}^M \lambda_i \right]^{M-p} \\ E(p) &= \frac{1}{2} p(2M-p) \log N \end{aligned} \quad (14.5.46)$$

N : number of samples used to estimate the M autocorrelation lags

Some results on the quality of this order selection criterion are given in the paper by Wax and Kailath (1985). The MDL criterion is guaranteed to be consistent.

14.5.6 Experimental Results

In this section we illustrate with an example the resolution characteristics of the eigenanalysis-based spectral estimation algorithms and compare their performance with the model-based methods and nonparametric methods. The signal sequence is

$$x(n) = \sum_{i=1}^4 A_i e^{j(2\pi f_i n + \phi_i)} + w(n)$$

where $A_i = 1$, $i = 1, 2, 3, 4$, $\{\phi_i\}$ are statistically independent random variables uniformly distributed on $(0, 2\pi)$, $\{w(n)\}$ is a zero-mean, white noise sequence with variance σ_w^2 , and the frequencies are $f_1 = -0.222$, $f_2 = -0.166$, $f_3 = 0.10$, and $f_4 = 0.122$. The sequence $\{x(n), 0 \leq n \leq 1023\}$ is used to estimate the number of frequency components and the corresponding values of their frequencies for $\sigma_w^2 = 0.1, 0.5, 1.0$, and $M = 12$ (length of the estimated autocorrelation).

Figures 14.5.1, 14.5.2, 14.5.3, and 14.5.4 illustrate the estimated power spectra of the signal using the Blackman–Tukey method, the minimum-variance method of Capon, the AR Yule–Walker method, and the MUSIC algorithm, respectively. The results from the ESPRIT algorithm are given in Table 14.2. From these results it is apparent that (1) the Blackman–Tukey method does not provide sufficient resolution to estimate the sinusoids from the data; (2) the minimum-variance method of Capon resolves only the frequencies f_1 , f_2 but not f_3 and f_4 ; (3) the AR methods resolve all

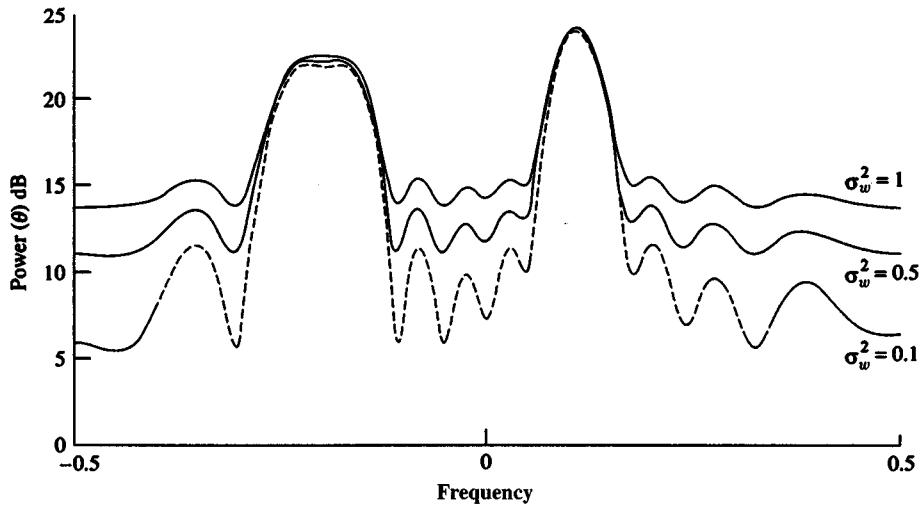


Figure 14.5.1 Power spectrum estimates from Blackman-Tukey method.

frequencies for $\sigma_w^2 = 0.1$ and $\sigma_w^2 = 0.5$; and (4) the MUSIC and ESPRIT algorithms not only recover all four sinusoids, but their performance for different values of σ_w^2 is essentially indistinguishable. We further observe that the resolution properties of the minimum variance method and the AR methods are functions of the noise variance. These results clearly demonstrate the power of the eigenanalysis-based algorithms in resolving sinusoids in additive noise.

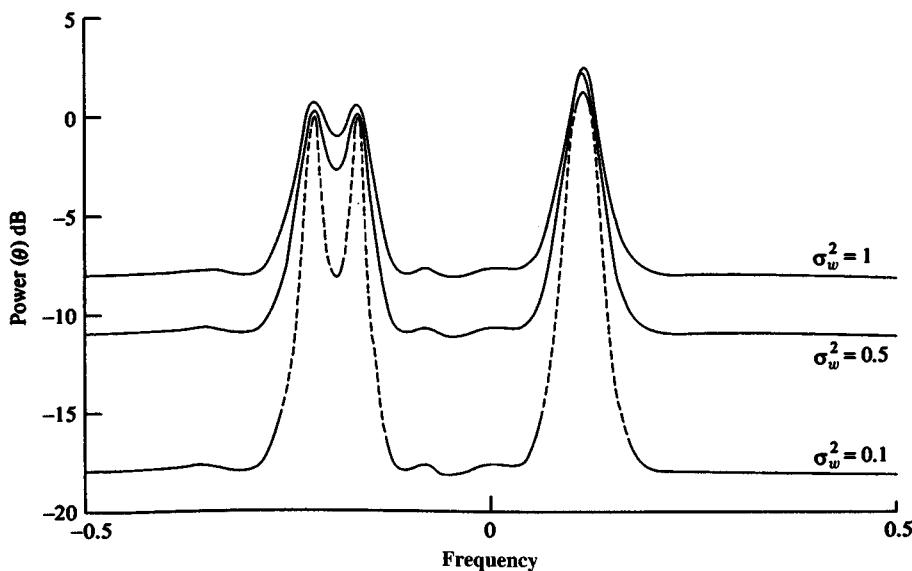


Figure 14.5.2 Power spectrum estimates from minimum-variance method.

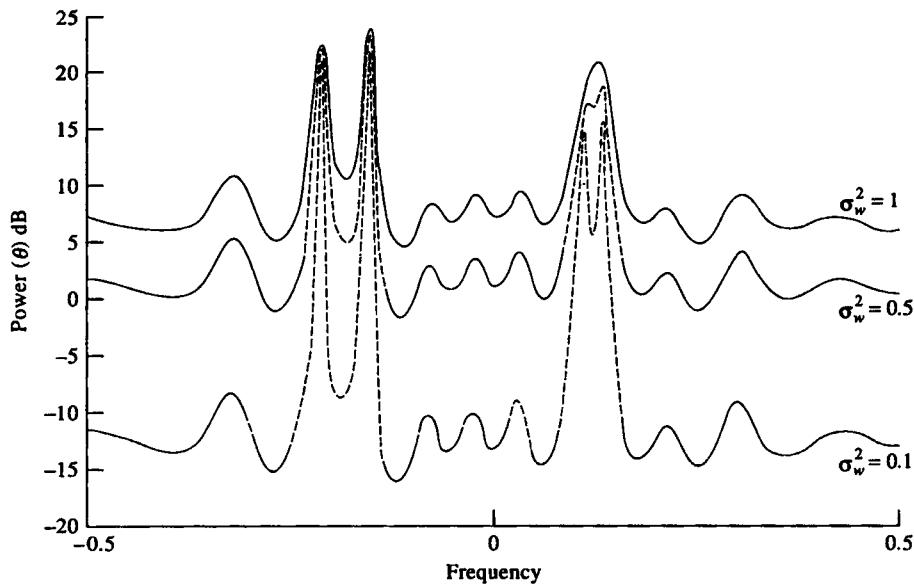


Figure 14.5.3 Power spectrum estimates from Yule–Walker AR method.

In conclusion, we should emphasize that the high-resolution, eigenanalysis-based spectral estimation methods described in this section, namely MUSIC and ESPRIT, are not only applicable to sinusoidal signals, but apply more generally to the estimation of narrowband signals.

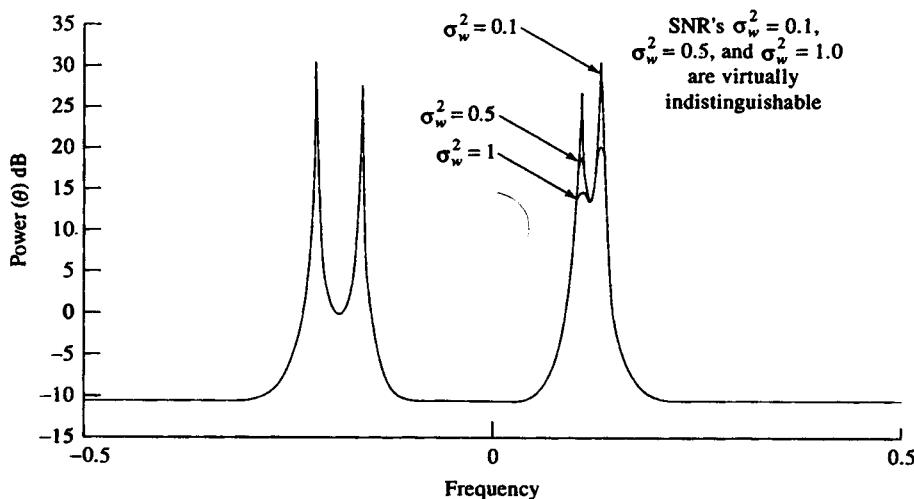


Figure 14.5.4 Power spectrum estimates from MUSIC algorithm.

TABLE 14.2 ESPRIT Algorithm

σ_w^2	\hat{f}_1	\hat{f}_2	\hat{f}_3	\hat{f}_4
0.1	-0.2227	-0.1668	-0.1224	-0.10071
0.5	-0.2219	-0.167	-0.121	0.0988
1.0	-0.222	-0.167	0.1199	0.1013
True values	-0.222	-0.166	0.122	0.100

14.6 Summary and References

Power spectrum estimation is one of the most important areas of research and applications in digital signal processing. In this chapter we have described the most important power spectrum estimation techniques and algorithms that have been developed over the past century, beginning with the nonparametric or classical methods based on the periodogram and concluding with the more modern parametric methods based on AR, MA, and ARMA linear models. Our treatment is limited in scope to single-time-series spectrum estimation methods, based on second moments (autocorrelation) of the statistical data.

The parametric and nonparametric methods that we described have been extended to multichannel and multidimensional spectrum estimation. The tutorial paper by McClellan (1982) treats the multidimensional spectrum estimation problem, while the paper by Johnson (1982) treats the multichannel spectrum estimation problem. Additional spectrum estimation methods have been developed for use with higher-order cumulants that involve the bispectrum and the trispectrum. A tutorial paper on these topics has been published by Nikias and Raghubeer (1987).

As evidenced from our previous discussion, power spectrum estimation is an area that has attracted many researchers and, as a result, thousands of papers have been published in the technical literature on this subject. Much of this work has been concerned with new algorithms and techniques, and modifications of existing techniques. Other work has been concerned with obtaining an understanding of the capabilities and limitations of the various power spectrum methods. In this context the statistical properties and limitations of the classical nonparametric methods have been thoroughly analyzed and are well understood. The parametric methods have also been investigated by many researchers, but the analysis of their performance is difficult and, consequently, fewer results are available. Some of the papers that have addressed the problem of performance characteristics of parametric methods are those of Kromer (1969), Lacoss (1971), Berk (1974), Baggeroer (1976), Sakai (1979), Swingler (1980), Lang and McClellan (1980), and Tufts and Kumaresan (1982).

In addition to the references already given in this chapter on the various methods for spectrum estimation and their performance, we should include for reference some of the tutorial and survey papers. In particular, we cite the tutorial paper by Kay and Marple (1981), which includes about 280 references, the paper by Brillinger (1974), and the Special Issue on Spectral Estimation of the *IEEE Proceedings*, September

1982. Another indication of the widespread interest in the subject of spectrum estimation and analysis is the publication of texts by Gardner (1987), Kay (1988), and Marple (1987), and the IEEE books edited by Chilvers (1978) and Kesler (1986).

Many computer programs as well as software packages that implement the various spectrum estimation methods described in this chapter are available. One software package is available through the IEEE (*Programs for Digital Signal Processing*, IEEE Press, 1979); others are available commercially.

Problems

- 14.1** (a) By expanding (14.1.23), taking the expected value, and finally taking the limit as $T_0 \rightarrow \infty$, show that the right-hand side converges to $\Gamma_{xx}(F)$.
 (b) Prove that

$$\sum_{m=-N}^N r_{xx}(m)e^{-j2\pi fm} = \frac{1}{N} \left| \sum_{n=0}^{N-1} x(n)e^{-j2\pi fn} \right|^2$$

- 14.2** For zero-mean, jointly Gaussian random variables, X_1, X_2, X_3, X_4 , it is well known [see Papoulis (1984)] that

$$E(X_1X_2X_3X_4) = E(X_1X_2)E(X_3X_4) + E(X_1X_3)E(X_2X_4) + E(X_1X_4)E(X_2X_3)$$

Use this result to derive the mean-square value of $r'_{xx}(m)$, given by (14.1.27) and the variance, which is

$$\text{var}[r'_{xx}(m)] = E[|r'_{xx}(m)|^2] - |E[r'_{xx}(m)]|^2$$

- 14.3** By use of the expression for the fourth joint moment for Gaussian random variables, show that

$$(a) E[P_{xx}(f_1)P_{xx}(f_2)] = \sigma_x^4 \left\{ 1 + \left[\frac{\sin \pi(f_1 + f_2)N}{N \sin \pi(f_1 + f_2)} \right]^2 + \left[\frac{\sin \pi(f_1 - f_2)N}{N \sin \pi(f_1 - f_2)} \right]^2 \right\}$$

$$(b) \text{cov}[P_{xx}(f_1)P_{xx}(f_2)] = \sigma_x^4 \left\{ \left[\frac{\sin \pi(f_1 + f_2)N}{N \sin \pi(f_1 + f_2)} \right]^2 + \left[\frac{\sin \pi(f_1 - f_2)N}{N \sin \pi(f_1 - f_2)} \right]^2 \right\}$$

$$(c) \text{var}[P_{xx}(f)] = \sigma_x^4 \left\{ 1 + \left(\frac{\sin 2\pi f N}{N \sin 2\pi f} \right)^2 \right\} \text{ under the condition that the sequence } x(n) \text{ is a zero-mean white Gaussian noise sequence with variance } \sigma_x^2.$$

- 14.4** Generalize the results in Problem 14.3 to a zero-mean Gaussian noise process with power density spectrum $\Gamma_{xx}(f)$. Then derive the variance of the periodogram $P_{xx}(f)$, as given by (14.1.38). (Hint: Assume that the colored Gaussian noise process is the output of a linear system excited by white Gaussian noise. Then use the appropriate relations given in Section 12.1.)

- 14.5** Show that the periodogram values at frequencies $f_k = k/L$, $k = 0, 1, \dots, L - 1$, given by (14.1.41) can be computed by passing the sequence through a bank of N IIR filters, where each filter has an impulse response

$$h_k(n) = e^{-j2\pi nk/N} u(n)$$

and then compute the magnitude-squared value of the filter outputs at $n = N$. Note that each filter has a pole on the unit circle at the frequency f_k .

- 14.6** Prove that the normalization factor given by (14.2.12) ensures that (14.2.19) is satisfied.
- 14.7** Let us consider the use of the DFT (computed via the FFT algorithm) to compute the autocorrelation of the complex-valued sequence $x(n)$, that is,

$$r_{xx}(m) = \frac{1}{N} \sum_{n=0}^{N-m-1} x^*(n)x(n+m), \quad m \geq 0$$

Suppose the size M of the FFT is much smaller than that of the data length N . Specifically, assume that $N = KM$.

- (a) Determine the steps needed to section $x(n)$ and compute $r_{xx}(m)$ for $-(M/2) + 1 \leq m \leq (M/2) - 1$, by using $4K$ M -point DFTs and one M -point IDFT.
- (b) Now consider the following three sequences $x_1(n)$, $x_2(n)$, and $x_3(n)$, each of duration M . Let the sequences $x_1(n)$ and $x_2(n)$ have arbitrary values in the range $0 \leq n \leq (M/2) - 1$, but be zero for $(M/2) \leq n \leq M - 1$. The sequence $x_3(n)$ is defined as

$$x_3(n) = \begin{cases} x_1(n), & 0 \leq \frac{M}{2} - 1 \\ x_2\left(n - \frac{M}{2}\right), & \frac{M}{2} \leq n \leq M - 1 \end{cases}$$

Determine a simple relationship among the M -point DFTs $X_1(k)$, $X_2(k)$, and $X_3(k)$.

- (c) By using the result in part (b), show how the computation of the DFTs in part (a) can be reduced in number from $4K$ to $2K$.
- 14.8** The Bartlett method is used to estimate the power spectrum of a signal $x(n)$. We know that the power spectrum consists of a single peak with a 3-dB bandwidth of 0.01 cycle per sample, but we do not know the location of the peak.

- (a) Assuming that N is large, determine the value of $M = N/K$ so that the spectral window is narrower than the peak.
- (b) Explain why it is not advantageous to increase M beyond the value obtained in part (a).

- 14.9** Suppose we have $N = 1000$ samples from a sample sequence of a random process.
- Determine the frequency resolution of the Bartlett, Welch (50% overlap), and Blackman–Tukey methods for a quality factor $Q = 10$.
 - Determine the record lengths (M) for the Bartlett, Welch (50% overlap), and Blackman–Tukey methods.
- 14.10** Consider the problem of continuously estimating the power spectrum from a sequence $x(n)$ based on averaging periodograms with exponential weighting into the past. Thus with $P_{xx}^{(0)}(f) = 0$, we have

$$P_{xx}^{(m)}(f) = w P_{xx}^{(m-1)}(f) + \frac{1-w}{M} \left| \sum_{n=0}^{M-1} x_m(n) e^{-j2\pi f n} \right|^2$$

where successive periodograms are assumed to be uncorrelated and w is the (exponential) weighting factor.

- Determine the mean and variance of $P_{xx}^{(m)}(f)$ for a Gaussian random process.
- Repeat the analysis of part (a) for the case in which the modified periodogram defined by Welch is used in the averaging with no overlap.

- 14.11** The periodogram in the Bartlett method can be expressed as

$$P_{xx}^{(i)}(f) = \sum_{m=-(M-1)}^{M-1} \left(1 - \frac{|m|}{M} \right) r_{xx}^{(i)}(m) e^{-j2\pi f m}$$

where $r_{xx}^{(i)}(m)$ is the estimated autocorrelation sequence obtained from the i th block of data. Show that $P_{xx}^{(i)}(f)$ can be expressed as

$$P_{xx}^{(i)}(f) = \mathbf{E}^H(f) \mathbf{R}_{xx}^{(i)} \mathbf{E}(f)$$

where

$$\mathbf{E}(f) = [1 \quad e^{j2\pi f} \quad e^{j4\pi f} \quad \dots \quad e^{j2\pi(M-1)f}]^T$$

and therefore,

$$P_{xx}^B(f) = \frac{1}{K} \sum_{k=1}^K \mathbf{E}^H(f) \mathbf{R}_{xx}^{(k)} \mathbf{E}(f)$$

- 14.12** Derive the recursive order-update equation given in (14.3.19).
- 14.13** Determine the mean and the autocorrelation of the sequence $x(n)$, which is the output of an ARMA (1, 1) process described by the difference equation

$$x(n) = \frac{1}{2}x(n-1) + w(n) - w(n-1)$$

where $w(n)$ is a white noise process with variance σ_w^2 .

- 14.14** Determine the mean and the autocorrelation of the sequence $x(n)$ generated by the MA(2) process described by the difference equation

$$x(n) = w(n) - 2w(n-1) + w(n-2)$$

where $w(n)$ is a white noise process with variance σ_w^2 .

- 14.15** An MA(2) process has the autocorrelation sequence

$$\gamma_{xx}(m) = \begin{cases} 6\sigma_w^2, & m = 0 \\ -4\sigma_w^2, & m = \pm 1 \\ -2\sigma_w^2, & m = \pm 2 \\ 0, & \text{otherwise} \end{cases}$$

- (a) Determine the coefficients of the MA(2) process that have the foregoing autocorrelation.

- (b) Is the solution unique? If not, give all the possible solutions.

- 14.16** An MA(2) process has the autocorrelation sequence

$$\gamma_{xx}(m) = \begin{cases} \sigma_w^2, & m = 0 \\ -\frac{35}{62}\sigma_w^2, & m = \pm 1 \\ \frac{6}{62}\sigma_w^2, & m = \pm 2 \end{cases}$$

- (a) Determine the coefficients of the minimum-phase system for the MA(2) process.

- (b) Determine the coefficients of the maximum-phase system for the MA(2) process.

- (c) Determine the coefficients of the mixed-phase system for the MA(2) process.

- 14.17** Consider the linear system described by the difference equation

$$y(n) = 0.8y(n-1) + x(n) + x(n-1)$$

where $x(n)$ is a wide-sense stationary random process with zero mean and autocorrelation

$$\gamma_{xx}(m) = \left(\frac{1}{2}\right)^{|m|}$$

- (a) Determine the power density spectrum of the output $y(n)$.

- (b) Determine the autocorrelation $\gamma_{yy}(m)$ of the output.

- (c) Determine the variance σ_y^2 of the output.

- 14.18** From (14.3.6) and (14.3.9) we note that an AR(p) stationary random process satisfies the equation

$$\gamma_{xx}(m) + \sum_{k=1}^p a_p(k)\gamma_{xx}(m-k) = \begin{cases} \sigma_w^2, & m = 0, \\ 0, & 1 \leq m \leq p, \end{cases}$$

where $a_p(k)$ are the prediction coefficients of the linear predictor of order p and σ_w^2 is the minimum mean-square prediction error. If the $(p+1) \times (p+1)$ autocorrelation matrix Γ_{xx} in (14.3.9) is positive definite, prove that:

- (a) The reflection coefficients $|K_m| < 1$ for $1 \leq m \leq p$.
 (b) The polynomial

$$A_p(z) = 1 + \sum_{k=1}^p a_p(k)z^{-k}$$

has all its roots inside the unit circle (i.e., it is minimum phase).

- 14.19** An AR(2) process is described by the difference equation

$$x(n) = 0.81x(n-2) + w(n)$$

where $w(n)$ is a white noise process with variance σ_w^2 .

- (a) Determine the parameters of the MA(2), MA(4), and MA(8) models which provide a minimum mean-square error fit to the data $x(n)$.
 (b) Plot the true spectrum and those of the MA(q), $q = 2, 4, 8$, spectra and compare the results. Comment on how well the MA(q) models approximate the AR(2) process.

- 14.20** An MA(2) process is described by the difference equation

$$x(n) = w(n) + 0.81w(n-2)$$

where $w(n)$ is a white noise process with variance σ_w^2 .

- (a) Determine the parameters of the AR(2), AR(4), and AR(8) models that provide a minimum mean-square error fit to the data $x(n)$.
 (b) Plot the true spectra and those of the AR(p), $p = 2, 4, 8$, and compare the results. Comment on how well the AR(p) models approximate the MA(2) process.

- 14.21** (a) Determine the power spectra for the random processes generated by the following difference equations.

1. $x(n) = -0.81x(n-2) + w(n) - w(n-1)$
2. $x(n) = w(n) - w(n-2)$
3. $x(n) = -0.81x(n-2) + w(n)$

where $w(n)$ is a white noise process with variance σ_w^2 .

- (b) Sketch the spectra for the processes given in part (a).
 (c) Determine the autocorrelation $\gamma_{xx}(m)$ for the processes in (2) and (3).

- 14.22** The Bartlett method is used to estimate the power spectrum of a signal from a sequence $x(n)$ consisting of $N = 2400$ samples.

- (a) Determine the smallest length M of each segment in the Bartlett method that yields a frequency resolution of $\Delta f = 0.01$.
 (b) Repeat part (a) for $\Delta f = 0.02$.
 (c) Determine the quality factors Q_B for parts (a) and (b).

14.23 A random process $x(n)$ is characterized by the power density spectrum

$$\Gamma_{xx}(f) = \sigma_w^2 \frac{|e^{j2\pi f} - 0.9|^2}{|e^{j2\pi f} - j0.9|^2 |e^{j2\pi f} + j0.9|^2}$$

where σ_w^2 is a constant (scale factor).

- (a) If we view $\Gamma_{xx}(f)$ as the power spectrum at the output of a linear pole-zero system $H(z)$ driven by white noise, determine $H(z)$.
- (b) Determine the system function of a stable system (noise-whitening filter) that produces a white noise output when excited by $x(n)$.

14.24 The N -point DFT of a random sequence $x(n)$ is

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi nk/N}$$

Assume that $E[x(n)] = 0$ and $E[x(n)x(n+m)] = \sigma_x^2 \delta(m)$ [i.e., $x(n)$ is a white noise process].

- (a) Determine the variance of $X(k)$.
- (b) Determine the autocorrelation of $X(k)$.

14.25 Suppose that we represent an ARMA(p, q) process as a cascade of an MA(q) followed by an AR(p) model. The input-output equation for the MA(q) model is

$$v(n) = \sum_{k=0}^q b_k w(n-k)$$

where $w(n)$ is a white noise process. The input-output equation for the AR(p) model is

$$x(n) + \sum_{k=1}^p a_k x(n-k) = v(n)$$

- (a) By computing the autocorrelation of $v(n)$, show that

$$\gamma_{vv}(m) = \sigma_w^2 \sum_{k=0}^{q-m} b_k^* b_{k+m}$$

- (b) Show that

$$\gamma_{vv}(m) = \sum_{k=0}^p a_k \gamma_{vx}(m+k), \quad a_0 = 1$$

where $\gamma_{vx}(m) = E[v(n+m)x^*(n)]$.

- 14.26** Determine the autocorrelation $\gamma_{xx}(m)$ of the random sequence

$$x(n) = A \cos(\omega_1 n + \phi)$$

where the amplitude A and the frequency ω_1 are (known) constants and ϕ is a uniformly distributed random phase over the interval $(0, 2\pi)$.

- 14.27** Suppose that the AR(2) process in Problem 14.19 is corrupted by an additive white noise process $v(n)$ with variance σ_v^2 . Thus we have

$$y(n) = x(n) + v(n)$$

- (a) Determine the difference equation for $y(n)$ and thus demonstrate that $y(n)$ is an ARMA(2, 2) process. Determine the coefficients of the ARMA process.
 (b) Generalize the result in part (a) to an AR(p) process

$$x(n) = - \sum_{k=1}^p a_k (x_{n-k}) + w(n)$$

and

$$y(n) = x(n) + v(n)$$

- 14.28 (a)** Determine the autocorrelation of the random sequence

$$x(n) = \sum_{k=1}^K A_k \cos(\omega_k n + \phi_k) + w(n)$$

where $\{A_k\}$ are constant amplitudes, $\{\omega_k\}$ are constant frequencies, and $\{\phi_k\}$ are mutually statistically independent and uniformly distributed random phases. The noise sequence $w(n)$ is white with variance σ_w^2 .

- (b) Determine the power density spectrum of $x(n)$.

- 14.29** The harmonic decomposition problem considered by Pisarenko can be expressed as the solution to the equation

$$\mathbf{a}^H \boldsymbol{\Gamma}_{yy} \mathbf{a} = \sigma_w^2 \mathbf{a}^H \mathbf{a}$$

The solution for \mathbf{a} can be obtained by minimizing the quadratic form $\mathbf{a}^H \boldsymbol{\Gamma}_{yy} \mathbf{a}$ subject to the constraint that $\mathbf{a}^H \mathbf{a} = 1$. The constraint can be incorporated into the performance index by means of a Lagrange multiplier. Thus the performance index becomes

$$\mathcal{E} = \mathbf{a}^H \boldsymbol{\Gamma}_{yy} \mathbf{a} + \lambda(1 - \mathbf{a}^H \mathbf{a})$$

By minimizing \mathcal{E} with respect to \mathbf{a} , show that this formulation is equivalent to the Pisarenko eigenvalue problem given in (14.5.9) with the Lagrange multiplier playing the role of the eigenvalue. Thus show that the minimum of \mathcal{E} is the minimum eigenvalue σ_w^2 .

- 14.30** The autocorrelation of a sequence consisting of a sinusoid with random phase in noise is

$$\gamma_{xx}(m) = P \cos 2\pi f_1 m + \sigma_w^2 \delta(m)$$

where f_1 is the frequency of the sinusoidal, P is its power, and σ_w^2 is the variance of the noise. Suppose that we attempt to fit an AR(2) model to the data.

- (a) Determine the optimum coefficients of the AR(2) model as a function of σ_w^2 and f_1 .
- (b) Determine the reflection coefficients K_1 and K_2 corresponding to the AR(2) model parameters.
- (c) Determine the limiting values of the AR(2) parameters and (K_1, K_2) as $\sigma_w^2 \rightarrow 0$.

- 14.31** The minimum variance power spectrum estimate described in Section 14.4 is determined by minimizing the variance

$$\sigma_y^2 = \mathbf{h}^H \boldsymbol{\Gamma}_{xx} \mathbf{h}$$

subject to the constraint

$$\mathbf{E}^H(f) \mathbf{h} = 1$$

where $\mathbf{E}(f)$ is defined as the vector

$$\mathbf{E}^t(f) = [1 \quad e^{j2\pi f} \quad e^{j4\pi f} \quad \dots \quad e^{j2\pi pf}]$$

To determine the optimum filter that minimizes σ_y^2 , let us define the function

$$\mathcal{E}(\mathbf{h}) = \mathbf{h}^H \boldsymbol{\Gamma}_{xx} \mathbf{h} + \mu(1 - \mathbf{E}^H(f) \mathbf{h}) + \mu^*(1 - \mathbf{h}^H \mathbf{E}(f))$$

where μ is a Lagrange multiplier

- (a) By differentiating $\mathcal{E}(\mathbf{h})$ and setting the derivative to zero, show that

$$\mathbf{h}_{\text{opt}} = \mu^* \boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f)$$

- (b) Solve for μ^* by using the constraint and, thus, show that

$$\mathbf{h}_{\text{opt}} = \frac{\boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f)}{\mathbf{E}^H(f) \boldsymbol{\Gamma}_{xx}^{-1} \mathbf{E}(f)}$$

- 14.32** The periodogram spectral estimate is expressed as

$$P_{xx}(f) = \frac{1}{N} |X(f)|^2$$

where

$$\begin{aligned} X(f) &= \sum_{n=0}^{N-1} x(n) e^{-j2\pi f n} \\ &= \mathbf{E}^H(f) \mathbf{X}(n) \end{aligned}$$

$\mathbf{X}(f)$ is the data vector and $\mathbf{E}(f)$ is defined as

$$\mathbf{E}^t(f) = [1 \quad e^{j2\pi f} \quad e^{j4\pi f} \quad \dots \quad e^{j2\pi f(N-1)}]$$

Show that

$$E[P_{xx}(f)] = \frac{1}{N} \mathbf{E}^H(f) \mathbf{\Gamma}_{xx} \mathbf{E}(f)$$

where $\mathbf{\Gamma}_{xx}$ is the autocorrelation matrix of the data vector $\mathbf{X}(n)$.

- 14.33** Determine the frequency and power of a single real sinusoid in white noise. The signal and noise correlation function is given as

$$\gamma_{yy}(m) = \begin{cases} 3, & m = 0 \\ 0, & m = 1 \\ -2, & m = 2 \\ 0, & |m| > 2 \end{cases}$$

- 14.34** The signal $y(n)$ consists of complex exponentials in white noise. Its autocorrelation matrix is

$$\mathbf{\Gamma}_{yy} = \begin{bmatrix} 2 & -j & -1 \\ j & 2 & -j \\ -1 & j & 2 \end{bmatrix}$$

Use the MUSIC algorithm to determine the frequencies of the exponentials and their power levels.

- 14.35** Show that $P_{\text{MUSIC}}(f)$ can be expressed as

$$P_{\text{MUSIC}}(f) = \frac{1}{\mathbf{s}^H(f) \left(\sum_{k=p+1}^M \mathbf{v}_k \mathbf{v}_k^H \right) \mathbf{s}(f)}$$

- 14.36 Root MUSIC algorithm** Let us define the noise subspace polynomials as

$$V_k(z) = \sum_{n=0}^{M-1} v_k(n+1) z^{-n}, \quad k = p+1, \dots, M$$

where $v_k(n)$ are the elements of the noise subspace vector \mathbf{v}_k .

- (a) Show that $P_{\text{MUSIC}}(f)$ can be expressed as

$$\begin{aligned} P_{\text{MUSIC}}(f) &= \frac{1}{\sum_{k=p+1}^M V_k(f) V_k^*(f)} \\ &= \frac{1}{\sum_{k=p+1}^M V_k(z) V_k^* \left(\frac{1}{z^*} \right) \Big|_{z=e^{j2\pi f}}} \end{aligned}$$

(b) For p complex sinusoids in white noise, the polynomial

$$Q(z) = \sum_{k=p+1}^M V_k(z) V_k^* \left(\frac{1}{z^*} \right)$$

goes to zero at $z = e^{j2\pi f_i}$, $i = 1, 2, \dots, p$. Hence, this polynomial has p roots on the unit circle in the z -plane, where each root is a double root. The determination of the frequencies by factoring the polynomial $Q(z)$ is called the *root MUSIC method*. For the correlation matrix given in Problem 14.34, determine the frequencies of the exponentials by the root MUSIC method. The extra roots obtained by this method are spurious roots and may be discarded.

- 14.37** This problem involves the use of crosscorrelation to detect a signal in noise and estimate the time delay in the signal. A signal $x(n)$ consists of a pulsed sinusoid corrupted by a stationary zero-mean white noise sequence. That is,

$$x(n) = y(n - n_0) + w(n), \quad 0 \leq n \leq N - 1$$

where $w(n)$ is the noise with variance σ_w^2 and the signal is

$$\begin{aligned} y(n) &= A \cos \omega_0 n, & 0 \leq n \leq M - 1 \\ &= 0, & \text{otherwise} \end{aligned}$$

The frequency ω_0 is known but the delay n_0 , which is a positive integer, is unknown, and is to be determined by crosscorrelating $x(n)$ with $y(n)$. Assume that $N > M + n_0$. Let

$$r_{xy}(m) = \sum_{n=0}^{N-1} y(n - m) x(n)$$

denote the crosscorrelation sequence between $x(n)$ and $y(n)$. In the absence of noise this function exhibits a peak at delay $m = n_0$. Thus n_0 is determined with no error. The presence of noise can lead to errors in determining the unknown delay.

- (a)** For $m = n_0$, determine $E[r_{xy}(n_0)]$. Also, determine the variance, $\text{var}[r_{xy}(n_0)]$, due to the presence of the noise. In both calculations, assume that the double frequency term averages to zero. That is, $M \gg 2\pi/\omega_0$.
- (b)** Determine the signal-to-noise ratio, defined as

$$\text{SNR} = \frac{\{E[r_{xy}(n_0)]\}^2}{\text{var}[r_{xy}(n_0)]}$$

- (c)** What is the effect of the pulse duration M on the SNR?

- 14.38** Generate 100 samples of a zero-mean white noise sequence $w(n)$ with variance $\sigma_w^2 = \frac{1}{12}$, by using a uniform random number generator.

- (a) Compute the autocorrelation of $w(n)$ for $0 \leq m \leq 15$.
- (b) Compute the periodogram estimate $P_{xx}(f)$ and plot it.
- (c) Generate 10 different realizations of $w(n)$ and compute the corresponding sample autocorrelation sequences $r_k(m)$, $1 \leq k \leq 10$ and $0 \leq m \leq 15$.
- (d) Compute and plot the average autocorrelation sequence for part (c):

$$r_{av}(m) = \frac{1}{10} \sum_{k=1}^{10} r_k(m)$$

and the periodogram corresponding to $r_{av}(m)$.

- (e) Comment on the results in parts (a) through (d).

- 14.39** A random signal is generated by passing zero-mean white Gaussian noise with unit variance through a filter with system function

$$H(z) = \frac{1}{(1 + az^{-1} + 0.99z^{-2})(1 - az^{-1} + 0.98z^{-2})}$$

- (a) Sketch a typical plot of the theoretical power spectrum $\Gamma_{xx}(f)$ for a small value of the parameter a (i.e., $0 < a < 0.1$). Pay careful attention to the value of the two spectral peaks and the value of $P_{xx}(\omega)$ for $\omega = \pi/2$.
- (b) Let $a = 0.1$. Determine the section length M required to resolve the spectral peaks of $\Gamma_{xx}(f)$ when using Bartlett's method.
- (c) Consider the Blackman–Tukey method of smoothing the periodogram. How many lags of the correlation estimate must be used to obtain resolution comparable to that of the Bartlett estimate considered in part (b)? How many data points must be used if the variance of the estimate is to be comparable to that of a four-section Bartlett estimate?
- (d) For $a = 0.05$, fit an AR(4) model to 100 samples of the data based on the Yule–Walker method and plot the power spectrum. Avoid transient effects by discarding the first 200 samples of the data.
- (e) Repeat part (d) with the Burg method.
- (f) Repeat parts (d) and (e) for 50 data samples and comment on similarities and differences in the results.

A

Random Number Generators

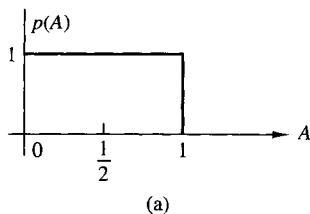
In some of the examples given in the text, random numbers are generated to simulate the effect of noise on signals and to illustrate how the method of correlation can be used to detect the presence of a signal buried in noise. In the case of periodic signals, the correlation technique also allowed us to estimate the period of the signal.

In practice, random number generators are often used to simulate the effect of noiselike signals and other random phenomena encountered in the physical world. Such noise is present in electronic devices and systems and usually limits our ability to communicate over large distances and to be able to detect relatively weak signals. By generating such noise on a computer, we are able to study its effects through simulation of communication systems, radar detection systems, and the like and to assess the performance of such systems in the presence of noise.

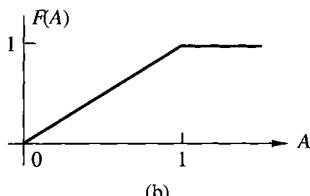
Most computer software libraries include a uniform random number generator. Such a random number generator generates a number between zero and 1 with equal probability. We call the output of the random number generator a random variable. If A denotes such a random variable, its range is the interval $0 \leq A \leq 1$.

We know that the numerical output of a digital computer has limited precision, and as a consequence, it is impossible to represent the continuum of numbers in the interval $0 \leq A \leq 1$. However, we can assume that our computer represents each output by a large number of bits in either fixed point or floating point. Consequently, for all practical purposes, the number of outputs in the interval $0 \leq A \leq 1$ is sufficiently large, so that we are justified in assuming that any value in the interval is a possible output from the generator.

The uniform probability density function for the random variable A , denoted as $p(A)$, is illustrated in Fig. A.1(a). We note that the average value or mean value of A , denoted as m_A , is $m_A = \frac{1}{2}$. The integral of the probability density function, which



(a)



(b)

Figure A.1

represents the area under $p(A)$, is called the probability distribution function of the random variable A and is defined as

$$F(A) = \int_{-\infty}^A p(x)dx \quad (\text{A.1})$$

For any random variable, this area must always be unity, which is the maximum value that can be achieved by a distribution function. Hence

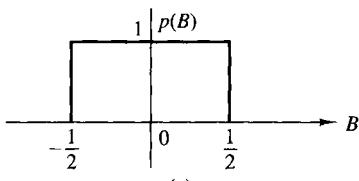
$$F(1) = \int_{-\infty}^1 p(x)dx = 1 \quad (\text{A.2})$$

and the range of $F(A)$ is $0 \leq F(A) \leq 1$ for $0 \leq A \leq 1$.

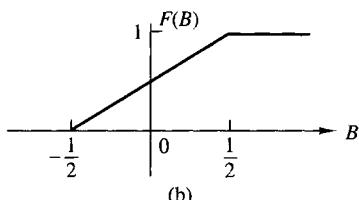
If we wish to generate uniformly distributed noise in an interval $(b, b + 1)$ it can simply be accomplished by using the output A of the random number generator and shifting it by an amount b . Thus a new random variable B can be defined as

$$B = A + b \quad (\text{A.3})$$

which now has a mean value $m_B = b + \frac{1}{2}$. For example, if $b = -\frac{1}{2}$, the random variable B is uniformly distributed in the interval $(-\frac{1}{2}, \frac{1}{2})$, as shown in Fig A.2(a). Its probability distribution function $F(B)$ is shown in Fig A.2(b).



(a)



(b)

Figure A.2

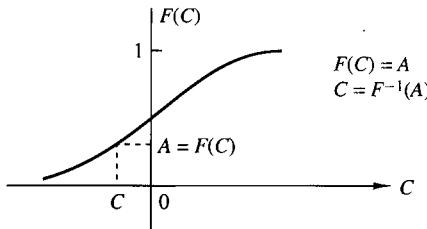


Figure A.3

A uniformly distributed random variable in the range $(0, 1)$ can be used to generate random variables with other probability distribution functions. For example, suppose that we wish to generate a random variable C with probability distribution function $F(C)$, as illustrated in Fig A.3. Since the range of $F(C)$ is the interval $(0, 1)$, we begin by generating a uniformly distributed random variable A in the range $(0, 1)$. If we set

$$F(C) = A \quad (\text{A.4})$$

then

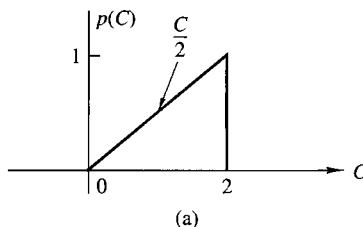
$$C = F^{-1}(A) \quad (\text{A.5})$$

Thus we solve (A.4) for C , and the solution in (A.5) provides the value of C for which $F(C) = A$. By this means we obtain a new random variable C with probability distribution $F(C)$. This inverse mapping from A to C is illustrated in Fig A.3.

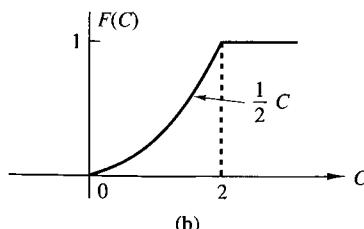
EXAMPLE A.1

Generate a random variable C that has the linear probability density function shown in Fig A.4(a), that is,

$$p(C) = \begin{cases} \frac{C}{2}, & 0 \leq C \leq 2 \\ 0, & \text{otherwise} \end{cases}$$



(a)



(b)

Figure A.4

Solution. This random variable has a probability distribution function

$$F(C) = \begin{cases} 0, & C < 0 \\ \frac{1}{4}C^2, & 0 \leq C \leq 2 \\ 1, & C > 2 \end{cases}$$

which is illustrated in Fig A.4(b). We generate a uniformly distributed random variable A and set $F(C) = A$. Hence

$$F(C) = \frac{1}{4}C^2 = A$$

Upon solving for C , we obtain

$$C = 2\sqrt{A}$$

Thus we generate a random variable C with probability function $F(C)$, as shown in Fig A.4(b).

In Example A.1 the inverse mapping $C = F^{-1}(A)$ was simple. In some cases it is not. This problem arises in trying to generate random numbers that have a normal distribution function.

Noise encountered in physical systems is often characterized by the normal or Gaussian probability distribution, which is illustrated in Fig A.5. The probability density function is given by

$$p(C) = \frac{1}{\sqrt{2\pi}\sigma} e^{-C^2/2\sigma^2}, \quad -\infty < C < \infty \quad (\text{A.6})$$

where σ^2 is the variance of C , which is a measure of the spread of the probability density function $p(C)$. The probability distribution function $F(C)$ is the area under $p(C)$ over the range $(-\infty, C)$. Thus

$$F(C) = \int_{-\infty}^C p(x)dx \quad (\text{A.7})$$

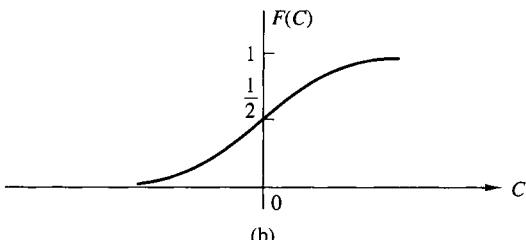
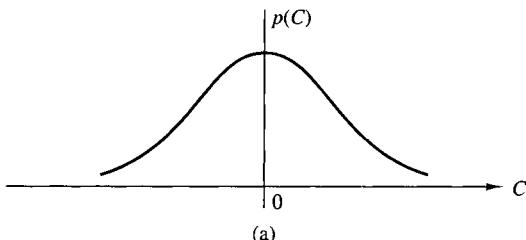


Figure A.5

Unfortunately, the integral in (A.7) cannot be expressed in terms of simple functions. Consequently, the inverse mapping is difficult to achieve.

A way has been found to circumvent this problem. From probability theory it is known that a (Rayleigh distributed) random variable R , with probability distribution function

$$F(R) = \begin{cases} 0, & R < 0 \\ 1 - e^{-R^2/2\sigma^2}, & R \geq 0 \end{cases} \quad (\text{A.8})$$

is related to a pair of Gaussian random variables C and D , through the transformation

$$C = R \cos \Theta \quad (\text{A.9})$$

$$D = R \sin \Theta \quad (\text{A.10})$$

where Θ is a uniformly distributed variable in the interval $(0, 2\pi)$. The parameter σ^2 is the variance of C and D . Since (A.8) is easily inverted, we have

$$F(R) = 1 - e^{-R^2/2\sigma^2} = A$$

and hence

$$R = \sqrt{2\sigma^2 \ln[1/(1 - A)]} \quad (\text{A.11})$$

where A is a uniformly distributed random variable in the interval $(0, 1)$. Now if we generate a second uniformly distributed random variable B and define

$$\Theta = 2\pi B \quad (\text{A.12})$$

then from (A.9) and (A.10), we obtain two statistically independent Gaussian distributed random variables C and D .

```

C      SUBROUTINE GAUSS CONVERTS A UNIFORM RANDOM
C      SEQUENCE XIN IN [0,1] TO A GAUSSIAN RANDOM
C      SEQUENCE WITH G(0,SIGMA**2)
C      PARAMETERS :
C          XIN    :UNIFORM IN [0,1] RANDOM NUMBER
C          B      :UNIFORM IN [0,1] RANDOM NUMBER
C          SIGMA  :STANDARD DEVIATION OF THE GAUSSIAN
C          YOUT   :OUTPUT FROM THE GENERATOR
C
C      SUBROUTINE GAUSS 9XIN,B,SIGMA,YOUT)
PI=4.0*ATAN (1.0)
B=2.0*PI*B
R=SQRT (2.0*(SIGMA**2)*ALOG(1.0/(1.0-XIN)))
YOUT=R*COS(B)
RETURN
END
C      NOTE: TO USE THE ABOVE SUBROUTINE FOR A
C      GAUSSIAN RANDOM NUMBER GENERATOR
C      YOU MUST PROVIDE AS INPUT TWO UNIFORM RANDOM NUMBERS
C      XIN AND B
C      XIN AND B MUST BE STATISTICALLY INDEPENDENT
C

```

Figure A.6 Subroutine for generating Gaussian random variables

The method described above is often used in practice to generate Gaussian distributed random variables. As shown in Fig A.5, these random variables have a mean value of zero and a variance σ^2 . If a nonzero mean Gaussian random variable is desired, then C and D can be translated by the addition of the mean value.

A subroutine implementing this method for generating Gaussian distributed random variables is given in Fig A.6.

B

Tables of Transition Coefficients for the Design of Linear-Phase FIR Filters

In Section 10.2.3 we described a design method for linear-phase FIR filters that involved the specification of $H_r(\omega)$ at a set of equally spaced frequencies $\omega_k = 2\pi(k + \alpha)/M$, where $\alpha = 0$ or $\alpha = \frac{1}{2}$, $k = 0, 1, \dots, (M - 1)/2$ for M odd and $k = 0, 1, 2, \dots, (M/2) - 1$ for M even, where M is the length of the filter. Within the passband of the filter, we select $H_r(\omega_k) = 1$, and in the stopband, $H_r(\omega_k) = 0$. For frequencies in the transition band, the values of $H_r(\omega_k)$ are optimized to minimize the maximum sidelobe in the stopband. This is called a *minimax optimization criterion*.

The optimization of the values of $H_r(\omega)$ in the transition band has been performed by Rabiner et al. (1970) and tables of transition values have been provided in the published paper. A selected number of the tables for lowpass FIR filters are included in this appendix.

Four tables are given. Table B.1 lists the transition coefficients for the case $\alpha = 0$ and one coefficient in the transition band for both M odd and M even. Table B.2 lists the transition coefficients for the case $\alpha = 0$, and two coefficients in the transition band for M odd and M even. Table B.3 lists the transition coefficients for the case $\alpha = \frac{1}{2}$, M even and one coefficient in the transition band. Finally, Table B.4 lists the transition coefficients for the case $\alpha = \frac{1}{2}$, M even, and two coefficients in the transition band. The tables also include the level of the maximum sidelobe and a bandwidth parameter, denoted as BW.

To use the tables, we begin with a set of specifications, including (1) the bandwidth of the filter, which can be defined as $(2\pi/M)(BW + \alpha)$, where BW is the number of consecutive frequencies at which $H(\omega_k) = 1$, (2) the width of the transition region, which is roughly $2\pi/M$ times the number of transition coefficients, and (3) the maximum tolerable sidelobe in the stopband. The length of the filter can be selected from the tables to satisfy the specifications.

TABLE B.1 Transition Coefficients for $\alpha = 0$

M Odd			M Even		
BW	Minimax	T_1	BW	Minimax	T_1
$M = 15$			$M = 16$		
1	-42.30932283	0.43378296	1	-39.75363827	0.42631836
2	-41.26299286	0.41793823	2	-37.61346340	0.40397949
3	-41.25333786	0.41047636	3	-36.57721567	0.39454346
4	-41.94907713	0.40405884	4	-35.87249756	0.38916626
5	-44.37124538	0.39268189	5	-35.31695461	0.38840332
6	-56.01416588	0.35766525	6	-35.51951933	0.40155639
$M = 33$			$M = 32$		
1	-43.03163004	0.42994995	1	-42.24728918	0.42856445
2	-42.42527962	0.41042481	2	-41.29370594	0.40773926
3	-42.40898275	0.40141601	3	-41.03810358	0.39662476
4	-42.45948601	0.39641724	4	-40.93496323	0.38925171
6	-42.52403450	0.39161377	5	-40.85183477	0.37897949
8	-42.44085121	0.39039917	8	-40.75032616	0.36990356
10	-42.11079407	0.39192505	10	-40.54562140	0.35928955
12	-41.92705250	0.39420166	12	-39.93450451	0.34487915
14	-44.69430351	0.38552246	14	-38.91993237	0.34407349
15	-56.18293285	0.35360718	$M = 64$		
1	-43.16935968	0.42919312	1	-42.96059322	0.42882080
2	-42.61945581	0.40903320	2	-42.30815172	0.40830689
3	-42.70906305	0.39920654	3	-42.32423735	0.39807129
4	-42.86997318	0.39335937	4	-42.43565893	0.39177246
5	-43.01999664	0.38950806	5	-42.55461407	0.38742065
6	-43.14578819	0.38679809	6	-42.66526604	0.38416748
10	-43.44808340	0.38129272	10	-43.01104736	0.37609863
14	-43.54684496	0.37946167	14	-43.28309965	0.37089233
18	-43.48173618	0.37955322	18	-43.56508827	0.36605225
22	-43.19538212	0.38162842	22	-43.96245098	0.35977783
26	-42.44725609	0.38746948	26	-44.60516977	0.34813232
30	-44.76228619	0.38417358	30	-43.81448936	0.29973144
31	-59.21673775	0.35282745	$M = 128$		
1	-43.20501566	0.42899170	1	-43.15302420	0.42889404
2	-42.66971111	0.40867310	2	-42.59092569	0.40847778
3	-42.77438974	0.39868774	3	-42.67634487	0.39838257
4	-42.95051050	0.39268189	4	-42.84038544	0.39226685
6	-43.25854683	0.38579101	5	-42.99805641	0.38812256
8	-43.47917461	0.38195801	7	-43.25537014	0.38281250
10	-43.63750410	0.37954102	10	-43.52547789	0.3782638
18	-43.95589399	0.37518311	18	-43.93180990	0.37251587
26	-44.05913115	0.37384033	26	-44.18097305	0.36941528
34	-44.05672455	0.37371826	34	-44.40153408	0.36686401
42	-43.94708776	0.37470093	42	-44.67161417	0.36394653
50	-43.58473492	0.37797851	50	-45.17186594	0.35902100
58	-42.14925432	0.39086304	58	-46.92415667	0.34273681
59	-42.60623264	0.39063110	62	-49.46298973	0.28751221
60	-44.78062010	0.38383713			
61	-56.22547865	0.35263062			

Source: Rabiner et al. (1970); © 1970 IEEE; reprinted with permission.

TABLE B.2 Transition Coefficients for $\alpha = 0$

BW	M Odd			BW	M Even		
	Minimax	T_1	T_2		Minimax	T_1	T_2
$M = 15$							
1	-70.60540585	0.09500122	0.58995418	1	-65.27693653	0.10703125	0.60559357
2	-69.26168156	0.10319824	0.59357118	2	-62.85937929	0.12384644	0.62201631
3	-69.91973495	0.10083618	0.58594327	3	-62.96594906	0.12827148	0.62855407
4	-75.51172256	0.08407953	0.55715312	4	-66.03942485	0.12130127	0.61952704
5	-103.45078300	0.05180206	0.49917424	5	-71.73997498	0.11066284	0.60979204
$M = 33$							
1	-70.60967541	0.09497070	0.58985167	1	-67.37020397	0.09610596	0.59045212
2	-68.16726971	0.10585937	0.59743846	2	-63.93104696	0.11263428	0.60560235
3	-67.13149548	0.10937500	0.59911696	3	-62.49787903	0.11931763	0.61192546
5	-66.53917217	0.10965576	0.59674101	5	-61.28204536	0.12541504	0.61824023
7	-67.23387909	0.10902100	0.59417456	7	-60.82049131	0.12907715	0.62307031
9	-67.85412312	0.10502930	0.58771575	9	-59.74928167	0.12068481	0.60685586
11	-69.08597469	0.10219727	0.58216391	11	-62.48683357	0.13004150	0.62821502
13	-75.86953640	0.08137207	0.54712777	13	-70.64571857	0.11017914	0.60670943
14	-104.04059029	0.05029373	0.49149549	$M = 64$			
$M = 65$							
1	-70.66014957	0.09472656	0.58945943	1	-70.26372528	0.09376831	0.58789222
2	-68.89622307	0.10404663	0.59476127	2	-67.20729542	0.10411987	0.59421778
3	-67.90234470	0.10720215	0.59577449	3	-65.80684280	0.10850220	0.59666158
4	-67.24003792	0.10726929	0.59415763	4	-64.95227051	0.11038818	0.59730067
5	-66.86065960	0.10689087	0.59253047	5	-64.42742348	0.11113281	0.59698496
9	-66.27561188	0.10548706	0.58845983	9	-63.41714096	0.10936890	0.59088884
13	-65.96417046	0.10466309	0.58660485	13	-62.72142410	0.10828857	0.58738641
17	-66.16404629	0.10649414	0.58862042	17	-62.37051868	0.11031494	0.58968142
21	-66.76456833	0.10701904	0.58894575	21	-62.04848146	0.11254273	0.59249461
25	-68.13407993	0.10327148	0.58320831	25	-61.88074064	0.11994629	0.60564501
29	-75.98313046	0.08069458	0.54500379	29	-70.05681992	0.10717773	0.59842159
30	-104.92083740	0.04978485	0.48965181	$M = 128$			
$M = 125$							
1	-70.68010235	0.09464722	0.58933268	1	-70.58992958	0.09445190	0.58900996
2	-68.94157696	0.10390015	0.59450024	2	-68.62421608	0.10349731	0.59379058
3	-68.19352627	0.10682373	0.59508549	3	-67.66701698	0.10701294	0.59506081
5	-67.34261131	0.10668945	0.59187505	4	-66.95196629	0.10685425	0.59298926
7	-67.09767151	0.10587158	0.59821869	6	-66.32718945	0.10596924	0.58953845
9	-67.05801296	0.10523682	0.58738706	9	-66.01315498	0.10471191	0.58593906
17	-67.17504501	0.10372925	0.58358265	17	-65.89422417	0.10288086	0.58097354
25	-67.22918987	0.10316772	0.58224835	25	-65.92644215	0.10182495	0.57812308
33	-67.11609936	0.10303955	0.58198956	33	-65.95577812	0.10096436	0.57576437
41	-66.71271324	0.10313721	0.58245499	41	-65.97698021	0.10094604	0.57451694
49	-66.62364197	0.10561523	0.58629534	49	-65.67919827	0.09865112	0.56927420
57	-69.28378487	0.10061646	0.57812192	57	-64.61514568	0.09845581	0.56604486
58	-70.35782337	0.09663696	0.57121235	61	-71.76589394	0.10496826	0.59452277
59	-75.94707718	0.08054886	0.54451285				
60	-104.09012318	0.04991760	0.48963264				

Source: Rabiner et al. (1970); © 1970 IEEE; reprinted with permission.

TABLE B.3 Transition Coefficients for $\alpha = \frac{1}{2}$

BW	Minimax	T_1
$M = 16$		
1	-51.60668707	0.26674805
2	-47.48000240	0.32149048
3	-45.19746828	0.34810181
4	-44.32862616	0.36308594
5	-45.68347692	0.36661987
6	-56.63700199	0.34327393
$M = 32$		
1	-52.64991188	0.26073609
2	-49.39390278	0.30878296
3	-47.72596645	0.32984619
4	-46.68811989	0.34217529
6	-45.33436489	0.35704956
8	-44.30730963	0.36750488
10	-43.11168003	0.37810669
12	-42.97900438	0.38465576
14	-56.32780266	0.35030518
$M = 64$		
1	-52.90375662	0.25923462
2	-49.74046421	0.30603638
3	-48.38088989	0.32510986
4	-47.47863007	0.33595581
5	-46.88655186	0.34287720
6	-46.46230555	0.34774170
10	-45.46141434	0.35859375
14	-44.85988188	0.36470337
18	-44.34302616	0.36983643
22	-43.69835377	0.37586059
26	-42.45641375	0.38624268
30	-56.25024033	0.35200195
$M = 128$		
1	-52.96778202	0.25885620
2	-49.82771969	0.30534668
3	-48.51341629	0.32404785
4	-47.67455149	0.33443604
5	-47.11462021	0.34100952
7	-46.43420267	0.34880371
10	-45.88529110	0.35493774
18	-45.21660566	0.36182251
26	-44.87959814	0.36521607
34	-44.61497784	0.36784058
42	-44.32706451	0.37066040
50	-43.87646437	0.37500000
58	-42.30969715	0.38807373
62	-56.23294735	0.35241699

Source: Rabiner et al. (1970); © 1970 IEEE; reprinted with permission.

TABLE B.4 Transition Coefficients for $\alpha = \frac{1}{2}$

BW	Minimax	T_1	T_2
$M = 16$			
1	-77.26126766	0.05309448	0.41784180
2	-73.81026745	0.07175293	0.49369211
3	-73.02352142	0.07862549	0.51966134
4	-77.95156193	0.07042847	0.51158076
5	-105.23953247	0.04587402	0.46967784
$M = 32$			
1	-80.49464130	0.04725342	0.40357383
2	-73.92513466	0.07094727	0.49129255
3	-72.40863037	0.08012695	0.52153983
5	-70.95047379	0.08935547	0.54805908
7	-70.22383976	0.09403687	0.56031410
9	-69.94402790	0.09628906	0.56637987
11	-70.82423878	0.09323731	0.56226952
13	-104.85642624	0.04882812	0.48479068
$M = 64$			
1	-80.80974960	0.04658203	0.40168723
2	-75.11772251	0.06759644	0.48390015
3	-72.66662025	0.07886963	0.51850058
4	-71.85610867	0.08393555	0.53379876
5	-71.34401417	0.08721924	0.54311474
9	-70.32861614	0.09371948	0.56020256
13	-69.34809303	0.09761963	0.56903714
17	-68.06440258	0.10051880	0.57543691
21	-67.99149132	0.10289307	0.58007699
25	-69.32065105	0.10068359	0.57729656
29	-105.72862339	0.04923706	0.48767025
$M = 128$			
1	-80.89347839	0.04639893	0.40117195
2	-77.22580583	0.06295776	0.47399521
3	-73.43786240	0.07648926	0.51361278
4	-71.93675232	0.08345947	0.53266251
6	-71.10850430	0.08880615	0.54769675
9	-70.53600121	0.09255371	0.55752959
17	-69.95890045	0.09628906	0.56676912
25	-69.29977322	0.09834595	0.57137301
33	-68.75139713	0.10077515	0.57594641
41	-67.89687920	0.10183716	0.57863142
49	-66.76120186	0.10264282	0.58123560
57	-69.21525860	0.10157471	0.57946395
61	-104.57432938	0.04970703	0.48900685

Source: Rabiner et al. (1970); © 1970 IEEE; reprinted with permission.

1052 Appendix B Tables of Transition Coefficients for the Design of Linear-Phase FIR Filters

As an illustration, the filter design for which $M = 15$ and

$$H_r\left(\frac{2\pi k}{M}\right) = \begin{cases} 1, & k = 0, 1, 2, 3 \\ T_1, & k = 4 \\ 0, & k = 5, 6, 7 \end{cases}$$

corresponds to $\alpha = 0$, $\text{BW} = 4$, since $H_r(\omega_k) = 1$ at the four consecutive frequencies $\omega_k = 2\pi k/15$, $k = 0, 1, 2, 3$, and the transition coefficient is T_1 at the frequency $\omega_k = 8\pi/15$. The value given in Table B.1 for $M = 15$ and $\text{BW} = 4$ is $T_1 = 0.40405884$. The maximum sidelobe is at -41.9 dB, according to Table B.1.

References and Bibliography

- AKAIKE, H. 1969. "Power Spectrum Estimation Through Autoregression Model Fitting," *Ann. Inst. Stat. Math.*, Vol. 21, pp. 407–149.
- AKAIKE, H. 1974. "A New Look at the Statistical Model Identification," *IEEE Trans. Automatic Control*, Vol. AC-19, pp. 716–723, December.
- ANDERSEN, N. O. 1978. "Comments on the Performance of Maximum Entropy Algorithm," *Proc. IEEE*, Vol. 66, pp. 1581–1582, November.
- ANTONIOU, A. 1979. *Digital Filters: Analysis and Design*, McGraw-Hill, New York.
- AUER, E. 1987. "A Digital Filter Structure Free of Limit Cycles," *Proc. 1987 ICASSP*, pp. 21.11.1–21.11.4, Dallas, TX, April.
- AVENHAUS, E., and SCHUESSLER, H. W. 1970. "On the Approximation Problem in the Design of Digital Filters with Limited Wordlength," *Arch. Elek. Übertragung*, Vol. 24, pp. 571–572.
- BAGGEROER, A. B. 1976. "Confidence Intervals for Regression (MEM) Spectral Estimates," *IEEE Trans. Information Theory*, Vol. IT-22, pp. 534–545, September.
- BANDLER, J. W., and BARDAKJIAN, B. J. 1973. "Least p th Optimization of Recursive Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-21, pp. 460–470, October.
- BARNES, C. W., and FAM, A. T. 1977. "Minimum Norm Recursive Digital Filters That Are Free of Overflow Limit Cycles," *IEEE Trans. Circuits and Systems*, Vol. CAS-24, pp. 569–574, October.
- BARTLETT, M. S. 1948. "Smoothing Periodograms from Time Series with Continuous Spectra," *Nature* (London), Vol. 161, pp. 686–687, May.
- BARTLETT, M. S. 1961. *Stochastic Processes*, Cambridge University Press, Cambridge, UK.
- BECKMAN, F. S. 1960. "The Solution of Linear Equations by the conjugate Gradient Method" in *Mathematical Methods for Digital Computers*, A. Ralston and H.S. Wilf, eds., Wiley, New York.
- BERGLAND, G. D. 1969. "A Guided Tour of the Fast Fourier Transform," *IEEE Spectrum*, Vol. 6, pp. 41–52, July.
- BERK, K. N. 1974. "Consistent Autoregressive Spectral Estimates," *Ann. Stat.*, Vol. 2, pp. 489–502.
- BERNHARDT, P. A., ANTONIADIS, D. A., and DA ROSA, A. V. 1976. "Lunar Perturbations in Columnar Electron Content and Their Interpretation in Terms of Dynamo Electrostatic Fields," *J. Geophys. Res.*, Vol. 81, pp. 5957–5963, December.
- BERRYMAN, J. G. 1978. "Choice of Operator Length for Maximum Entropy Spectral Analysis," *Geophysics*, Vol. 43, pp. 1384–1391, December.
- BIERMAN, G. J. 1977. *Factorization Methods for Discrete Sequential Estimation*, Academic, New York.
- BLACKMAN, R. B., and TUKEY, J. W. 1958. *The Measurement of Power Spectra*, Dover, New York.
- BLAHUT, R. E. 1985. *Fast Algorithms for Digital Signal Processing*, Addison-Wesley, Reading, MA.
- BLUESTEIN, L. I. 1970. "A Linear Filtering Approach to the Computation of the Discrete Fourier Transform," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-18, pp. 451–455, December.
- BOLT, B. A. 1988. *Earthquakes*, W. H. Freeman and Co., New York.
- BOMAR, B. W. 1985. "New Second-Order State-Space Structures for Realizing Low Roundoff Noise Digital Filters," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-33, pp. 106–110, February.
- BRACEWELL, R. N. 1978. *The Fourier Transform and Its Applications*, 2d ed., McGraw-Hill, New York.

- BRIGHAM, E. O. 1988. *The Fast Fourier Transform and Its Applications*, Prentice Hall, Upper Saddle River, NJ.
- BRIGHAM, E. O., and MORROW, R. E. 1967. "The Fast Fourier Transform," *IEEE Spectrum*, Vol. 4, pp. 63–70, December.
- BRILLINGER, D. R. 1974. "Fourier Analysis of Stationary Processes," *Proc. IEEE*, Vol. 62, pp. 1628–1643, December.
- BROPHY, F., and SALAZAR, A. C. 1973. "Considerations of the Padé Approximant Technique in the Synthesis of Recursive Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-21, pp. 500–505, December.
- BROWN, J. L., JR., 1980. "First-Order Sampling of Bandpass Signals—A New Approach," *IEEE Trans. Information Theory*, Vol. IT-26, pp. 613–615, September.
- BROWN, R. C. 1983. *Introduction to Random Signal Analysis and Kalman Filtering*, Wiley, New York.
- BRUBAKER, T. A., and GOWDY, J. N. 1972. "Limit Cycles in Digital Filters," *IEEE Trans. Automatic Control*, Vol. AC-17, pp. 675–677, October.
- BRUZZONE, S. P., and KAVEH, M. 1980. "On Some Suboptimum ARMA Spectral Estimators," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 753–755, December.
- BURG, J. P. 1967. "Maximum Entropy Spectral Analysis," *Proc. 37th Meeting of the Society of Exploration Geophysicists*, Oklahoma City, OK, October. Reprinted in *Modern Spectrum Analysis*, D. G. Childers, ed., IEEE Press, New York.
- BURG, J. P. 1968. "A New Analysis Technique for Time Series Data," NATO Advanced Study Institute on Signal Processing with Emphasis on Underwater Acoustics, August 12–23. Reprinted in *Modern Spectrum Analysis*, D. G. Childers, ed., IEEE Press, New York.
- BURG, J. P. 1972. "The Relationship Between Maximum Entropy and Maximum Likelihood Spectra," *Geophysics*, Vol. 37, pp. 375–376, April.
- BURG, J. P. 1975. "Maximum Entropy Spectral Analysis," Ph.D. dissertation, Department of Geophysics, Stanford University, Stanford, CA, May.
- BURRUS, C. S., and PARKS, T. W. 1970. "Time-Domain Design of Recursive Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. 18, pp. 137–141, June.
- BURRUS, C. S. and PARKS, T. W. 1985. *DFT/FFT and Convolution Algorithms*, Wiley, New York.
- BUTTERWECK, H. J., VAN MEER, A. C. P., and VERKROOST, G. 1984. "New Second-Order Digital Filter Sections Without Limit Cycles," *IEEE Trans. Circuits and Systems*, Vol. CAS-31, pp. 141–146, February.
- CADZOW, J. A. 1979. "ARMA Spectral Estimation: An Efficient Closed-Form Procedure," *Proc. RADC Spectrum Estimation Workshop*, pp. 81–97, Rome, NY, October.
- CADZOW, J. A. 1981. "Autoregressive–Moving Average Spectral Estimation: A Model Equation Error Procedure," *IEEE Trans. Geoscience Remote Sensing*, Vol. GE-19, pp. 24–28, January.
- CADZOW, J. A. 1982. "Spectral Estimation: An Overdetermined Rational Model Equation Approach," *Proc. IEEE*, Vol. 70, pp. 907–938, September.
- CANDY, J. C. 1986. "Decimation for Sigma Delta Modulation," *IEEE Trans. Communications*, Vol. COM-34, pp. 72–76, January.
- CANDY, J. C., WOOLEY, B. A., and BENJAMIN, D. J. 1981. "A Voiceband Codec with Digital Filtering," *IEEE Trans. Communications*, Vol. COM-29, pp. 815–830, June.
- CAPON, J. 1969. "High-Resolution Frequency–Wavenumber Spectrum Analysis," *Proc. IEEE*, Vol. 57, pp. 1408–1418, August.
- CAPON, J. 1983. "Maximum-Likelihood Spectral Estimation," in *Nonlinear Methods of Spectral Analysis*, 2d ed., S. Haykin, ed., Springer-Verlag, New York.
- CAPON, J., and GOODMAN, N. R. 1971. "Probability Distribution for Estimators of the Frequency–Wavenumber Spectrum," *Proc. IEEE*, Vol. 58, pp. 1785–1786, October.
- CARAISCOS, C., and LIU, B. 1984. "A Roundoff Error Analysis of the LMS Adaptive Algorithm," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 34–41, January.
- CARAYANNIS, G., MANOLAKIS, D. G., and KALOUPTSIDIS, N. 1983. "A Fast Sequential Algorithm for Least-Squares Filtering and Prediction," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-31, pp. 1394–1402, December.

- CARAYANNIS, G., MANOLAKIS, D. G., and KALOUPTSIDIS, N. 1986. "A Unified View of Parametric Processing Algorithms for Prewindowed Signals," *Signal Processing*, Vol. 10, pp. 335–368, June.
- CARLSON, N. A., and CULMONE, A. F. 1979. "Efficient Algorithms for On-Board Array Processing," *Record 1979 International Conference on Communications*, pp. 58.1.1–58.1.5, Boston, June 10–14.
- CHAN, D. S. K., and RABINER, L. R. 1973a. "Theory of Roundoff Noise in Cascade Realizations of Finite Impulse Response Digital Filters," *Bell Syst. Tech. J.*, Vol. 52, pp. 329–345, March.
- CHAN, D. S. K., and RABINER, L. R. 1973b. "An Algorithm for Minimizing Roundoff Noise in Cascade Realizations of Finite Impulse Response Digital Filters," *Bell Syst. Tech. J.*, Vol. 52, pp. 347–385, March.
- CHAN, D. S. K., and RABINER, L. R. 1973c. "Analysis of Quantization Errors in the Direct Form for Finite Impulse Response Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-21, pp. 354–366, August.
- CHANG, T. 1981. "Suppression of Limit Cycles in Digital Filters Designed with One Magnitude-Truncation Quantizer," *IEEE Trans. Circuits and Systems*, Vol. CAS-28, pp. 107–111, February.
- CHEN, C. T. 1970. *Introduction to Linear System Theory*, Holt, Rinehart and Winston, New York.
- CHEN, W. Y., and STEGEN, G. R. 1974. "Experiments with Maximum Entropy Power Spectra of Sinusoids," *J. Geophys. Res.*, Vol. 79, pp. 3019–3022, July.
- CHILDERS, D. G., ed. 1978. *Modern Spectrum Analysis*, IEEE Press, New York.
- CHOW, J. C. 1972a. "On the Estimation of the Order of a Moving-Average Process," *IEEE Trans. Automatic Control*, Vol. AC-17, pp. 386–387, June.
- CHOW, J. C. 1972b. "On Estimating the Orders of an Autoregressive-Moving Average Process with Uncertain Observations," *IEEE Trans. Automatic Control*, Vol. AC-17, pp. 707–709, October.
- CHOW, Y., and CASSIGNOL, E. 1962. *Linear Signal Flow Graphs and Applications*, Wiley, New York.
- CHUI, C. K., and CHEN, G. 1987. *Kalman Filtering*, Springer-Verlag, New York.
- CIOFFI, J. M. 1987. "Limited Precision Effects in Adaptive Filtering," *IEEE Trans. Circuits and Systems*, Vol. CAS-34, pp. 821–833, July.
- CIOFFI, J. M., and KAILATH, T. 1984. "Fast Recursive-Least-Squares Transversal Filters for Adaptive Filtering," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 304–337, April.
- CIOFFI, J. M., and KAILATH, T. 1985. "Windowed Fast Transversal Filters Adaptive Algorithms with Normalization," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-33, pp. 607–625, June.
- CLAASEN, T. A. C. M., MECKLENBRAUKER, W. F. G., and PEEK, J. B. H. 1973. "Second-Order Digital Filter with Only One Magnitude-Truncation Quantizer and Having Practically No Limit Cycles," *Electron. Lett.*, Vol. 9, November.
- CLARKE, R. J. 1985. *Transform Coding of Images*. Academic Press, London, UK.
- COCHRAN, W. T., COOLEY, J. W., FAVIN, D. L., HELMS, H. D., KAENEL, R. A., LANG, W. W., MALING, G. C., NELSON, D. E., RADER, C. E., and WELCH, P. D. 1967. "What Is the Fast Fourier Transform," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-15, pp. 45–55, June.
- CONSTANTINIDES, A. G. 1967. "Frequency Transformations for Digital Filters," *Electron. Lett.*, Vol. 3, pp. 487–489, November.
- CONSTANTINIDES, A. G. 1968. "Frequency Transformations for Digital Filters," *Electron. Lett.*, Vol. 4, pp. 115–116, April.
- CONSTANTINIDES, A. G. 1970. "Spectral Transformations for Digital Filters," *Proc. IEEE*, Vol. 117, pp. 1585–1590, August.
- COOLEY, J. W., and TUKEY, J. W. 1965. "An Algorithm for the Machine Computation of Complex Fourier Series," *Math. Comp.*, Vol. 19, pp. 297–301, April.
- COOLEY, J. W., LEWIS, P., and WELCH, P. D. 1967. "Historical Notes on the Fast Fourier Transform," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-15, pp. 76–79, June.
- COOLEY, J. W., LEWIS, P., and WELCH, P. D. 1969. "The Fast Fourier Transform and Its Applications," *IEEE Trans. Education*, Vol. E-12, pp. 27–34, March.
- COULSON, A. 1995. "A Generalization of Nonuniform Bandpass Sampling," *IEEE Trans. on Signal Processing*, Vol. 43(3), pp. 694–704, March.
- COULSON, A., VAUGHAM, R., and POULETTI, M. 1994. "Frequency Shifting Using Bandpass Sampling," *IEEE Trans. on Signal Processing*, Vol 42(6), pp. 1556–1559, June.

- CROCHIERE, R. E. 1977. "On the Design of Sub-Band Coders for Low Bit Rate Speech Communication," *Bell Syst. Tech. J.*, Vol. 56, pp. 747–711, May–June.
- CROCHIERE, R. E. 1981. "Sub-Band Coding," *Bell Syst. Tech. J.*, Vol. 60, pp. 1633–1654, September.
- CROCHIERE, R. E., and RABINER, L. R. 1975. "Optimum FIR Digital Filter Implementations for Decimation, Interpolation, and Narrowband Filtering," *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. ASSP-23, pp. 444–456, October.
- CROCHIERE, R. E., and RABINER, L. R. 1976. "Further Considerations in the Design of Decimators and Interpolators," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-24, pp. 296–311, August.
- CROCHIERE, R. E., and RABINER, L. R. 1981. "Interpolation and Decimation of Digital Signals—A Tutorial Review," *Proc. IEEE*, Vol. 69, pp. 300–331, March.
- CROCHIERE, R. E., and RABINER, L. R. 1983. *Multirate Digital Signal Processing*, Prentice Hall, Upper Saddle River, NJ.
- DANIELS, R. W. 1974. *Approximation Methods for the Design of Passive, Active and Digital Filters*, McGraw-Hill, New York.
- DAVENPORT, W. B., JR. 1970. *Probability and Random Processes: An Introduction for Applied Scientists and Engineers*, McGraw-Hill, New York.
- DAVIS, H. F. 1963. *Fourier Series and Orthogonal Functions*, Allyn and Bacon, Boston.
- DECZKY, A. G. 1972. "Synthesis of Recursive Digital Filters Using the Minimum p -Error Criterion," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-20, pp. 257–263, October.
- DELLER, J. R. Jr., HANSEN, J. H. L., and PROAKIS, J. G. 2000. *Discrete-Time Processing of Speech Signals*, Wiley, New York.
- DEL SARTE, P., and GENIN, Y. 1986. "The Split Levinson Algorithm," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, pp. 470–478, June.
- DEL SARTE, P., GENIN, Y., and KAMP, Y. 1978. "Orthogonal Polynomial Matrices on the Unit Circle," *IEEE Trans. Circuits and Systems*, Vol. CAS-25, pp. 149–160, January.
- DERUSSO, P. M., ROY, R. J., and CLOSE, C. M. 1965. *State Variables for Engineers*, Wiley, New York.
- DUHAMEL, P. 1986. "Implementation of Split-Radix FFT Algorithms for Complex, Real, and Real-Symmetric Data," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, pp. 285–295, April.
- DUHAMEL, P., and HOLLMANN, H. 1984. "Split-Radix FFT Algorithm," *Electron. Lett.*, Vol. 20, pp. 14–16, January.
- DURBIN, J. 1959. "Efficient Estimation of Parameters in Moving-Average Models," *Biometrika*, Vol. 46, pp. 306–316.
- DWIGHT, H. B. 1957. *Tables of Integrals and Other Mathematical Data*, 3d ed., Macmillan, New York.
- DYM, H., and MCKEAN, H. P. 1972. *Fourier Series and Integrals*, Academic, New York.
- EBERT, P. M., MAZO, J. E., and TAYLOR, M. G. 1969. "Overflow Oscillations in Digital Filters," *Bell Syst. Tech. J.*, Vol. 48, pp. 2999–3020, November.
- ELEFTHERIOU, E., and FALCONER, D. D. 1987. "Adaptive Equalization Techniques for HF Channels," *IEEE J. Selected Areas in Communications*, Vol. SAC-5, pp. 238–247, February.
- ELUP, L., GARDNER, F. M., and HARRIS F. A., 1993 "Interpolation in digital Modems, Part II: Fundamentals and performance." *IEEE trans. on Communications*, Vol. 41(6), pp. 998–1008, June.
- FALCONER, D. D., and LJUNG, L. 1978. "Application of Fast Kalman Estimation to Adaptive Equalization," *IEEE Trans. Communications*, Vol. COM-26, pp. 1439–1446, October.
- FAM, A. T., and BARNES, C. W. 1979. "Non-minimal Realizations of Fixed-Point Digital Filters That Are Free of All Finite Wordlength Limit Cycles," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-27, pp. 149–153, April.
- FARROW, C. W. 1998. "A Continuously Variable Digital Delay Element." *Proc. IEEE Intern. Symposium on Circuits and Systems*, pp. 2641–2645.
- FETTWEIS, A. 1971. "Some Principles of Designing Digital Filters Imitating Classical Filter Structures," *IEEE Trans. Circuit Theory*, Vol. CT-18, pp. 314–316, March.
- FLETCHER, R., and POWELL, M. J. D. 1963. "A Rapidly Convergent Descent Method for Minimization," *Comput. J.*, Vol. 6, pp. 163–168.

- FOUGERE, P. F., ZAWALICK, E. J., and RADOSKI, H. R. 1976. "Spontaneous Line Splitting in Maximum Entropy Power Spectrum Analysis," *Phys. Earth Planet. Inter.*, Vol. 12, 201–207, August.
- FRERKING, M. E. 1994. *Digital Signal Processing in Communication Systems*, Kluwer Academic Publishers, Boston.
- FRIEDLANDER, B. 1982a. "Lattice Filters for Adaptive Processing," *Proc. IEEE*, Vol. 70, pp. 829–867, August.
- FRIEDLANDER, B. 1982b. "Lattice Methods for Spectral Estimation," *Proc. IEEE*, Vol. 70, pp. 990–1017, September.
- FUCHS, J. J. 1988. "Estimating the Number of Sinusoids in Additive White Noise," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-36, pp. 1846–1853, December.
- GANTMACHER, F. R. 1960. *The Theory of Matrices*, Vol. I., Chelsea, New York.
- GARDNER, F. M. 1993. "Interpolation in Digital Modems, Part I: Fundamentals." *IEEE Trans. on Communications*, Vol. 41(3), pp. 502–508, March.
- GARDNER, W. A. 1984. "Learning Characteristics of Stochastic-Gradient-Descent Algorithms: A General Study, Analysis and Critique," *Signal Processing*, Vol. 6, pp. 113–133, April.
- GARDNER, W. A. 1987. *Statistical Spectral Analysis: A Nonprobabilistic Theory*, Prentice Hall, Upper Saddle River, NJ.
- GARLAN, C., and ESTEBAN, D. 1980. "16 Kbps Real-Time QMF Sub-Band Coding Implementation," *Proc. 1980 International Conference on Acoustics, Speech, and Signal Processing*, pp. 332–335, April.
- GEORGE, D. A., BOWEN, R. R., and STOREY, J. R. 1971. "An Adaptive Decision-Feedback Equalizer," *IEEE Trans. Communication Technology*, Vol. COM-19, pp. 281–293, June.
- GERONIMUS, L. Y. 1958. *Orthogonal Polynomials* (in Russian) (English translation by Consultants Bureau, New York, 1961).
- GERSCHE, W., and SHARPE, D. R. 1973. "Estimation of Power Spectra with Finite-Order Autoregressive Models," *IEEE Trans. Automatic Control*, Vol. AC-18, pp. 367–369, August.
- GERSHO, A. 1969. "Adaptive Equalization of Highly Dispersive Channels for Data Transmission," *Bell Syst. Tech. J.*, Vol. 48, pp. 55–70, January.
- GIBBS, A. J. 1969. "An Introduction to Digital Filters," *Aust. Telecommun. Res.*, Vol. 3, pp. 3–14, November.
- GIBBS, A. J. 1970. "The Design of Digital Filters," *Aust. Telecommun. Res.*, Vol. 4, pp. 29–34, May.
- GITLIN, R. D., and WEINSTEIN, S. B. 1979. "On the Required Tap-Weight Precision for Digitally Implemented Mean-Squared Equalizers," *Bell Syst. Tech. J.*, Vol. 58, pp. 301–321, February.
- GITLIN, R. D., MEADORS, H. C., and WEINSTEIN, S. B. 1982. "The Tap-Leakage Algorithm: An Algorithm for the Stable Operation of a Digitally Implemented Fractionally Spaced, Adaptive Equalizer," *Bell Syst. Tech. J.*, Vol. 61, pp. 1817–1839, October.
- GOERTZEL, G. 1968. "An Algorithm for the Evaluation of Finite Trigonometric Series," *Am. Math. Monthly*, Vol. 65, pp. 34–35, January.
- GOHBERG, I., ed. 1986. *I. Schur Methods in Operator Theory and Signal Processing*, Birkhauser Verlag, Stuttgart, Germany.
- GOLD, B., and JORDAN, K. L., JR. 1986. "A Note on Digital Filter Synthesis," *Proc. IEEE*, Vol. 56, pp. 1717–1718, October.
- GOLD, B., and JORDAN, K. L., JR. 1969. "A Direct Search Procedure for Designing Finite Duration Impulse Response Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-17, pp. 33–36, March.
- GOLD, B., and RADER, C. M. 1966. "Effects of Quantization Noise in Digital Filters." *Proc. AFIPS 1966 Spring Joint Computer Conference*, Vol. 28, pp. 213–219.
- GOLD, B., and RADER, C. M. 1969. *Digital Processing of Signals*, McGraw-Hill, New York.
- GOLDEN, R. M., and KAISER, J. F. 1964. "Design of Wideband Sampled Data Filters," *Bell Syst. Tech. J.*, Vol. 43, pp. 1533–1546, July.
- GOOD, I. J. 1971. "The Relationship Between Two Fast Fourier Transforms," *IEEE Trans. Computers*, Vol. C-20, pp. 310–317.
- GORSKI-POPIEL, J., ed. 1975. *Frequency Synthesis: Techniques and Applications*, IEEE Press, New York.

- GOYAL, V. 2001. "Theoretical Foundations of Transform Coding." *IEEE Signal Processing Magazine*, pp 9-21, September.
- GRACE, O. D., and PITT, S. P. 1969. "Sampling and Interpolation of Bandlimited Signals by Quadrature Methods." *J. Acoust. Soc. Amer.*, Vol. 48(6), pp. 1311-1318, November.
- GRAUPE, D., KRAUSE, D. J., and MOORE, J. B. 1975. "Identification of Autoregressive-Moving Average Parameters of Time Series," *IEEE Trans. Automatic Control*, Vol. AC-20, pp. 104-107, February.
- GRAY, A. H., and MARKEI, J. D. 1973. "Digital Lattice and Ladder Filter Synthesis," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-21, pp. 491-500, December.
- GRAY, A. H., and MARKEI, J. D. 1976. "A Computer Program for Designing Digital Elliptic Filters," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-24, pp. 529-538, December.
- GRAY, R. M. 1990. *Source Coding Theory*, Kluwer, Boston, MA.
- GRENANDER, O., and SZEGO, G. 1958. *Toeplitz Forms and Their Applications*, University of California Press, Berkeley, CA.
- GRIFFITHS, L. J. 1975. "Rapid Measurements of Digital Instantaneous Frequency," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-23, pp. 207-222, April.
- GRIFFITHS, L. J. 1978. "An Adaptive Lattice Structure for Noise Cancelling Applications," Proc. ICASSP-78, pp. 87-90. Tulsa, OK, April.
- GUILLEMIN, E. A. 1957. *Synthesis of Passive Networks*, Wiley, New York.
- GUPTA, S. C. 1966. *Transform and State Variable Methods in Linear Systems*, Wiley, New York.
- HAMMING, R. W. 1962. *Numerical Methods for Scientists and Engineers*, McGraw-Hill, New York.
- HARRIS, F. 1997. *Performance and Design of Farrow Filter Used for Arbitrary Resampling*. 31st Conference on Signals, Systems, and Computers, Pacific Grove, CA, pp. 595-599.
- HAYKIN, S. 1991. *Adaptive Filter Theory*, 2d ed., Prentice Hall, Upper Saddle River, NJ.
- HELME, B., and NIKIAS, C. S. 1985. "Improved Spectrum Performance via a Data-Adaptive Weighted Burg Technique," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-33, pp. 903-910, August.
- HELM, H. D. 1967. "Fast Fourier Transforms Method of Computing Difference Equations and Simulating Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-15, pp. 85-90, June.
- HELM, H. D. 1968. "Nonrecursive Digital Filters: Design Methods for Achieving Specifications on Frequency Response," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-16, pp. 336-342, September.
- HELSTROM, C. W. 1990. *Probability and Stochastic Processes for Engineers*, 2d ed., Macmillan, New York.
- HERRING, R. W. 1980. "The Cause of Line Splitting in Burg Maximum-Entropy Spectral Analysis," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 692-701, December.
- HERMANN, O. 1970. "Design of Nonrecursive Digital Filters with Linear Phase," *Electron. Lett.*, Vol. 6, pp. 328-329, November.
- HERMANN, O., and SCHÜESSLER, H. W. 1970a. "Design of Nonrecursive Digital Filters with Minimum Phase," *Electron. Lett.*, Vol. 6, pp. 329-330, November.
- HERMANN, O., and SCHÜESSLER, H. W. 1970b. "On the Accuracy Problem in the Design of Nonrecursive Digital Filters," *Arch. Elek. Übertragung*, Vol. 24, pp. 525-526.
- HERMANN, O., RABINER, L. R., and CHAN, D. S. K. 1973. "Practical Design Rules for Optimum Finite Impulse Response Lowpass Digital Filters," *Bell Syst. Tech. J.*, Vol. 52, pp. 769-799, July-August.
- HILDEBRAND, F. B. 1952. *Methods of Applied Mathematics*, Prentice Hall, Upper Saddle River, NJ.
- HOFSTETTER, E., OPPENHEIM, A. V., and SIEGEL, J. 1971. "A New Technique for the Design of Nonrecursive Digital Filters," *Proc. 5th Annual Princeton Conference on Information Sciences and Systems*, pp. 64-72.
- HOGENAUER, E.B. 1981. "An Economical Class of Digital Filters for Decimation and Interpolation" *IEEE Trans. on ASSP*, Vol. 29(2), pp. 155-162, April.
- HOUSEHOLDER, A. S. 1964. *The Theory of Matrices in Numerical Analysis*, Blaisdell, Waltham, MA.
- HSU, F. M. 1982. "Square-Root Kalman Filtering for High-Speed Data Received Over Fading Dispersive HF Channels," *IEEE Trans. Information Theory*, Vol. IT-28, pp. 753-763, September.
- HSU, F. M., and GIORDANO, A. A. 1978. "Digital Whitening Techniques for Improving Spread Spectrum Communications Performance in the Presence of Narrowband Jamming and Interference," *IEEE Trans. Communications*, Vol. COM-26, pp. 209-216, February.

- HWANG, S. Y. 1977. "Minimum Uncorrelated Unit Noise in State Space Digital Filtering," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-25, pp. 273–281, August.
- INCINBONO, B. 1978. "adaptive Signal Processing for Detection and Communication," in *communication Systems and Random Process Theory*, J.K. Skwirzynski, ed., Sijthoff en Noordhoff, Alphen aan den Rijn, The Netherlands.
- JAYANT, N. S., and NOLL, P. 1984. *Digital Coding of waveforms*, Prentice Hall, Upper Saddle River, NJ.
- JACKSON, L. B. 1969. "An Analysis of Limit Cycles Due to Multiplication Rounding in Recursive Digital (Sub) Filters," *Proc. 7th Annual Allerton Conference on Circuit and System Theory*, pp. 69–78.
- JACKSON, L. B. 1970a. "On the Interaction of Roundoff Noise and Dynamic Range in Digital Filters," *Bell Syst. Tech. J.*, Vol. 49, pp. 159–184, February.
- JACKSON, L. B. 1970b. "Roundoff Noise Analysis for Fixed-Point Digital Filters Realized in Cascade or Parallel Form," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-18, pp. 107–122, June.
- JACKSON, L. B. 1976. "Roundoff Noise Bounds Derived from Coefficients Sensitivities in Digital Filters," *IEEE Trans. Circuits and Systems*, Vol. CAS-23, pp. 481–485, August.
- JACKSON, L. B. 1979. "Limit Cycles on State-Space Structures for Digital Filters," *IEEE Trans. Circuits and Systems*, Vol. CAS-26, pp. 67–68, January.
- JACKSON, L. B., LINDGREN, A. G., and KIM, Y. 1979. "Optimal Synthesis of Second-Order State-Space Structures for Digital Filters," *IEEE Trans. Circuits and Systems*, Vol. CAS-26, pp. 149–153, March.
- JACKSON, M., and MATTHEWSON, P. 1986. "Digital Processing of Bandpass Signals." *GEC Journal of Research*, Vol. 4(1), pp. 32–41.
- JAHNKE, E., and EMDE, F. 1945. *Tables of Functions*, 4th ed., Dover, New York.
- JAIN, V. K., and CROCHIERE, R. E. 1984. "Quadrature Mirror Filter Design in the Time Domain," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 353–361, April.
- JANG, Y., and YANG, S. 2001. *Recursive Cascaded Integrator-Comb Decimation Filters with Integer Multiple factors*, 44th IEEE Midwest Symposium on circuits and Systems, Daytona, OH, August.
- JENKINS, G. M., and WATTS, D. G. 1968. *Spectral Analysis and Its Applications*, Holden-Day, San Francisco.
- JOHNSON, D. H. 1982. "The Application of Spectral Estimation Methods to Bearing Estimation Problems," *Proc. IEEE*, Vol. 70, pp. 1018–1028, September.
- JOHNSTON, J. D. 1980. "A Filter Family Designed for Use in Quadrature Mirror Filter Banks," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 291–294, April.
- JONES, R. H. 1976. "Autoregression Order Selection," *Geophysics*, Vol. 41, pp. 771–773, August.
- JONES, S. K., CAVIN, R. K., and REED, W. M. 1982. "Analysis of Error-Gradient Adaptive Linear Equalizers for a Class of Stationary-Dependent Processes," *IEEE Trans. Information Theory*, Vol. IT-28, pp. 318–329, March.
- JURY, E. I. 1964. *Theory and Applications of the z-Transform Method*, Wiley, New York.
- KAILATH, T. 1974. "A View of Three Decades of Linear Filter Theory," *IEEE Trans. Information Theory*, Vol. IT-20, pp. 146–181, March.
- KAILATH, T. 1981. *Lectures on Wiener and Kalman Filtering*, 2d printing, Springer-Verlag, New York.
- KAILATH, T. 1985. "Linear Estimation for Stationary and Near-Stationary Processes," in *Modern Signal Processing*, T. Kailath, ed., Hemisphere Publishing Corp., Washington, DC.
- KAILATH, T. 1986. "A Theorem of I. Schür and Its Impact on Modern Signal Processing," in Gohberg 1986.
- KAILATH, T., VIEIRA, A. C. G., and MORF, M. 1978. "Inverses of Toeplitz Operators, Innovations, and Orthogonal Polynomials," *SIAM Rev.*, Vol. 20, pp. 1006–1019.
- KAISER, J. F. 1963. "Design Methods for Sampled Data Filters," *Proc. First Allerton Conference on Circuit System Theory*, pp. 221–236, November.
- KAISER, J. F. 1966. "Digital Filters," in *System Analysis by Digital Computer*, F. F. Kuo and J. F. Kaiser, eds., Wiley, New York.
- KALOUPTSIDIS, N., and THEODORIDIS, S. 1987. "Fast Adaptive Least-Squares Algorithms for Power Spectral Estimation," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-35, pp. 661–670, May.
- KALMAN, R. E. 1960. "A New Approach to Linear Filtering and Prediction Problems," *Trans. ASME, J. Basic Eng.*, Vol. 82D, pp. 35–45, March.

- KALMAN, R. E., and BUCY, R. S. 1961. "New Results in Linear Filtering Theory," *Trans. ASME, J. Basic Eng.*, Vol. 83, pp. 95–108.
- KASHYAP, R. L. 1980. "Inconsistency of the AIC Rule for Estimating the Order of Autoregressive Models," *IEEE Trans. Automatic Control*, Vol. AC-25, pp. 996–998, October.
- KAVEH, J., and BRUZZONE, S. P. 1979. "Order Determination for Autoregressive Spectral Estimation," *Record of the 1979 RADC Spectral Estimation Workshop*, pp. 139–145, Griffon Air Force Base, Rome, NY.
- KAVEH, M., and LIPPERT, G. A. 1983. "An Optimum Tapered Burg Algorithm for Linear Prediction and Spectral Analysis," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-31, pp. 438–444, April.
- KAY, S. M. 1980. "A New ARMA Spectral Estimator," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 585–588, October.
- KAY, S. M. 1988. *Modern Spectral Estimation*, Prentice Hall, Upper Saddle River, NJ.
- KAY, S. M., and MARPLE, S. L., JR. 1979. "Sources of and Remedies for Spectral Line Splitting in Autoregressive Spectrum Analysis," *Proc. 1979 ICASSP*, pp. 151–154.
- KAY, S. M., and MARPLE, S. L., JR. 1981. "Spectrum Analysis: A Modern Perspective," *Proc. IEEE*, Vol. 69, pp. 1380–1419, November.
- KESLER, S. B., ed. 1986. *Modern Spectrum Analysis II*, IEEE Press, New York.
- KETCHUM, J. W., and PROAKIS, J. G. 1982. "Adaptive Algorithms for Estimating and Suppressing Narrow-Band Interference in PN Spread-Spectrum Systems," *IEEE Trans. Communications*, Vol. COM-30, pp. 913–923, May.
- KNOWLES, J. B., and OLCAYTO, E. M. 1968. "Coefficient Accuracy and Digital Filter Response," *IEEE Trans. Circuit Theory*, Vol. CT-15, pp. 31–41, March.
- KOHLENBURG, A. 1953. "Exact Interpolation of Bandlimited Functions." *Journal of Applied Physics*, Vol. 24(12), pp. 1432–1436, May.
- KRISHNA, H. 1988. "New Split Levinson, Schür, and Lattice Algorithms for Digital Signal Processing," *Proc. 1988 International Conference on Acoustics, Speech, and Signal Processing*, pp. 1640–1642, New York, April.
- KROMER, R. E. 1969. "Asymptotic Properties of the Autoregressive Spectral Estimator," Ph.D. dissertation, Department of Statistics, Stanford University, Stanford, CA.
- KUNG, H. T. 1982. "Why Systolic Architectures?" *IEEE Computer*, Vol. 15, pp. 37–46.
- KUNG, S. Y., and HU, Y. H. 1983. "A Highly Concurrent Algorithm and Pipelined Architecture for Solving Toeplitz Systems," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-31, pp. 66–76, January.
- KUNG, S. Y., WHITEHOUSE, H. J., and KAILATH, T., eds. 1985. *VLSI and Modern Signal Processing*, Prentice Hall, Upper Saddle River, NJ.
- LAAKSO, T., VALIMAKI, V., KARJALAINEN, M., and LAINE, U. 1996. "Splitting the Unit Delay." *IEEE Signal Processing Magazine*, No. 1, pp. 30–54, January.
- LACOSS, R. T. 1971. "Data Adaptive Spectral Analysis Methods," *Geophysics*, Vol. 36, pp. 661–675, August.
- LANG, S. W., and MCCLELLAN, J. H. 1980. "Frequency Estimation with Maximum Entropy Spectral Estimators," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 716–724, December.
- LEVINSON, N. 1947. "The Wiener RMS Error Criterion in Filter Design and Prediction," *J. Math. Phys.*, Vol. 25, pp. 261–278.
- LEVY, H., and LESSMAN, F. 1961. *Finite Difference Equations*, Macmillan, New York.
- LIN, D. W. 1984. "On Digital Implementation of the Fast Kalman Algorithm," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 998–1005, October.
- LINDEN, D. A. 1959. "A Discussion of Sampling Theorems." *Proc. of the IRE*, Vol. 47(11), pp. 1219–1226, November.
- LING, F., and PROAKIS, J. G. (1984a). "Numerical Accuracy and Stability: Two Problems of Adaptive Estimation Algorithms Caused by Round-Off Error," *Proc. ICASSP-84*, pp. 30.3.1–30.3.4, San Diego, CA, March.

- LING, F., and PROAKIS, J. G. (1984b). "Nonstationary Learning Characteristics of Least-Squares Adaptive Estimation Algorithms," *Proc. ICASSP-84*, pp. 3.7.1–3.7.4, San Diego, CA, March.
- LING, F., MANOLAKIS, D., and PROAKIS, J. G. 1985, "New Forms of LS Lattice Algorithms and Analysis of Their Round-Off Error Characteristics," *Proc. ICASSP-85*, pp. 1739–1742, Tampa, FL, April.
- LING, F., MANOLAKIS, D., and PROAKIS, J. G. 1986. "Numerically Robust Least-Squares Lattice-Ladder Algorithms with Direct Updating of the Reflection Coefficients," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, pp. 837–845, August.
- LIU, B. 1971. "Effect of Finite Word Length on the Accuracy of Digital Filters—A Review," *IEEE Trans. Circuit Theory*, Vol. CT-18, pp. 670–677, November.
- LJUNG, S., and LJUNG, L. 1985. "Error Propagation Properties of Recursive Least-Squares Adaptation Algorithms," *Automatica*, Vol. 21, pp. 157–167.
- LUCKY, R. W. 1965. "Automatic Equalization for Digital Communications," *Bell Syst. Tech. J.*, Vol. 44, pp. 547–588, April.
- MAGEE, F. R. and PROAKIS, J. G. 1973. "Adaptive Maximum-Likelihood Sequence Estimation for Digital Signaling in the Presence of Intersymbol Interference," *IEEE Trans. Information Theory*, Vol. IT-19, pp. 120–124, January.
- MAKHOUL, J. 1975. "Linear Prediction: A Tutorial Review," *Proc. IEEE*, Vol. 63, pp. 561–580, April.
- MAKHOUL, J. 1978. "A Class of All-Zero Lattice Digital Filters: Properties and Applications," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-26, pp. 304–314, August.
- MAKHOUL, J. 1980. "A Fast Cosine Transform In One and Two Dimensions." *IEEE Trans. on ASSP*, Vol. 28(1), pp. 27–34, February.
- MARKEL, J. D., and GRAY, A. H., JR. 1976. *Linear Prediction of Speech*, Springer-Verlag, New York.
- MANOLAKIS, D., LING, F., and PROAKIS, J. G. 1987. "Efficient Time-Recursive Least-Squares Algorithms for Finite-Memory Adaptive Filtering," *IEEE Trans. Circuits and Systems*, Vol. CAS-34, pp. 400–408, April.
- MARPLE, S. L., JR. 1980. "A New Autoregressive Spectrum Analysis Algorithm," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 441–454, August.
- MARPLE, S. L., JR. 1987. *Digital Spectral Analysis with Applications*, Prentice Hall, Upper Saddle River, NJ.
- MARTUCCI, S. A. 1994. "Symmetric Convolution and the Discrete Sine and Cosine Transforms." *IEEE Trans. on Signal Processing*, Vol. 42(5), pp. 1038–1051, May.
- MARZETTA, T. L. 1983. "A New Interpretation for Capon's Maximum Likelihood Method of Frequency-Wavenumber Spectral Estimation," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-31, pp. 445–449, April.
- MARZETTA, T. L., and LANG, S. W. 1983. "New Interpretations for the MLM and DASE Spectral Estimators," *Proc. 1983 ICASSP*, pp. 844–846, Boston, April.
- MARZETTA, T. L., and LANG, S. W. 1984. "Power Spectral Density Bounds," *IEEE Trans. Information Theory*, Vol. IT-30, pp. 117–122, January.
- MASON, S. J., and ZIMMERMAN, H. J. 1960. *Electronic Circuits, Signals and Systems*, Wiley, New York.
- MAZO, J. E. 1979. "On the Independence Theory of Equalizer Convergence," *Bell Syst. Tech. J.*, Vol. 58, pp. 963–993, May.
- MCCLELLAN, J. H. 1982. "Multidimensional Spectral Estimation," *Proc. IEEE*, Vol. 70, pp. 1029–1039, September.
- MCDONOUGH, R. N. 1983. "Application of the Maximum-Likelihood Method and the Maximum Entropy Method to Array Processing," in *Nonlinear Methods of Spectral Analysis*, 2d ed., S. Haykin, ed., Springer-Verlag, New York.
- MCGILLEM, C. D., and COOPER, G. R. 1984. *Continuous and Discrete Signal and System Analysis*, 2d ed., Holt Rinehart and Winston, New York.
- MEDITCH, J. E. 1969. *Stochastic Optimal Linear Estimation and Control*, McGraw-Hill, New York.
- MEYER, R., and BURRUS, S. 1975. "A Unified Analysis of Multirate and Periodically Time-Varying Digital Filters." *IEEE Trans. on Circuits and Systems*, Vol. 22(3), pp. 162–168, March.
- MILLS, W. L., MULLIS, C. T., and ROBERTS, R. A. 1981. "Low Roundoff Noise and Normal Realizations of Fixed-Point IIR Digital Filters," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-29, pp. 893–903, August.

- MOORER, J. A. 1977. "Signal Aspects of Computer Music; A Survey," *Proc. IEEE*, Vol. 65, pp. 1108–1137, August.
- MORF, M., VIEIRA, A., and LEE, D. T. 1977. "Ladder Forms for Identification and Speech Processing," *Proc. 1977 IEEE Conference Decision and Control*, pp. 1074–1078, New Orleans, LA, December.
- MULLIS, C. T., and ROBERTS, R. A. 1976a. "Synthesis of Minimum Roundoff Noise Fixed-Point Digital Filters," *IEEE Trans. Circuits and Systems*, Vol. CAS-23, pp. 551–561, September.
- MULLIS, C. T., and ROBERTS, R. A. 1976b. "Roundoff Noise in Digital Filters: Frequency Transformations and Invariants," *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. ASSP-24, pp. 538–549, December.
- MURRAY, W., ed. 1972 *Numerical Methods for Unconstrained Minimization*, Academic, New York.
- MUSICUS, B. 1985. "Fast MLM Power Spectrum Estimation from Uniformly Spaced Correlations," *IEEE Trans. Acoustics, Speech, and Signal Proc.*, Vol. ASSP-33, pp. 1333–1335, October.
- NEWMAN, W. I. 1981. "Extension to the Maximum Entropy Method III," *Proc. 1st ASSP Workshop on Spectral Estimation*, pp. 1.7.1–1.7.6, Hamilton, ON, August.
- NICHOLS, H. E., GIORDANO, A. A., and PROAKIS, J. G. 1977. "MLD and MSE Algorithms for Adaptive Detection of Digital Signals in the Presence of Interchannel Interference," *IEEE Trans. Information Theory*, Vol. IT-23, pp. 563–575, September.
- NIKIAS, C. L., and RAGHUVeer, M. R. 1987. "Bispectrum Estimation: A Digital Signal Processing Framework," *Proc. IEEE*, Vol. 75, pp. 869–891, July.
- NIKIAS, C. L., and SCOTT, P. D. 1982. "Energy-Weighted Linear Predictive Spectral Estimation: A New Method Combining Robustness and High Resolution," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-30, pp. 287–292, April.
- NUTTALL, A. H. 1976. "Spectral Analysis of a Univariate Process with Bad Data Points, via Maximum Entropy and Linear Predictive Techniques," *NUSC Technical Report TR-5303*, New London, CT, March.
- NYQUIST, H. 1928. "Certain Topics in Telegraph Transmission Theory," *Trans. AIEE*, Vol. 47, pp. 617–644, April.
- OPPENHEIM, A. V. 1978. *Applications of Digital Signal Processing*, Prentice Hall, Upper Saddle River, NJ.
- OPPENHEIM, A. V., and SCHAFER, R. W. 1989. *Discrete-Time Signal Processing*, Prentice Hall, Upper Saddle River, NJ.
- OPPENHEIM, A. V., and WEINSTEIN, C. W. 1972. "Effects of Finite Register Length in Digital Filters and the Fast Fourier Transform," *Proc. IEEE*, Vol. 60, pp. 957–976, August.
- OPPENHEIM, A. V., and WILLSKY, A. S. 1983. *Signals and Systems*, Prentice Hall, Upper Saddle River, NJ.
- PAPOULIS, A. 1962 *The Fourier Integral and Its Applications*, McGraw-Hall, New York.
- PAPOULIS, A. 1984. *Probability, Random Variables, and Stochastic Processes*, 2d ed., McGraw-Hill, New York.
- PARKER, S. R., and HESS, S. F. 1971. "Limit-Cycle Oscillations in Digital Filters," *IEEE Trans. Circuit Theory*, Vol. CT-18, pp. 687–696, November.
- PARKS, T. W., and MCCLELLAN, J. H. 1972a. "Chebyshev-Approximation for Nonrecursive Digital Filters with Linear Phase," *IEEE Trans. Circuit Theory*, Vol. CT-19, pp. 189–194, March.
- PARKS, T. W., and MCCLELLAN, J. H. 1972b. "A Program for the Design of Linear Phase Finite Impulse Response Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-20, pp. 195–199, August.
- PARZEN, E. 1957. "On Consistent Estimates of the Spectrum of a Stationary Time Series," *Am. Math. Stat.*, Vol. 28, pp. 329–348.
- PARZEN, E. 1974. "Some Recent Advances in Time Series Modeling," *IEEE Trans. Automatic Control*, Vol. AC-19, pp. 723–730, December.
- PEACOCK, K. L., and TREITEL, S. 1969. "Predictive Deconvolution—Theory and Practice," *Geophysics*, Vol. 34, pp. 155–169.
- PEEBLES, P. Z., JR. 1987. *Probability, Random Variables, and Random Signal Principles*, 2d ed., McGraw-Hill, New York.

- PICINBONO, B. 1978. "Adaptive Signal Processing for Detection and Communication," in *Communication Systems and Random Process Theory*, J. K. Skwirzynski, ed., Sijthoff en Noordhoff, Alphen aan den Rijn, The Netherlands.
- PISARENKO, V. F. 1973. "The Retrieval of Harmonics from a Covariance Function," *Geophys. J. R. Astron. Soc.*, Vol. 33, pp. 347-366.
- POOLE, M. A. 1981. *Autoregressive Methods of Spectral Analysis*, E.E. degree thesis, Department of Electrical and Computer Engineering, Northeastern University, Boston, May.
- PRICE, R. 1990. "Mirror FFT and Phase FFT Algorithms," unpublished work, Raytheon Research Division, May.
- PROAKIS, J. G. 1970. "Adaptive Digital Filters for Equalization of Telephone Channels," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-18, pp. 195-200, June.
- PROAKIS, J. G. 1974. "Channel Identification for High Speed Digital Communications," *IEEE Trans. Automatic Control*, Vol. AC-19, pp. 916-922, December.
- PROAKIS, J. G. 1975. "Advances in Equalization for Intersymbol Interference," in *Advances in Communication Systems*, Vol. 4, A. J. Viterbi, ed., Academic, New York.
- PROAKIS, J. G. 1989. *Digital Communications*, 2nd ed., McGraw-Hill, New York.
- PROAKIS, J. G., and MILLER, J. H. 1969. "Adaptive Receiver for Digital Signaling through Channels with Intersymbol Interference," *IEEE Trans. Information Theory*, Vol. IT-15, pp. 484-497, July.
- PROAKIS, J. G., RADER, C. M., LING, F., NIKIAS, C. L., MOONEN, M. and PROUDLER, I. K. 2002. *Algorithms for Statistical Signal Processing*, Prentice-Hall, Upper Saddle River, NJ.
- PROAKIS, J. G. 2001. *Digital Communications*, 4th ed., McGraw-Hill, New York.
- QI, R., COAKLEY, F., and EVANS, B. 1996. "Practical Consideration for Bandpass Sampling," *Electronics Letters*, Vol. 32(20), pp. 1861-1862, September.
- RABINER, L. R., and SCHAEFER, R. W. 1974a. "On the Behavior of Minimax Relative Error FIR Digital Differentiators," *Bell Syst. Tech. J.*, Vol. 53, pp. 333-362, February.
- RABINER, L. R., and SCHAEFER, R. W. 1974b. "On the Behavior of Minimax FIR Digital Hilbert Transformers," *Bell Syst. Tech. J.*, Vol. 53, pp. 363-394, February.
- RABINER, L. R., and SCHAEFER, R. W. 1978. *Digital Processing of Speech Signals*, Prentice Hall, Upper Saddle River, NJ.
- RABINER, L. R., SCHAFER, R. W., and RADER, C. M. 1969. "The Chirp z-Transform Algorithm and Its Applications," *Bell Syst. Tech. J.*, Vol. 48, pp. 1249-1292, May-June.
- RABINER, L. R., GOLD, B., and MCGONEGAL, C. A. 1970. "An Approach to the Approximation Problem for Nonrecursive Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-18, pp. 83-106, June.
- RABINER, L. R., MCCLELLAN, J. H., and PARKS, T. W. 1975. "FIR Digital Filter Design Techniques Using Weighted Chebyshev Approximation," *Proc. IEEE*, Vol. 63, pp. 595-610, April.
- RADER, C. M. 1970. "An Improved Algorithm for High-Speed Auto-correlation with Applications to Spectral Estimation," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-18, pp. 439-441, December.
- RADER, C. M., and BRENNER, N. M. 1976. "A New Principle for Fast Fourier Transformation," *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. ASSP-24, pp. 264-266, June.
- RADER, C. M., and GOLD, B. 1967a. "Digital Filter Design Techniques in the Frequency Domain," *Proc. IEEE*, Vol. 55, pp. 149-171, February.
- RADER, C. M., and GOLD, B. 1967b. "Effects of Parameter Quantization on the Poles of a Digital Filter," *Proc. IEEE*, Vol. 55, pp. 688-689, May.
- RAMSTAD, T. A. 1984. "Digital Methods for Conversion Between Arbitrary Sampling Frequencies," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 577-591, June.
- RAO, K., and HUANG, J. 1996. *Techniques and Standards for Image, Video, and Audio Coding*, Prentice Hall, Upper Saddle River, NJ.
- RAO, K. R., and YIP, P. 1990. *Discrete Cosine Transform*, Academic Press, Boston.
- REMEZ, E. YA. 1957. *General Computational Methods of Chebyshev Approximation*, Atomic Energy Translation 4491, Kiev, USSR.
- RICE, D., and WU, K. 1982. "Quatature Sampling with High Dynamic Range." *IEEE Trans. Aerospace and Electronic Systems*, Vol. 18(4), pp. 736-739, November.

- RISSANEN, J. 1983. "A Universal Prior for the Integers and Estimation by Minimum Description Length," *Ann. Stat.*, Vol. 11, pp. 417–431.
- ROBERTS, R. A., and MULLIS, C. T. 1987. *Digital Signal Processing*, Addison-Wesley, Reading, MA.
- ROBINSON, E. A. 1962. *Random Wavelets and Cybernetic Systems*, Charles Griffin, London.
- ROBINSON, E. A. 1982. "A Historical Perspective of Spectrum Estimation," *Proc. IEEE*, Vol. 70, pp. 885–907, September.
- ROBINSON, E. A., and TREITEL, S. 1978. "Digital Signal Processing in Geophysics," in *Applications of Digital Signal Processing*, A. V. Oppenheim, ed., Prentice Hall, Upper Saddle River, NJ.
- ROBINSON, E. A., and TREITEL, S. 1980. *Geophysical Signal Analysis*, Prentice Hall, Upper Saddle River, NJ.
- ROY, R., PAULRAJ, A., and KAILATH, T. 1986. "ESPRIT: A Subspace Rotation Approach to Estimation of Parameters of Cisoids in Noise," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, pp. 1340–1342, October.
- SAFRANEK, R. J., MACKAY, K., JAYANT, N. W., and KIM, T. 1988. "Image Coding Based on Selective Quantization of the Reconstruction Noise in the Dominant Sub-Band," *Proc. 1988 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 765–768, April.
- SAKAI, H. 1979. "Statistical Properties of AR Spectral Analysis," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-27, pp. 402–409, August.
- SANDBERG, I. W., and KAISER, J. F. 1972. "A Bound on Limit Cycles in Fixed-Point Implementations of Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-20, pp. 110–112, June.
- SATORIUS, E. H., and ALEXANDER J. T. 1978. "High Resolution Spectral Analysis of Sinusoids in Correlated Noise," *Proc. 1978 ICASSP*, pp. 349–351, Tulsa, OK, April 10–12.
- SAYED, A. H. 2003 *Adaptive Filters*, Wiley, New York.
- SCHAFER, R. W., and RABINER, L. R. 1973. "A Digital Signal Processing Approach to Interpolation," *Proc. IEEE*, Vol. 61, pp. 692–702, June.
- SCHEUERMANN, H., and GOCKLER, H. 1981. "A Comprehensive Survey of Digital Transmultiplexing Methods," *Proc. IEEE*, Vol. 69, pp. 1419–1450.
- SCHMIDT, R. D. 1981. "A Signal Subspace Approach to Multiple Emitter Location and Spectral Estimation," Ph.D. dissertation, Department of Electrical Engineering, Stanford University, Stanford, CA, November.
- SCHMIDT, R. D. 1986. "Multiple Emitter Location and Signal Parameter Estimation," *IEEE Trans. Antennas and Propagation*, Vol. AP 34, pp. 276–280, March.
- SCHOTT, J. P., and MCCLELLAN, J. H. 1984. "Maximum Entropy Power Spectrum Estimation with Uncertainty in Correlation Measurements," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 410–418, April.
- SCHUR, I. 1917. "On Power Series Which Are Bounded in the Interior of the Unit Circle," *J. Reine Angew. Math.*, Vol. 147, pp. 205–232, Berlin. For an English translation of the paper, see Gohberg 1986.
- SCHUSTER, SIR ARTHUR. 1898. "On the Investigation of Hidden Periodicities with Application to a Supposed Twenty-Six-Day Period of Meteorological Phenomena," *Terr. Mag.*, Vol. 3, pp. 13–41, March.
- SEDLMEYER, A., and FETTWEIS, A. 1973. "Digital Filters with True Ladder Configuration," *Int. J. Circuit Theory Appl.*, Vol. 1, pp. 5–10, March.
- SHANKS, J. L. 1967. "Recursion Filters for Digital Processing," *Geophysics*, Vol. 32, pp. 33–51, February.
- SHANNON, C. E. 1949. "Communication in the Presence of Noise," *Proc. IRE*, pp. 10–21, January.
- SHEINGOLD, D. H., ed. 1986. *Analog-Digital Conversion Handbook*, Prentice Hall, Upper Saddle River, NJ.
- SIEBERT, W. M. 1986. *Circuits, Signals and Systems*, McGraw-Hill, New York.
- SINGLETON, R. C. 1967. "A Method for Computing the Fast Fourier Transform with Auxiliary Memory and Limit High Speed Storage," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-15, pp. 91–98, June.
- SINGLETON, R. C. 1969. "An Algorithm for Computing the Mixed Radix Fast Fourier Transform," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-17, pp. 93–103, June.

- SLOCK, D. T. M., and KAILATH, T. 1988. "Numerically Stable Fast Recursive Least Squares Transversal Filters," *Proc. 1988 Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 1364–1368, NY, April.
- SLOCK, D. T. M., and KAILATH, T. 1991. "Numerically Stable Fast Transversal Filters for Recursive Least Squares Adaptive Filtering," *IEEE Trans. Signal Processing*, Vol. 39, pp. 92–114, January.
- SMITH, M. J. T., and BARWELL, T. P. 1984. "A Procedure for Designing Exact Reconstruction Filter Banks for Tree Structured Subband Coders," *Proc. 1984 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 27.1.1–27.1.4, San Diego, March.
- SMITH, M. J. T., and EDDINS, S. L. 1988. "Subband Coding of Images with Octave Band Tree Structures," *Proc. 1987 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1382–1385, Dallas, April.
- STARK, H., and WOODS, J. W. 1994. *Probability, Random Processes, and Estimation Theory for Engineers*, 2nd Ed., Prentice Hall, Upper Saddle River, NJ.
- STEIGLITZ, K. 1965. "The Equivalence of Digital and Analog Signal Processing," *Inf. Control*, Vol. 8, pp. 455–467, October.
- STEIGLITZ, K. 1970. "Computer-Aided Design of Recursive Digital Filters," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-18, pp. 123–129, June.
- STOCKHAM, T. G. 1966. "High Speed Convolution and Correlation," *1966 Spring Joint Computer Conference, AFIPS Proc.*, Vol. 28, pp. 229–233.
- STORER, J. E. 1957. *Passive Network Synthesis*, McGraw-Hill, New York.
- STRANG, G. 1999. "The Discrete Cosine Transform." *SIAM Review*, Vol. 41(1), pp. 135–137.
- SWARZTRAUBER, P. 1986. "Symmetric FFT's," *Mathematics of Computation*, Vol. 47, pp. 323–346, July.
- SWINGLER, D. N. 1979a. "A Comparison Between Burg's Maximum Entropy Method and a Nonrecursive Technique for the Spectral Analysis of Deterministic Signals," *J. Geophys. Res.*, Vol. 84, pp. 679–685, February.
- SWINGLER, D. N. 1979b. "A Modified Burg Algorithm for Maximum Entropy Spectral Analysis," *Proc. IEEE*, Vol. 67, pp. 1368–1369, September.
- SWINGLER, D. N. 1980. "Frequency Errors in MEM Processing," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-28, pp. 257–259, April.
- SZEGO, G. 1967. *Orthogonal Polynomials*, 3d ed., Colloquium Publishers, No. 23, American Mathematical Society, Providence, RI.
- THORVALDSEN, T. 1981. "A Comparison of the Least-Squares Method and the Burg Method for Autoregressive Spectral Analysis," *IEEE Trans. Antennas and Propagation*, Vol. AP-29, pp. 675–679, July.
- TONG, H. 1975. "Autoregressive Model Fitting with Noisy Data by Akaike's Information Criterion," *IEEE Trans. Information Theory*, Vol. IT-21, pp. 476–480, July.
- TONG, H. 1977. "More on Autoregressive Model Fitting with Noise Data by Akaike's Information Criterion," *IEEE Trans. Information Theory*, Vol. IT-23, pp. 409–410, May.
- TRETTNER, S. A. 1976. *Introduction to Discrete-Time Signal Processing*, Wiley, New York.
- TUFTS, D. W., and KUMARESAN, R. 1982. "Estimation of Frequencies of Multiple Sinusoids: Making Linear Prediction Perform Like Maximum Likelihood," *Proc. IEEE*, Vol. 70, pp. 975–989, September.
- ULRYCH, T. J., and BISHOP, T. N. 1975. "Maximum Entropy Spectral Analysis and Autoregressive Decomposition," *Rev. Geophys. Space Phys.*, Vol. 13, pp. 183–200, February.
- ULRYCH, T. J., and CLAYTON, R. W. 1976. "Time Series Modeling and Maximum Entropy," *Phys. Earth Planet. Inter.*, Vol. 12, pp. 188–200, August.
- UNGERBOECK, G. 1972. "Theory on the Speed of Convergence in Adaptive Equalizers for Digital Communication," *IBM J. Res. Devel.*, Vol. 16, pp. 546–555, November.
- VAIDYANATHAN, P. P. 1990. "Multirate Digital Filters, Filter Banks, Polyphase Networks, and Applications: A Tutorial," *Proc. IEEE*, Vol. 78, pp. 56–93, January.
- VAIDYANATHAN, P. P. 1993. *Multirate Systems and Filter Banks*, Prentice Hall, Upper Saddle River, NJ.
- VAUGHAN, R., SCOTT, N., and WHITE, D. 1991. "The Theory of Bandpass Sampling," *IEEE Trans. on signal Procesing*, Vol. 39(9), pp. 1973–1984.
- VETTERLI, J. 1984. "Multi-dimensional Sub-Band Coding: Some Theory and Algorithms," *Signal Processing*, Vol. 6, pp. 97–112, April.

- VETTERLI, J. 1987. "A Theory of Multirate Filter Banks," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-35, pp. 356–372, March.
- VIEIRA, A. C. G. 1977. "Matrix Orthogonal Polynomials with Applications to Autoregressive Modeling and Ladder Forms," Ph.D. dissertation, Department of Electrical Engineering, Stanford University, Stanford, CA, December.
- WALKER, G. 1931. "On Periodicity in Series of Related Terms," *Proc. R. Soc., Ser. A*, Vol. 313, pp. 518–532.
- WANG, Z. 1984. "Fast Algorithms for the Discrete W Transform for the Discrete Fourier Transform." *IEEE Trans. on ASSP*, Vol. 32(4), pp. 803–816, August.
- WATERS, W., and JARRETT, B. 1982. "Bandpass Signal Sampling and Coherent Detection." *IEEE Trans. on Aerospace and Electronic Systems*, Vol. 18(4), pp. 731–736, November.
- WAX, M., and KAILATH, T. 1985. "Detection of Signals by Information Theoretic Criteria," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, pp. 387–392, April.
- WEINBERG, L. 1962. *Network Analysis and Synthesis*, McGraw-Hill, New York.
- WELCH, P. D. 1967. "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging over Short Modified Periodograms," *IEEE Trans. Audio and Electroacoustics*, Vol. AU-15, pp. 70–73, June.
- WIDROW, B., and HOFF, M. E., Jr. 1960. "Adaptive Switching Circuits," *IRE WESCON Conv. Rec.*, pt. 4, pp. 96–104.
- WIDROW, B., MANTEY, P., and GRIFFITHS, L. J. 1967. "Adaptive Antenna Systems," *Proc. IEEE*, Vol. 55, pp. 2143–2159, December.
- WIDROW, B. et al. 1975. "Adaptive Noise Cancelling Principles and Applications," *Proc. IEEE*, Vol. 63, pp. 1692–1716, December.
- WIDROW, B., MANTEY, P., and GRIFFITHS, L. J. 1967. "Adaptive Antenna systems." *Proc. IEEE*, Vol. 55, pp. 2143–2159, December.
- WIENER, N. 1949. *Extrapolation, Interpolation and Smoothing of Stationary Time Series with Engineering Applications*, Wiley, New York.
- WIENER, N., and PALEY, R. E. A. C. 1934. *Fourier Transforms in the Complex Domain*, American Mathematical Society, Providence, RI.
- WINOGRAD, S. 1976. "On Computing the Discrete Fourier Transform," *Proc. Natl. Acad. Sci.*, Vol. 73, pp. 105–106.
- WINOGRAD, S. 1978. "On Computing the Discrete Fourier Transform," *Math. Comp.*, Vol. 32, pp. 177–199.
- WOLD, H. 1938. *A Study in the Analysis of Stationary Time Series*, reprinted by Almqvist & Wiksell, Stockholm, 1954.
- WOOD, L. C., and TREITEL, S. 1975. "Seismic Signal Processing," *Proc. IEEE*, Vol. 63, pp. 649–661, April.
- WOODS, J. W., and O'NEIL, S. D. 1986. "Subband Coding of Images," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, pp. 1278–1288, October.
- YOULA, D., and KAZANJIAN, N. 1978. "Bauer-Type Factorization of Positive Matrices and the Theory of Matrices Orthogonal on the Unit Circle," *IEEE Trans. Circuits and Systems*, Vol. CAS-25, pp. 57–69, January.
- YULE, G. U. 1927. "On a Method of Investigating Periodicities in Disturbed Series with Special References to Wolfer's Sunspot Numbers," *Philos. Trans. R. Soc. London*, Ser. A, Vol. 226, pp. 267–298, July.
- ZADEH, L. A., and DESOER, C. A. 1963. *Linear System Theory: The State-Space Approach*, McGraw-Hill, New York.
- ZVEREV, A. I. 1967. *Handbook of Filter Synthesis*, Wiley, New York.

Answers to Selected Problems

Chapter 1

- 1.1** (a) One dimensional, multichannel, discrete time, and digital.
(b) Multi dimensional, single channel, continuous-time, analog.
(c) One dimensional, single channel, continuous-time, analog.
(d) One dimensional, single channel, continuous-time, analog.
(e) One dimensional, multichannel, discrete-time, digital.
- 1.3** (a) Periodic with period $T_p = \frac{2\pi}{5}$.
(c) $f = \frac{1}{12\pi} \Rightarrow$ non-periodic.
(e) $\cos(\frac{\pi n}{2})$ is periodic with period $N_p = 4$; $\sin(\frac{\pi n}{8})$ is periodic with period $N_p = 16$; $\cos(\frac{\pi n}{4} + \frac{\pi}{3})$ is periodic with period $N_p = 8$. Therefore, $x(n)$ is periodic with period $N_p = 16$ (16 is the least common multiple of 4, 8, 16).
- 1.4** (b) $N = 7; k = 0 1 2 3 4 5 6 7; \text{GCD}(k, N) = 7 1 1 1 1 1 1 7; N_p = 1 7 7 7 7 7 1.$
- 1.5** (a) Yes. $x(1) = 3 = 3 \sin\left(\frac{100\pi}{F_s}\right) \Rightarrow 200$ samples/sec.
- 1.6** (b) If $x(n)$ is periodic, then $f = k/N$ where N is the period. Then, $T_d = \left(\frac{k}{f}T\right) = k\left(\frac{T_p}{f}\right)T = kT_p$. Thus, it takes k periods (kT_p) of the analog signal to make 1 period (T_d) of the discrete signal.
- 1.8** (a) $F_{\max} = 100\text{Hz}, F_s \geq 2F_{\max} = 200 \text{ Hz}.$
(b) $F_{fold} = \frac{F_s}{2} = 125 \text{ Hz}.$
- 1.10** (a) $F_s = 1000 \text{ samples/sec}; F_{fold} = 500 \text{ Hz}.$
(b) $F_{\max} = 900 \text{ Hz}; F_N = 2F_{\max} = 1800 \text{ Hz}.$
(c) $f_1 = 0.3; f_2 = 0.9$; But $f_2 = 0.9 > 0.5 \Rightarrow f_2 = 0.1$. Hence, $x(n) = 3 \cos[(2\pi)(0.3)n] + 2 \cos[(2\pi)(0.1)n]$.
(d) $\Delta = \frac{10}{1023}.$

Chapter 2

- 2.7** (a) Static, nonlinear, time invariant, causal, stable.
(c) Static, linear, time variant, causal, stable.
(e) Static, nonlinear, time invariant, causal, stable.
(h) Static, linear, time invariant, causal, stable.
(k) Static, nonlinear, time invariant, causal, stable.

2.11 Since the system is linear and $x_1(n) + x_2(n) = \delta(n)$, it follows that the impulse response of the system is $y_1(n) + y_2(n) = \{0, 3, -1, 2, 1\}$.

If the system were time invariant, the response to $x_3(n)$ would be $\{3, 2, 1, 3, 1\}$. But this is not the case.

2.16 (b) (1) $y(n) = h(n) * x(n) = \{1, 3, 7, 7, 7, 6, 4\}; \sum_n y(n) = 35, \sum_k h(k) = 5, \sum_k x(k) = 7$.

$$(4) y(n) = \{1, 2, 3, 4, 5\}; \sum_n y(n) = 15, \sum_n h(n) = 1, \sum_n x(n) = 15$$

$$(7) y(n) = \{0, 1, 4, -4, -5, -1, 3\} \sum_n y(n) = -2, \sum_n h(n) = -1, \sum_n x(n) = 2$$

$$(10) y(n) = \{1, 4, 4, 4, 10, 4, 4, 1, 1\}; \sum_n y(n) = 36, \sum_n h(n) = 6, \sum_n x(n) = 6$$

2.19 $y(n) = \sum_{k=0}^4 h(k)x(n-k), x(n) = \left\{a^{-3}, a^{-2}, a^{-1}, \underset{\uparrow}{1}, a, \dots, a^5\right\}, h(n) = \left\{1, 1, 1, 1, 1, \underset{\uparrow}{1}\right\}; y(n) = \sum_{k=0}^4 x(n-k), -3 \leq n \leq 9; y(n) = 0, \text{ otherwise}$

2.22 (a) $y_1(n) = x(n) + x(n-1) = \{1, 5, 6, 5, 8, 8, 6, 7, 9, 12, 15, 9\}$
 $y_4(n) = \{0.25, 1.5, 2.75, 2.75, 3.25, 4, 3.5, 3.25, 3.75, 5.25, 6.25, 7, 6, 2.25\}$

2.26 With $x(n) = 0$, we have $y(n-1) + \frac{4}{3}y(n-2) = 0$ $y(-1) = -\frac{4}{3}y(-2)$; $y(0) = (-\frac{4}{3})^2 y(-2)$; $y(1) = (-\frac{4}{3})^3 y(-2)$

Therefore, $y(k) = (-\frac{4}{3})^{k+2} y(-2) \leftarrow \text{zero-input response.}$

2.32 (a) $L_1 = N_1 + M_1$ and $L_2 = N_2 + M_2$

(c) $N_1 = -2, N_2 = 4, M_1 = -1, M_2 = 2$

Partial overlap from left: $n = -3$ $n = -1$ $L_1 = -3$; Full overlap: $n = 0$ $n = 3$; Partial overlap from right: $n = 4$ $n = 6$ $L_2 = 6$

2.34 $h(n) = \left\{1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}\right\}; y(n) = \left\{1, 2, 2, 5, 3, 3, 3, 2, 1, 0\right\}$

Then, $x(n) = \left\{1, \frac{3}{2}, \frac{3}{2}, \frac{7}{4}, \frac{3}{2}\right\}$

2.38 $\sum_{n=-\infty}^{\infty} |h(n)| = \sum_{n=0, \text{neven}}^{\infty} |a|^n; \sum_{n=-\infty}^{\infty} |a|^{2n} = \frac{1}{1-|a|^2}$

Stable if $|a| < 1$

2.40 $y(n) = 2[1 - (\frac{1}{2})^{n+1}]u(n) - 2[1 - (\frac{1}{2})^{n-9}]u(n-10)$

2.41 (a) $y(n) = \frac{2}{3}[2^{n+1} - (\frac{1}{2})^{n+1}]u(n)$

2.42 (a) $h_c(n) = h_1(n) * h_2(n) * h_3(n) = [\delta(n) - \delta(n-1)] * u(n) * h(n) = h(n)$

(b) No.

2.45 $y(n) = -\frac{1}{2}y(n-1) + z(n) + 2x(n-2); y(-2) = 1, y(-1) = \frac{3}{2}, y(0) = \frac{17}{4}, y(1) = \frac{47}{8}, \dots$

2.48 (a) $y(n) = ay(n-1) + bx(n) \Rightarrow h(n) = ba^n u(n) \sum_{n=0}^{\infty} h(n) = \frac{b}{1-a} = 1 \Rightarrow b = 1 - a$.

(b) $s(n) = \sum_{k=0}^n h(n-k) = b \left[\frac{1-a^{n+1}}{1-a} \right] u(n)$

$s(\infty) = \frac{b}{1-a} = 1 \Rightarrow b = 1 - a$

2.51 (a) $y(n) = \frac{1}{3}x(n) + \frac{1}{3}x(n-3) + y(n-1)$ for $x(n) = \delta(n)$, we have
 $h(n) = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \dots\right\}$

(b) $y(n) = \frac{1}{2}y(n-1) + \frac{1}{8}y(n-2) + \frac{1}{2}x(n-2), y(-1) = y(-2) = 0$
with $x(n) = \delta(n), h(n) = \{0, 0, \frac{1}{2}, \frac{1}{4}, \frac{3}{16}, \frac{1}{8}, \frac{11}{128}, \frac{15}{256}, \frac{41}{1024}, \dots\}$

(c) $y(n) = 1.4y(n-1) - 0.48y(n-2) + x(n), y(-1) - y(-2) = 0$
with $x(n)\delta(n), h(n) = \{1, 1, 4, 1.48, 1.4, 1.2496, 1.0774, 0.9086, \dots\}$

(d) All three systems are IIR.

2.54 (a) convolution: $y_1(n) = \left\{ \begin{array}{l} 13, 7, 7, 7, 7, 4 \\ \downarrow \end{array} \right\}$
 correlation: $\gamma_1(n) = \left\{ \begin{array}{l} 1, 3, 7, 7, 7, 4 \\ \downarrow \end{array} \right\}$

(c) convolution: $y_4(n) = \left\{ \begin{array}{l} 1, 4, 10, 20, 25, 24, 16 \\ \downarrow \end{array} \right\}$
 correlation: $\gamma_4(n) = \left\{ \begin{array}{l} 4, 11, 20, 30, 20, 11, 4 \\ \downarrow \end{array} \right\}$

2.58 $h(n) = [c_1 2^n + c_2 n 2^n] u(n)$
 With $y(0) = 1$, $y(1) = 3$, we have, $c_1 = 1$, and $c_2 = \frac{1}{2}$.

2.61 $x(n) = \begin{cases} 1, & n_0 - N \leq n \leq n_0 + N \\ 0 & \text{otherwise} \end{cases}$

$$r_{xx}(l) = \begin{cases} 2N+1-|l|, & -2N \leq l \leq 2N \\ 0, & \text{otherwise} \end{cases}$$

2.63 $r_{xx}(l) = \sum_{n=-\infty}^{\infty} x(n)x(n-l) = \begin{cases} 2N+1-|l|, & -2N \leq l \leq 2N \\ 0, & \text{otherwise} \end{cases}$

Since $r_{xx}(0) = 2N+1$, the normalized autocorrelation is

$$\rho_{xx}(l) = \begin{cases} \frac{1}{2N+1} (2N+1-|l|), & -2N \leq l \leq 2N \\ 0, & \text{otherwise} \end{cases}$$

Chapter 3

3.1 (a) $X(z) = 3z^5 + 6 + z^{-1} - 4z^{-2}$ ROC : $0 < |z| < \infty$

3.2 (a) $X(z) = \frac{1}{(1-z^{-1})^2}$

(d) $X(z) = \frac{[az^{-1}-(az^{-1})^3]\sin w_0}{[1-(2a\cos w_0)z^{-1}+a^2z^{-2}]^2}, |z| < a$

(h) $X(z) = \frac{1-(\frac{1}{2}z^{-1})^{10}}{1-\frac{1}{2}z^{-1}}, |z| > \frac{1}{2}$

3.4 (a) $X(z) = \frac{z^{-1}}{(1+z^{-1})^2}, |z| > 1$

(f) $X(z) = 1 - z^{-2} + z^{-4} - z^{-5}, z \neq 0$

3.8 (a) $Y(z) = \frac{X(z)}{1-z^{-1}}$

3.12 (a) $x(n) = [4(2)^n - 3 - n]u(n)$

3.14 (a) $x(n) = [2(-1)^n - (-2)^n]u(n)$

(e) $x(n) = -\left[\frac{3}{5}\left(\frac{1}{\sqrt{2}}\right)^n \cos \frac{\pi}{4}n + \frac{23}{10}\left(1\sqrt{2}\right)^n \sin \frac{\pi}{4}n + \frac{17}{20}(12)^n\right]u(n)$

(j) $x(n) = \left(-\frac{1}{a}\right)^{n+1}u(n) + \left(\frac{1}{a}\right)^{n-1}u(n-1)$

3.16 (a) $y(n) = \left[-\frac{4}{3}\left(\frac{1}{4}\right)^n + \frac{1}{3} + \left(\frac{1}{2}\right)^n\right]u(n)$

(d) $y(n) = [-2(n+1) + 2^{n+1}]u(n)$

3.19 (b) $x(n) = -\left(\frac{1}{n}\right)\left(\frac{1}{2}\right)^n u(n-1)$

3.24 (a) $x(n) = [0.136(0.28)^n + 0.864(-1.78)^n]u(n)$

3.35 (a) $y(n) = \left[\frac{1}{7}\left(\frac{1}{3}\right)^n \frac{6}{7}\left(\frac{1}{2}\right)^n \cos \frac{\pi n}{3} + \frac{3\sqrt{3}}{7}\left(\frac{1}{2}\right)^n \sin \frac{\pi n}{3}\right]u(n)$

(d) $y(n) = \frac{10}{\sqrt{2}} \sin\left(\frac{\pi n}{2} + \frac{\pi}{4}\right)u(n)$

(h) $y(n) = \left[4\left(\frac{1}{2}\right)^n - n\left(\frac{1}{4}\right)^n - 3\left(\frac{1}{4}\right)^n\right]u(n)$

3.38 (a) $h(n) = \left[2\left(\frac{1}{2}\right)^n - \left(\frac{1}{4}\right)^n \right] u(n)$
 $y(n) = \left[\frac{8}{3} - 2\left(\frac{1}{2}\right)^n + \frac{1}{3}\left(\frac{1}{4}\right)^n \right] u(n)$

(d) $h(n) = \left[2\left(\frac{2}{5}\right)^n - \left(\frac{1}{5}\right)^n \right] u(n)$
 $y(n) = \left[\frac{25}{12} + \frac{1}{4}\left(\frac{1}{5}\right)^n - \frac{4}{3}\left(\frac{2}{5}\right)^n \right] u(n)$

3.42 (a) $h(n) = \left[-\frac{7}{2}\left(\frac{1}{5}\right)^{n-1} + \frac{9}{2}\left(\frac{2}{5}\right)^{n-1} \right] u(n-1)$

(b) $y(n) = \left[\frac{b_0+b_1}{1+a_1} + \frac{a_1b_0-b_1}{1+a_1}(-a_1)^n \right] u(n)$

3.49 (d) $y(n) = \left[\frac{4}{3} - \frac{3}{8}\left(\frac{1}{2}\right)^n + \frac{7}{24}\left(-\frac{1}{2}\right)^n \right] u(n)$

3.56 (a) $x(n) = \left(\frac{1}{2}\right)^n u(n)$

(d) $x(n) = \left[\frac{3}{10}\left(\frac{1}{2}\right)^n \frac{7}{10}\left(-\frac{1}{3}\right)^n \right] u(n)$

3.58 ROC: $a < |z| < 1/a$
 $x(-18) = -32/15309$

Chapter 4

4.2 (a) $X_a(f) = \frac{A}{a+j2\pi F}; |X_a(F)| = \frac{A}{\sqrt{a^2+(2\pi F)^2}}; \angle X_a(f) = -\tan^{-1} \frac{2\pi f}{a}$

4.5 (a) $x(n) = \left\{ \frac{11}{2}, 2 + \frac{3}{4}\sqrt{2}, 1, 2 - \frac{3}{4}\sqrt{2}, \frac{1}{2}, 2 - \frac{3}{4}\sqrt{2}, 1, 2 + \frac{3}{4}\sqrt{2} \right\}$
Hence, $c_0 = 2, c_1 = c_7 = 1, c_2 = c_6 = \frac{1}{2}, c_3 = c_5 = \frac{1}{4}, c_4 = 0$

(b) $P = \frac{53}{8}$

4.6 (a) $c_k = \begin{cases} \frac{1}{2j}, & k = 3 \\ \frac{1}{2}, & k = 5 \\ \frac{1}{2}, & k = 10 \\ \frac{-1}{2j}, & k = 12 \\ 0, & \text{otherwise} \end{cases}$

(d) $c_0 = 0,$
 $c_1 = \frac{2j}{5} \left[-\sin\left(\frac{2\pi}{5}\right) + 2\sin\left(\frac{4\pi}{5}\right) \right]$
 $c_2 = \frac{2j}{5} \left[\sin\left(\frac{4\pi}{5}\right) - 2\sin\left(\frac{2\pi}{5}\right) \right];$
 $c_3 = -c_2; c_4 = -c_1$

(h) $N = 2; c_k = \frac{1}{2}(1 - e^{-j\pi k}); c_0 = 0, c_1 = -1$

4.9 (a) $X(\omega) = \frac{1-e^{-j6\omega}}{1-e^{-j\omega}}$

(e) $\sum_{n=-\infty}^{\infty} |x(n)| \rightarrow \infty.$
Therefore, the Fourier transform does not exist.

(b) $X(\omega) = (2M+1)A + 2A \sum_{k=1}^M (2M+1-k) \cos wk$

4.14 (a) $X(0) = \sum_n x(n) = -1;$

(e) $\int_{-\pi}^{\pi} |X(\omega)|^2 d\omega = 2\pi \sum_n |x(n)|^2 = (2\pi)(19) = 38\pi$

4.17 (a) $\sum_n x^*(n)e^{-j\omega n} = (\sum_n x(n)e^{-j(-\omega)n})^* = X^*(-\omega)$

(c) $Y(\omega) = X(\omega) + X(\omega)e^{-j\omega} = (1 - e^{-j\omega})X(\omega)$

(f) $Y(\omega) = X(2\omega)$

- 4.19** (a) $X_1(\omega) = \frac{1}{2j} [X(\omega - \frac{\pi}{4}) + X(\omega + \frac{\pi}{4})]$
 (d) $X_4(\omega) = \frac{1}{2} [X(\omega - \pi) + X(\omega + \pi)] = X(\omega - \pi)$
- 4.22** (a) $X_1(\omega) = \frac{e^{j\omega/2}}{1 - ae^{j\omega/2}}$
 (d) $X_4(\omega) = \frac{1}{2} [X(\omega - 0.3\pi) + X(\omega + 0.3\pi)]$
- 4.23** (b) $Y_2(\omega) = X\left(\frac{\omega}{2}\right)$

Chapter 5

- 5.1** (a) Because the range of n is $(-\infty, \infty)$, the Fourier transforms of $x(n)$ and $y(n)$ do not exist. However, the relationship implied by the forms of $x(n)$ and $y(n)$ is $y(n) = x^3(n)$. In this case, the system H_1 is non-linear.

(b) In this case, $H(\omega) = \frac{1 - \frac{1}{4}e^{-j\omega}}{1 - \frac{1}{8}e^{-j\omega}}$, so the system is LTI.

5.3 (a) $|H(\omega)| = \frac{1}{[\frac{5}{4} - \cos \omega]^{\frac{1}{2}}}; \angle H(\omega) = -\tan^{-1} \frac{\frac{1}{2} \sin \omega}{1 - \frac{1}{2} \cos \omega}$

- 5.4** (a) $H(\omega) = (\sin \omega)e^{j\pi/2}$
 (j) $H(\omega) = (\cos \omega) (\cos \frac{\omega}{2}) e^{-j3\omega/2}$
 (n) $H(\omega) = (\sin^2 \omega/2)e^{-j(\omega-\pi)}$

5.8 $y_{ss}(n) = 3 \cos(\frac{\pi}{2}n + 60^\circ); y_{tr}(n) = 0$

5.12 (a) $H(\omega) = \frac{b}{1 - 0.9e^{-j\omega}}; |H(0)| = 1, \Rightarrow b = \pm 0.1$

$$\Theta(\omega) = \begin{cases} -\frac{\omega M}{2}, & \cos \frac{\omega M}{2} > 0 \\ \pi - \frac{\omega M}{2}, & \cos \frac{\omega M}{2} < 0 \end{cases}$$

(b) $|H(\omega_0)|^2 = \frac{1}{2} \Rightarrow \frac{b^2}{1.81 - 1.8 \cos \omega_0} = \frac{1}{2} \Rightarrow \omega_0 = 0.105$

5.16 (a) $|H(\omega)| = \frac{4}{5 - 3 \cos \omega}; \angle H(\omega) = 0$

5.18 (a) $H(\omega) = 2e^{-j2\omega} e^{j\pi/2} \sin 2\omega$

(b) $y(n) = 2 \cos \pi n/4, H\left(\frac{\pi}{4}\right) = 2, \angle H\left(\frac{\pi}{4}\right) = 0$

5.20 (a) $Y(\omega) = X\left(\frac{\omega}{2}\right) = \begin{cases} 1, & |\omega| \leq \frac{\pi}{2} \\ 0, & \frac{\pi}{2} \leq |\omega| \leq p \end{cases}$

(b) $Y(\omega) = \frac{1}{2} [X(\omega - \pi) + X(\omega + \pi)] = \begin{cases} 0, & |\omega| \leq \frac{3\pi}{4} \\ \frac{1}{2}, & \frac{3\pi}{4} \leq |\omega| < \pi \end{cases}$

5.22 (a) $|H(\omega)| = 2|\sin 5\omega|,$
 $\Theta(\omega) = \begin{cases} \frac{\pi}{2} - 5\omega & \text{for } \sin 5\omega > 0 \\ \frac{\pi}{2} - 5\omega + \pi, & \text{for } \sin 5\omega < 0 \end{cases}$

(b) $|H\left(\frac{\pi}{10}\right)| = 2, \angle H\left(\frac{\pi}{10}\right) = 0; |H\left(\frac{\pi}{3}\right)| = \sqrt{3}, \Theta\left(\frac{\pi}{3}\right) = \angle H\left(\frac{\pi}{3}\right) = -\frac{\pi}{6}$

(1) Hence, $y(n) = 2 \cos \frac{\pi}{10} n + 3\sqrt{3} \sin \left(\frac{\pi}{3} n - \frac{\pi}{15}\right)$

(2) $H(0) = 0, H\left(\frac{2\pi}{5}\right) = 0$; Hence, $y(n) = 0$

5.24 (a) $H(z) = \frac{2}{1 - \frac{1}{2}z^{-1}} - 1; h(n) = 2\left(\frac{1}{2}\right)^n u(n) - \delta(n)$

5.30 $|H(\omega)| = \cos^2 \frac{\omega}{2}; \Theta(\omega) = \angle H(\omega) = -\omega$

5.33 (a) $H(\omega) = \frac{1}{2M+1} \left[1 + 2 \sum_{k=1}^{M-1} \cos \omega k \right]$

(b) $H(\omega) = \frac{1}{2M} \cos M\omega + \frac{1}{2M} \left[1 + 2 \sum_{k=1}^{M-1} \cos \omega k \right]$

- 5.39** (a) $|H_1(\omega)|^2 = \frac{1}{2} \Rightarrow \cos w_2 = \frac{4a-1-a^2}{2a}$
 (b) $|H_2(\omega)|^2 = (\frac{1}{2}) \Rightarrow \cos \omega \frac{2a}{1+a^2}$

By comparing the results of (a) and (b) we conclude that $\omega_2 < \omega_1$. Therefore, the second filter has a smaller 3dB bandwidth.

- 5.49** (a) Replace z by z^8 . We need 8 zeros at the frequencies $\omega = 0, \pm\frac{\pi}{4}, \pm\frac{\pi}{3}, \pm\frac{3\pi}{4}, \pi$. Hence,
 $H(z) = \frac{1-z^8}{1-a z^{-8}}; y(n) = ay(n-8) + x(n) - x(n-8)$

5.53

$$H_r(\omega) = 2[h(0)\sin\frac{3\omega}{2} + h(1)\sin\frac{\omega}{2}]; H_r\left(\frac{\pi}{4}\right) = \frac{1}{2}; H_r\left(\frac{3\pi}{4}\right) = 1 \\ H_r\left(\frac{3\pi}{4}\right) = 2h(0)\sin\frac{9\pi}{8} + 2h(1)\sin\frac{3\pi}{8} = 1 \\ 1.85h(0) + 0.765h(1) = \frac{1}{2} \\ -0.765h(0) + 1.85h(1) = 1 \\ h(1) = 0.56, h(0) = 0.04$$

- 5.59** (a) $H(z) = \frac{0.1}{1-0.9z^{-1}}; H_{bp}(\omega) = H\left(\omega - \frac{\pi}{2}\right) = \frac{0.1}{1-0.9e^{-j\pi-\frac{\pi}{2}}}$
 (b) $h(n) = 0.1(0.9e^{j\pi/2})^n u(n)$

- 5.61** (a) $H(z) = \frac{b}{1-a z^{-1}}; H(\omega) = \frac{b}{1-a e^{-j\omega}}; b = \pm(1-a)$
 (b) $|H(\omega)|^2 = \frac{b^2}{1+a^2-2a\cos\omega} = \frac{1}{2}; \omega_3 = \cos^{-1}\left(\frac{4a-1-a^2}{2a}\right)$

- 5.66** (a) $h(n) = \left[-\frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n\right]u(-n-1)$
 $y(n) = y(n-1) + y(n-2) + x(n-1)$

- 5.68** $r_{xx}(n) = \frac{16}{15}\left(\frac{1}{4}\right)^n u(n) + \frac{16}{15}(4)^n u(-n-1); r_{hh}(n) = \frac{4}{3}\left(\frac{1}{2}\right)^n u(n) + \frac{4}{3}(2)^n u(-n-1)$
 $r_{xy}(n) = \frac{16}{15}(2)^n u(n-1) - \frac{16}{15}(4)^n u(-n-1) - \frac{128}{105}\left(\frac{1}{4}\right)^n u(n)$
 $r_{yy}(n) = \frac{64}{21}(2)^n u(-n-1) - \frac{128}{105}(4)^n u(-n-1) + \frac{64}{21}\left(\frac{1}{2}\right)^n u(n) - \frac{128}{105}\left(\frac{1}{4}\right)^n u(n)$

- 5.77** (a) $H(z)H(z^{-1}) = \frac{\frac{5}{4}-\frac{1}{2}(z+z^{-1})}{\frac{10}{9}-\frac{1}{3}(z+z^{-1})}$
 $H(z) = \frac{1-\frac{1}{2}z^{-1}}{1-\frac{1}{3}z^{-1}}$

- 5.82** (a) $B = 10 \text{ kHz}; F_s = 20 \text{ kHz}; z_1 = \frac{10k}{20k} = 0.5; z_2 = \frac{7.777k}{20k} = 0.3889; z_3 = \frac{8.889k}{20k} = 0.4445;$
 $z_4 = \frac{6.667k}{20k} = 0.3334;$
 $H(z) = (z-0.5)(z-0.3889)(z-0.4445)(z-0.3334)$

Chapter 6

- 6.9** (a) $X_a(F) = \frac{1}{j2\pi(F+F_0)}$
 (b) $X(f) = \frac{1}{1-e^{-j2\pi(F+F_0/F_s)}}$

- 6.10** Since $\frac{F_c+\frac{B}{2}}{B} = \frac{50+10}{20} = 3$ is an integer, then $F_s = 2B = 40 \text{ Hz}$

- 6.14** $\sum_{n=-\infty}^{\infty} x^2(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} |X(w)|^2 dw = \frac{E_a}{T}$

- 6.18** Let P_d denote the power spectral density of the quantization noise. Then

$$P_n = \int_{-\frac{B}{F_s}}^{\frac{B}{F_s}} P_d df = \frac{2B}{F_s} P_d = \sigma_e^2$$

$$\text{SQNR} = 10 \log 10 \frac{\sigma_x^2}{\sigma_e^2} = 10 \log_{10} \frac{\sigma_x^2 F_s}{2B P_d} = 10 \log_{10} \frac{\sigma_x^2 F_s}{2B P_d} + 10 \log 10 F_s$$

Thus, SQNR will increase by 3 dB if F_s is doubled.

- 6.20** $H_s(z) = z^{-1}; H_n(z) = (1-z^{-1})^2$

Chapter 7

- 7.1** Since $x(n)$ is real, the real part of the DFT is even, imaginary part odd. Thus, the remaining points are $\{0.125 + j0.0518, 0, 0.125 + j0.3018\}$

7.2 (a) $\tilde{x}_2(l) = \sin\left(\frac{3\pi}{8}\right)|l|, |l| \leq 7$

Therefore, $x_1(n) \circ x_2(n) = \{1.25, 2.55, 2.55, 1.25, 0.25, -1.06, -1.06, 0.25\}$

(c) $\tilde{R}_{xx}(k) = X_1(k)X_1^*(k) = \frac{N^2}{4}[\delta(k-1) + \delta(k+1)]$
 $\Rightarrow \tilde{r}_{xx}(n) = \frac{N}{2} \cos\left(\frac{2\pi}{N}n\right)$

(d) $\tilde{R}_{yy}(k) = X_2(k)X_2^*(k) = \frac{N^2}{4}[\delta(k-1) + \delta(k+1)]$
 $\Rightarrow \tilde{r}_{yy}(n) = \frac{N}{2} \cos\left(\frac{2\pi}{N}n\right)$

7.5 (a) $\sum_{n=0}^{N-1} x_1(n)x_2^*(n) = \frac{1}{4} \sum_{n=0}^{N-1} \left(e^{j\frac{2\pi}{N}n} + e^{-j\frac{2\pi}{N}n}\right)^2 = \frac{N}{2}$

7.9 $X_3(k) = \{17, 19, 22, 19\}$

7.12 (a) $s(k) = W_2^k X(k)$
 $s(n) = \{3, 4, 0, 0, 1, 2\}$

7.14 (a) $y(n) = x_1(n) \circ x_2(n) = \{4, 0, 1, 2, 3\}$

7.21 (a) $F_s \equiv F_N = 2B = 6000$ samples/sec

(b) $L = 120$ samples

(c) $LT = \frac{1}{6000} \times 120 = 0.02$ seconds

7.23 (a) $X(k) = \sum_{n=0}^{N-1} \delta(n)e^{-j\frac{2\pi}{N}kn} = 1, 0 \leq k \leq N-1$

(e) $X(k) = N\delta(k-k_0)$

(h) $X(k) = \frac{1}{1-e^{-j\frac{2\pi}{N}k}}$

7.25 (a) $X(w) = 3 + 2\cos(2w) + 4\cos(4w)$

7.31 (a) $c_k = \left(\frac{2}{\pi}, -\frac{1}{\pi}, \frac{2}{3\pi}, -\frac{1}{2\pi}, \dots\right)$

7.32 (a) $Y(j\Omega) = T_0 \sin c\left(\frac{T_0(\Omega-\Omega_0)}{2}\right) e^{-j\frac{T_0(\Omega-\Omega_0)}{2}}$

(c) $Y(w) = \frac{\sin \frac{N}{2}(w-w_0)}{\sin \frac{w-\Omega_0}{2}} e^{-j\frac{N-1}{2}(w-w_0)}$

Chapter 8

8.5 $X(k) = 2 + 2e^{-j\frac{2\pi}{5}} + e^{-j\frac{4\pi}{5}} + \dots + e^{-j\frac{8\pi}{5}}$

$x'(n) = \{2, 2, 1, 1, 1\}$

$x'(n) = \sum_m x(n+7m), n = 0, 1, \dots, 4$

8.8 $W_8 = \frac{1}{\sqrt{2}}(1-j)$

Refer to Fig. 8.1.9. The first stage of butterflies produces $\{2, 2, 2, 2, 0, 0, 0, 0\}$. The phase factor multiplications do not change this sequence. The next stage produces $\{4, 4, 0, 0, 0, 0, 0, 0\}$ which again remains unchanged by the phase factors. The last stage produces $\{8, 0, 0, 0, 0, 0, 0, 0\}$. The bit reversal to permute the sequence into proper order unscrambles only zeros so the result remains $\{8, 0, 0, 0, 0, 0, 0, 0\}$.

8.13 (a) “gain” = $W_8^0 W_8^0 (-1) W_8^2 = -W_8^2 = j$

(b) Given a certain output sample, there is one path from every input leading to it. This is true for every output.

(c) $X(3) = x(0) + W_8^3 x(1) - W_8^2 x(2) + W_8^2 W_8^3 x(3) - W_8^0 x(4) - W_8^0 W_8^3 x(5) + W_8^0 W_8^2 x(6) + W_8^0 W_8^2 W_8^3 x(7)$

8.16 $x = x_R + jx_I = (a + jb)(c + jd)$

$$e = (a - b)d \quad 1 \text{ add } 1 \text{ mult}$$

$$x_R = e + (c - d)a \quad 2 \text{ adds } 1 \text{ mult}$$

$$x_I = e + (c + d)b \quad 2 \text{ adds } 1 \text{ mult}$$

Total 5 adds 3 mult

8.19 $X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}$

Let $F(t)$, $t = 0, 1, \dots, N - 1$ be the DFT of the sequence on k $X(k)$.

$$F(t) = \sum_{k=0}^{N-1} X(k) W_N^{tk} = \{x(N-1), x(N-2), \dots, x(1), x(0)\}$$

8.21 (a) $W(z) = \frac{1}{2} \frac{1 - z^{-N}}{1 - z^{-1}} - \frac{1}{4} \frac{1 - \left(z^{-1} e^{j \frac{2\pi}{N-1}}\right)^N}{1 - z^{-1} e^{j \frac{2\pi}{N-1}}} - \frac{1}{4} \frac{1 - \left(z^{-1} e^{-j \frac{2\pi}{N-1}}\right)^N}{1 - z^{-1} e^{-j \frac{2\pi}{N-1}}}$

(b) $x_w(n) = w(n)x(n); X_w(k) = W(k)N X(k)$

8.32 $X(k) = \sum_{m=0}^{N-1} x(m) W_N^{-k(N-m)}$

This can be viewed as the convolution of the N -length sequence $x(n)$ with the impulse response of a linear filter.

$h_k(n) \triangleq W_N^{kn} u(n)$, evaluated at time N

$$H_k(z) = \sum_{n=0}^{\infty} W_N^{kn} z^{-n} = \frac{1}{1 - W_N^k z^{-1}} = \frac{Y_u(z)}{X(z)}$$

Then, $y_k(n) = W_N^k y_k(n-1) + x(n)$, $y_k(-1) = 0$ and $y_k(N) = X(k)$

8.34 In the DIF case, the number of butterflies affecting a given output is $\frac{N}{2}$ in the first stage, $\frac{N}{4}$ in the second, \dots . The total number is $N - 1$.

Every butterfly requires 4 real multiplies, and the error variance is $\frac{\delta^2}{12}$. Under the assumption that the errors are uncorrelated, the variance of the total output quantization error is

$$\sigma_q^2 = 4(N-1) \frac{\delta^2}{12} = \frac{N\delta^2}{3}$$

Chapter 9

9.3 $H(z) = \frac{8-z^{-1}}{1-0.5z^{-1}}$

$$h(n) = 8(0.5)^n u(n) - (0.5)^{n-1} u(n-1)$$

9.6

$$c_0 = 1; c_1 = -(b_1 + b_2); d_1 = b_1; c_2 = b_2$$

9.8 (a) $h(n) = \frac{1}{2} \left[\left(\frac{1}{2}\right)^n + \left(-\frac{1}{2}\right) \right]^n u(n)$

9.11 $H(z) = C \left(1 - \frac{53}{150}z^{-1} + 0.52z^{-2} - 0.74z^{-3} + \frac{1}{3}z^{-4}\right)$, where C is a constant

9.15 $k_2 = \frac{1}{3}; k_1 = \frac{3}{2}$

9.17 $h(n) = \delta(n) + 0.157\delta(n-1) + 0.0032\delta(n-2) + 0.88(n-3)$

9.21 $H(z) = 1 + 1.38z^{-1} + 1.311z^{-2} + 1.337z^{-3} + 0.9z^{-4}$

9.30 (a) $|H(e^{jw})|^2 = \frac{a^2 - 2a \cos w + 1}{1 - 2a \cos w + a^2} = 1 \forall w$

9.32 $y(n) = 0.999y(n-1) + e(n)$

where $e(n)$ is white noise, uniformly distributed in the interval $[-\frac{1}{2^9}, \frac{1}{2^9}]$. Then $E\{y^2(n)\} = 0.999^2 E\{y^2(n-1)\} + E\{e^2(n)\} = 6.361 \times 10^{-4}$

9.35 (a) $y(n) = Q[0.1\delta(n)] + Q[0.5y(n-1)]; y(0) = Q[0.1] = \frac{1}{8}; y(1) = Q[\frac{1}{16}] = 0$ and $y(2) = y(3) = y(4) = 0$; no limit cycle

9.37 Define $\rho_c = r \cos \theta, \rho_s = r \sin \theta$ for convenience, (a)

$$-\rho_s y(n-1) + e_1(n) + x(n) + \rho_c v(n-1) + e_2(n) = v(n)$$

$$\rho_s v(n-1) + e_3(n) + \rho_c y(n-1) + e_4(n) = y(n)$$

9.38 (a) $h(n) = \left[2 \left(\frac{1}{2}\right)^n - \left(\frac{1}{4}\right)^n \right] u(n)$
 $\sigma_q^2 = \frac{64}{35} \sigma_{e1}^2 + \frac{16}{15} \sigma_{e2}^2$

9.40 (a) $G_1 \frac{\left(1-0.8e^{j\frac{\pi}{4}}\right)\left(1-0.8e^{-j\frac{\pi}{4}}\right)}{(1-0.5)(1+\frac{1}{3})} = 1; G_1 = 1.1381$
 $G_2 \frac{(1+0.25)\left(1-\frac{5}{8}\right)}{\left(1-0.8e^{j\frac{\pi}{3}}\right)\left(1-0.8e^{-j\frac{\pi}{3}}\right)} = 1; G_2 = 1.7920$

9.41 (a) $k_1 = -\frac{4}{9}; k_2 = \frac{5}{32}$

(b) $k_2 = -\frac{1}{6}; k_1 = -\frac{1}{5}$

Chapter 10

10.1 $h_d(n) = \frac{\sin \frac{\pi}{6}(n-12)}{\pi(n-12)}; h(n) = h_d(n)w(n)$
 where $w(n)$ is a rectangular window of length $N = 25$.

10.2 $h_d(n) = \delta(n) - \frac{\sin \frac{\pi}{3}(n-2)}{\pi(n-12)} + \frac{\sin \frac{\pi}{6}(n-12)}{\pi(n-12)}; h(n) = h_d(n)w(n)$
 where $w(n)$ is a rectangular window of length 25.

10.5 $H_r(\omega) = 2 \sum_{n=0}^1 h(n) \cos [\omega (\frac{3}{2} - n)]$
 From $H_r(0) = 1$ and $H_r(\frac{\pi}{2}) = 1/2$, we obtain $h(0) = 0.073, h(1) = 0.427, h(2) = h(1)$ and $h(3) = h(0)$

10.7 $h(n) = \{0.3133, -0.0181, -0.0914, 0.0122, 0.0400, -0.0019, -0.0141, 0.52, 0.52, -0.0141, -0.0019, 0.0400, 0.0122, -0.0914, -0.0181, 0.3133\}$

10.9 $h_d(n) = \frac{\cos \pi(n-10)}{(n-10)}, \quad 0 \leq n \leq 20, n \neq 10$

$$= 0, \quad n = 10$$

Then $h(n) = h_d(n)w(n)$, where $w(n)$ is the Hamming window of length 21.

10.12 (a) Let $T = 2$. Then $H(z) = \frac{1+z^{-1}}{1-z^{-1}} \Rightarrow y(n) = y(n-1) + x(n) + x(n-1)$

10.13 $H(z) = A \frac{(1+z^{-1})(1+2z^{-1}+z^{-2})}{(1-\frac{1}{2}z^{-1})(1-\frac{1}{2}z^{-1}+\frac{1}{4}z^{-2})}$
 $H(z)|_{z=1} = 1; A = \frac{3}{64}, b_1 = 2, b_2 = 1, a_1 = 1, c_1 = -\frac{1}{2}, d_1 = -\frac{1}{2}, d_2 = \frac{1}{4}$

10.15 From the design specifications we obtain $\varepsilon = 0.509$, $\delta = 99.995$, $f_p = \frac{1}{6}$, $f_s = \frac{1}{4}$

$$\text{Butterworth filter: } N_{min} \geq \frac{\log \eta}{\log k} = 9.613 \Rightarrow N = 10$$

$$\text{Chebyshev filter: } N_{min} \geq \frac{\cos h^{-1} \eta}{\cos h^{-1} k} = 5.212 \Rightarrow N = 6$$

$$\text{Elliptic filter: } N_{min} \geq \frac{k(\sin \alpha)}{k(\cos \alpha)}, \frac{k(\cos \beta)}{k(\sin \beta)} = 3.78 \Rightarrow N = 4$$

10.19 (a) MATLAB is used to design the FIR filter using the Remez algorithm. We find that a filter of length $M = 37$ meets the specifications. We note that in MATLAB, the frequency scale is normalized to $\frac{1}{2}$ of the sampling frequency.

(b) $\delta_1 = 0.02$, $\delta_2 = 0.01$, $\Delta f = \frac{20}{100} - \frac{15}{100} = 0.05$

With equation (10.2.94) we obtain $\hat{M} = \frac{-20 \log_{10}(\sqrt{\delta_1 \delta_2}) - 13}{14.6 \Delta f} + 1 \approx 34$

With equation (10.2.95) we obtain $D_\infty(\delta_1 \delta_2) = 1.7371$; $f(\delta_1 \delta_2) = 11.166$

and $\hat{M} = \frac{D_\infty(\delta_1 \delta_2) - f(\delta_1 \delta_2)(\Delta f)^2}{\Delta f} + 1 \approx 36$

Note (10.2.95) is a better approximation of M .

10.21 (a) dc gain: $H_a(0) = 1$; 3 dB frequency: $\Omega_c = \alpha$

For all Ω , only $H(j\infty) = 0$; $h_a(\tau) = \frac{1}{e} h_a(0) = \frac{1}{e}; r = \frac{1}{\alpha}$

10.24 $H(z) = \frac{1}{6}(1 - z^{-6})(1 - z^{-1})(2 + z^{-1} + \frac{3}{2}z^{-\alpha} + \frac{1}{2}z^{-3} + z^{-4})$

This filter is FIR with zeros at $z = 1, e^{\pm j\frac{\pi}{6}}, e^{\pm j\frac{\pi}{2}}, e^{\pm j\frac{5\pi}{6}}$, $-0.55528 \pm j0.6823$ and $0.3028 \pm j0.7462$

10.25 (a) $f_L = \frac{900}{2500} = 0.36$; $f_H = \frac{1100}{2500} = 0.44$

$$h_d(n) = \frac{2 \sin 0.08\pi(n-15)}{(n-15)}; h(n) = h_d(n)w_H(n)$$

$$w_H(n) = 0.54 - 0.46 \cos \frac{2\pi(n-15)}{30}$$

Index

A

Accumulator 55
Adaptive arrays 900–902
Adaptive filters 880–959
 applications of 880–902
 for antenna arrays 900–902
 for channel equalization 883–887
 for echo cancellation 887–891
 for interference suppression 891–895
 for linear predictive coding 897–900
 for noise cancellation 896–897
 for system identification 882–883
 for system modeling 880–882
direct-form FIR 902–927
 properties of 925–927
 with FAEST algorithm 946
 with fast RLS algorithm 923–925, 945–947
 with MSE algorithm 903–907
 with RLS algorithm 907–927
 with square-root (LDU) algorithms 921–923
lattice-ladder filters 927–954
 properties of 951–954
 with a priori LS algorithm 940
 with error-feedback algorithm 944

with gradient algorithm 950–951
with normalized algorithm 949–950
Adaptive line enhancer 895–896
Adaptive noise cancelling 896–897
Akaike information criterion (AIC) 997
Algorithms 4
 Chirp-z 544–549
 FFT 511–537
 Goertzel 542–544
 Remez 686–691
Aliasing, frequency-domain 20, 389–394
 time-domain 391
Alternation theorem 684
Amplitude 14
Analog signals (*see* Signals)
Analog-to-digital (A/D) converter 5, 19, 401–410
Autocorrelation, of deterministic signals 116–128
 of random signals 323–326, 826
Autocovariance 826
Autoregressive (AR) process 836, 987
 autocorrelation of 837
Autoregressive-moving average (ARMA) process 837, 987
 autocorrelation of 837
Averages, autocorrelation 826
 autocovariance 826
 ensemble 825–828

expected value 825
for discrete-time signals 829–830
moments 825
power 826
time-averages 830–833

B

Backward predictor 578, 841
Bandlimited signals 266
Bandpass filter 327, 332–334
Bandpass signal 266
Bandwidth 265–267, 660
Bartlett's method (*see* Power spectrum estimation)
Bessel filter 726–727
Bilinear transformation 712–717
Binary codes 405
Blackman–Tukey method (*see* Power spectrum estimation)
Burg algorithm (*see* Power spectrum estimation)
Butterworth filters 717–720

C

Canonic form 112
Capon method (*see* Power spectrum estimation)
Cauchy integral theorem 156
Causality, implications of 654–659
Causal signals 85
Causal systems 66–67, 83–85, 196–198
Cepstral coefficients 835
Cepstrum 261–262

- Characteristic polynomial 99
 Chebyshev filters 720–724
 Chirp signal 547
 Chirp-z transform algorithm 544–549
 Circular convolution 470–476
 Coding 20–35
 Comb filter 341–344
 Conjugate-gradient algorithm 906
 Constant-coefficient difference equations 93–108
 solution of 98–108
 Continuous-time signals 17
 exponentials 17
 sampling of 17–22, 384–394
 sampling theorem for 26–31, 384–394
 Convolution (linear) 69–80
 circular 470–476
 properties 80–83
 sum 69
 Correlation 116–128, 827
 autocorrelation 120,
 321–326, 826
 computation 123–125
 cross-correlation 118,
 321–323
 of periodic signals 123
 properties 120–123
 Coupled-form oscillator 347–349
 Cross-power density spectrum 829
- D**
- Dead band 625
 Decimation 754–760
 Deconvolution 262, 349–354,
 358–360
 homomorphic 262, 360–362
 Delta modulation 434
 Difference equations 89–108
 constant coefficient 98–108
 solution 98–108
 for recursive systems 93,
 108
 from one-sided z -transform 210–211
- homogeneous solution 98–101
 particular solution 101–103
 total solution 103–108
 Differentiator 691
 design of 691–693
 Digital resonator 335
 Digital sinusoidal oscillator 347–349
 Digital-to-analog (D/A) converter 5, 20, 36,
 408–440
 Dirichlet conditions, for Fourier series 228
 for Fourier transform 237
 Discrete Fourier transform (DFT) 454–461
 computation 480–488,
 511–536
 butterfly 522, 526, 528
 decimation-in-frequency FFT algorithm 524–526
 decimation-in-time FFT algorithm 519–526
 direct 480–488, 511–513
 divide-and-conquer method 513–536
 in-place computations 523
 radix-2 FFT algorithms 519–526
 radix-4 FFT algorithms 526–532
 shuffling of data 523
 split radix 532–536
 via linear filtering 537–549
 definition 456
 IDFT 456
 implementation of FFT algorithm 536–539
 properties 464–480
 circular convolution 470–476
 circular correlation 478
 circular frequency shift 478
 circular time shift 477
 complex conjugate 478
 linearity 465
 multiplication 470–476
- Parseval's theorem 479
 periodicity 465
 symmetry 465–470
 table 470
 time reversal 476
 relationship to Fourier series 461, 463
 relationship to Fourier transform 461
 relationship to z -transform 462
 use in frequency analysis 488–495
 use in linear filtering 480–488
- Discrete-time signals 9, 36–52
 antisymmetric (odd) 48
 correlation 116–128
 definition 9, 42
 exponential 44–45
 frequency analysis of 241–259
 nonperiodic 48
 periodic 11–17
 random 11
 representation of 36
 sinusoidal 14–16
 symmetric (even) 48
 unit ramp 44
 unit sample 43
 unit step 43
- Discrete-time systems 53–69
 causal 66–67, 83–85
 dynamic 60
 finite-duration impulse response 88–89
 finite memory 60, 88–89
 implementation of 563–601
 infinite-duration impulse response 88–89
 infinite memory 60, 88–89
 linear 63
 memoryless 57
 noncausal 66–67
 nonlinear 64
 nonrecursive 92–93
 recursive 90–93
 relaxed 56
 representation 36
 shift-invariant 60–61
 stability triangle 202

Discrete-time systems (*continued*)

- stable (BIBO) 67, 85–88
- static 60
- time-invariant 60–61
- unit sample (impulse)
 - response 73–80
- unstable 67, 85–88

Distortion, amplitude

- 312 delay 328
- harmonic 366
- phase 312

Down sampling 52

(*see also* Sampling rate conversion) 52

Dynamic range 33, 404, 605

E

Echo, far-end

888 near-end 888

Echo canceller 888

Echo suppressor 888

Eigenfunction 302

Eigenvalue 302

Elliptic filters 724–726

Energy

definition 45

density spectrum 238–241, 254–259

partial 382

signal 45–46

Energy density spectrum

238–241, 254–259

computation 961–966

Ensemble 825

averages 825–828

Envelope delay 328

Ergodic 830

correlation-ergodic 832–833

mean-ergodic 831–832

Estimate (properties)

asymptotic bias 969

asymptotic variance 968

bias 969

consistent 968

(*see also* Power spectrum estimation)

Estimate (properties), variance

968

F

Fast Fourier transform (FFT)

algorithms 511–537

application to 511

application to, correlation 537–539

application to, efficient computation of DFT 511–537

application to, linear filtering 537–539

implementation 536–537

mirror FFT 536

phase FFT 536

radix-2 algorithm 519–526 decimation-in-frequency 524–526

decimation-in-time 519–526

radix-4 algorithm 526–532

split-radix 532–536

Fast Kalman algorithms 945

Fibonacci sequence 210

difference equation 210–211

Filter 326

bandpass 327, 332–334

definition 312, 326–329

design of IIR filters 326–349, 701–727

all pass 345–347

by pole-zero placement 329–349

comb 341–344

notch 338–341

resonators (digital) 335–338

design of linear-phase FIR 660–701

transition coefficient for 1047–1052

distortion 312

distortionless 328

frequency-selective 326

highpass 327, 329–332

ideal 327–329

lowpass 327, 329–332

nonideal, passband ripple 659 stopband ripple

660

transition band 660

prediction error filter 575, 839

smoothing 36

structures 563–582

Wiener filter 862–873

Filter banks 790–796

critically sampled 794

quadrature mirror 798–799

uniform DFT 790–796

Filtering 480

of long data sequences 485–488

overlap-add method for 487–488

overlap-save method for 485–487

via DFT 481–488

Filter transformations 334–338, 727–734

analog domain 730–732

digital domain 732–734

lowpass-to-highpass 334–338

Final prediction error (FPE) criterion 996–997

Final value theorem 209

FIR filters 660

antisymmetric 660–670

design 660–701

comparison of methods 699–701

differentiators 691–693

equiripple (Chebyshev) approximation 678–699

frequency sampling

method 671–678

Hilbert transformers

693–699

window method 678–691

linear phase property 664–670

symmetric 660–670

FIR filter structures 565–582

cascade form 567–568

direct form 566–567

conversion to lattice form 581–582

frequency sampling form 569–574

lattice form 574–605, 859

conversion to direct form 579–581

FIR filter structures (*continued*)
 transposed form 586–588
 FIR systems 88, 92, 113
 Fixed-point representation 601–605
 Floating-point representation 605–608
 Flowgraphs 584–586
 Folding frequency 25, 389–391
 Forced response 95
 Forward predictor 578, 838–841
 Fourier series 18, 226–233, 241–245
 coefficients of 226–229, 241–245
 for continuous-time periodic signals 226–233
 for discrete-time periodic signals 241–245
 Fourier transform 234–238, 248–251
 convergence of 251–254
 inverse 236
 of continuous-time aperiodic signals 234–238
 of discrete-time aperiodic signals 248–251
 properties 283–291
 convolution 283–284
 correlation 284
 differentiation 289–291
 frequency shifting 286
 linearity 279–281
 modulation 286–287
 multiplication 288–289
 of signals with poles on unit circle 262–265
 Parseval's theorem 287
 relationship to z -transform 259–261
 symmetry 271–279
 table 290
 time-reversal 282
 time-shifting 281
 Frequency 11–16
 alias 15, 23
 content 26
 folding 25, 389–391
 fundamental range 17

highest 16
 negative 13
 normalized 22
 positive 13
 relative 22
 Frequency analysis 234–251
 continuous-time aperiodic signals 234–238
 continuous-time periodic signals 226–233
 discrete-time aperiodic signals 248–251
 discrete-time periodic signals 241–245
 dualities 267–268
 for LTI systems 300–326
 table of formulas for 269
 Frequency response 306
 computation 317–321
 geometric interpretation of 317–321
 magnitude of 304
 phase of 304
 relation to system function 314–317
 to exponentials 301–309
 to sinusoids 306–309
 Frequency transformations (*see* Filter transformations)
 Full duplex transmission 887
 Fundamental period 15

G

Gaussian random variable 1045–1046
 subroutine for 1045
 Gibbs phenomenon 254, 669
 Goertzel algorithm 542–544
 Granular noise 407
 Group (envelope) delay 328

H

Harmonic distortion 366, 446–447
 High-frequency signal 265
 Hilbert transform 658
 Hilbert transformer 693–699
 Homomorphic 360, 362
 deconvolution 360–362
 system 361
 Hybrid 888–891

I

IIR filters 701–727
 design from analog filters 701–727
 by approximation of derivatives 703–707
 by bilinear transformation 712–717, 727
 by impulse invariance 707–712
 by matched-z transformation 717
 least-squares design methods 746–747
 Padé approximation 747–748
 pole-zero placement 329–349
 Prony's (least squares) 746–747
 Shanks' (least squares) 748–749
 IIR filter structures 582–601
 cascade form 588–591
 direct form 582–584
 lattice-ladder 594–601, 860–862
 parallel form 591–594
 second-order modules 589
 transposed forms 584–588
 Impulse response 106–108
 Initial value theorems 168
 Innovations process 833–836
 Interpolation 28–31, 389, 751, 760–762
 function 28
 ideal 28–31, 389
 linear 36, 427–440
 Inverse filter 349–350
 Inverse Fourier transform 236, 251
 Inverse system 350–351
 Inverse z -transform 156–170, 179–193
 by contour integration 156–157, 180–182
 integral formula 156
 partial fraction expansion 184–193
 power series 182–184

- J**
- Joint-process estimate 938–940
- K**
- Kalman gain vector 918
- L**
- Lattice filter algorithms 927–954
 - a posteriori form 940
 - a priori form 940
 - error-feedback form 944
 - gradient form 950–951
 - joint process estimate 938–940
 - modified form 940–946
 - normalized form 949–950
 - properties of 951–954
 - square-foot form 949–950
- Lattice filters 574–579, 594–601, 858–862, 874
 - ARMA structure 860–862
 - AR structures 858–860
 - MA structure 838–841
- LDU decomposition 921–923
- Leakage 489, 964
- Learning curves 911
- Least squares 746–747
 - filter design 746–747
- Least-squares estimation 907
- Levinson–Durbin algorithm 846–850
 - generalized 850, 876
 - split Levinson 874
- Limit cycle oscillations 624–629
- Linear filtering 480–488
 - based on DFT 480–488
 - overlap-add method 487–488
 - overlap-save method 485–487
- Linear interpolation 427–440
- Linear prediction 578, 838–858
 - backward 841–845
 - forward 838–841
 - lattice filter for 845
 - normal equations for 846
 - properties of 855–858
- Linear prediction filter (*see* Linear prediction) 575
- Linear predictive coding 897
 - of speech 897–900
- LMS algorithm 905–907
 - excess mean-square error of 908
 - properties of 907
- Local loop 888
- Low-frequency signal 265
- Lowpass filter 327
- LTI systems 112–116
 - moving average 113
 - second order 112–113
 - structures 108–116
 - canonic form 112
 - direct form I 108–109
 - direct form II 109–113
 - nonrecursive 112–116
 - recursive 112–116
 - weighted moving average 112
- M**
- Maximal ripple filters 685
- Maximum entropy method 993
- Maximum-phase system 354–357
- Mean square estimation 866, 903–905
 - orthogonality principle 866–867
- Minimum description length (MDL) 997
- Minimum-phase system 354–357
- Minimum variance estimate 1012–1015
- Mixed-phase system 354–357
- Moving-average filter 304
- Moving-average (MA) process 837, 987
 - autocorrelation of 837
- Moving-average signal 115
- Multichannel signal 8
- Multidimensional signal 6–9
- N**
- Narrowband signal 266
- Natural response 95
- Natural signals 267–268
- Noise subspace 1017
- Noise whitening filter 835
- Normal equations 846
 - solution of 846–854
- Levinson–Durbin algorithm 846–850
- Schur algorithm 850–853
- Number representation 601–608
 - fixed-point 601–605
 - floating point 605–608
- Nyquist rate 28
- O**
- One's complement. 603
- One-sided z -transform 205–211
- Orthogonality principle 866–867
- Oscillators (sinusoidal generators) 347
- CORDIC algorithm for 349
 - coupled-form 347–349
 - digital 347
- Overflow 629–631
- Overlap-add method 487–488
- Overlap-save method 485–487
- Overload noise 407
- Oversampling A/D 433–440
- Oversampling D/A 439
- P**
- Paley–Wiener theorem 656
- Parseval's relations 238, 255, 287, 479
 - aperiodic (energy) signals 238, 255, 287
 - DFT 479
 - periodic (power) signals 230, 246
- Partial energy 358
- Partial fraction expansion (*see* Inverse z -transform)
- Periodogram 966–971
 - estimation of 966–971
 - mean value 968
 - variance 968

- Phase 12
 maximum 354–357
 minimum 354–357
 mixed 354–357
 response 306
- Pisarenko method 1015–1019
- Poles 170
 complex conjugate 189–192, 204–205
 distinct 173–198
 location 173–198
 multiple-order 187–188
- Polyphase filters 766–767
 for decimation 768
 for interpolation 773
- Power 46
 definition 46
 signal 47
- Power density spectrum 229–233
 definition 230
 estimation of (*see also* Power spectrum estimation) 230
 periodic signals 229–233, 245–248
 random signals 828–830
 rectangular pulse train 232–233
- Power spectrum estimation 963
 Capon (minimum variance) method 1012–1015
 direct method 963
 eigenanalysis algorithms 1019–1028
 ESPRIT 1022–1025
 MUSIC 1021
 order selection 1025–1026
 Pisarenko 1015–1019
 experimental results 1001–1009
 from finite data 966–973
 indirect method 963
 leakage 964
 nonparametric methods 973–985
 Bartlett 974–975
 Blackman–Tukey 977–981, 983–984
- computational requirements 984–985
 performance characteristics 981–984
 Welch 975–977, 982–983
- parametric (model-based) methods 985–1009
 ARMA model 988, 999–1001
 AR model 989
 AR model order selection 996–997
 Burg method 990–994
 least-squares 994–995
 MA model 990, 997–999
 maximum entropy method 993
 model parameters 988–990
 modified Burg 991
 relation to linear prediction 988–990
 sequential least squares 995–996
 Yule–Walker 990
 use of DFT 971–973
- Prediction coefficients 839
 Prediction-error filter 574, 839
 properties of 855–858
 Principal eigenvalues 1020
 Probability density function 824–825
 Probability distribution function 1041–1044
 Prony's method 746–747
 Pseudorandom sequences 145
 Barker sequence 145
 maximal-length shift register sequences 144–145
- Q**
- Quadrature mirror filters 788
 for perfect reconstruction 798–799
 for subband coding 788
- Quality 981–984
 of Bartlett estimate 982
 of Blackman–Tukey estimate 983–984
- of Welch estimate 982–983
- Quantization 20, 31–35, 403–406
 differential 433
 differential predictive 434
 dynamic range 33, 404, 605
 error 31, 34, 613–639
 in A/D conversion 403–406
 in filter coefficients 613–620
 level 32, 403
 resolution 32, 605
 rounding 32, 608–612
 step size 32, 605
 truncation 32, 608–612
- Quantization effects 31, 405, 549, 601
 fixed-point numbers 601–605
 one's complement 603
 sign-magnitude 603
 table of bipolar codes 406
 two's complement 603–605
 floating-point numbers 605–608
 in A/D conversion 34–35, 406–408
 in computation of DFT 549–555
 direct computation 549–551
 FFT algorithms 551–555
 in filter coefficients 613–620
 limit cycles 624–629
 dead band 625
 overflow 629–631
 zero-input 625
 scaling to prevent overflow 629–631
 statistical characterization 631–639
- Quantizer 403
 midrise 404
 midtread 404
 resolution 403–405
 uniform 404
- R**
- Random number generators 1041–1046

- Random processes 323–326,
824–833
averages 825–831
autocorrelation 826
autocovariance 826
expected value 825
for discrete-time signals
829–830
moments 825
power 826
correlation-ergodic 832–833
discrete-time 829–830
ergodic 830
jointly stationary 825
mean-ergodic 831–832
power density spectrum
828–829
response of linear systems
323–326
autocorrelation 323–326
expected value 323
power density spectrum
324–326
sample function 825
stationary 825
wide-sense 826
time-averages 830–832
- Random signals (*see* Random processes)
- Rational z -transforms 184–193
poles 170–173
zeros 170–173
- Recursive least squares
907–954
direct-form FIR algorithms
907–927
fast LS 923–925, 945–947
properties of 925–927
- lattice algorithms 928–954
a posteriori form 940
a priori form 940
error-feedback form
944
gradient 950–951
joint process estimate
938–940
modified form 940–946
normalized form 949–950
properties of 951–954
square-root form 921–923
- Recursive systems 112–116
- Reflection coefficients 575,
598, 845–846, 927
- Resonator (*see* Digital resonator)
- Reverse (reciprocal) polynomial 579, 842
backward system function
579, 842
- Round-off error 608–612,
631–639
- S**
- Sample-and-hold 402–403, 409
- Sample function 825
- Sampling 9, 19, 21, 384–394
aliasing effects 25–26
frequency 21
frequency domain 449–454
interval 21
Nyquist rate 28
of analog signals 21–31,
384–394
of discrete-time signals
751–806
of sinusoidal signals 22–26
period 21
periodic 21
rate 21
theorem 26–28
time-domain 22–26, 384–394
uniform 21
- Sampling-rate conversion
762–806
(*see also* Sampling-rate conversion)
applications of 784–806
for DFT filter banks
790–796
for interfacing 785
for lowpass filters 786
for phase shifters 784–785
for subband coding
787–788
for transmultiplexing
796–798
by arbitrary factor 781–782
by rational factor 762–766
decimation 751–760
filter design for 762–775
interpolation 751, 760–762
multistage 775–779
- of bandpass signals 779–781
polyphase filters for
766–767
- Sampling theorem 26–28,
384–394
- Schur algorithm 850–853
pipelined architecture for
853–854
split-Schur algorithm 874
- Shanks' method 748–749
- Sigma-delta modulation 436
- Signal flowgraphs 584–588
- Signals 2–4
analog 9
antisymmetric 48
aperiodic 48
bandpass 266
continuous-time 9
deterministic 11
digital 10
discrete-time 9, 36–52
electrocardiogram (ECG)
8
harmonically related 17
multichannel 8
multidimensional 9
natural 267
frequency ranges 267–268
periodic 13
random 11, 824–833
correlation-ergodic
832–833
ergodic 824
expected value of 825
mean-ergodic 832–833
moments of 826–830
statistically independent
827
strict-sense stationary
825
time-averages 830–833
unbiased 831
uncorrelated 827
wide-sense stationary
826
seismic 268
sinusoidal 12
speech 2–4
symmetric 48
- Signal subspace 1020

Sign magnitude representation 603
 Sinusoidal generators (*see* Oscillators)
 Spectrum 225–226
 analysis 226
 estimation of 226, 961–1028
 (*see also* Power spectrum estimation) 226
 Split-radix algorithms 532–536
 Spread-spectrum signal 892
 Stability of LTI systems 196–203
 of second-order systems 201–203
 Stability triangle 202
 Steady-state response 195–196, 311–312
 Structures 108–116
 direct form I 108–109
 direct form II 109–113
 Subband coding 787–789
 Superposition principle 63
 Superposition summation 73
 System 3, 53–56
 dynamic 60
 finite memory 60
 infinite memory 60
 inverse 350
 invertible 350
 relaxed 56
 System function 177–179, 314–317
 of all-pole system 179
 of all-zero system 177–179
 of LTI systems 177–179
 relation to frequency response 314–317
 System identification 350, 358–360
 System modeling 836
 System responses 94
 forced 95
 impulse 106–108
 natural (free) 95, 212
 of relaxed pole-zero systems 170–179
 of systems with initial conditions 211–214

steady-state 195–196
 transient 104, 195–196
 zero-input 95
 zero-state 94

T

Time averages 830–833
 Time-limited signals 266
 Toeplitz matrix 847, 864
 Transient response 104, 195–196, 309–311
 Transition band 660
 Transposed structures 584–588
 Truncation error 32, 608–612
 Trunk lines 888
 Two's complement representation 603

U

Uniform distribution 407, 549–551, 608–612
 Unit circle 261–265
 Unit sample (impulse) response 106–108
 Unit sample sequence 43

V

Variability 981
 Variance 549–551, 631–639

W

Welch method 975–977, 982–985
 Wideband signal 266
 Wiener filters 862–873, 904
 FIR structure 864–866
 for filtering 863
 for prediction 863
 for smoothing 863
 IIR structure 867–871
 noncausal 871–873
 Wiener–Hopf equation 864
 Wiener–Khintchine theorem 285
 Window functions 668
 Wold representation 836
 Wolfer sunspot numbers 10
 autocorrelation 125

Y

Yule–Walker equations 846
 modified 1000
 Yule–Walker method 990

Z

Zero-input linear 96
 Zero-input response 95
 Zero-order hold 34, 409
 Zero padding 456
 Zeros 170
 Zero-state linear 96
 Zoom frequency analysis 821–822
 z-transforms 147
 definition 147–148
 bilateral (two-sided) 147–148
 unilateral (one-sided) 205–211
 inverse 156–170, 179–193
 by contour integration 156–157, 180–182
 by partial fraction-expansion 184–193
 by power series 182–184
 properties 157–170
 convolution 164–166
 correlation 166–167
 differentiation 163–164
 initial value theorem 168
 linearity 157–159
 multiplication 167–168
 Parseval's relation 168
 scaling 161–162
 table of 169
 time reversal 162
 time shifting 159–161
 rational 170–179
 region of convergence (ROC) 147–156
 relationship of Fourier transform 259–261
 table of 169