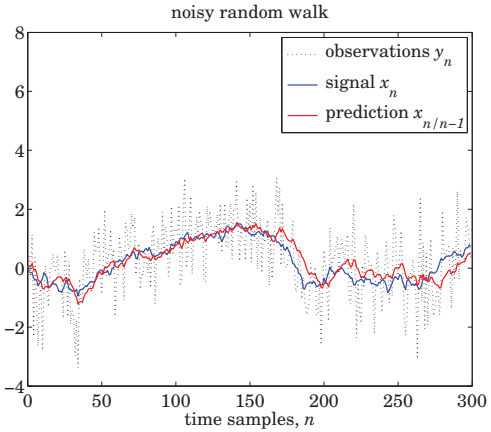


12

Linear Prediction

12.1 Pure Prediction and Signal Modeling

In Sec. 1.17, we discussed the connection between linear prediction and signal modeling. Here, we rederive the same results by considering the linear prediction problem as a special case of the Wiener filtering problem, given by Eq. (11.4.6). Our aim is to cast the results in a form that will suggest a practical way to solve the prediction problem and hence also the modeling problem. Consider a stationary signal y_n having a signal model

$$S_{yy}(z) = \sigma_\epsilon^2 B(z) B(z^{-1}) \qquad \epsilon_n \longrightarrow \boxed{B(z)} \longrightarrow y_n \tag{12.1.1}$$

as guaranteed by the spectral factorization theorem. Let $R_{yy}(k)$ denote the autocorrelation of y_n:

$$R_{yy}(k) = E[y_{n+k} y_n]$$

The linear prediction problem is to predict the current value y_n on the basis of all the past values $Y_{n-1} = \{y_i\,,\ -\infty < i \le n-1\}$. If we define the delayed signal $y_1(n) = y_{n-1}$, then the linear prediction problem is equivalent to the optimal Wiener filtering problem of estimating y_n from the related signal $y_1(n)$. The optimal estimation filter $H(z)$ is given by Eq. (11.4.6), where we must identify x_n and y_n with y_n and $y_1(n)$ of the present notation. Using the filtering equation $Y_1(z) = z^{-1} Y(z)$, we find that y_n and $y_1(n)$ have the same spectral factor $B(z)$

$$S_{y_1 y_1}(z) = (z^{-1})(z) S_{yy}(z) = S_{yy}(z) = \sigma_\epsilon^2 B(z) B(z^{-1})$$

and also that

$$S_{y y_1}(z) = S_{yy}(z) z = z \sigma_\epsilon^2 B(z) B(z^{-1})$$

Inserting these into Eq. (11.4.6), we find for the optimal filter $H(z)$

$$H(z) = \frac{1}{\sigma_\epsilon^2 B(z)} \left[\frac{S_{y y_1}(z)}{B(z^{-1})} \right]_+ = \frac{1}{\sigma_\epsilon^2 B(z)} \left[\frac{z \sigma_\epsilon^2 B(z) B(z^{-1})}{B(z^{-1})} \right]_+, \quad \text{or,}$$

$$H(z) = \frac{1}{B(z)} [z B(z)]_+ \tag{12.1.2}$$

The causal instruction can be removed as follows: Noting that $B(z)$ is a causal and stable filter, we may expand it in the power series

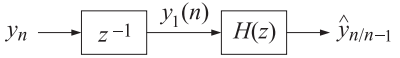
$$B(z) = 1 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + \cdots$$

The causal part of $zB(z)$ is then

$$\begin{aligned}[zB(z)]_+ &= [z + b_1 + b_2 z^{-1} + b_3 z^{-2} + \cdots]_+ = b_1 + b_2 z^{-1} + b_3 z^{-2} + \cdots \\ &= z(b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + \cdots) = z(B(z) - 1)\end{aligned}$$

The prediction filter $H(z)$ then becomes

$$H(z) = \frac{1}{B(z)}\, z(B(z) - 1) = z\left[1 - \frac{1}{B(z)}\right] \tag{12.1.3}$$



The input to this filter is $y_1(n)$ and the output is the prediction $\hat{y}_{n/n-1}$.

Example 12.1.1: Suppose that y_n is generated by driving the all-pole filter

$$y_n = 0.9y_{n-1} - 0.2y_{n-2} + \epsilon_n$$

by zero-mean white noise ϵ_n. Find the best predictor $\hat{y}_{n/n-1}$. The signal model in this case is $B(z) = 1/(1 - 0.9z^{-1} + 0.2z^{-2})$ and Eq. (12.1.3) gives

$$z^{-1}H(z) = 1 - \frac{1}{B(z)} = 1 - (1 - 0.9z^{-1} + 0.2z^{-2}) = 0.9z^{-1} - 0.2z^{-2}$$

The I/O equation for the prediction filter is obtained by

$$\hat{Y}(z) = H(z)Y_1(z) = z^{-1}H(z)Y(z) = [0.9z^{-1} - 0.2z^{-2}]Y(z)$$

and in the time domain

$$\hat{y}_{n/n-1} = 0.9y_{n-1} - 0.2y_{n-2}$$

Example 12.1.2: Suppose that

$$S_{yy}(z) = \frac{(1 - 0.25z^{-2})(1 - 0.25z^2)}{(1 - 0.8z^{-1})(1 - 0.8z)}$$

Determine the best predictor $\hat{y}_{n/n-1}$. Here, the minimum phase factor is

$$B(z) = \frac{1 - 0.25z^{-2}}{1 - 0.8z^{-1}}$$

and therefore the prediction filter is

$$z^{-1}H(z) = 1 - \frac{1}{B(z)} = 1 - \frac{1 - 0.8z^{-1}}{1 - 0.25z^{-2}} = \frac{0.8z^{-1} - 0.25z^{-2}}{1 - .25z^{-2}}$$

The I/O equation of this filter is conveniently given recursively by the difference equation

$$\hat{y}_{n/n-1} = 0.25\hat{y}_{n-2/n-3} + 0.8y_{n-1} - 0.25y_{n-2}$$ □

The prediction error

$$e_{n/n-1} = y_n - \hat{y}_{n/n-1}$$

is identical to the whitening sequence ϵ_n driving the signal model (12.1.1) of y_n, indeed,

$$\begin{aligned}E(z) &= Y(z) - \hat{Y}(z) = Y(z) - H(z)Y_1(z) = Y(z) - H(z)z^{-1}Y(z) \\ &= [1 - z^{-1}H(z)]Y(z) = \frac{1}{B(z)}Y(z) = \epsilon(z)\end{aligned}$$

Thus, in accordance with the results of Sec. 1.13 and Sec. 1.17

$$e_{n/n-1} = y_n - \hat{y}_{n/n-1} = \epsilon_n \tag{12.1.4}$$

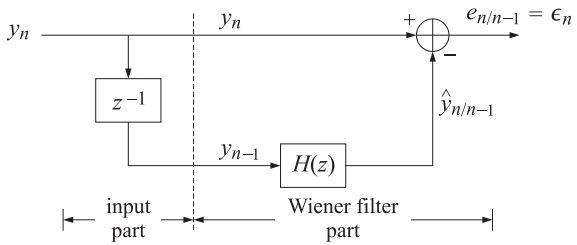


Fig. 12.1.1 Linear Predictor.

An overall realization of the linear predictor is shown in Fig. 12.1.1. The indicated dividing line separates the linear predictor into the Wiener filtering part and the input part which provides the proper input signals to the Wiener part. The transfer function from y_n to $e_{n/n-1}$ is the *whitening inverse filter*

$$A(z) = \frac{1}{B(z)} = 1 - z^{-1}H(z)$$

which is *stable and causal* by the minimum-phase property of the spectral factorization (12.1.1). In the z-domain we have

$$E(z) = \epsilon(z) = A(z)Y(z)$$

and in the time domain

$$e_{n/n-1} = \epsilon_n = \sum_{m=0}^{\infty} a_m y_{n-m} = y_n + a_1 y_{n-1} + a_2 y_{n-2} + \cdots$$

The predicted estimate $\hat{y}_{n/n-1} = y_n - e_{n/n-1}$ is

$$\hat{y}_{n/n-1} = -[a_1 y_{n-1} + a_2 y_{n-2} + \cdots]$$

These results are identical to Eqs. (1.17.2) and (1.17.3). The relationship noted above between linear prediction and signal modeling can also be understood in terms of the

gapped-function approach of Sec. 11.7. Rewriting Eq. (12.1.1) in terms of the prediction-error filter $A(z)$ we have

$$S_{yy}(z) = \frac{\sigma_\epsilon^2}{A(z)A(z^{-1})} \tag{12.1.5}$$

from which we obtain

$$A(z)S_{yy}(z) = \frac{\sigma_\epsilon^2}{A(z^{-1})} \tag{12.1.6}$$

Since we have the filtering equation $\epsilon(z) = A(z)Y(z)$, it follows that

$$S_{\epsilon y}(z) = A(z)S_{yy}(z)$$

and in the time domain

$$R_{ey}(k) = E[\epsilon_n y_{n-k}] = \sum_{i=0}^{\infty} a_i R_{yy}(k-i) \tag{12.1.7}$$

which is recognized as the gapped function (11.7.1). By construction, ϵ_n is the orthogonal complement of y_n with respect to the entire past subspace $Y_{n-1} = \{y_{n-k},\ k = 1, 2, \dots\}$, therefore, ϵ_n will be orthogonal to each y_{n-k} for $k = 1, 2, \dots$. These are precisely the gap conditions. Because the prediction is based on the entire past, the gapped function develops an infinite right-hand side gap. Thus, Eq. (12.1.7) implies

$$R_{ey}(k) = E[\epsilon_n y_{n-k}] = \sum_{i=0}^{\infty} a_i R_{yy}(k-i) = 0\,, \quad \text{for all } k = 1, 2, \dots \tag{12.1.8}$$

The same result, of course, also follows from the z-domain equation (12.1.6). Both sides of the equation are stable, but since $A(z)$ is minimum-phase, $A(z^{-1})$ will be maximum phase, and therefore it will have a stable but anticausal inverse $1/A(z^{-1})$. Thus, the right-hand side of Eq. (12.1.6) has no strictly causal part. Equating to zero all the coefficients of positive powers of z^{-1} results in Eq. (12.1.8).

The value of the gapped function at $k = 0$ is equal to σ_ϵ^2. Indeed, using the gap conditions (12.1.8) we find

$$\begin{aligned}\sigma_\epsilon^2 &= E[\epsilon_n^2] = E[\epsilon_n(y_n + a_1 y_{n-1} + a_2 y_{n-2} + \cdots)] \\ &= R_{\epsilon y}(0) + a_1 R_{\epsilon y}(1) + a_2 R_{\epsilon y}(2) + \cdots = R_{\epsilon y}(0) = E[\epsilon_n y_n]\end{aligned}$$

Using Eq. (12.1.7) with $k = 0$ and the symmetry property $R_{yy}(i) = R_{yy}(-i)$, we find

$$\sigma_\epsilon^2 = E[\epsilon_n^2] = E[\epsilon_n y_n] = R_{yy}(0) + a_1 R_{yy}(1) + a_2 R_{yy}(2) + \cdots \tag{12.1.9}$$

Equations (12.1.8) and (12.1.9) may be combined into one:

$$\sum_{i=0}^{\infty} a_i R_{yy}(k-i) = \sigma_\epsilon^2 \delta(k)\,, \quad \text{for all } k \geq 0 \tag{12.1.10}$$

which can be cast in the matrix form:

$$\begin{bmatrix} R_{yy}(0) & R_{yy}(1) & R_{yy}(2) & R_{yy}(3) & \cdots \\ R_{yy}(1) & R_{yy}(0) & R_{yy}(1) & R_{yy}(2) & \cdots \\ R_{yy}(2) & R_{yy}(1) & R_{yy}(0) & R_{yy}(1) & \cdots \\ R_{yy}(3) & R_{yy}(2) & R_{yy}(1) & R_{yy}(0) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} \sigma_\epsilon^2 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix} \tag{12.1.11}$$

These equations are known as the *normal equations* of linear prediction [915–928]. They provide the solution to both the signal modeling and the linear prediction problems. They determine the model parameters $\{a_1, a_2, \dots; \sigma_\epsilon^2\}$ of the signal y_n directly in terms of the experimentally accessible quantities $R_{yy}(k)$. To render them computationally manageable, the infinite matrix equation (12.1.11) must be reduced to a finite one, and furthermore, the quantities $R_{yy}(k)$ must be estimated from actual data samples of y_n. We discuss these matters next.

12.2 Autoregressive Models

In general, the number of prediction coefficients $\{a_1, a_2, \dots\}$ is infinite since the predictor is based on the infinite past. However, there is an important exception to this; namely, when the process y_n is autoregressive. In this case, the signal model $B(z)$ is an all-pole filter of the type

$$B(z) = \frac{1}{A(z)} = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_p z^{-p}} \tag{12.2.1}$$

which implies that the prediction filter is a polynomial

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_p z^{-p} \tag{12.2.2}$$

The signal generator for y_n is the following difference equation, driven by the uncorrelated sequence ϵ_n:

$$y_n + a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_p y_{n-p} = \epsilon_n \tag{12.2.3}$$

and the optimal prediction of y_n is simply given by:

$$\hat{y}_{n/n-1} = -[a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_p y_{n-p}] \tag{12.2.4}$$

In this case, the best prediction of y_n depends only on the past p samples $\{y_{n-1}, y_{n-2}, \dots, y_{n-p}\}$. The infinite set of equations (12.1.10) or (12.1.11) are still satisfied even though only the first $p+1$ coefficients $\{1, a_1, a_2, \dots, a_p\}$ are nonzero.

The $(p+1)\times(p+1)$ portion of Eq. (12.1.11) is sufficient to determine the $(p+1)$ model parameters $\{a_1, a_2, \dots, a_p; \sigma_\epsilon^2\}$:

$$\begin{bmatrix} R_{yy}(0) & R_{yy}(1) & R_{yy}(2) & \cdots & R_{yy}(p) \\ R_{yy}(1) & R_{yy}(0) & R_{yy}(1) & \cdots & R_{yy}(p-1) \\ R_{yy}(2) & R_{yy}(1) & R_{yy}(0) & \cdots & R_{yy}(p-2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R_{yy}(p) & R_{yy}(p-1) & R_{yy}(p-2) & \cdots & R_{yy}(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma_\epsilon^2 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{12.2.5}$$

Such equations may be solved efficiently by Levinson's algorithm, which requires $O(p^2)$ operations and $O(p)$ storage locations to obtain the a_is instead of $O(p^3)$ and $O(p^2)$, respectively, that would be required if the inverse of the autocorrelation matrix

R_{yy} were to be computed. The finite set of model parameters $\{a_1, a_2, \ldots, a_p; \sigma_\epsilon^2\}$ determines the signal model of y_n completely. Setting $z = e^{j\omega}$ into Eq. (12.1.5)) we find a simple *parametric representation* of the power spectrum of the AR signal y_n

$$S_{yy}(\omega) = \frac{\sigma_\epsilon^2}{|A(\omega)|^2} = \frac{\sigma_\epsilon^2}{|1 + a_1 e^{-j\omega} + a_2 e^{-2j\omega} + \cdots + a_p e^{-j\omega p}|^2} \tag{12.2.6}$$

In practice, the normal equations (12.2.5) provide a means of determining approximate estimates for the model parameters $\{a_1, a_2, \ldots, a_p; \sigma_\epsilon^2\}$. Typically, a block of length N of recorded data is available

$$\boxed{y_0, y_1, y_2, \ldots, y_{N-1}}$$

There are many different methods of extracting reasonable estimates of the model parameters using this block of data. We mention: (1) the autocorrelation or Yule-Walker method, (2) the covariance method, and (3) Burg's method. There are also some variations of these methods. The first method, the Yule-Walker method, is perhaps the most obvious and straightforward one. In the normal equations (12.2.5), one simply replaces the ensemble autocorrelations $R_{yy}(k)$ by the corresponding sample autocorrelations computed from the given block of data; that is,

$$\hat{R}_{yy}(k) = \frac{1}{N} \sum_{n=0}^{N-1-k} y_{n+k} y_n, \quad \text{for } 0 \le k \le p \tag{12.2.7}$$

where only the first $p+1$ lags are needed in Eq. (12.2.5). We must have, of course, $p \le N-1$. As discussed in Sec. 1.13, the resulting estimates of the model parameters $\{\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_p; \hat{\sigma}_\epsilon^2\}$ may be used now in a number of ways; examples include obtaining an *estimate* of the power spectrum of the sequence y_n

$$\hat{S}_{yy}(\omega) = \frac{\hat{\sigma}_\epsilon^2}{|\hat{A}(\omega)|^2} = \frac{\hat{\sigma}_\epsilon^2}{|1 + \hat{a}_1 e^{-j\omega} + \hat{a}_2 e^{-2j\omega} + \cdots + \hat{a}_p e^{-j\omega p}|^2}$$

or, representing the block of N samples y_n in terms of a few (i.e., $p+1$) filter parameters. To synthesize the original samples one would generate white noise ϵ_n of variance $\hat{\sigma}_\epsilon^2$ and send it through the generator filter whose coefficients are the estimated values; that is, the filter

$$\hat{B}(z) = \frac{1}{\hat{A}(z)} = \frac{1}{1 + \hat{a}_1 z^{-1} + \hat{a}_2 z^{-2} + \cdots + \hat{a}_p z^{-p}}$$

The Yule-Walker analysis procedure, also referred to as the autocorrelation method of linear prediction [917], is summarized in Fig. 12.2.1.

12.3 Linear Prediction and the Levinson Recursion

In the last section, we saw that if the signal being predicted is autoregressive of order p, then the optimal linear predictor collapses to a pth order predictor. The infinite dimensional Wiener filtering problem collapses to a finite dimensional one. A geometrical way to understand this property is to say that the projection of y_n on the subspace

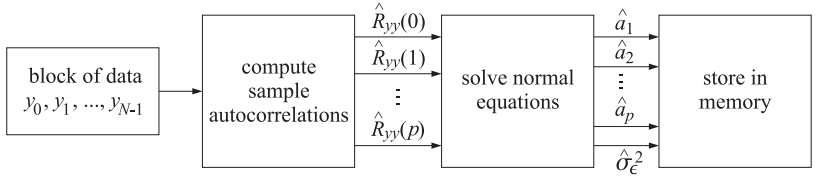


Fig. 12.2.1 Yule-Walker Analysis Algorithm.

spanned by the entire past $\{y_{n-i},\ 1 \le i < \infty\}$ is the same as the projection of y_n onto the subspace spanned only by the past p samples; namely, $\{y_{n-i},\ 1 \le i \le p\}$. This is a consequence of the difference equation (12.2.3) generating y_n.

If the process y_n is not autoregressive, these two projections will be different. For any given p, the projection of y_n onto the past p samples will still provide the best linear prediction of y_n that can be made on the basis of these p samples. As p increases, more and more past information is taken into account, and we expect the prediction of y_n to become better and better in the sense of yielding a smaller mean-square prediction error.

In this section, we consider the finite-past prediction problem and discuss its efficient solution via the Levinson recursion [915–928]. For sufficiently large values of p, it may be considered to be an adequate approximation to the full prediction problem and hence also to the modeling problem.

Consider a stationary time series y_n with autocorrelation function $R(k) = E[y_{n+k} y_n]$. For any given p, we seek the best linear predictor of the form

$$\hat{y}_n = -[a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_p y_{n-p}] \tag{12.3.1}$$

The p prediction coefficients $\{a_1, a_2, \ldots, a_p\}$ are chosen to minimize the mean-square prediction error

$$\mathcal{E} = E[e_n^2] \tag{12.3.2}$$

where e_n is the prediction error

$$e_n = y_n - \hat{y}_n = y_n + a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_p y_{n-p} \tag{12.3.3}$$

Differentiating Eq. (12.3.2) with respect to each coefficient a_i, $i = 1, 2, \ldots, p$, yields the orthogonality equations

$$E[e_n y_{n-i}] = 0, \quad \text{for } i = 1, 2, \ldots, p \tag{12.3.4}$$

which express the fact that the optimal predictor $\hat{y}_n$ is the projection onto the span of the past p samples; that is, $\{y_{n-i},\ i = 1, 2, \ldots, p\}$. Inserting the expression (12.3.3) for e_n into Eq. (12.3.4), we obtain p linear equations for the coefficients

$$\sum_{j=0}^{p} a_j E[y_{n-j} y_{n-i}] = \sum_{j=0}^{p} R(i-j)\, a_j = 0, \quad \text{for } i = 1, 2, \ldots, p \tag{12.3.5}$$

Using the conditions (12.3.4) we also find for the *minimized* value of

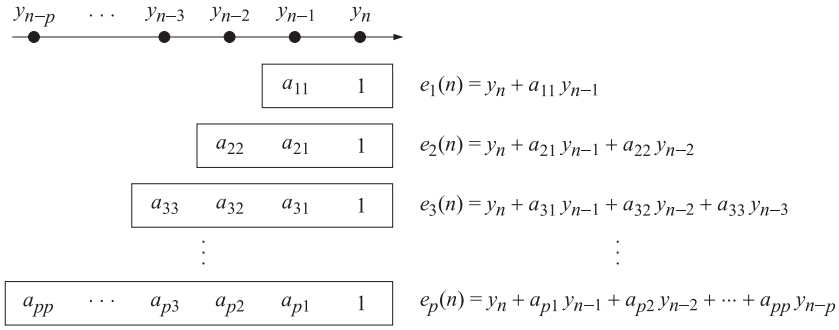
$$\sigma_e^2 = \mathcal{E} = E[e_n^2] = E[e_n y_n] = \sum_{j=0}^{p} R(j)\, a_j \tag{12.3.6}$$

Equations (12.3.5) and (12.3.6) can be combined into the $(p+1)\times(p+1)$ matrix equation

$$\begin{bmatrix} R(0) & R(1) & R(2) & \cdots & R(p) \\ R(1) & R(0) & R(1) & \cdots & R(p-1) \\ R(2) & R(1) & R(0) & \cdots & R(p-2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R(p) & R(p-1) & R(p-2) & \cdots & R(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma_e^2 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{12.3.7}$$

which is identical to Eq. (12.2.5) for the autoregressive case. It is also the *truncated* version of the infinite matrix equation (12.1.11) for the full prediction problem.

Instead of solving the normal equations (12.3.7) directly, we would like to embed this problem into a whole class of similar problems; namely, those of determining the best linear predictors of orders $p = 1$, $p = 2$, $p = 3, \ldots$, and so on. This approach will lead to Levinson's algorithm and to the so-called lattice realizations of linear prediction filters. Pictorially this class of problems is illustrated below



where $[1, a_{11}]$, $[1, a_{21}, a_{22}]$, $[1, a_{31}, a_{32}, a_{33}], \ldots$, represent the best predictors of orders $p = 1, 2, 3, \ldots$, respectively. It was necessary to attach an extra index indicating the order of the predictor. Levinson's algorithm is an iterative procedure that constructs the next predictor from the previous one. In the process, all optimal predictors of lower orders are also computed. Consider the predictors of orders p and $p+1$, below

y_{n-p-1}	y_{n-p}	$\cdots$	y_{n-2}	y_{n-1}	y_n
	a_{pp}	$\cdots$	a_{p2}	a_{p1}	1
$a_{p+1,p+1}$	$a_{p+1,p}$	$\cdots$	$a_{p+1,2}$	$a_{p+1,1}$	1

$$e_p(n) = y_n + a_{p1} y_{n-1} + a_{p2} y_{n-2} + \cdots + a_{pp} y_{n-p}$$

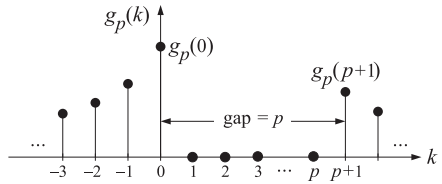
$$e_{p+1}(n) = y_n + a_{p+1,1} y_{n-1} + a_{p+1,2} y_{n-2} + \cdots + a_{p+1,p+1} y_{n-p-1}$$

Our objective is to construct the latter in terms of the former. We will use the approach of Robinson and Treitel, based on gapped functions [925]. Suppose that the best predictor of order p, $[1, a_{p1}, a_{p2}, \ldots, a_{pp}]$, has already been constructed. The corresponding gapped function is

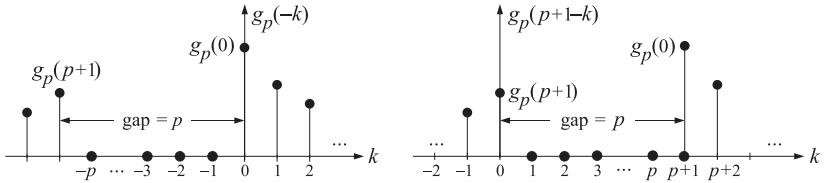
$$g_p(k) = E[e_p(n) y_{n-k}] = E\left[\left(\sum_{i=0}^{p} a_{pi} y_{n-i}\right) y_{n-k}\right] = \sum_{i=0}^{p} a_{pi} R(k-i) \tag{12.3.8}$$

It has a gap of length p as shown , that is,

$$g_p(k) = 0\,, \quad \text{for } 1 \le k \le p$$



These gap conditions are the same as the orthogonality equations (12.3.4). Using $g_p(k)$ we now construct a new gapped function $g_{p+1}(k)$ of gap $p+1$. To do this, first we reflect $g_p(k)$ about the origin; that is, $g_p(k) \to g_p(-k)$. The reflected function has a gap of length p but at negatives times. A delay of $(p+1)$ time units will realign this gap with the original gap. This follows because if $1 \le k \le p$, then $1 \le p+1-k \le p$. The reflected-delayed function will be $g_p(p+1-k)$. These operations are shown in the following figure



Since both $g_p(k)$ and $g_p(p+1-k)$ have exactly the same gap, it follows that so will any linear combination of them. Therefore,

$$g_{p+1}(k) = g_p(k) - \gamma_{p+1} g_p(p+1-k) \tag{12.3.9}$$

will have a gap of length at least p. We now select the parameter γ_{p+1} so that $g_{p+1}(k)$ acquires an *extra* gap point; its gap is now of length $p+1$. The extra gap condition is

$$g_{p+1}(p+1) = g_p(p+1) - \gamma_{p+1} g_p(0) = 0$$

which may be solved for

$$\gamma_{p+1} = \frac{g_p(p+1)}{g_p(0)}$$

Evaluating Eq. (12.3.8) at $k = p+1$, and using the fact that the value of the gapped function at $k = 0$ is the minimized value of the mean-squared error, that is,

$$E_p = E[e_p^2(n)] = E[e_p(n) y_n] = g_p(0) \tag{12.3.10}$$

we finally find

$$\gamma_{p+1} = \frac{\Delta_p}{E_p} \tag{12.3.11}$$

where we set $\Delta_p = g_p(p+1)$

$$\Delta_p = \sum_{i=0}^{p} a_{pi} R(p+1-i) = [R(p+1), R(p), R(p-1), \ldots, R(1)] \begin{bmatrix} 1 \\ a_{p1} \\ a_{p2} \\ \vdots \\ a_{pp} \end{bmatrix} \tag{12.3.12}$$

The coefficients γ_{p+1} are called *reflection, PARCOR*, or *Schur coefficients*. This terminology will become clear later. Evaluating Eq. (12.3.9) at $k = 0$ and using $g_p(p+1) = \gamma_{p+1} g_p(0)$, we also find a recursion for the quantity $E_{p+1} = g_{p+1}(0)$

$$E_{p+1} = g_{p+1}(0) = g_p(0) - \gamma_{p+1} g_p(p+1) = g_p(0) - \gamma_{p+1} \cdot \gamma_{p+1} g_p(0)\,, \quad \text{or,}$$

$$E_{p+1} = (1 - \gamma_{p+1}^2) E_p \tag{12.3.13}$$

This represents the minimum value of the mean-square prediction error $E[e_{p+1}^2(n)]$ for the predictor of order $p+1$. Since both E_p and E_{p+1} are nonnegative, it follows that the factor $(1 - \gamma_{p+1}^2)$ will be nonnegative and less than one. It represents the improvement in the prediction obtained by using a predictor of order $p+1$ instead of a predictor of order p. It also follows that γ_{p+1} has magnitude less than one, $|\gamma_{p+1}| \leq 1$.

To find the new prediction coefficients, $a_{p+1,i}$, we use the fact that the gapped functions are equal to the convolution of the corresponding prediction-error filters with the autocorrelation function of y_n:

$$g_p(k) = \sum_{i=0}^{p} a_{pi} R(k-i) \qquad \Rightarrow \quad G_p(z) = A_p(z) S_{yy}(z)$$

$$g_{p+1}(k) = \sum_{i=0}^{p+1} a_{p+1,i} R(k-i) \quad \Rightarrow \quad G_{p+1}(z) = A_{p+1}(z) S_{yy}(z)$$

where $S_{yy}(z)$ represents the power spectral density of y_n. Taking z-transforms of both sides of Eq. (12.3.9), we find

$$G_{p+1}(z) = G_p(z) - \gamma_{p+1} z^{-(p+1)} G_p(z^{-1})\,, \quad \text{or,}$$

$$A_{p+1}(z) S_{yy}(z) = A_p(z) S_{yy}(z) - \gamma_{p+1} z^{-(p+1)} A_p(z^{-1}) S_{yy}(z^{-1})$$

where we used the fact that the reflected gapped function $g_p(-k)$ has z-transform $G_p(z^{-1})$, and therefore the delayed (by $p+1$) as well as reflected gapped function $g_p(p+1-k)$ has z-transform $z^{-(p+1)} G_p(z^{-1})$. Since $S_{yy}(z) = S_{yy}(z^{-1})$ because of the symmetry relations $R(k) = R(-k)$, it follows that $S_{yy}(z)$ is a common factor in all the terms. Therefore, we obtain a relationship between the new best prediction-error filter $A_{p+1}(z)$ and the old one $A_p(z)$

$$A_{p+1}(z) = A_p(z) - \gamma_{p+1} z^{-(p+1)} A_p(z^{-1}) \qquad \text{(Levinson recursion)} \tag{12.3.14}$$

Taking inverse z-transforms, we find

$$\begin{bmatrix} 1 \\ a_{p+1,1} \\ a_{p+1,2} \\ \vdots \\ a_{p+1,p} \\ a_{p+1,p+1} \end{bmatrix} = \begin{bmatrix} 1 \\ a_{p1} \\ a_{p2} \\ \vdots \\ a_{pp} \\ 0 \end{bmatrix} - \gamma_{p+1} \begin{bmatrix} 0 \\ a_{pp} \\ a_{p,p-1} \\ \vdots \\ a_{p1} \\ 1 \end{bmatrix} \tag{12.3.15}$$

which can also be written as

$$a_{p+1,i} = a_{pi} - \gamma_{p+1} a_{p,p+1-i}\,, \quad \text{for } 1 \leq i \leq p$$

$$a_{p+1,p+1} = -\gamma_{p+1}$$

Introducing the *reverse* polynomial $A_p^R(z) = z^{-p} A_p(z^{-1})$, we may write Eq. (12.3.14) as

$$A_{p+1}(z) = A_p(z) - \gamma_{p+1} z^{-1} A_p^R(z) \tag{12.3.16}$$

Taking the reverse of both sides, we find

$$A_{p+1}(z^{-1}) = A_p(z^{-1}) - \gamma_{p+1} z^{p+1} A_p(z)$$

$$A_{p+1}^R(z) = z^{-(p+1)} A_{p+1}(z^{-1}) = z^{-(p+1)} A_p(z^{-1}) - \gamma_{p+1} A_p(z)\,, \quad \text{or,}$$

$$A_{p+1}^R(z) = z^{-1} A_p^R(z) - \gamma_{p+1} A_p(z) \tag{12.3.17}$$

Equation (12.3.17) is, in a sense, redundant, but it will prove convenient to think of the Levinson recursion as a recursion on both the forward, $A_p(z)$, and the reverse, $A_p^R(z)$, polynomials. Equations (12.3.16) and Eq. (12.3.17) may be combined into a 2×2 matrix recursion equation, referred to as the *forward Levinson recursion*:

$$\begin{bmatrix} A_{p+1}(z) \\ A_{p+1}^R(z) \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_{p+1} z^{-1} \\ -\gamma_{p+1} & z^{-1} \end{bmatrix} \begin{bmatrix} A_p(z) \\ A_p^R(z) \end{bmatrix} \quad \text{(forward recursion)} \tag{12.3.18}$$

The recursion is initialized at $p = 0$ by setting

$$A_0(z) = A_0^R(z) = 1 \quad \text{and} \quad E_0 = R(0) = E[y_n^2] \tag{12.3.19}$$

which corresponds to no prediction at all. We summarize the computational steps of the Levinson algorithm:

1. Initialize at $p = 0$ using Eq. (12.3.19).
2. At stage p, the filter $A_p(z)$ and error E_p are available.
3. Using Eq. (12.3.11), compute γ_{p+1}.
4. Using Eq. (12.3.14) or Eq. (12.3.18), determine the new polynomial $A_{p+1}(z)$.
5. Using Eq. (12.3.13), update the mean-square prediction error to E_{p+1}.
6. Go to stage $p+1$.

The iteration may be continued until the final desired order is reached. The dependence on the autocorrelation $R(k)$ of the signal y_n is entered through Eq. (12.3.11) and $E_0 = R(0)$. To reach stage p, only the $p+1$ autocorrelation lags $\{R(0), R(1), \ldots, R(p)\}$ are required. At the pth stage, the iteration already has provided all the prediction filters of lower order, and all the previous reflection coefficients. Thus, an alternative parametrization of the pth order predictor is in terms of the sequence of reflection coefficients $\{\gamma_1, \gamma_2, \ldots, \gamma_p\}$ and the prediction error E_p

$$\{E_p, a_{p1}, a_{p2}, \ldots, a_{pp}\} \quad \Leftrightarrow \quad \{E_p, \gamma_1, \gamma_2, \ldots, \gamma_p\}$$

One may pass from one parameter set to another. And both sets are equivalent to the autocorrelation set $\{R(0), R(1), \ldots, R(p)\}$. The alternative parametrization of the autocorrelation function $R(k)$ of a stationary random sequence in terms of the equivalent set of reflection coefficients is a general result [929,930], and has also been extended to the multichannel case [931].

If the process y_n is autoregressive of order p, then as soon as the Levinson recursion reaches this order, it will provide the autoregressive coefficients $\{a_1, a_2, \ldots, a_p\}$ which are also the best prediction coefficients for the full (i.e., based on the infinite past) prediction problem. Further continuation of the Levinson recursion will produce nothing new—all prediction coefficients (and all reflection coefficients) of order higher than p will be zero, so that $A_q(z) = A_p(z)$ for all $q > p$.

The four functions **lev**, **frwlev**, **bkwlev**, and **rlev** allow the passage from one parameter set to another. The function **lev** is an implementation of the computational sequence outlined above. The input to the function is the final desired order of the predictor, say M, and the vector of autocorrelation lags $\{R(0), R(1), ..., R(M)\}$. Its output is the lower-triangular matrix L whose rows are the *reverse* of all the lower order prediction-error filters. For example, for $M = 4$ the matrix L would be

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ a_{11} & 1 & 0 & 0 & 0 \\ a_{22} & a_{21} & 1 & 0 & 0 \\ a_{33} & a_{32} & a_{31} & 1 & 0 \\ a_{44} & a_{43} & a_{42} & a_{41} & 1 \end{bmatrix} \tag{12.3.20}$$

The first column of L contains the negatives of all the reflection coefficients. This follows from the Levinson recursion (12.3.14) which implies that the negative of the highest coefficient of the pth prediction-error filter is the pth reflection coefficient; namely,

$$\gamma_p = -a_{pp}\,, \quad p = 1, 2, \ldots, M \tag{12.3.21}$$

This choice for L is justified below and in Sec. 12.9. The function **lev** also produces the vector of mean-square prediction errors $\{E_0, E_1, \ldots, E_M\}$ according to the recursion (12.3.13).

The function **frwlev** is an implementation of the forward Levinson recursion (12.3.18) or (12.3.15). Its input is the set of reflection coefficients $\{\gamma_1, \gamma_2, \ldots, \gamma_M\}$ and its output is the set of all prediction-error filters up to order M, that is, $A_p(z)$, $p = 1, 2, \ldots, M$. Again, this output is arranged into the matrix L.

The function **bkwlev** is the inverse operation to **frwlev**. Its input is the vector of prediction-error filter coefficients $[1, a_{M1}, a_{M2}, \ldots, a_{MM}]$ of the final order M, and its output is the matrix L containing all the lower order prediction-error filters. The set of reflection coefficients are extracted from the first column of L. This function is based on the inverse of the matrix equation (12.3.18). Shifting p down by one unit, we write Eq. (12.3.18) as

$$\begin{bmatrix} A_p(z) \\ A^R_p(z) \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_p z^{-1} \\ -\gamma_p & z^{-1} \end{bmatrix} \begin{bmatrix} A_{p-1}(z) \\ A^R_{p-1}(z) \end{bmatrix} \tag{12.3.22}$$

Its inverse is

$$\begin{bmatrix} A_{p-1}(z) \\ A^R_{p-1}(z) \end{bmatrix} = \frac{1}{1-\gamma_p^2} \begin{bmatrix} 1 & \gamma_p \\ \gamma_p z & z \end{bmatrix} \begin{bmatrix} A_p(z) \\ A^R_p(z) \end{bmatrix} \quad \text{(backward recursion)} \tag{12.3.23}$$

At each stage p, start with $A_p(z)$ and extract $\gamma_p = -a_{pp}$ from the highest coefficient of $A_p(z)$. Then, use Eq. (12.3.23) to obtain the polynomial $A_{p-1}(z)$. The iteration begins at the given order M and proceeds downwards to $p = M-1, M-2, \ldots, 1, 0$.

The function **rlev** generates the set of autocorrelation lags $\{R(0), R(1), ..., R(M)\}$ from the knowledge of the final prediction-error filter $A_M(z)$ and final prediction error E_M. It calls **bkwlev** to generate all the lower order prediction-error filters, and then it reconstructs the autocorrelation lags using the gapped function condition $g_p(p) = \sum_{i=0}^{p} a_{pi} R(p-i) = 0$, which may be solved for $R(p)$ in terms of $R(p-i)$, $i = 1, 2, \ldots, p$, as follows:

$$R(p) = -\sum_{i=1}^{p} a_{pi} R(p-i)\,, \quad p = 1, 2, \ldots, M \tag{12.3.24}$$

For example, the first few iterations of Eq. (12.3.24) will be:

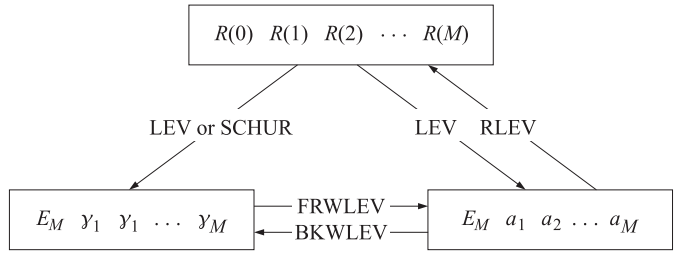
$$\begin{aligned} R(1) &= -[a_{11} R(0)] \\ R(2) &= -[a_{21} R(1) + a_{22} R(0)] \\ R(3) &= -[a_{31} R(2) + a_{32} R(1) + a_{33} R(0)] \end{aligned}$$

To get this recursion started, the value of $R(0)$ may be obtained from Eq. (12.3.13). Using Eq. (12.3.13) repeatedly, and $E_0 = R(0)$ we find

$$E_M = (1-\gamma_1^2)(1-\gamma_2^2)\cdots(1-\gamma_M^2) R(0) \tag{12.3.25}$$

Since the reflection coefficients are already known (from the call to **bkwlev**) and E_M is given, this equation provides the right value for $R(0)$.

The function **schur**, based on the Schur algorithm and discussed in Sec. 12.10, is an alternative to **lev**. The logical interconnection of these functions is shown below.



Example 12.3.1: Given the autocorrelation lags

$$\{R(0), R(1), R(2), R(3), R(4)\} = \{128, -64, 80, -88, 89\}$$

Find all the prediction-error filters $A_p(z)$ up to order four, the four reflection coefficients, and the corresponding mean-square prediction errors. Below, we simply state the results obtained using the function **lev**:

$$A_1(z) = 1 + 0.5z^{-1}$$
$$A_2(z) = 1 + 0.25z^{-1} - 0.5z^{-2}$$
$$A_3(z) = 1 - 0.375z^{-2} + 0.5z^{-3}$$
$$A_4(z) = 1 - 0.25z^{-1} - 0.1875z^{-2} + 0.5z^{-3} - 0.5z^{-4}$$

The reflection coefficients are the negatives of the highest coefficients; thus,

$$\{\gamma_1, \gamma_2, \gamma_3, \gamma_4\} = \{-0.5,\ 0.5,\ -0.5,\ 0.5\}$$

The vector of mean-squared prediction errors is given by

$$\{E_0, E_1, E_2, E_3, E_4\} = \{128,\ 96,\ 72,\ 54,\ 40.5\}$$

Sending the above vector of reflection coefficients through the function **frwlev** would generate the above set of polynomials. Sending the coefficients of $A_4(z)$ through **bkwlev** would generate the same set of polynomials. Sending the coefficients of $A_4(z)$ and $E_4 = 40.5$ through **rlev** would recover the original autocorrelation lags $R(k)$, $k = 0, 1, 2, 3, 4$. □

The *Yule-Walker* method (see Sec. 12.2) can be used to extract the linear prediction parameters from a given set of signal samples. From a given length-N block of data

$$\boxed{y_0, y_1, y_2, \ldots, y_{N-1}}$$

compute the sample autocorrelations $\{\hat{R}(0), \hat{R}(1), \ldots, \hat{R}(M)\}$ using, for example, Eq. (12.2.7), and send them through the Levinson recursion. The **yw** implements the Yule-Walker method. The input to the function is the data vector of samples $\{y_0, y_1, \ldots, y_{N-1}\}$ and the desired final order M of the predictor. Its output is the set of all prediction-error filters up to order M, arranged in the matrix L, and the vector of mean-squared prediction errors up to order M, that is, $\{E_0, E_1, \ldots, E_M\}$

Example 12.3.2: Given the signal samples

$$\{y_0, y_1, y_2, y_3, y_4\} = \{1, 1, 1, 1, 1\}$$

determine all the prediction-error filters up to order four. Using the fourth order predictor, predict the sixth value in the above sequence, i.e., the value of y_5.

The sample autocorrelation of the above signal is easily computed using the methods of Chapter 1. We find (ignoring the $1/N$ normalization factor):

$$\{\hat{R}(0), \hat{R}(1), \hat{R}(2), \hat{R}(3), \{\hat{R}(4)\} = \{5, 4, 3, 2, 1\}$$

Sending these lags through the function **lev** we find the prediction-error filters:

$$A_1(z) = 1 - 0.8z^{-1}$$
$$A_2(z) = 1 - 0.889z^{-1} + 0.111z^{-2}$$
$$A_3(z) = 1 - 0.875z^{-1} + 0.125z^{-3}$$
$$A_4(z) = 1 - 0.857z^{-1} + 0.143z^{-4}$$

Therefore, the fourth order prediction of y_n given by Eq. (12.3.1) is

$$\hat{y}_n = 0.857y_{n-1} - 0.143y_{n-4}$$

which gives $\hat{y}_5 = 0.857 - 0.143 = 0.714$. □

The results of this section can also be derived from those of Sec. 1.8 by invoking stationarity and making the proper identification of the various quantities. The data vector $\mathbf{y}$ and the subvectors $\tilde{\mathbf{y}}$ and $\tilde{\tilde{\mathbf{y}}}$ are identified with $\mathbf{y} = \mathbf{y}_{p+1}(n)$, $\tilde{\mathbf{y}} = \mathbf{y}_p(n)$, and $\tilde{\tilde{\mathbf{y}}} = \mathbf{y}_p(n-1)$, where

$$\mathbf{y}_{p+1}(n) = \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-p} \\ y_{n-p-1} \end{bmatrix}, \quad \mathbf{y}_p(n) = \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-p} \end{bmatrix}, \quad \mathbf{y}_p(n-1) = \begin{bmatrix} y_{n-1} \\ y_{n-2} \\ \vdots \\ y_{n-p-1} \end{bmatrix} \tag{12.3.26}$$

It follows from stationarity that the autocorrelation matrices of these vectors are independent of the absolute time instant n; therefore, we write

$$R_p = E[\mathbf{y}_p(n)\mathbf{y}_p(n)^T] = E[\mathbf{y}_p(n-1)\mathbf{y}_p(n-1)^T], \quad R_{p+1} = E[\mathbf{y}_{p+1}(n)\mathbf{y}_{p+1}(n)^T]$$

It is easily verified that R_p is the order-p autocorrelation matrix defined in Eq. (12.3.7) and that the order-$(p+1)$ autocorrelation matrix R_{p+1} admits the block decompositions

$$R_{p+1} = \left[\begin{array}{c|ccc} R(0) & R(1) & \cdots & R(p+1) \\ \hline R(1) & & & \\ \vdots & & R_p & \\ R(p+1) & & & \end{array}\right] = \left[\begin{array}{ccc|c} & & & R(p+1) \\ & R_p & & \vdots \\ & & & R(1) \\ \hline R(p+1) & \cdots & R(1) & R(0) \end{array}\right]$$

It follows, in the notation of Sec. 1.8, that $\bar{R} = \tilde{R} = R_p$ and $\rho_a = \rho_b = R(0)$, and

$$\mathbf{r}_a = \begin{bmatrix} R(1) \\ \vdots \\ R(p+1) \end{bmatrix}, \quad \mathbf{r}_b = \begin{bmatrix} R(p+1) \\ \vdots \\ R(1) \end{bmatrix}$$

Thus, $\mathbf{r}_a$ and $\mathbf{r}_b$ are the reverse of each other. It follows that the backward predictors are the reverse of the forward ones. Therefore, Eq. (12.3.14) is the same as Eq. (1.8.40), with the identifications

$$\mathbf{a} = \mathbf{a}_{p+1}, \quad \mathbf{b} = \mathbf{b}_{p+1}, \quad \bar{\mathbf{a}} = \tilde{\mathbf{a}} = \mathbf{a}_p, \quad \bar{\mathbf{b}} = \tilde{\mathbf{b}} = \mathbf{b}_p$$

where

$$\mathbf{a}_{p+1}=\begin{bmatrix}1\\ a_{p+1,1}\\ \vdots\\ a_{p+1,p}\\ a_{p+1,p+1}\end{bmatrix},\quad \mathbf{b}_{p+1}=\begin{bmatrix}a_{p+1,p+1}\\ a_{p+1,p}\\ \vdots\\ a_{p+1,1}\\ 1\end{bmatrix}\quad \mathbf{a}_p=\begin{bmatrix}1\\ a_{p1}\\ \vdots\\ a_{pp}\end{bmatrix},\quad \mathbf{b}_p=\begin{bmatrix}a_{pp}\\ \vdots\\ a_{p1}\\ 1\end{bmatrix}$$

Symbolically, $\mathbf{b}_p=\mathbf{a}_p^R$, $\mathbf{b}_{p+1}=\mathbf{a}_{p+1}^R$. We have $\bar{E}_a=\bar{E}_b=E_p$ and $\gamma_a=\gamma_b=\gamma_{p+1}$. Thus, Eq. (12.3.15) may be written as

$$\mathbf{a}_{p+1}=\begin{bmatrix}\mathbf{a}_p\\ 0\end{bmatrix}-\gamma_{p+1}\begin{bmatrix}0\\ \mathbf{b}_p\end{bmatrix}=\begin{bmatrix}\mathbf{a}_p\\ 0\end{bmatrix}-\gamma_{p+1}\begin{bmatrix}0\\ \mathbf{a}_p^R\end{bmatrix} \tag{12.3.27}$$

The normal Eqs. (12.3.7) can be written for orders p and $p+1$ in the compact form of Eqs. (1.8.38) and (1.8.12)

$$R_p\mathbf{a}_p=E_p\mathbf{u}_p\,,\quad R_{p+1}\mathbf{a}_{p+1}=E_{p+1}\mathbf{u}_{p+1}\,,\quad \mathbf{u}_p=\begin{bmatrix}1\\ \mathbf{0}\end{bmatrix},\quad \mathbf{u}_{p+1}=\begin{bmatrix}\mathbf{u}_p\\ 0\end{bmatrix} \tag{12.3.28}$$

Recognizing that Eq. (12.3.12) can be written as $\Delta_p=\mathbf{a}_p^T\mathbf{r}_b$, it follows that the reflection coefficient equation (12.3.11) is the same as (1.8.42). The rows of the matrix L defined by Eq. (12.3.20) are the reverse of the forward predictors; that is, the backward predictors of successive orders. Thus, L is the same as that defined in Eq. (1.8.13). The rows of the matrix U defined in Eq. (1.8.30) are the forward predictors, with the first row being the predictor of highest order. For example,

$$U=\begin{bmatrix}1 & a_{41} & a_{42} & a_{43} & a_{44}\\ 0 & 1 & a_{31} & a_{32} & a_{33}\\ 0 & 0 & 1 & a_{21} & a_{22}\\ 0 & 0 & 0 & 1 & a_{11}\\ 0 & 0 & 0 & 0 & 1\end{bmatrix}$$

Comparing L with U, we note that one is obtained from the other by reversing its rows and then its columns; formally, $U=JLJ$, where J is the corresponding reversing matrix.

12.4 Levinson's Algorithm in Matrix Form

In this section, we illustrate the mechanics of the Levinson recursion—cast in matrix form—by explicitly carrying out a few of the recursions given in Eq. (12.3.15). The objective of such recursions is to solve normal equations of the type

$$\begin{bmatrix}R_0 & R_1 & R_2 & R_3\\ R_1 & R_0 & R_1 & R_2\\ R_2 & R_1 & R_0 & R_1\\ R_3 & R_2 & R_1 & R_0\end{bmatrix}\begin{bmatrix}1\\ a_{31}\\ a_{32}\\ a_{33}\end{bmatrix}=\begin{bmatrix}E_3\\ 0\\ 0\\ 0\end{bmatrix}$$

for the unknowns $\{E_3, a_{31}, a_{32}, a_{33}\}$. The corresponding prediction-error filter is

$$A_3(z)=1+a_{31}z^{-1}+a_{32}z^{-2}+a_{33}z^{-3}$$

and the minimum value of the prediction error is E_3. The solution is obtained in an iterative manner, by solving a family of similar matrix equations of lower dimensionality. Starting at the upper left corner,

$$\left[\begin{array}{cc|cc} R_0 & R_1 & R_2 & R_3\\ R_1 & R_0 & R_1 & R_2\\ \hline R_2 & R_1 & R_0 & R_1\\ R_3 & R_2 & R_1 & R_0\end{array}\right]$$

the R matrices are successively enlarged until the desired dimension is reached (4×4 in this example). Therefore, one successively solves the matrix equations

$$[R_0][1]=[E_0]\,,\quad \begin{bmatrix}R_0 & R_1\\ R_1 & R_0\end{bmatrix}\begin{bmatrix}1\\ a_{11}\end{bmatrix}=\begin{bmatrix}E_1\\ 0\end{bmatrix},\quad \begin{bmatrix}R_0 & R_1 & R_2\\ R_1 & R_0 & R_1\\ R_2 & R_1 & R_0\end{bmatrix}\begin{bmatrix}1\\ a_{21}\\ a_{22}\end{bmatrix}=\begin{bmatrix}E_2\\ 0\\ 0\end{bmatrix}$$

The solution of each problem is obtained *in terms of* the solution of the previous one. In this manner, the final solution is gradually built up. In the process, one also finds all the lower order prediction-error filters.

The iteration is based on two key properties of the autocorrelation matrix: first, the autocorrelation matrix of a given size contains as subblocks all the lower order autocorrelation matrices; and second, the autocorrelation matrix is reflection invariant. That is, it remains invariant under interchange of its columns and then its rows. This interchanging operation is equivalent to the similarity transformation by the "reversing" matrix J having 1's along its anti-diagonal, e.g.,

$$J=\begin{bmatrix}0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 1 & 0 & 0 & 0\end{bmatrix} \tag{12.4.1}$$

The invariance property means that the autocorrelation matrix commutes with the matrix J

$$JRJ^{-1}=R \tag{12.4.2}$$

This property immediately implies that if the matrix equation is satisfied:

$$\begin{bmatrix}R_0 & R_1 & R_2 & R_3\\ R_1 & R_0 & R_1 & R_2\\ R_2 & R_1 & R_0 & R_1\\ R_3 & R_2 & R_1 & R_0\end{bmatrix}\begin{bmatrix}a_0\\ a_1\\ a_2\\ a_3\end{bmatrix}=\begin{bmatrix}b_0\\ b_1\\ b_2\\ b_3\end{bmatrix}$$

then the following equation is also satisfied:

$$\begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} b_3 \\ b_2 \\ b_1 \\ b_0 \end{bmatrix}$$

The steps of the Levinson algorithm are explicitly as follows:

Step 0

Solve $R_0 \cdot 1 = E_0$. This defines E_0. Then enlarge to the next size by padding a zero, that is,

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} E_0 \\ \Delta_0 \end{bmatrix}, \quad \text{this defines } \Delta_0. \text{ Then, also}$$

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta_0 \\ E_0 \end{bmatrix}, \quad \text{by reversal invariance}$$

These are the preliminaries to Step 1.

Step 1

We wish to solve

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{11} \end{bmatrix} = \begin{bmatrix} E_1 \\ 0 \end{bmatrix} \tag{12.4.3}$$

Try an expression of the form

$$\begin{bmatrix} 1 \\ a_{11} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \gamma_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Acting on both sides by $\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix}$ and using the results of Step 0, we obtain

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{11} \end{bmatrix} = \begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \gamma_1 \begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \text{or,}$$

$$\begin{bmatrix} E_1 \\ 0 \end{bmatrix} = \begin{bmatrix} E_0 \\ \Delta_0 \end{bmatrix} - \gamma_1 \begin{bmatrix} \Delta_0 \\ E_0 \end{bmatrix}, \quad \text{or,}$$

$$E_1 = E_0 - \gamma_1 \Delta_0, \quad 0 = \Delta_0 - \gamma_1 E_0, \quad \text{or}$$

$$\gamma_1 = \frac{\Delta_0}{E_0}, \quad E_1 = E_0 - \gamma_1 \Delta_0 = (1 - \gamma_1^2) E_0, \quad \text{where } \Delta_0 = R_1$$

These define γ_1 and E_1. As a preliminary to Step 2, enlarge Eq. (12.4.3) to the next size by padding a zero

$$\begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{11} \\ 0 \end{bmatrix} = \begin{bmatrix} E_1 \\ 0 \\ \Delta_1 \end{bmatrix}, \quad \text{this defines } \Delta_1. \text{ Then, also}$$

$$\begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 0 \\ a_{11} \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta_1 \\ 0 \\ E_1 \end{bmatrix}, \quad \text{by reversal invariance}$$

Step 2

We wish to solve

$$\begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{21} \\ a_{22} \end{bmatrix} = \begin{bmatrix} E_2 \\ 0 \\ 0 \end{bmatrix} \tag{12.4.4}$$

Try an expression of the form:

$$\begin{bmatrix} 1 \\ a_{21} \\ a_{22} \end{bmatrix} = \begin{bmatrix} 1 \\ a_{11} \\ 0 \end{bmatrix} - \gamma_2 \begin{bmatrix} 0 \\ a_{11} \\ 1 \end{bmatrix}, \quad \text{with } \gamma_2 \text{ to be determined}$$

Acting on both sides by the 3×3 autocorrelation matrix and using Step 1, we find

$$\begin{bmatrix} E_2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} E_1 \\ 0 \\ \Delta_1 \end{bmatrix} - \gamma_2 \begin{bmatrix} \Delta_1 \\ 0 \\ E_1 \end{bmatrix}, \quad \text{or,}$$

$$E_2 = E_1 - \gamma_2 \Delta_1, \quad 0 = \Delta_1 - \gamma_2 E_1, \quad \text{or}$$

$$\gamma_2 = \frac{\Delta_1}{E_1}, \quad E_2 = (1 - \gamma_2^2) E_1, \quad \text{where } \Delta_1 = [R_2, R_1] \begin{bmatrix} 1 \\ a_{11} \end{bmatrix}$$

These define γ_2 and E_2. As a preliminary to Step 3, enlarge Eq. (12.4.4) to the next size by padding a zero

$$\begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{21} \\ a_{22} \\ 0 \end{bmatrix} = \begin{bmatrix} E_2 \\ 0 \\ 0 \\ \Delta_2 \end{bmatrix}, \quad \text{this defines } \Delta_2. \text{ Then, also}$$

$$\begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 0 \\ a_{22} \\ a_{21} \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta_2 \\ 0 \\ 0 \\ E_2 \end{bmatrix}, \quad \text{by reversal invariance}$$

Step 3

We wish to solve

$$\begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} = \begin{bmatrix} E_3 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{12.4.5}$$

Try an expression of the form:

$$\begin{bmatrix} 1 \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} = \begin{bmatrix} 1 \\ a_{21} \\ a_{22} \\ 0 \end{bmatrix} - \gamma_3 \begin{bmatrix} 0 \\ a_{22} \\ a_{21} \\ 1 \end{bmatrix}, \quad \text{with } \gamma_3 \text{ to be determined}$$

Acting on both sides by the 4×4 autocorrelation matrix and using Step 2, we obtain

$$\begin{bmatrix} E_3 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} E_2 \\ 0 \\ 0 \\ \Delta_2 \end{bmatrix} - \gamma_3 \begin{bmatrix} \Delta_2 \\ 0 \\ 0 \\ E_2 \end{bmatrix}, \quad \text{or,}$$

$$E_3 = E_2 - \gamma_3 \Delta_2\,, \quad 0 = \Delta_2 - \gamma_3 E_2\,, \quad \text{or}$$

$$\gamma_3 = \frac{\Delta_2}{E_2}\,, \quad E_3 = (1-\gamma_3^2)E_2\,, \quad \text{where } \Delta_2 = [R_3, R_2, R_1] \begin{bmatrix} 1 \\ a_{21} \\ a_{22} \end{bmatrix}$$

Clearly, the procedure can be continued to higher and higher orders, as required in each problem. Note that at each step, we used the order-updating Eqs. (1.8.40) in conjunction with Eq. (1.8.47).

12.5 Autocorrelation Sequence Extensions

In this section, we discuss the problem of extending an autocorrelation function and the related issues of singular autocorrelation matrices. The equivalence between an autocorrelation function and the set of reflection coefficients provides a convenient and systematic way to (a) test whether a given finite set of numbers are the autocorrelation lags of a stationary signal and (b) extend a given finite set of autocorrelation lags to arbitrary lengths while preserving the autocorrelation property.

For a finite set of numbers $\{R(0), R(1), \ldots, R(p)\}$ to be the lags of an autocorrelation function, it is necessary and sufficient that all reflection coefficients, extracted from this set via the Levinson recursion, have magnitude less than one; that is, $|\gamma_i| < 1$, for $i = 1, 2, \ldots, p$, and also that $R(0) > 0$. These conditions are equivalent to the positive definiteness of the autocorrelation matrix R_p. The proof follows from the fact that the positivity of R_p is equivalent to the conditions on the prediction errors $E_i > 0$, for $i = 1, 2, \ldots, p$. In turn, these conditions are equivalent to $E_0 = R(0) > 0$ and and, through Eq. (12.3.13), to the reflection coefficients having magnitude less than one.

The problem of extending a finite set $\{R(0), R(1), \ldots, R(p)\}$ of autocorrelation lags is to find a number $R(p+1)$ such that the extended set $\{R(0), R(1), \ldots, R(p), R(p+1)\}$ is still an autocorrelation sequence. This can be done by parametrizing $R(p+1)$ in terms of the next reflection coefficient γ_{p+1}. Solving Eq. (12.3.12) for $R(p+1)$ and using Eq. (12.3.11), we obtain

$$R(p+1) = \gamma_{p+1} E_p - \left[a_{p1} R(p) + a_{p2} R(p-1) + \cdots + a_{pp} R(1)\right] \tag{12.5.1}$$

Any number γ_{p+1} in the range $-1 < \gamma_{p+1} < 1$ will give rise to an acceptable value for $R(p+1)$. The choice $\gamma_{p+1} = 0$ is special and corresponds to the so-called autoregressive or *maximum entropy extension* of the autocorrelation function (see Problem 12.16). If this choice is repeated to infinity, we will obtain the set of reflection coefficients

$$\{\gamma_1, \gamma_2, \ldots, \gamma_p, 0, 0, \ldots\}$$

It follows from the Levinson recursion that all prediction-error filters of order greater than p will remain equal to the pth filter, $A_p(z) = A_{p+1}(z) = A_{p+2}(z) = \cdots$. Therefore, the corresponding whitening filter will be $A(z) = A_p(z)$, that is, an autoregressive model of order p. With the exception of the above autoregressive extension that leads to an all-pole signal model, the extendibility conditions $|\gamma_{p+i}| < 1$, $i \geq 1$, do not necessarily guarantee that the resulting signal model will be a rational (pole-zero) model.

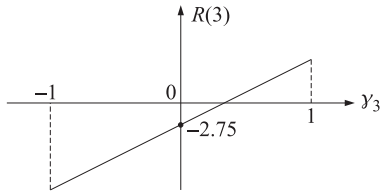
Example 12.5.1: Consider the three numbers $\{R(0), R(1), R(2)\} = \{8, 4, -1\}$. The Levinson recursion gives $\{\gamma_1, \gamma_2\} = \{0.5, -0.5\}$ and $\{E_1, E_2\} = \{6, 4.5\}$. Thus, the above numbers qualify to be autocorrelation lags. The corresponding prediction-error filters are

$$\mathbf{a}_1 = \begin{bmatrix} 1 \\ a_{11} \end{bmatrix} = \begin{bmatrix} 1 \\ -0.5 \end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix} 1 \\ a_{21} \\ a_{22} \end{bmatrix} = \begin{bmatrix} 1 \\ -0.75 \\ 0.5 \end{bmatrix}$$

The next lag in this sequence can be chosen according to Eq. (12.5.1)

$$R(3) = \gamma_3 E_2 - \left[a_{21} R(2) + a_{22} R(1)\right] = 4.5\gamma_3 - 2.75$$

where γ_3 is any number in the interval $-1 < \gamma_3 < 1$. The resulting possible values of $R(3)$ are plotted below versus γ_3. In particular, the autoregressive extension corresponds to $\gamma_3 = 0$, which gives $R(3) = -2.75$. □



The end-points, $\gamma_{p+1} = \pm 1$, of the allowed interval $(-1, 1)$ correspond to the two possible extreme values of $R(p+1)$:

$$R(p+1) = \pm E_p - \left[a_{p1} R(p) + a_{p2} R(p-1) + \cdots + a_{pp} R(1)\right]$$

In this case, the corresponding prediction error vanishes $E_{p+1} = (1 - \gamma_{p+1}^2) E_p = 0$. This makes the resulting order-$(p+1)$ autocorrelation matrix R_{p+1} singular. The prediction filter becomes either the symmetric (if $\gamma_{p+1} = -1$) or antisymmetric (if $\gamma_{p+1} = 1$) combination

$$\mathbf{a}_{p+1} = \begin{bmatrix} \mathbf{a}_p \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{a}_p^R \end{bmatrix}, \quad A_{p+1}(z) = A_p(z) + z^{-1} A_p^R(z)\,, \quad \text{or,}$$

$$\mathbf{a}_{p+1} = \begin{bmatrix} \mathbf{a}_p \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ \mathbf{a}_p^R \end{bmatrix}, \quad A_{p+1}(z) = A_p(z) - z^{-1} A_p^R(z)$$

In either case, it can be shown that the zeros of the polynomial $A_{p+1}(z)$ lie on the unit circle, and that the prediction filter $\mathbf{a}_{p+1}$ becomes an eigenvector of R_{p+1} with zero eigenvalue; namely, $R_{p+1}\mathbf{a}_{p+1} = 0$. This follows from the normal Eqs. (12.3.28) $R_{p+1}\mathbf{a}_{p+1} = E_{p+1}\mathbf{u}_{p+1}$ and $E_{p+1} = 0$.

Example 12.5.2: Consider the extended autocorrelation sequence of Example 12.5.1 defined by the singular choice $\gamma_3 = -1$. Then, $R(3) = -4.5 - 2.75 = -7.25$. The corresponding order-3 prediction-error filter is computed using the order-2 predictor and the Levinson recursion

$$\mathbf{a}_3 = \begin{bmatrix} 1 \\ a_{31} \\ a_{32} \\ a_{33} \end{bmatrix} = \begin{bmatrix} 1 \\ -0.75 \\ 0.5 \\ 0 \end{bmatrix} - \gamma_3 \begin{bmatrix} 0 \\ 0.5 \\ -0.75 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -0.25 \\ -0.25 \\ 1 \end{bmatrix}$$

It is symmetric about its middle. Its zeros, computed as the solutions of $(1 - 0.25z^{-1} - 0.25z^{-2} + z^{-3}) = (1 + z^{-1})(1 - 1.25z^{-1} + z^{-2}) = 0$ are

$$z = -1\,, \quad z = \frac{5 \pm j\sqrt{39}}{8}$$

and lie on the unit circle. Finally, we verify that $\mathbf{a}_3$ is an eigenvector of R_3 with zero eigenvalue:

$$R_3\mathbf{a}_3 = \begin{bmatrix} 8 & 4 & -1 & -7.25 \\ 4 & 8 & 4 & -1 \\ -1 & 4 & 8 & 4 \\ -7.25 & -1 & 4 & 8 \end{bmatrix} \begin{bmatrix} 1 \\ -0.25 \\ -0.25 \\ 1 \end{bmatrix} = 0 \qquad \square$$

Singular autocorrelation matrices, and the associated symmetric or antisymmetric prediction filters with zeros on the unit circle, find application in the method of *line spectrum pairs* (LSP) of speech analysis [937]. They are also intimately related to the eigenvector methods of spectrum estimation, such as Pisarenko's method of harmonic retrieval, discussed in Sec. 14.2. This connection arises from the property that singular autocorrelation matrices (with nonsingular principal minors) admit a *representation* as a sum of sinusoidal components [938], the frequencies of which are given precisely by the zeros, on the unit circle, of the corresponding prediction filter. This sinusoidal representation is equivalent to the eigen-decomposition of the matrix. The prediction filter can, alternatively, be computed as the eigenvector belonging to zero eigenvalue. The proof of these results can be derived as a limiting case; namely, the noise-free case, of the more general eigenvector methods that deal with sinusoids in noise. A direct proof is suggested in Problem 14.10.

Example 12.5.3: Consider the autocorrelation matrix $R = \begin{bmatrix} 2 & 1 & -1 \\ 1 & 2 & 1 \\ -1 & 1 & 2 \end{bmatrix}$. It is easily verified that the corresponding autocorrelation lags $R(k)$ admit the sinusoidal representation

$$R(k) = 2\cos(\omega_1 k) = e^{j\omega_1 k} + e^{-j\omega_1 k}\,, \quad \text{for } k = 0, 1, 2$$

where $\omega_1 = \pi/3$. Sending these lags through the Levinson recursion, we find $\{\gamma_1, \gamma_2\} = \{0.5, -1\}$ and $\{E_1, E_2\} = \{1.5, 0\}$. Thus, R singular. Its eigenvalues are $\{0, 3, 3\}$. The corresponding prediction filters are $\mathbf{a}_1 = [1, -0.5]^T$ and $\mathbf{a}_2 = [1, -1, 1]^T$. It is easily verified that $\mathbf{a}_2$ is an eigenvector of R with zero eigenvalue, i.e., $R\mathbf{a}_2 = 0$. The corresponding eigenfilter $A_2(z) = 1 - z^{-1} + z^{-2}$, is symmetric about its middle and has zeros on the unit circle coinciding with the sinusoids present in R, namely, $z = e^{\pm j\omega_1}$. The other two eigenvectors of R are

$$\mathbf{c} = \begin{bmatrix} 1 \\ \cos\omega_1 \\ \cos 2\omega_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \\ -0.5 \end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix} 0 \\ \sin\omega_1 \\ \sin 2\omega_1 \end{bmatrix} = \begin{bmatrix} 0 \\ \sqrt{3}/2 \\ \sqrt{3}/2 \end{bmatrix}$$

both belonging to eigenvalue $\lambda = 3$. Their norm is $\|\mathbf{c}\| = \|\mathbf{d}\| = \sqrt{3}/2$. The three eigenvectors $\mathbf{a}_2, \mathbf{c}, \mathbf{d}$ are mutually orthogonal. It is easily verified that the matrix R may be represented in the form $R = 2\mathbf{c}\mathbf{c}^T + 2\mathbf{d}\mathbf{d}^T$, which, after normalizing $\mathbf{c}$ and $\mathbf{d}$ to unit norm, is recognized as the eigendecomposition of R, We can also express R in terms of its complex sinusoidal components in the form $R = \mathbf{s}\mathbf{s}^\dagger + \mathbf{s}^*\mathbf{s}^T$, where

$$\mathbf{s} = \mathbf{c} + j\mathbf{d} = \begin{bmatrix} 1 \\ e^{j\omega_1} \\ e^{2j\omega_1} \end{bmatrix}, \quad \mathbf{s}^\dagger = \mathbf{s}^{*T} = \left[1,\, e^{-j\omega_1},\, e^{-2j\omega_1}\right]$$

Example 12.5.4: Similarly, one can verify that the four autocorrelation lags $\{8, 4, -1, -7.25\}$ of the singular matrix of Example 12.5.2 can be represented in the sinusoidal form

$$R(k) = P_1 e^{j\omega_1 k} + P_2 e^{j\omega_2 k} + P_3 e^{j\omega_3 k}\,, \quad \text{for } k = 0, 1, 2, 3$$

where $P_1 = 8/13$, $P_2 = P_3 = 96/13$, and ω_i correspond to the zeros of the prediction filter $\mathbf{a}_3$, namely,

$$e^{j\omega_1} = -1\,, \quad e^{j\omega_2} = \frac{5 + j\sqrt{39}}{8}\,, \quad e^{j\omega_3} = \frac{5 - j\sqrt{39}}{8}\,, \quad \text{so that, } \omega_3 = -\omega_2$$

The matrix itself has the sinusoidal representation

$$R = P_1\mathbf{s}_1\mathbf{s}_1^\dagger + P_2\mathbf{s}_2\mathbf{s}_2^\dagger + P_3\mathbf{s}_3\mathbf{s}_3^\dagger\,, \quad \text{where } \mathbf{s}_i = \begin{bmatrix} 1 \\ e^{j\omega_i} \\ e^{2j\omega_i} \\ e^{3j\omega_i} \end{bmatrix}$$

Had we chosen the value $\gamma_3 = 1$ in Example 12.5.2, we would have found the extended lag $R(3) = 1.75$ and the antisymmetric order-3 prediction-error filter $\mathbf{a}_3 = [1, -1.25, 1.25, -1]^T$, whose zeros are on the unit circle:

$$e^{j\omega_1} = 1\,, \quad e^{j\omega_2} = \frac{1 + j\sqrt{63}}{8}\,, \quad e^{j\omega_3} = \frac{1 - j\sqrt{63}}{8}$$

with $R(k)$ admitting the sinusoidal representation

$$R(k) = P_1 + 2P_2\cos(\omega_2 k) = [8, 4, -1, 1.75]\,, \quad \text{for } k = 0, 1, 2, 3$$

where $P_1 = 24/7$ and $P_2 = 16/7$. $\square$

12.6 Split Levinson Algorithm

The main computational burden of Levinson's algorithm is $2p$ multiplications per stage, arising from the p multiplications in Eq. (12.3.15) and in the computation of the inner product (12.3.12). Thus, for M stages, the algorithm requires

$$2\sum_{p=1}^{M} p = M(M+1)$$

or, $O(M^2)$ multiplications. This represents a factor of M savings over solving the normal equations (12.3.7) by direct matrix inversion, requiring $O(M^3)$ operations. The savings can be substantial considering that in speech processing $M = 10$–15, and in seismic processing $M = 100$–200. Progress in VLSI hardware has motivated the development of efficient parallel implementations of Levinson's algorithm and its variants [939–958]. With M parallel processors, the complexity of the algorithm is typically reduced by another factor of M to $O(M)$ or $O(M\log M)$ operations.

An interesting recent development is the realization that Levinson's algorithm has some inherent redundancy, which can be exploited to derive more efficient versions of the algorithm allowing an additional 50% reduction in computational complexity. These versions were motivated by a new stability test for linear prediction polynomials by Bistritz [959], and have been termed *Split Levinson* or *Immitance-Domain Levinson algorithms* [960–967]. They are based on efficient *three-term recurrence* relations for the symmetrized or antisymmetrized prediction polynomials. Following [960], we define the order-p symmetric polynomial

$$F_p(z) = A_{p-1}(z) + z^{-1}A^R_{p-1}(z)\,, \quad \mathbf{f}_p = \begin{bmatrix} \mathbf{a}_{p-1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{a}^R_{p-1} \end{bmatrix} \tag{12.6.1}$$

The coefficient vector $\mathbf{f}_p$ is symmetric about its middle; that is, $f_{p0} = f_{pp} = 1$ and $f_{pi} = a_{p-1,i} + a_{p-1,p-i} = f_{p,p-i}$, for $i = 1, 2, \ldots, p-1$. Thus, only half of the vector $\mathbf{f}_p$, is needed to specify it completely. Using the backward recursion (12.3.22) to write $A_{p-1}(z)$ in terms of $A_p(z)$, we obtain the alternative expression

$$F_p = \frac{1}{1-\gamma_p^2}\left[(A_p + \gamma_p A^R_p) + z^{-1}(\gamma_p z A_p + z A^R_p)\right] = \frac{1}{1-\gamma_p}\left[A_p + A^R_p\right], \quad \text{or,}$$

$$(1-\gamma_p)F_p(z) = A_p(z) + A^R_p(z)\,, \quad (1-\gamma_p)\mathbf{f}_p = \mathbf{a}_p + \mathbf{a}^R_p \tag{12.6.2}$$

The polynomial $A_p(z)$ and its reverse may be recovered from the knowledge of the symmetric polynomials $F_p(z)$. Writing Eq. (12.6.1) for order $p+1$, we obtain $F_{p+1}(z) = A_p(z) + z^{-1}A^R_p(z)$. This equation, together with Eq. (12.6.2), may be solved for $A_p(z)$ and $A^R_p(z)$, yielding

$$A_p(z) = \frac{F_{p+1}(z) - (1-\gamma_p)z^{-1}F_p(z)}{1-z^{-1}}\,, \quad A^R_p(z) = \frac{(1-\gamma_p)F_p(z) - F_{p+1}(z)}{1-z^{-1}} \tag{12.6.3}$$

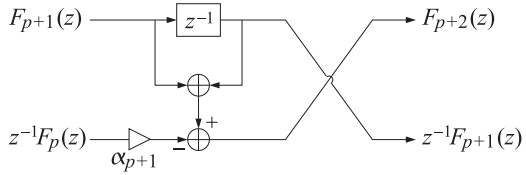
Inserting these expressions into the forward Levinson recursion (12.3.16) and canceling the common factor $1/(1-z^{-1})$, we obtain a three-term recurrence relation for $F_p(z)$:

$$F_{p+2} - (1-\gamma_{p+1})z^{-1}F_{p+1} = \left[F_{p+1} - (1-\gamma_p)z^{-1}F_p\right] - \gamma_{p+1}z^{-1}\left[(1-\gamma_p)F_p - F_{p+1}\right]$$

or,

$$F_{p+2}(z) = (1+z^{-1})F_{p+1}(z) - \alpha_{p+1}z^{-1}F_p(z) \tag{12.6.4}$$

where $\alpha_{p+1} = (1+\gamma_{p+1})(1-\gamma_p)$. In block diagram form



Because $F_p(z)$ has order p and is delayed by z^{-1}, the coefficient form of (12.6.4) is

$$\mathbf{f}_{p+2} = \begin{bmatrix} \mathbf{f}_{p+1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{f}_{p+1} \end{bmatrix} - \alpha_{p+1}\begin{bmatrix} 0 \\ \mathbf{f}_p \\ 0 \end{bmatrix} \tag{12.6.5}$$

The recursion is initialized by $F_0(z) = 2$ and $F_1(z) = 1 + z^{-1}$. Because of the symmetric nature of the polynomial $F_p(z)$ only half of its coefficients need be updated by Eqs. (12.6.4) or (12.6.5). To complete the recursion, we need an efficient way to update the coefficients α_{p+1}. Taking the dot product of both sides of Eq. (12.6.2) with the row vector $[R(0), R(1), \ldots, R(p)]$, we obtain

$$[R(0),\ldots,R(p)]\mathbf{a}_p + [R(0),\ldots,R(p)]\mathbf{a}^R_p = (1-\gamma_p)[R(0),\ldots,R(p)]\mathbf{f}_p$$

The first term is recognized as the gapped function $g_p(0) = E_p$, and the second term as $g_p(p) = 0$. Dividing by $1-\gamma_p$ and denoting $\tau_p = E_p/(1-\gamma_p)$, we obtain

$$\tau_p = [R(0), R(1), \ldots, R(p)]\mathbf{f}_p = \sum_{i=0}^{p} R(i) f_{pi} \tag{12.6.6}$$

Because of the symmetric nature of $\mathbf{f}_p$ the quantity τ_p can be computed using only half of the terms in the above inner product. For example, if p is odd, the above sum may be folded to half its terms

$$\tau_p = \sum_{i=0}^{(p-1)/2} [R(i) + R(p-i)] f_{pi}$$

Because Eqs. (12.6.5) and (12.6.6) can be folded in half, the total number of multiplications per stage will be $2(p/2) = p$, as compared with $2p$ for the classical Levinson algorithm. This is how the 50% reduction in computational complexity arises. The recursion is completed by noting that α_{p+1} can be computed in terms of τ_p by

$$\alpha_{p+1} = \frac{\tau_{p+1}}{\tau_p} \tag{12.6.7}$$

This follows from Eq. (12.3.13),

$$\frac{\tau_{p+1}}{\tau_p} = \frac{E_{p+1}}{1-\gamma_{p+1}}\frac{1-\gamma_p}{E_p} = \frac{1-\gamma^2_{p+1}}{1-\gamma_{p+1}}(1-\gamma_p) = (1+\gamma_{p+1})(1-\gamma_p) = \alpha_{p+1}$$

A summary of the algorithm, which also includes a recursive computation of the reflection coefficients, is as follows:

1. Initialize with $\tau_0 = E_0 = R(0)$, $\gamma_0 = 0$, $\mathbf{f}_0 = [2]$, $\mathbf{f}_1 = [1,1]^T$.
2. At stage p, the quantities $\tau_p, \gamma_p, \mathbf{f}_p, \mathbf{f}_{p+1}$ are available.
3. Compute τ_{p+1} from Eq. (12.6.6), using only half the terms in the sum.
4. Compute α_{p+1} from Eq. (12.6.7), and solve for $\gamma_{p+1} = -1 + \alpha_{p+1}/(1-\gamma_p)$.
5. Compute $\mathbf{f}_{p+2}$ from Eq. (12.6.5), using half of the coefficients.
6. Go to stage $p+1$.

After the final desired order is reached, the linear prediction polynomial can be recovered from Eq. (12.6.3), which can be written recursively as

$$a_{pi} = a_{p,i-1} + f_{p+1,i} - (1-\gamma_p) f_{p,i-1}\,, \quad i = 1, 2, \ldots, p \tag{12.6.8}$$

with $a_{p0} = 1$, or vectorially,

$$\begin{bmatrix} \mathbf{a}_p \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{a}_p \end{bmatrix} + \mathbf{f}_{p+1} - (1-\gamma_p) \begin{bmatrix} 0 \\ \mathbf{f}_p \end{bmatrix} \tag{12.6.9}$$

Using the three-term recurrence (12.6.5), we may replace $\mathbf{f}_{p+1}$ in terms of $\mathbf{f}_p$ and $\mathbf{f}_{p-1}$, and rewrite Eq. (12.6.9) as

$$\begin{bmatrix} \mathbf{a}_p \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{a}_p \end{bmatrix} + \begin{bmatrix} \mathbf{f}_p \\ 0 \end{bmatrix} + \gamma_p \begin{bmatrix} 0 \\ \mathbf{f}_p \end{bmatrix} - \alpha_p \begin{bmatrix} 0 \\ \mathbf{f}_{p-1} \\ 0 \end{bmatrix} \tag{12.6.10}$$

and in the z-domain

$$A_p(z) = z^{-1} A_p(z) + (1+\gamma_p z^{-1}) F_p(z) - \alpha_p z^{-1} F_{p-1}(z) \tag{12.6.11}$$

Example 12.6.1: We rederive the results of Example 12.3.1 using this algorithm, showing explicitly the computational savings. Initialize with $\tau_0 = R(0) = 128$, $\mathbf{f}_0 = [2]$, $\mathbf{f}_1 = [1,1]^T$. Using Eq. (12.6.6), we compute

$$\tau_1 = [R(0), R(1)]\mathbf{f}_1 = [R(0)+R(1)]f_{10} = 128 - 64 = 64$$

Thus, $\alpha_1 = \tau_1/\tau_0 = 64/128 = 0.5$ and $\gamma_1 = -1 + \alpha_1 = -0.5$. Using Eq. (12.6.5) we find

$$\mathbf{f}_2 = \begin{bmatrix} \mathbf{f}_1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{f}_1 \end{bmatrix} - \alpha_1 \begin{bmatrix} 0 \\ \mathbf{f}_0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} - 0.5 \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

and compute τ_2

$$\tau_2 = [R(0), R(1), R(2)]\mathbf{f}_2 = [R(0)+R(2)]f_{20} + R(1) f_{21} = 144$$

Thus, $\alpha_2 = \tau_2/\tau_1 = 144/64 = 2.25$ and $\gamma_2 = -1 + \alpha_2/(1-\gamma_1) = -1 + 2.25/1.5 = 0.5$. Next, compute $\mathbf{f}_3$ and τ_3

$$\mathbf{f}_3 = \begin{bmatrix} \mathbf{f}_2 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{f}_2 \end{bmatrix} - \alpha_2 \begin{bmatrix} 0 \\ \mathbf{f}_1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} - 2.25 \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ -0.25 \\ -0.25 \\ 1 \end{bmatrix}$$

$$\tau_3 = [R(0), R(1), R(2), R(3)]\mathbf{f}_3 = [R(0)+R(3)]f_{30} + [R(1)+R(2)]f_{31} = 36$$

which gives $\alpha_3 = \tau_3/\tau_2 = 36/144 = 0.25$ and $\gamma_3 = -1 + \alpha_3/(1-\gamma_2) = -0.5$. Next, we compute $\mathbf{f}_4$ and τ_4

$$\mathbf{f}_4 = \begin{bmatrix} \mathbf{f}_3 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{f}_3 \end{bmatrix} - \alpha_3 \begin{bmatrix} 0 \\ \mathbf{f}_2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ -0.25 \\ -0.25 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ -0.25 \\ -0.25 \\ 1 \end{bmatrix} - 0.25 \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \\ -0.75 \\ 0.5 \\ 1 \end{bmatrix}$$

$$\begin{aligned} \tau_4 &= [R(0), R(1), R(2), R(3), R(4)]\mathbf{f}_4 \\ &= [R(0)+R(4)]f_{40} + [R(1)+R(3)]f_{41} + R(2) f_{42} = 81 \end{aligned}$$

which gives $\alpha_4 = \tau_4/\tau_3 = 81/36 = 2.25$ and $\gamma_4 = -1 + \alpha_4/(1-\gamma_3) = 0.5$. The final prediction filter $\mathbf{a}_4$ can be computed using Eq. (12.6.9) or (12.6.10). To avoid computing $\mathbf{f}_5$ we use Eq. (12.6.10), which gives

$$\begin{bmatrix} 1 \\ a_{41} \\ a_{42} \\ a_{43} \\ a_{44} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ a_{41} \\ a_{42} \\ a_{43} \\ a_{44} \end{bmatrix} + \begin{bmatrix} 1 \\ 0.5 \\ -0.75 \\ 0.5 \\ 1 \\ 0 \end{bmatrix} + 0.5 \begin{bmatrix} 0 \\ 1 \\ 0.5 \\ -0.75 \\ 0.5 \\ 1 \end{bmatrix} - 2.25 \begin{bmatrix} 0 \\ 1 \\ -0.25 \\ -0.25 \\ 1 \\ 0 \end{bmatrix}$$

with solution $\mathbf{a}_4 = [1, -0.25, -0.1875, 0.5, -0.5]^T$. □

12.7 Analysis and Synthesis Lattice Filters

The Levinson recursion, expressed in the 2×2 matrix form of Eq. (12.3.18) forms the basis of the so-called lattice, or ladder, realizations of the prediction-error filters and their inverses [917]. Remembering that the prediction-error sequence $e_p(n)$ is the convolution of the prediction-error filter $[1, a_{p1}, a_{p2}, \ldots, a_{pp}]$ with the original data sequence y_n, that is,

$$e_p^+(n) = y_n + a_{p1} y_{n-1} + a_{p2} y_{n-2} + \cdots + a_{pp} y_{n-p} \tag{12.7.1}$$

we find in the z-domain

$$E_p^+(z) = A_p(z) Y(z) \tag{12.7.2}$$

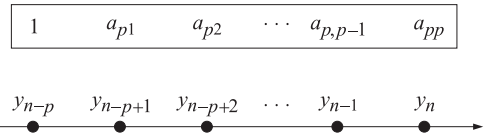
where we changed the notation slightly and denoted $e_p(n)$ by $e_p^+(n)$. At this point, it proves convenient to introduce the *backward* prediction-error sequence, defined in terms of the *reverse* of the prediction-error filter, as follows:

$$\begin{aligned} E_p^-(z) &= A_p^R(z) Y(z) \\ e_p^-(n) &= y_{n-p} + a_{p1} y_{n-p+1} + a_{p2} y_{n-p+2} + \cdots + a_{pp} y_n \end{aligned} \tag{12.7.3}$$

where $A_p^R(z)$ is the reverse of $A_p(z)$, namely,

$$A_p^R(z) = z^{-p} A_p(z^{-1}) = a_{pp} + a_{p,p-1} z^{-1} + a_{p,p-2} z^{-2} + \cdots + a_{p1} z^{-(p-1)} + z^{-p}$$

The signal sequence $e_p^-(n)$ may be interpreted as the *postdiction* error in postdicting the value of y_{n-p} on the basis of the *future* p samples $\{y_{n-p+1}, y_{n-p+2}, \ldots, y_{n-1}, y_n\}$, as shown below



Actually, the above choice of postdiction coefficients is the optimal one that minimizes the mean-square postdiction error

$$E[e_p^-(n)^2] = \min \tag{12.7.4}$$

This is easily shown by inserting Eq. (12.7.3) into (12.7.4) and using stationarity

$$E[e_p^-(n)^2] = E\left[\left(\sum_{m=0}^{p} a_{pm} y_{n-p+m}\right)^2\right] = \sum_{m,k=0}^{p} a_{pm} E[y_{n-p+m} y_{n-p+k}] a_{pk}$$

$$= \sum_{m,k=0}^{p} a_{pm} R(m-k) a_{pk} = E[e_p^+(n)^2]$$

which shows that the forward and the backward prediction error criteria are the same, thus, having the *same* solution for the optimal coefficients. We can write Eqs. (12.7.1) and (12.7.3) vectorially

$$e_p^+(n) = [1, a_{p1}, \ldots, a_{pp}] \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-p} \end{bmatrix} = \mathbf{a}_p^T \mathbf{y}_p(n) \tag{12.7.5a}$$

$$e_p^-(n) = [a_{pp}, a_{p,p-1}, \ldots, 1] \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-p} \end{bmatrix} = \mathbf{a}_p^{RT} \mathbf{y}_p(n) = \mathbf{b}_p^T \mathbf{y}_p(n) \tag{12.7.5b}$$

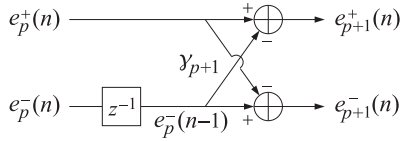
They are recognized as the forward and backward prediction errors e_a and e_b of Eq. (1.8.9). Multiplying both sides of the Levinson recursion (12.3.18) by $Y(z)$, we cast it in the equivalent form in terms of the forward and backward prediction-error sequences:

$$\begin{bmatrix} E_{p+1}^+(z) \\ E_{p+1}^-(z) \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_{p+1} z^{-1} \\ -\gamma_{p+1} & z^{-1} \end{bmatrix} \begin{bmatrix} E_p^+(z) \\ E_p^-(z) \end{bmatrix} \tag{12.7.6}$$

and in the time domain

$$\begin{aligned} e_{p+1}^+(n) &= e_p^+(n) - \gamma_{p+1} e_p^-(n-1) \\ e_{p+1}^-(n) &= e_p^-(n-1) - \gamma_{p+1} e_p^+(n) \end{aligned} \tag{12.7.7}$$

and in block diagram form



These recursions are initialized at $p = 0$ by

$$E_0^\pm(z) = A_0(z) Y(z) = Y(z) \quad \text{and} \quad e_0^\pm(n) = y_n \tag{12.7.8}$$

Eqs. (12.7.7) are identical to (1.8.50), with the identifications $e_a \to e_{p+1}^+(n)$, $\bar{e}_a \to e_p^+(n)$, $e_b \to e_{p+1}^-(n)$, $\tilde{e}_b \to e_p^-(n-1)$ the last following from Eq. (12.3.26).

The lattice realization of the prediction-error filter is based on the recursion (12.7.7). Starting at $p = 0$, the output of the pth stage of (12.7.7) becomes the input of the $(p+1)$th stage, up to the final desired order $p = M$. This is depicted in Fig. 12.7.1.

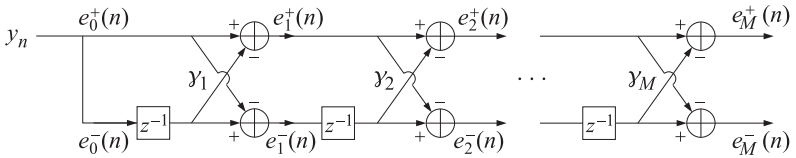


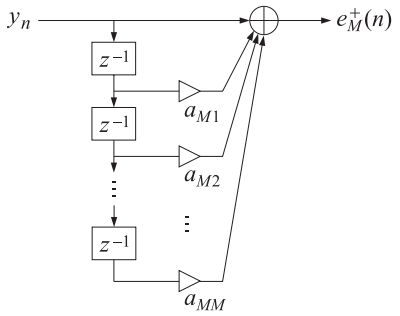
Fig. 12.7.1 Analysis lattice filter.

At each time instant n the numbers held in the M delay registers of the lattice can be taken as the internal state of the lattice. The function **lattice** is an implementation of Fig. 12.7.1. At each instant n, the function takes two overall inputs $e_0^\pm(n)$, makes M calls to the function **section** that implements the single lattice section (12.7.7), produces the two overall outputs $e_M^\pm(n)$, and updates the internal state of the lattice in preparation for the next call. By allowing the reflection coefficients to change between calls, the function can also be used in adaptive lattice filters.

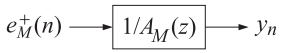
Eqs. (12.7.3) imply that the transfer function from the input y_n to the output $e_M^+(n)$ is the desired prediction-error filter $A_M(z)$, whereas the transfer function from y_n to $e_M^-(n)$ is the reversed filter $A_M^R(z)$. The lattice realization is therefore equivalent to the direct-form realization

$$e_M^+(n) = y_n + a_{M1} y_{n-1} + a_{M2} y_{n-2} + \cdots a_{MM} y_{n-M} \qquad y_n \longrightarrow \boxed{A_M(z)} \longrightarrow e_M^+(n)$$

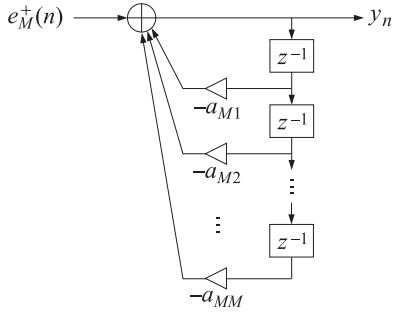
realized directly in terms of the prediction coefficients. It is depicted below



The synthesis filter $1/A_M(z)$ can also be realized in a lattice form. The input to the synthesis filter is the prediction error sequence $e_M^+(n)$ and its output is the original sequence y_n:



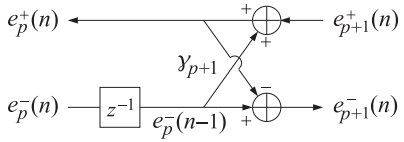
Its direct-form realization is:



For the lattice realization, since y_n corresponds to $e_0^+(n)$, we must write Eq. (12.7.7) in an order-decreasing form, starting at $e_M^+(n)$ and ending with $e_0^+(n) = y_n$. Rearranging the terms of the first of Eq. (12.7.7), we have

$$\begin{aligned} e_p^+(n) &= e_{p+1}^+(n) + \gamma_{p+1} e_p^-(n-1) \\ e_{p+1}^-(n) &= e_p^-(n-1) - \gamma_{p+1} e_p^+(n) \end{aligned} \tag{12.7.9}$$

which can be realized as shown below:



Note the difference in signs in the upper and lower adders. Putting together the stages from $p = M$ to $p = 0$, we obtain the synthesis lattice filter shown in Fig. 12.7.2.

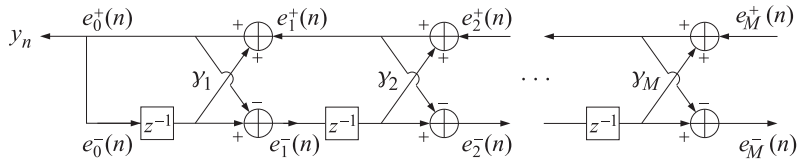


Fig. 12.7.2 Synthesis lattice filter.

Lattice structures based on the split Levinson algorithm can also be developed [960,961]. They are obtained by cascading the block diagram realizations of Eq. (12.6.4) for different values of α_p. The output signals from each section are defined by

$$e_p(n) = \sum_{i=0}^{p} f_{pi} y_{n-i}\,, \quad E_p(z) = F_p(z)\,Y(z)$$

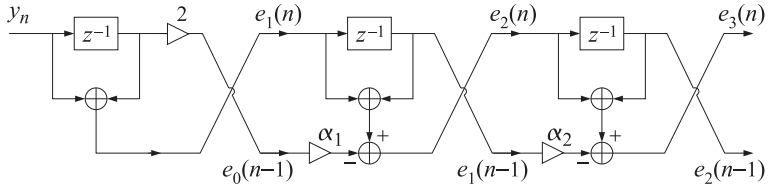
Multiplying both sides of Eq. (12.6.1) by $Y(z)$ we obtain the time-domain expression

$$e_p(n) = e_{p-1}^+(n) + e_{p-1}^-(n-1)$$

Similarly, multiplying both sides of Eq. (12.6.4) by $Y(z)$ we obtain the recursions

$$e_{p+2}(n) = e_{p+1}(n) + e_{p+1}(n-1) - \alpha_p e_p(n-1)$$

They are initialized by $e_0(n) = 2y_n$ and $e_1(n) = y_n + y_{n-1}$. Putting together the various sections we obtain the lattice-type realization



The forward prediction error may be recovered from Eq. (12.6.3) or Eq. (12.6.11) by multiplying both sides with $Y(z)$; for example, using Eq. (12.6.11) we find

$$e_p^+(n) = e_{p-1}^+(n) + e_p(n) + \gamma_p e_p(n) - \alpha_p e_{p-1}(n-1)$$

12.8 Alternative Proof of the Minimum-Phase Property

The synthesis filter $1/A_M(z)$ must be stable and causal, which requires all the M zeros of the prediction-error filter $A_M(z)$ to lie inside the unit circle on the complex z-plane. We have already presented a proof of this fact which was based on the property that the coefficients of $A_M(z)$ minimized the mean-squared prediction error $E[e_M^+(n)^2]$. Here, we present an alternative proof based on the Levinson recursion and the fact that

all reflection coefficients γ_p have magnitude less than one [920]. From the definition (12.7.3), it follows that

$$e_p^-(n-1) = y_{n-p-1} + a_{p1}y_{n-p} + a_{p2}y_{n-p+1} + \cdots + a_{pp}y_{n-1} \tag{12.8.1}$$

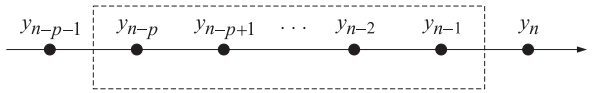
This quantity represents the estimation error of postdicting y_{n-p-1} on the basis of the p future samples $\{y_{n-p}, y_{n-p+1}, \ldots, y_{n-1}\}$. Another way to say this is that the linear combination of these p samples is the projection of y_{n-p-1} onto the subspace of random variables spanned by $\{y_{n-p}, y_{n-p+1}, \ldots, y_{n-1}\}$; that is,

$$e_p^-(n-1) = y_{n-p-1} - (\text{projection of } y_{n-p-1} \text{ onto } \{y_{n-p}, y_{n-p+1}, \ldots, y_{n-1}\}) \tag{12.8.2}$$

On the other hand, $e_p^+(n)$ given in Eq. (12.7.1) is the estimation error of y_n based on the same set of samples. Therefore,

$$e_p^+(n) = y_n - (\text{projection of } y_n \text{ onto } \{y_{n-p}, y_{n-p+1}, \ldots, y_{n-1}\}) \tag{12.8.3}$$

The samples $\{y_{n-p}, y_{n-p+1}, \ldots, y_{n-1}\}$ are the *intermediate* set of samples between y_{n-p-1} and y_n as shown below:



Therefore, according to the discussion in Sec. 1.7, the PARCOR coefficient between y_{n-p-1} and y_n with the effect of intermediate samples removed is given by

$$\text{PARCOR} = \frac{E[e_p^+(n)\, e_p^-(n-1)]}{E[e_p^-(n-1)^2]}$$

This is precisely the reflection coefficient γ_{p+1} of Eq. (12.3.11). Indeed, using Eq. (12.8.1) and the gap conditions, $g_p(k) = 0$, $k = 1, 2, \ldots, p$, we find

$$\begin{aligned} E[e_p^+(n)\, e_p^-(n-1)] &= E[e_p^+(n)\,(y_{n-p-1} + a_{p1}y_{n-p} + a_{p2}y_{n-p+1} + \cdots + a_{pp}y_{n-1})] \\ &= g_p(p+1) + a_{p1}g_p(p) + a_{p2}g_p(p-1) + \cdots a_{pp}g_p(1) \\ &= g_p(p+1) \end{aligned}$$

Similarly, invoking stationarity and Eq. (12.7.4),

$$E[e_p^-(n-1)^2] = E[e_p^-(n)^2] = E[e_p^+(n)^2] = g_p(0)$$

Thus, the reflection coefficient γ_{p+1} is really a PARCOR coefficient:

$$\gamma_{p+1} = \frac{E[e_p^+(n)\, e_p^-(n-1)]}{E[e_p^-(n-1)^2]} = \frac{E[e_p^+(n)\, e_p^-(n-1)]}{\sqrt{E[e_p^-(n-1)^2]\, E[e_p^+(n)^2]}} \tag{12.8.4}$$

Using the Schwarz inequality with respect to the inner product $E[uv]$, that is,

$$|E[uv]|^2 \le E[u^2]\, E[v^2]$$

then Eq. (12.8.4) implies that γ_{p+1} will have magnitude less than one:

$$|\gamma_{p+1}| \le 1\,, \quad \text{for each } p = 0, 1, \ldots \tag{12.8.5}$$

To prove the minimum-phase property of $A_M(z)$ we must show that all of its M zeros are inside the unit circle. We will do this by induction. Let Z_p and N_p denote the number of zeros and poles of $A_p(z)$ that lie inside the unit circle. Levinson's recursion, Eq. (12.3.13), expresses $A_{p+1}(z)$ as the sum of $A_p(z)$ and a correction term $F(z) = -\gamma_{p+1}z^{-1}A_p^R(z)$, that is,

$$A_{p+1}(z) = A_p(z) + F(z)$$

Using the inequality (12.8.5) and the fact that $A_p(z)$ has the same magnitude spectrum as $A_p^R(z)$, we find the inequality

$$|F(z)| = |-\gamma_{p+1}z^{-1}A_p^R(z)| = |\gamma_{p+1}A_p(z)| \le |A_p(z)|$$

for $z = e^{j\omega}$ on the unit circle. Then, the argument principle and Rouche's theorem imply that the addition of the function $F(z)$ will not affect the difference $N_p - Z_p$ of poles and zeros contained inside the unit circle. Thus,

$$N_{p+1} - Z_{p+1} = N_p - Z_p$$

Since the only pole of $A_p(z)$ is the multiple pole of order p at the origin arising from the term z^{-p}, it follows that $N_p = p$. Therefore,

$$(p+1) - Z_{p+1} = p - Z_p\,, \quad \text{or,}$$

$$Z_{p+1} = Z_p + 1$$

Starting at $p = 0$ with $A_0(z) = 1$, we have $Z_0 = 0$. It follows that

$$Z_p = p$$

which states that all the p zeros of the polynomial $A_p(z)$ lie inside the unit circle.

Another way to state this result is: "A necessary and sufficient condition for a polynomial $A_M(z)$ to have all of its M zeros strictly inside the unit circle is that all reflection coefficients $\{\gamma_1, \gamma_2, \ldots, \gamma_M\}$ resulting from $A_M(z)$ via the backward recursion Eq. (12.3.21) have magnitude strictly less than one." This is essentially equivalent to the well-known *Schur-Cohn test* of stability [968–971]. The function **bkwlev** can be used in this regard to obtain the sequence of reflection coefficients. The Bistritz test [959], mentioned in Sec. 12.6, is an alternative stability test.

Example 12.8.1: Test the minimum phase property of the polynomials

(a) $A(z) = 1 - 2.60z^{-1} + 2.55z^{-2} - 2.80z^{-3} + 0.50z^{-4}$
(b) $A(z) = 1 - 1.40z^{-1} + 1.47z^{-2} - 1.30z^{-3} + 0.50z^{-4}$

Sending the coefficients of each through the function **bkwlev**, we find the set of reflection coefficients

(a) $\{0.4, -0.5, 2.0, -0.5\}$
(b) $\{0.4, -0.5, 0.8, -0.5\}$

Since among (a) there is one reflection coefficient of magnitude greater than one, case (a) will not be minimum phase, whereas case (b) is. □

12.9 Orthogonality of Backward Prediction Errors—Cholesky Factorization

Another interesting structural property of the lattice realizations is that, in a certain sense, the backward prediction errors $e_p^-(n)$ are orthogonal to each other. To see this, consider the case $M = 3$, and form the matrix product

$$\underbrace{\begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix}}_{R} \underbrace{\begin{bmatrix} 1 & a_{11} & a_{22} & a_{33} \\ 0 & 1 & a_{21} & a_{32} \\ 0 & 0 & 1 & a_{31} \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{L^T} = \underbrace{\begin{bmatrix} E_0 & 0 & 0 & 0 \\ * & E_1 & 0 & 0 \\ * & * & E_2 & 0 \\ * & * & * & E_3 \end{bmatrix}}_{L_1}$$

Because the normal equations (written upside down) are satisfied by each prediction-error filter, the right-hand side will be a lower-triangular matrix. The "don't care" entries have been denoted by $*$s. Multiply from the left by L to get

$$LRL^T = LL_1 = \begin{bmatrix} E_0 & 0 & 0 & 0 \\ * & E_1 & 0 & 0 \\ * & * & E_2 & 0 \\ * & * & * & E_3 \end{bmatrix}$$

Since L is by definition lower-triangular, the right-hand side will still be lower triangular. But the left-hand side is symmetric. Thus, so is the right-hand side and as a result it must be diagonal. We have shown that

$$LRL^T = D = \text{diag}\{E_0, E_1, E_2, E_3\} \tag{12.9.1}$$

or, written explicitly

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ a_{11} & 1 & 0 & 0 \\ a_{22} & a_{21} & 1 & 0 \\ a_{33} & a_{32} & a_{31} & 1 \end{bmatrix} \begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 & a_{11} & a_{22} & a_{33} \\ 0 & 1 & a_{21} & a_{32} \\ 0 & 0 & 1 & a_{31} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} E_0 & 0 & 0 & 0 \\ 0 & E_1 & 0 & 0 \\ 0 & 0 & E_2 & 0 \\ 0 & 0 & 0 & E_3 \end{bmatrix}$$

This is identical to Eq. (1.8.17). The pqth element of this matrix equation is then

$$\mathbf{b}_p^T R \mathbf{b}_q = \delta_{pq} E_p \tag{12.9.2}$$

where $\mathbf{b}_p$ and $\mathbf{b}_q$ denote the pth and qth columns of L^T. These are recognized as the *backward prediction-error filters* of orders p and q. Eq. (12.9.2) implies then the orthogonality of the backward prediction-error filters with respect to an inner product $\mathbf{x}^T R \mathbf{y}$.

The backward prediction errors $e_p^-(n)$ can be expressed in terms of the $\mathbf{b}_p$s and the vector of samples $\mathbf{y}(n) = [y_n, y_{n-1}, y_{n-2}, y_{n-3}]^T$, as follows:

$$\begin{aligned} e_0^-(n) &= [1, \quad 0, \quad 0, \quad 0]\mathbf{y}(n) = \mathbf{b}_0^T \mathbf{y}(n) = y_n \\ e_1^-(n) &= [a_{11}, \quad 1, \quad 0, \quad 0]\mathbf{y}(n) = \mathbf{b}_1^T \mathbf{y}(n) = a_{11} y_n + y_{n-1} \\ e_2^-(n) &= [a_{22}, a_{21}, \quad 1, \quad 0]\mathbf{y}(n) = \mathbf{b}_2^T \mathbf{y}(n) = a_{22} y_n + a_{21} y_{n-1} + y_{n-2} \\ e_3^-(n) &= [a_{33}, a_{32}, a_{31}, 1]\mathbf{y}(n) = \mathbf{b}_3^T \mathbf{y}(n) = a_{33} y_n + a_{32} y_{n-1} + a_{31} y_{n-2} + y_{n-3} \end{aligned} \tag{12.9.3}$$

which can be rearranged into the vector form

$$\mathbf{e}^-(n) = \begin{bmatrix} e_0^-(n) \\ e_1^-(n) \\ e_2^-(n) \\ e_3^-(n) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ a_{11} & 1 & 0 & 0 \\ a_{22} & a_{21} & 1 & 0 \\ a_{33} & a_{32} & a_{31} & 1 \end{bmatrix} \begin{bmatrix} y_n \\ y_{n-1} \\ y_{n-2} \\ y_{n-3} \end{bmatrix} = L\mathbf{y}(n) \tag{12.9.4}$$

It is identical to Eq. (1.8.15). Using Eq. (12.9.1), it follows now that the covariance matrix of $\mathbf{e}^-(n)$ is diagonal; indeed, since $R = E[\mathbf{y}(n)\mathbf{y}(n)^T]$,

$$R_{e^-e^-} = E[\mathbf{e}^-(n)\mathbf{e}^-(n)^T] = LRL^T = D \tag{12.9.5}$$

which can also be expressed component-wise as the zero-lag cross-correlation

$$R_{e_p^- e_q^-}(0) = E[e_p^-(n) e_q^-(n)] = \delta_{pq} E_p \tag{12.9.6}$$

Thus, at each time instant n, the backward prediction errors $e_p^-(n)$ are mutually uncorrelated (orthogonal) with each other. The orthogonality conditions (12.9.6) and the lower-triangular nature of L render the transformation (12.9.4) equivalent to the *Gram-Schmidt orthogonalization* of the data vector $\mathbf{y}(n) = [y_n, y_{n-1}, y_{n-2}, y_{n-3}]^T$. Equation (12.9.1), written as

$$R = L^{-1} D L^{-T}$$

corresponds to an *LU Cholesky factorization* of the covariance matrix R.

Since the backward errors $e_p^-(n)$, $p = 0, 1, 2, \ldots, M$, for an Mth order predictor are generated at the output of each successive lattice segment of Fig. 12.7.1, we may view the analysis lattice filter as an implementation of the Gram-Schmidt orthogonalization of the vector $\mathbf{y}(n) = [y_n, y_{n-1}, y_{n-2}, \ldots, y_{n-M}]^T$.

It is interesting to note, in this respect, that this implementation requires only knowledge of the reflection coefficients $\{\gamma_1, \gamma_2, \ldots, \gamma_M\}$.

The data vector $\mathbf{y}(n)$ can also be orthogonalized by means of the forward predictors, using the matrix U. This representation, however, is not as conveniently realized by the lattice structure because the resulting orthogonalized vector consists of forward prediction errors that are orthogonal, but *not* at the same time instant. This can be seen from the definition of the forward errors

$$U\mathbf{y}(n) = \begin{bmatrix} 1 & a_{31} & a_{32} & a_{33} \\ 0 & 1 & a_{21} & a_{22} \\ 0 & 0 & 1 & a_{11} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_n \\ y_{n-1} \\ y_{n-2} \\ y_{n-3} \end{bmatrix} = \begin{bmatrix} e_3^+(n) \\ e_2^+(n-1) \\ e_1^+(n-2) \\ e_0^+(n-3) \end{bmatrix}$$

Thus, additional delays must be inserted at the forward outputs of the lattice structure to achieve orthogonalization. For this reason, the backward outputs, being mutually orthogonal at the *same* time instant n, are preferred. The corresponding UL factorization of R is in this basis

$$URU^T = \text{diag}\{E_3, E_2, E_1, E_0\}$$

This is the reverse of Eq. (12.9.1)) obtained by acting on both sides by the reversing matrix J and using the fact that $U = JLJ$, the invariance of $R = JRJ$, and $J^2 = I$.

The above orthogonalization may also be understood in the z-domain: since the backward prediction error $e_p^-(n)$ is the output of the reversed prediction-error filter $A_p^R(z)$ driven by the data sequence y_n, we have for the cross-density

$$S_{e_p^- e_q^-}(z) = A_p^R(z) S_{yy}(z) A_q^R(z^{-1})$$

Integrating this expression over the unit circle and using Eq. (12.9.6), we find

$$\begin{aligned}\oint_{\text{u.c.}} A_p^R(z) S_{yy}(z) A_q^R(z^{-1}) \frac{dz}{2\pi jz} &= \oint_{\text{u.c.}} S_{e_p^- e_q^-}(z) \frac{dz}{2\pi jz} \\ &= R_{e_p^- e_q^-}(0) = E\big[e_p^-(n) e_q^-(n)\big] = \delta_{pq} E_p\end{aligned} \tag{12.9.7}$$

that is, the reverse polynomials $A_p^R(z)$ are *mutually orthogonal* with respect to the above inner product defined by the (positive-definite) weighting function $S_{yy}(z)$. Equation (12.9.7) is the z-domain expression of Eq. (12.9.2). This result establishes an intimate connection between the linear prediction problem and the theory of *orthogonal polynomials* on the unit circle developed by Szegö [972,973].

The LU factorization of R implies a UL factorization of the inverse of R; that is, solving Eq. (12.9.1) for R^{-1} we have:

$$R^{-1} = L^T D^{-1} L \tag{12.9.8}$$

Since the Levinson recursion generates all the lower order prediction-error filters, it essentially generates the inverse of R.

The computation of this inverse may also be done *recursively* in the order, as follows. To keep track of the order let us use an extra index

$$R_3^{-1} = L_3^T D_3^{-1} L_3 \tag{12.9.9}$$

The matrix L_3 contains as a submatrix the matrix L_2; in fact,

$$L_3 = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ a_{11} & 1 & 0 & 0 \\ a_{22} & a_{21} & 1 & 0 \\ \hline a_{33} & a_{32} & a_{31} & 1 \end{array}\right] = \left[\begin{array}{c|c} L_2 & \mathbf{0} \\ \hline \boldsymbol{\alpha}_3^{RT} & 1 \end{array}\right] \tag{12.9.10}$$

where $\boldsymbol{\alpha}_3^{RT}$ denotes the transpose of the reverse of the vector of prediction coefficients; namely, $\boldsymbol{\alpha}_3^{RT} = [a_{33}, a_{32}, a_{21}]$. The diagonal matrix D_3^{-1} may also be block divided in the same manner:

$$D_3^{-1} = \left[\begin{array}{c|c} D_2^{-1} & \mathbf{0} \\ \hline \mathbf{0}^T & 1 \end{array}\right]$$

Inserting these block decompositions into Eq. (12.9.9) and using the lower order result $R_2^{-1} = L_2^T D_2^{-1} L_2$, we find

$$R_3^{-1} = \left[\begin{array}{c|c} R_2^{-1} + \frac{1}{E_3}\boldsymbol{\alpha}_3^R \boldsymbol{\alpha}_3^{RT} & \frac{1}{E_3}\boldsymbol{\alpha}_3^R \\ \hline \frac{1}{E_3}\boldsymbol{\alpha}_3^{RT} & \frac{1}{E_3} \end{array}\right] = \left[\begin{array}{c|c} R_2^{-1} & \mathbf{0} \\ \hline \mathbf{0}^T & 0 \end{array}\right] + \frac{1}{E_3}\mathbf{b}_3\mathbf{b}_3^T \tag{12.9.11}$$

where $\mathbf{b}_3 = \mathbf{a}_3^R = [\boldsymbol{\alpha}_3^{RT}, 1]^T = [a_{33}, a_{32}, a_{31}, 1]^T$. This is identical to Eq. (1.8.28).

Thus, through Levinson's algorithm, as the prediction coefficients $\boldsymbol{\alpha}_3$ and error E_3 are obtained, the inverse of R may be *updated* to the next higher order. Eq. (12.9.11) also suggests an efficient way of solving more general normal equations of the type

$$R_3\mathbf{h}_3 = \begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} h_{30} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{bmatrix} = \mathbf{r}_3 \tag{12.9.12}$$

for a given right-hand vector $\mathbf{r}_3$. Such normal equations arise in the design of FIR Wiener filters; for example, Eq. (11.3.9). The solution for $\mathbf{h}_3$ is obtained recursively from the solution of similar linear equations of lower order. For example, let $\mathbf{h}_2$ be the solution of the previous order

$$R_2\mathbf{h}_2 = \begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ r_2 \end{bmatrix} = \mathbf{r}_2$$

where the right-hand side vector $\mathbf{r}_2$ is part of $\mathbf{r}_3$. Then, Eq. (12.9.11) implies a recursive relationship between $\mathbf{h}_3$ and $\mathbf{h}_2$:

$$\mathbf{h}_3 = R_3^{-1}\mathbf{r}_3 = \left[\begin{array}{c|c} R_2^{-1} + \frac{1}{E_3}\boldsymbol{\alpha}_3^R \boldsymbol{\alpha}_3^{RT} & \frac{1}{E_3}\boldsymbol{\alpha}_3^R \\ \hline \frac{1}{E_3}\boldsymbol{\alpha}_3^{RT} & \frac{1}{E_3} \end{array}\right] \left[\begin{array}{c} \mathbf{r}_2 \\ \hline r_3 \end{array}\right] = \left[\begin{array}{c} \mathbf{h}_2 + \frac{1}{E_3}\boldsymbol{\alpha}_3^R (r_3 + \boldsymbol{\alpha}_3^{RT}\mathbf{r}_2) \\ \hline \frac{1}{E_3}(r_3 + \boldsymbol{\alpha}_3^{RT}\mathbf{r}_2) \end{array}\right]$$

In terms of the reverse prediction-error filter $\mathbf{b}_3 = \mathbf{a}_3^R = [a_{33}, a_{32}, a_{31}, 1]^T = [\boldsymbol{\alpha}_3^{RT}, 1]^T$, we may write

$$\mathbf{h}_3 = \begin{bmatrix} \mathbf{h}_2 \\ \hline 0 \end{bmatrix} + c\,\mathbf{b}_3\,, \quad \text{where } c = \frac{1}{E_3}(r_3 + \boldsymbol{\alpha}_3^{RT}\mathbf{r}_2) = \frac{1}{E_3}\mathbf{b}_3^T\mathbf{r}_3 \tag{12.9.13}$$

Thus, the recursive updating of the solution $\mathbf{h}$ must be done by carrying out the auxiliary updating of the prediction-error filters. The method requires $O(M^2)$ operations, compared to $O(M^3)$ if the inverse of R were to be computed directly.

This recursive method of solving general normal equations, developed by Robinson and Treitel, has been reviewed elsewhere [921,922,974–976] Some additional insight into the properties of these recursions can be gained by using the Toeplitz property of R. This property together with the symmetric nature of R imply that R commutes with the reversing matrix:

$$J_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} = J_3^{-1}\,, \quad J_3 R_3 J_3 = R_3 \tag{12.9.14}$$

Therefore, even though the inverse R_3^{-1} is not Toeplitz, it still commutes with this reversing matrix; that is,

$$J_3 R_3^{-1} J_3 = R_3^{-1} \tag{12.9.15}$$

The effect of this symmetry property on the block decomposition Eq. (12.9.11) may be seen by decomposing J_3 also as

$$J_3 = \left[\begin{array}{c|c} \mathbf{0} & J_2 \\ \hline 1 & \mathbf{0}^T \end{array}\right] = \left[\begin{array}{c|c} \mathbf{0}^T & 1 \\ \hline J_2 & \mathbf{0} \end{array}\right]$$

where J_2 is the lower order reversing matrix. Combining Eq. (12.9.15) with Eq. (12.9.11), we find

$$R_3^{-1} = J_3 R_3^{-1} J_3 = \left[\begin{array}{c|c} \mathbf{0}^T & 1 \\ \hline J_2 & \mathbf{0} \end{array}\right] \left[\begin{array}{c|c} R_2^{-1} + \dfrac{1}{E_3}\boldsymbol{\alpha}_3^R \boldsymbol{\alpha}_3^{RT} & \dfrac{1}{E_3}\boldsymbol{\alpha}_3^R \\ \hline \dfrac{1}{E_3}\boldsymbol{\alpha}_3^{RT} & \dfrac{1}{E_3} \end{array}\right] \left[\begin{array}{c|c} \mathbf{0} & J_2 \\ \hline 1 & \mathbf{0}^T \end{array}\right]$$

or, since R_2 commutes with J_2, and $J_2\boldsymbol{\alpha}_3^R = \boldsymbol{\alpha}_3$, we have

$$R_3^{-1} = \left[\begin{array}{c|c} \dfrac{1}{E_3} & \dfrac{1}{E_3}\boldsymbol{\alpha}_3^T \\ \hline \dfrac{1}{E_3}\boldsymbol{\alpha}_3 & R_2^{-1} + \dfrac{1}{E_3}\boldsymbol{\alpha}_3\boldsymbol{\alpha}_3^T \end{array}\right] = \left[\begin{array}{cc} 0 & \mathbf{0}^T \\ \mathbf{0} & R_2^{-1} \end{array}\right] + \frac{1}{E_3}\mathbf{a}_3\mathbf{a}_3^T \tag{12.9.16}$$

which is the same as Eq. (1.8.35). Both ways of expressing R_3^{-1} given by Eqs. (12.9.16) and (12.9.11), are useful. They may be combined as follows: Eq. (12.9.16) gives for the ijth matrix element:

$$(R_3^{-1})_{ij} = (R_2^{-1} + \boldsymbol{\alpha}_3\boldsymbol{\alpha}_3^T E_3^1)_{i-1,j-1} = (R_2^{-1})_{i-1,j-1} + \alpha_{3i}\alpha_{3j}E_3^{-1}$$

which valid for $1 \le i,j \le 3$. On the other hand, from Eq. (12.9.11) we have

$$(R_3^{-1})_{i-1,j-1} = (R_2^{-1})_{i-1,j-1} + \alpha_{3i}^R\alpha_{3j}^R E_3^{-1}$$

which is valid also for $1 \le i,j \le 3$. Subtracting the two to cancel the common term $(R_2^{-1})_{i-1,j-1}$, we obtain the Goberg-Semencul-Trench-Zohar recursion [977–981]:

$$(R_3^{-1})_{ij} = (R_3^{-1})_{i-1,j-1} + (\boldsymbol{\alpha}_3\boldsymbol{\alpha}_3^T - \boldsymbol{\alpha}_3^R\boldsymbol{\alpha}_3^{RT})_{ij}E_3^{-1}\,, \quad 1 \le i,j \le 3 \tag{12.9.17}$$

which allows the building-up of R_3^{-1} along each diagonal, provided one knows the "boundary" values to get these recursions started. But these are:

$$(R_3^{-1})_{00} = E_3^{-1}\,, \quad (R_3^{-1})_{i0} = (R_3^{-1})_{0i} = a_{3i}E_3^{-1}\,, \quad 1 \le i,j \le 3 \tag{12.9.18}$$

Thus, from the prediction-error filter $\mathbf{a}_3$ and its reverse, the entire inverse of the autocorrelation matrix may be built up. Computationally, of course, the best procedure is to use Eq. (12.9.8), where L and D are obtained as byproducts of the Levinson recursion. The function **lev** of the appendix starts with the $M+1$ autocorrelation lags $\{R(0), R(1), \ldots, R(M)\}$ and generates the required matrices L and D. The main reason for the existence of fast algorithms for Toeplitz matrices can be traced to the nesting property that the principal submatrices of a Toeplitz matrix are simply the lower order Toeplitz submatrices. Similar fast algorithms have been developed for other types of structured matrices, such as Hankel and Vandermonde matrices [982–984].

12.10 Schur Algorithm

The Schur algorithm has its roots in the original work of Schur on the theory of functions bounded in the unit disk [985,986]. It is an important signal processing tool in a variety of contexts, such as linear prediction and signal modeling, fast matrix factorizations, filter synthesis, inverse scattering, and other applications [987–1007].

In linear prediction, Schur's algorithm is an efficient alternative to Levinson's algorithm and can be used to compute the set of reflection coefficients from the autocorrelation lags and also to compute the conventional LU Cholesky factorization of the autocorrelation matrix. The Schur algorithm is essentially the gapped function recursion (12.3.9). It proves convenient to work simultaneously with Eq. (12.3.9) and its reverse. We define the forward and backward gapped functions of order p

$$g_p^+(k) = E[e_p^+(n)y_{n-k}]\,, \quad g_p^-(k) = E[e_p^-(n)y_{n-k}] \tag{12.10.1}$$

The forward one is identical to that of Eq. (12.3.8). The backward one is the convolution of the backward filter $\mathbf{b}_p = \mathbf{a}_p^R$ with the autocorrelation function; that is,

$$g_p^+(k) = \sum_{i=0}^{p} a_{pi}R(k-i)\,, \quad g_p^-(k) = \sum_{i=0}^{p} b_{pi}R(k-i) \tag{12.10.2}$$

where $b_{pi} = a_{p,p-i}$. In the z-domain, we have

$$G_p^+(z) = A_p(z)S_{yy}(z)\,, \quad G_p^-(z) = A_p^R(z)S_{yy}(z) \tag{12.10.3}$$

Using $S_{yy}(z) = S_{yy}(z^{-1})$, it follows that

$$G_p^-(z) = A_p^R(z)S_{yy}(z) = z^{-p}A_p(z^{-1})S_{yy}(z^{-1}) = z^{-p}G_p^+(z^{-1})$$

and in the time domain:

$$g_p^-(k) = g_p^+(p-k) \tag{12.10.4}$$

Thus, the backward gapped function is the reflected and delayed version of the forward one. However, the delay is only p units—one less than required to completely align the gaps. Therefore, the forward and backward gapped functions have slightly different gaps of length p; namely,

$$\begin{aligned} g_p^+(k) &= 0\,, \quad \text{for } k = 1,2,\ldots,p \\ g_p^-(k) &= 0\,, \quad \text{for } k = 0,1,\ldots,p-1 \end{aligned} \tag{12.10.5}$$

By the definition (12.10.1), the gap conditions of the backward function are equivalent to the orthogonality conditions for the backward predictor; namely, that the estimation error $e_p^-(n)$ be orthogonal to the observations $\{y_{n-k},\ k = 0,1,\ldots,p-1\}$ that make up the estimate of y_{n-p}. Inserting the lattice recursions (12.7.7) into (12.10.1), or using the polynomial recursions (12.3.18) into (12.10.3), we obtain the lattice recursions for the gapped functions, known as the *Schur recursions*

$$\begin{aligned} g_{p+1}^+(k) &= g_p^+(k) - \gamma_{p+1}g_p^-(k-1) \\ g_{p+1}^-(k) &= g_p^-(k-1) - \gamma_{p+1}g_p^+(k) \end{aligned} \tag{12.10.6}$$

or, in matrix form

$$\begin{bmatrix} g^+_{p+1}(k) \\ g^-_{p+1}(k) \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_{p+1} \\ -\gamma_{p+1} & 1 \end{bmatrix} \begin{bmatrix} g^+_p(k) \\ g^-_p(k-1) \end{bmatrix}$$

They are initialized by $g^\pm_0(k) = R(k)$. The first term of Eq. (12.10.6) is identical to Eq. (12.3.9) and the second term is the reverse of Eq. (12.3.9) obtained by the substitution $k \to p+1-k$. The forward gap condition $g^+_{p+1}(p+1) = 0$ can be solved for the reflection coefficient

$$\gamma_{p+1} = \frac{g^+_p(p+1)}{g^-_p(p)} \tag{12.10.7}$$

Note that Eq. (12.10.4) implies $g^-_p(p) = g^+_p(0) = E_p$, and therefore, Eq. (12.10.7) is the same as Eq. (12.3.11). For an Mth order predictor, we only need to consider the values $g^\pm_p(k)$, for $k = 0, 1, \ldots, M$. We arrange these values (for the backward function) into the column vector

$$\mathbf{g}^-_p = \begin{bmatrix} g^-_p(0) \\ g^-_p(1) \\ \vdots \\ g^-_p(M) \end{bmatrix} \tag{12.10.8}$$

By virtue of the gap conditions (12.10.5), the first p entries, $k = 0, 1, \ldots, p-1$, of this vector are zero. Therefore, we may construct the lower-triangular matrix having the $\mathbf{g}^-_p$s as columns

$$G = [\mathbf{g}^-_0, \mathbf{g}^-_1, \cdots, \mathbf{g}^-_M] \tag{12.10.9}$$

For example, if $M = 3$,

$$G = \begin{bmatrix} g^-_0(0) & 0 & 0 & 0 \\ g^-_0(1) & g^-_1(1) & 0 & 0 \\ g^-_0(2) & g^-_1(2) & g^-_2(2) & 0 \\ g^-_0(3) & g^-_1(3) & g^-_2(3) & g^-_3(3) \end{bmatrix}$$

The first column of G consists simply of the $M+1$ autocorrelation lags:

$$\mathbf{g}^-_0 = \begin{bmatrix} R(0) \\ R(1) \\ \vdots \\ R(M) \end{bmatrix} \tag{12.10.10}$$

The main diagonal consists of the prediction errors of successive orders, namely, $g^-_p(p) = E_p$, for $p = 0, 1, \ldots, M$. Stacking the values of definition (12.10.1) into a vector, we can write compactly,

$$\mathbf{g}^-_p = E[e_p(n)\mathbf{y}(n)] \tag{12.10.11}$$

where $\mathbf{y}(n) = [y_n, y_{n-1}, \ldots, y_{n-M}]^T$ is the data vector for an Mth order predictor. Thus, the matrix G can be written as in Eq. (1.8.56)

$$G = E\Big[\mathbf{y}(n)\,[e^-_0(n), e^-_1(n), \ldots, e^-_M(n)]\Big] = E[\mathbf{y}(n)\mathbf{e}^-(n)^T] \tag{12.10.12}$$

where $\mathbf{e}^-(n) = [e^-_0(n), e^-_1(n), \ldots, e^-_M(n)]^T$ is the decorrelated vector of backward prediction errors. Following Eq. (1.8.57), we multiply (12.10.12) from the left by the lower triangular matrix L, and using the transformation $\mathbf{e}^-(n) = L\mathbf{y}(n)$ and Eq. (12.9.5), we obtain

$$LG = LE[\mathbf{y}(n)\mathbf{e}^-(n)^T] = E[\mathbf{e}^-(n)\mathbf{e}^-(n)^T] = D$$

Therefore, G is essentially the inverse of L

$$G = L^{-1}D \tag{12.10.13}$$

Using Eq. (12.9.1), we obtain the conventional LU Cholesky factorization of the autocorrelation matrix R in the form

$$R = L^{-1}DL^{-T} = (GD^{-1})D(D^{-1}G^T) = GD^{-1}G^T \tag{12.10.14}$$

The backward gapped functions are computed by iterating the Schur recursions (12.10.6) for $0 \le k \le M$ and $0 \le p \le M$. One computational simplification is that, because of the presence of the gap, the functions $g^\pm_p(k)$ need only be computed for $p \le k \le M$ (actually, $g^+_p(p) = 0$ could also be skipped). This gives rise to the *Schur algorithm*:

0. Initialize in order by $g^\pm_0(k) = R(k)$, $k = 0, 1, \ldots, M$.
1. At stage p, we have available $g^\pm_p(k)$ for $p \le k \le M$.
2. Compute $\gamma_{p+1} = \dfrac{g^+_p(p+1)}{g^-_p(p)}$.
3. For $p+1 \le k \le M$, compute

$$g^+_{p+1}(k) = g^+_p(k) - \gamma_{p+1} g^-_p(k-1)$$
$$g^-_{p+1}(k) = g^-_p(k-1) - \gamma_{p+1} g^+_p(k)$$

4. Go to stage $p+1$.
5. At the final order M, set $E_M = g^-_M(M)$.

The function **schur** is an implementation of this algorithm. The inputs to the function are the order M and the lags $\{R(0), R(1), \ldots, R(M)\}$. The outputs are the parameters $\{E_M, \gamma_1, \gamma_2, \ldots, \gamma_M\}$. This function is a simple alternative to **lev**. It may be used in conjunction with **frwlev**, **bkwlev**, and **rlev**, to pass from one linear prediction parameter set to another. The function **schur1** is a small modification of **schur** that, in addition to the reflection coefficients, outputs the lower triangular Cholesky factor G. The prediction errors can be read off from the main diagonal of G, that is, $E_P = G(p,p)$, $p = 0, 1, \ldots, M$.

Example 12.10.1: Sending the five autocorrelation lags, $\{128, -64, 80, -88, 89\}$, of Example 12.3.1 through **schur1** gives the set of reflection coefficients $\{\gamma_1, \gamma_2, \gamma_3, \gamma_4\} = \{-0.5, 0.5, -0.5, 0.5\}$, and the matrix G

$$G = \begin{bmatrix} 128 & 0 & 0 & 0 & 0 \\ -64 & 96 & 0 & 0 & 0 \\ 80 & -24 & 72 & 0 & 0 \\ -88 & 36 & 0 & 54 & 0 \\ 89 & -43.5 & 13.5 & 13.5 & 40.5 \end{bmatrix}$$

Recall that the first column should be the autocorrelation lags and the main diagonal should consist of the mean square prediction errors. It is easily verified that $GD^{-1}G^T = R$. □

The computational bottleneck of the classical Levinson recursion is the computation of the inner product (12.3.12). The Schur algorithm avoids this step by computing γ_{p+1} as the ratio of the two gapped function values (12.10.7). Moreover, at each stage p, the computations indicated in step 3 of the algorithm can be done in parallel. Thus, with M parallel processors, the overall computation can be reduced to $O(M)$ operations. As formulated above, the Schur algorithm is essentially equivalent to the Le Roux-Gueguen fixed-point algorithm [990]. The possibility of a fixed-point implementation arises from the fact that all gapped functions have a fixed dynamic range, bounded by

$$\left| g_p^{\pm}(k) \right| \leq R(0) \tag{12.10.15}$$

This is easily seen by applying the Schwarz inequality to definition (12.10.1) and using $E_p \leq R(0)$

$$\left| g_p^{\pm}(k) \right|^2 = \left| E[e_p^{\pm}(n) y_{n-k}] \right|^2 \leq E[e_p^{\pm}(n)^2] E[y_{n-k}^2] \leq E_p R(0) \leq R(0)^2$$

The Schur algorithm admits a nice filtering interpretation in terms of the lattice structure. By definition, the gapped functions are the convolution of the forward/backward pth order prediction filters with the autocorrelation sequence $R(k)$. Therefore, $g_p^{\pm}(k)$ will be the outputs from the pth section of the lattice filter, Fig. 12.7.1, driven by the input $R(k)$. Moreover, Eq. (12.10.6) states that the $(p+1)$st reflection coefficient is obtainable as the ratio of the two *inputs* to the $(p+1)$st lattice section, at time instant $p+1$ (note that $g_p^-(p) = g_p^-(p+1-1)$ is outputted at time p from the pth section and is delayed by one time unit before it is inputted to the $(p+1)$st section at time $p+1$.) The correct values of the gapped functions $g_p^{\pm}(k)$ are obtained when the input to the lattice filter is the infinite double-sided sequence $R(k)$. If we send in the finite *causal* sequence

$$x(k) = \{R(0), R(1), \ldots, R(M), 0, 0, \ldots\}$$

then, because of the initial and final transient behavior of the filter, the outputs of the pth section will agree with $g_p^{\pm}(k)$ only for $p \leq k \leq M$. To see this, let $y_p^{\pm}(k)$ denote the two outputs. Because of the causality of the input and filter and the finite length of the input, the convolutional filtering equation will be

$$y_p^+(k) = \sum_{i=\max\{0,k-M\}}^{\min\{p,k\}} a_{pi}\, x(k-i) = \sum_{i=\max\{0,k-M\}}^{\min\{p,k\}} a_{pi}\, R(k-i)$$

This agrees with Eq. (12.10.2) only after time p and before time M, that is,

$$y_p^{\pm}(k) = g_p^{\pm}(k)\,, \quad \text{only for } p \leq k \leq M$$

The column vector $\mathbf{y}_p^- = [y_p^-(0), y_p^-(1), \ldots, y_p^-(M)]^T$, formed by the first M backward output samples of the pth section, will agree with $\mathbf{g}_p^-$ only for the entries $p \leq k \leq M$. Thus, the matrix of backward outputs $Y^- = [\mathbf{y}_0^-, \mathbf{y}_1^-, \ldots, \mathbf{y}_M^-]$ formed by the columns $\mathbf{y}_p^-$ will agree with G only in its *lower-triangular* part. But this is enough to determine G because its upper part is zero.

Example 12.10.2: Send the autocorrelation lags of Example 12.10.1 into the lattice filter of Fig. 12.7.1 (with all its delay registers initialized to zero), arrange the forward/backward outputs from the pth section into the column vectors, $\mathbf{y}_p^{\pm}$, and put these columns together to form the output matrices $Y^{\pm}$. The result is,

$$Y^- = \begin{bmatrix} 128 & 64 & -64 & 64 & -64 \\ -64 & 96 & 64 & -80 & 96 \\ 80 & -24 & 72 & 64 & -96 \\ -88 & 36 & 0 & 54 & 64 \\ 89 & -43.5 & 13.5 & 13.5 & 40.5 \end{bmatrix}, \quad Y^+ = \begin{bmatrix} 128 & 128 & 128 & 128 & 128 \\ -64 & 0 & -32 & -64 & -96 \\ 80 & 48 & 0 & 32 & 72 \\ -88 & -48 & -36 & 0 & -32 \\ 89 & 45 & 27 & 27 & 0 \end{bmatrix}$$

The lower-triangular part of Y^- agrees with G. The forward/backward outputs $\mathbf{y}_p^{\pm}$ can be computed using, for example, the function **lattice**. They can also be computed directly by convolving the prediction filters with the input. For example, the backward filter of order 4 given in Example 12.3.1 is $\mathbf{a}_4^R = [-0.5, 0.5, -0.1875, -0.25, 1]^T$. Convolving it with the autocorrelation sequence gives the last column of Y^-

$$[128, -64, 80, -88, 89] * [-0.5, 0.5, -0.1875, -0.25, 1] = [-64, 96, -96, 64, 40.5, \ldots]$$

Convolving the forward filter $\mathbf{a}_4$ with the autocorrelation sequence gives the last column of the matrix Y^+

$$[128, -64, 80, -88, 89] * [1, -0.25, -0.1875, 0.5, -0.5] = [128, -96, 72, -32, 0, \ldots]$$

Note that we are interested only in the outputs for $0 \leq k \leq M = 4$. The last 4 outputs (in general, the last p outputs for a pth order filter) of these convolutions were not shown. They correspond to the transient behavior of the filter after the input is turned off. □

It is also possible to derive a *split* or *immitance-domain* version of the Schur algorithm that achieves a further 50% reduction in computational complexity [962,963]. Thus, with M parallel processors, the complexity of the Schur algorithm can be reduced to $O(M/2)$ operations. We define a symmetrized or split gapped function in terms of the symmetric polynomial $F_p(z)$ defined in Eq. (12.6.1)

$$g_p(k) = \sum_{i=0}^{p} f_{pi}\, R(k-i)\,, \quad G_p(z) = F_p(z) S_{yy}(z) \tag{12.10.16}$$

It can be thought of as the output of the filter $F_p(z)$ driven by the autocorrelation sequence. Multiplying both sides of Eq. (12.6.1) by $S_{yy}(z)$ and using the definition (12.10.3), we obtain $G_p(z) = G_{p-1}^+(z) + z^{-1} G_{p-1}^-(z)$, or, in the time domain

$$g_p(k) = g_{p-1}^+(k) + g_{p-1}^-(k-1) \tag{12.10.17}$$

Similarly, Eq. (12.6.2) gives

$$(1-\gamma_p)\, g_p(k) = g_p^+(k) + g_p^-(k) \tag{12.10.18}$$

It follows from Eqs. (12.10.4) and (12.10.18) or from the symmetry property of $F_p(z)$ that $g_p(k) = g_p(p-k)$, and in particular, $g_p(0) = g_p(p)$. The split Levinson algorithm of Sec. 12.6 requires the computation of the coefficients $\alpha_{p+1} = \tau_{p+1}/\tau_p$. Setting $k = 0$ in

the definition (12.10.16) and using the reflection symmetry $R(i) = R(-i)$, we recognize that the inner product of Eq. (12.6.6) is $\tau_p = g_p(0) = g_p(p)$. Therefore, the coefficient α_{p+1} can be written as the ratio of the two gapped function values

$$\alpha_{p+1} = \frac{g_{p+1}(p+1)}{g_p(p)} \tag{12.10.19}$$

Because the forward and backward gapped functions have overlapping gaps, it follows that $g_p(k)$ will have gap $g_p(k) = 0$, for $k = 1, 2, \ldots, p-1$. Therefore, for an Mth order predictor, we only need to know the values of $g_p(k)$ for $p \le k \le M$. These can be computed by the following three-term recurrence, obtained by multiplying the recurrence (12.6.4) by $S_{yy}(z)$

$$g_{p+2}(k) = g_{p+1}(k) + g_{p+1}(k-1) - \alpha_{p+1} g_p(k-1) \tag{12.10.20}$$

Using $F_0(z) = 2$ and $F_1(z) = 1 + z^{-1}$, it follows from the definition that $g_0(k) = 2R(k)$ and $g_1(k) = R(k) + R(k-1)$. To initialize τ_0 correctly, however, we must choose $g_0(0) = R(0)$, so that $\tau_0 = g_0(0) = R(0)$. Thus, we are led to the following *split Schur algorithm*:

0. Initialize by $g_0(k) = 2R(k)$, $g_1(k) = R(k) + R(k-1)$, for $k = 1, 2, \ldots, M$, and $g_0(0) = R(0)$, $\gamma_0 = 0$.
1. At stage p, we have available γ_p, $g_p(k)$ for $p \le k \le M$, and $g_{p+1}(k)$ for $p+1 \le k \le M$.
2. Compute α_{p+1} from Eq. (12.10.19) and solve for $\gamma_{p+1} = -1 + \alpha_{p+1}/(1-\gamma_p)$.
3. For $p+2 \le k \le M$, compute $g_{p+2}(k)$ using Eq. (12.10.20)
4. Go to stage $p+1$.

Recalling that $E_p = \tau_p(1-\gamma_p)$, we may set at the final order $E_M = \tau_M(1-\gamma_M) = g_M(M)(1-\gamma_M)$. Step 3 of the algorithm requires only one multiplication for each k, whereas step 3 of the ordinary Schur algorithm requires *two*. This reduces the computational complexity by 50%. The function **schur2** (see Appendix B) is an implementation of this algorithm. The inputs to the function are the order M and the lags $\{R(0), R(1), \ldots, R(M)\}$. The outputs are the parameters $\{E_M, \gamma_1, \gamma_2, \ldots, \gamma_M\}$. The function can be modified easily to include the computation of the backward gapped functions $g_p^-(k)$, which are the columns of the Cholesky matrix G. This can be done by the recursion

$$g_p^-(k) = g_p^-(k-1) + (1-\gamma_p) g_p(k) - g_{p+1}(k) \tag{12.10.21}$$

where $p+1 \le k \le M$, with starting value $g_p^-(p) = E_p = g_p(p)(1-\gamma_p)$. This recursion will generate the lower-triangular part of G. Equation (12.10.21) follows by writing Eq. (12.10.17) for order $(p+1)$ and subtracting it from Eq. (12.10.18). Note, also, that Eq. (12.10.17) and the bound (12.10.15) imply the bound $|g_p(k)| \le 2R(0)$, which allows a fixed-point implementation.

We finish this section by discussing the connection of the Schur algorithm to Schur's original work. It follows from Eq. (12.10.3) that the ratio of the two gapped functions

$G_p^+(z)$ is an *all-pass stable* transfer function, otherwise known as a *lossless bounded real* function [971]:

$$S_p(z) = \frac{G_p^-(z)}{G_p^+(z)} = \frac{A_p^R(z)}{A_p(z)} = \frac{a_{pp} + a_{p,p-1} z^{-1} + \cdots + z^{-p}}{1 + a_{p1} z^{-1} + \cdots + a_{pp} z^{-p}} \tag{12.10.22}$$

The all-pass property follows from the fact that the reverse polynomial $A^R(z)$ has the same magnitude response as $A_p(z)$. The stability property follows from the minimum-phase property of the polynomials $A_p(z)$, which in turn is equivalent to all reflection coefficients having magnitude less than one. Such functions satisfy the boundedness property

$$|S_p(z)| \le 1, \quad \text{for } |z| \ge 1 \tag{12.10.23}$$

with equality attained on the unit circle. Taking the limit $z \to \infty$, it follows from Eq. (12.10.22) that the reflection coefficient γ_p is obtainable from $S_p(z)$ by

$$S_p(\infty) = a_{pp} = -\gamma_p \tag{12.10.24}$$

Using the backward Levinson recursion reqs5.3.23, we obtain a new all-pass function

$$S_{p-1}(z) = \frac{G_{p-1}^-(z)}{G_{p-1}^+(z)} = \frac{A_{p-1}^R(z)}{A_{p-1}(z)} = \frac{z(\gamma_p A_p + A_p^R)}{A_p + \gamma_p A_p^R}$$

or, dividing numerator and denominator by $A_p(z)$

$$S_{p-1}(z) = z \frac{S_p(z) + \gamma_p}{1 + \gamma_p S_p(z)} \tag{12.10.25}$$

This is Schur's original recursion [985]. Applying this recursion repeatedly from some initial value $p = M$ down to $p = 0$, with $S_0(z) = 1$, will give rise to the set of reflection or Schur coefficients $\{\gamma_1, \gamma_2, \ldots, \gamma_M\}$. The starting all-pass function $S_M(z)$ will be stable if and only if all reflection coefficients have magnitude less than one. We note finally that there is an intimate connection between the Schur algorithm and inverse scattering problems [991,995,1001,1002,1005–1007,1053].

In Sec. 12.13, we will see that the lattice recursions (12.10.6) describe the forward and backward moving waves incident on a layered structure. The Schur function $S_p(z)$ will correspond to the overall reflection response of the structure, and the recursion (12.10.25) will describe the successive removal of the layers. The coefficients γ_p will represent the elementary reflection coefficients at the layer interfaces. This justifies the term *reflection coefficients* for the γs.

12.11 Lattice Realizations of FIR Wiener Filters

In this section, we combine the results of Sections 11.3 and 12.9 to derive alternative realizations of Wiener filters that are based on the Gram-Schmidt lattice structures. Consider the FIR Wiener filtering problem of estimating a desired signal x_n, on the basis of the related signal y_n, using an Mth order filter. The I/O equation of the optimal filter is given by Eq. (11.3.8). The vector of optimal weights is determined by solving the set of

normal equations, given by Eq. (11.3.9). The discussion of the previous section suggests that Eq. (11.3.9) can be solved efficiently using the Levinson recursion. Defining the data vector

$$\mathbf{y}(n)=\begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-M} \end{bmatrix} \tag{12.11.1}$$

we rewrite Eq. (11.3.9) in the compact matrix form

$$R_{yy}\mathbf{h}=\mathbf{r}_{xy} \tag{12.11.2}$$

where R_{yy} is the $(M+1)\times(M+1)$ autocorrelation matrix of $\mathbf{y}(n)$, and $\mathbf{r}_{xy}$, the $(M+1)$-vector of cross-correlations between x_n, and $\mathbf{y}(n)$, namely,

$$R_{yy}=E\big[\mathbf{y}(n)\mathbf{y}(n)^T\big], \quad \mathbf{r}_{xy}=E[x_n\mathbf{y}(n)]=\begin{bmatrix} R_{xy}(0) \\ R_{xy}(1) \\ \vdots \\ R_{xy}(M) \end{bmatrix} \tag{12.11.3}$$

and $\mathbf{h}$ is the $(M+1)$-vector of optimal weights

$$\mathbf{h}=\begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_M \end{bmatrix} \tag{12.11.4}$$

The I/O equation of the filter, Eq. (12.9.4), is

$$\hat{x}_n=\mathbf{h}^T\mathbf{y}(n)=h_0y_n+h_1y_{n-1}+\cdots+h_My_{n-M} \tag{12.11.5}$$

Next, consider the Gram-Schmidt transformation of Eq. (12.9.4) from the data vector $\mathbf{y}(n)$ to the decorrelated vector $\mathbf{e}^-(n)$:

$$\mathbf{e}^-(n)=L\mathbf{y}(n) \quad \text{or,} \quad \begin{bmatrix} e_0^-(n) \\ e_1^-(n) \\ \vdots \\ e_M^-(n) \end{bmatrix}=L\begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_{n-M} \end{bmatrix} \tag{12.11.6}$$

Inserting (12.11.6) into (12.11.5), we find

$$\hat{x}_n=\mathbf{h}^TL^{-1}\mathbf{e}^-(n)$$

Defining the $(M+1)$-vector

$$\mathbf{g}=L^{-T}\mathbf{h} \tag{12.11.7}$$

we obtain the alternative I/O equation for the Wiener filter:

$$\hat{x}_n=\mathbf{g}^T\mathbf{e}^-(n)=\sum_{p=0}^{M}g_pe_p^-(n)=g_0e_0^-(n)+g_1e_1^-(n)+\cdots+g_Me_M^-(n) \tag{12.11.8}$$

This is easily recognized as the projection of x_n onto the subspace spanned by $\{e_0^-(n),e_1^-(n),\ldots,e_M^-(n)\}$, which is the same as that spanned by the data vector $\{y_n, y_{n-1},\ldots,y_{n-M}\}$. Indeed, it follows from Eqs. (12.11.7) and (12.11.2) that

$$\begin{aligned}\mathbf{g}^T=\mathbf{h}^TL^{-1}&=E[x_n\mathbf{y}(n)^T]E[\mathbf{y}(n)\mathbf{y}(n)^T]^{-1}L^{-1}\\&=E[x_n\mathbf{e}^-(n)^T]L^{-T}\big(L^{-1}E[\mathbf{e}^-(n)\mathbf{e}^-(n)^T]L^{-T}\big)^{-1}L^{-1}\\&=E[x_n\mathbf{e}^-(n)^T]E[\mathbf{e}^-(n)\mathbf{e}^-(n)^T]^{-1}\\&=\big[E[x_ne_0^-(n)]/E_0,\,E[x_ne_1^-(n)]/E_1,\ldots,E[x_ne_M^-(n)]/E_M\big]\end{aligned}$$

so that the estimate of x_n can be expressed as

$$\hat{x}_n=E[x_n\mathbf{e}^-(n)^T]E[\mathbf{e}^-(n)\mathbf{e}^-(n)^T]^{-1}\mathbf{e}^-(n)=E[x_n\mathbf{y}(n)^T]E[\mathbf{y}(n)\mathbf{y}(n)^T]^{-1}\mathbf{y}(n)$$

The key to the lattice realization of the optimal filtering equation (12.11.8) is the observation that the *analysis lattice filter* of Fig. 12.7.1 for the process y_n, provides, in its successive lattice stages, the signals $e_p^-(n)$ which are required in the sum (12.11.8). Thus, if the weight vector $\mathbf{g}$ is known, an alternative realization of the optimal filter will be as shown in Fig. 12.11.1. By comparison, the direct form realization using Eq. (12.11.5) operates directly on the vector $\mathbf{y}(n)$, which, at each time instant n, is available at the tap registers of the filter. This is depicted in Fig. 12.11.2.

Both types of realizations can be formulated *adaptively*, without requiring prior knowledge of the filter coefficients or the correlation matrices R_{yy} and $\mathbf{r}_{xy}$. We will discuss adaptive implementations in Chap. 16. If R_{yy} and $\mathbf{r}_{xy}$ are known, or can be estimated, then the design procedure for both the lattice and the direct form realizations is implemented by the following three steps:

1. Using Levinson's algorithm, implemented by the function **lev**, perform the LU Cholesky factorization of R_{yy}, to determine the matrices L and D.
2. The vector of weights $\mathbf{g}$ can be computed in terms of the known quantities $L, D, \mathbf{r}_{xy}$ as follows:
$$\mathbf{g}=L^{-T}\mathbf{h}=L^{-T}R_{yy}^{-1}\mathbf{r}_{xy}=L^{-T}(L^TD^{-1}L)\mathbf{r}_{xy}=D^{-1}L\mathbf{r}_{xy}$$
3. The vector $\mathbf{h}$ can be recovered from $\mathbf{g}$ by $\mathbf{h}=L^T\mathbf{g}$.

The function **firw** is an implementation of this design procedure. The inputs to the function are the order M and the correlation lags $\{R_{yy}(0),R_{yy}(1),\ldots,R_{yy}(M)\}$ and $\{R_{xy}(0),R_{xy}(1),\ldots,R_{xy}(M)\}$. The outputs are the quantities $L, D, \mathbf{g}$, and $\mathbf{h}$. The estimate (12.11.8) may also be written recursively in the order of the filter. If we denote,

$$\hat{x}_p(n)=\sum_{i=0}^{p}g_ie_i^-(n) \tag{12.11.9}$$

we obtain the recursion

$$\hat{x}_p(n)=\hat{x}_{p-1}(n)+g_pe_p^-(n), \quad p=0,1,\ldots,M \tag{12.11.10}$$

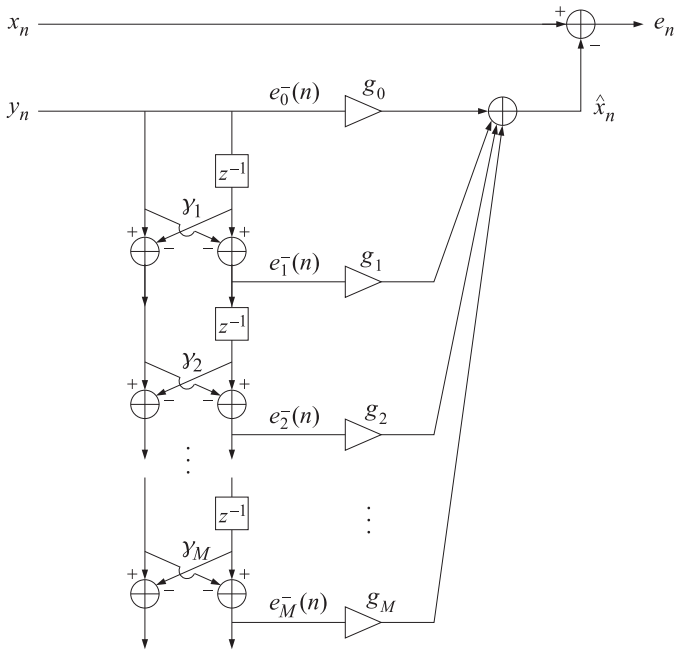


Fig. 12.11.1 Lattice realization of FIR Wiener filter.

initialized as $\hat{x}_{-1}(n) = 0$. The quantity $\hat{x}_p(n)$ is the *projection* of x_n on the subspace spanned by $\{e_0^-(n), e_1^-(n), \ldots, e_p^-(n)\}$, which by virtue of the lower-triangular nature of the matrix L is the same space as that spanned by $\{y_n, y_{n-1}, \ldots, y_{n-p}\}$. Thus, $\hat{x}_p(n)$ represents the optimal estimate of x_n based on a pth order filter. Similarly, $\hat{x}_{p-1}(n)$ represents the optimal estimate of x_n based on the $(p-1)$th order filter; that is, based on the past $p-1$ samples $\{y_n, y_{n-1}, \ldots, y_{n-p+1}\}$. These two subspaces differ by y_{n-p}.

The term $e_p^-(n)$ is by construction the best postdiction error of estimating y_{n-p} from the samples $\{y_n, y_{n-1}, \ldots, y_{n-p+1}\}$; that is, $e_p^-(n)$ is the *orthogonal complement* of y_{n-p} projected on that subspace. Therefore, the term $g_p e_p^-(n)$ in Eq. (12.11.10) represents the improvement in the estimate of x_n that results by taking into account the *additional* past value y_{n-p}; it represents that part of x_n that cannot be estimated in terms of the subspace $\{y_n, y_{n-1}, \ldots, y_{n-p+1}\}$. The estimate $\hat{x}_p(n)$ of x_n is *better* than $\hat{x}_{p-1}(n)$ in the sense that it produces a smaller mean-squared estimation error. To see this, define the estimation errors in the two cases

$$e_p(n) = x_n - \hat{x}_p(n)\,, \quad e_{p-1}(n) = x_n - \hat{x}_{p-1}(n)$$

Using the recursion (12.11.10), we find

$$e_p(n) = e_{p-1}(n) - g_p e_p^-(n) \tag{12.11.11}$$

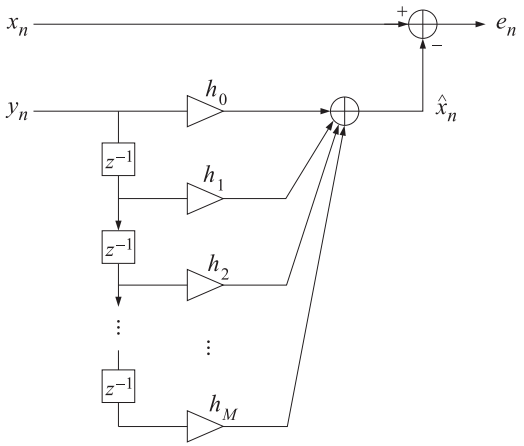


Fig. 12.11.2 Direct-form realization of FIR Wiener filter.

Using $g_p = E[x_n e_p^-(n)]/E_p$, we find for $\mathcal{E}_p = E[e_p(n)^2]$

$$\mathcal{E}_p = E[x_n^2] - \sum_{i=0}^{p} g_i E[x_n e_i^-(n)] = \mathcal{E}_{p-1} - g_p E[x_n e_p^-(n)]$$

$$= \mathcal{E}_{p-1} - \big(E[x_n e_p^-(n)]\big)^2 / E_p = \mathcal{E}_{p-1} - g_p^2 E_p$$

Thus, $\mathcal{E}_p$ is smaller than $\mathcal{E}_{p-1}$. This result shows explicitly how the estimate is constantly improved as the length of the filter is increased. The nice feature of the lattice realization is that the filter length can be increased simply by adding more lattice sections without having to recompute the weights g_p of the previous sections. A realization equivalent to Fig. 12.11.1, but which shows explicitly the recursive construction (12.11.10) of the estimate of x_n and of the estimation error (12.11.11), is shown in Fig. 12.11.3.

The function **lwf** is an implementation of the lattice Wiener filter of Fig. 12.11.3. The function **dwf** implements the direct-form Wiener filter of Fig. 12.11.2. Each call to these functions transforms a pair of input samples $\{x, y\}$ into the pair of output samples $\{\hat{x}, e\}$ and updates the internal state of the filter. Successive calls over $n = 0, 1, 2, \ldots$, will transform the input sequences $\{x_n, y_n\}$ into the output sequences $\{\hat{x}_n, e_n\}$. In both realizations, the internal state of the filter is taken to be the vector of samples stored in the delays of the filter; that is, $w_p(n) = e_{p-1}^-(n-1)$, $p = 1, 2, \ldots, M$ for the lattice case, and $w_p(n) = y_{n-p}$, $p = 1, 2, \ldots, M$ for the direct-form case. By allowing the filter coefficients to change between calls, these functions can be used in adaptive implementations.

Next, we present a Wiener filter design example for a noise canceling application. The primary and secondary signals $x(n)$ and $y(n)$ are of the form

$$x(n) = s(n) + v_1(n)\,, \quad y(n) = v_2(n)$$

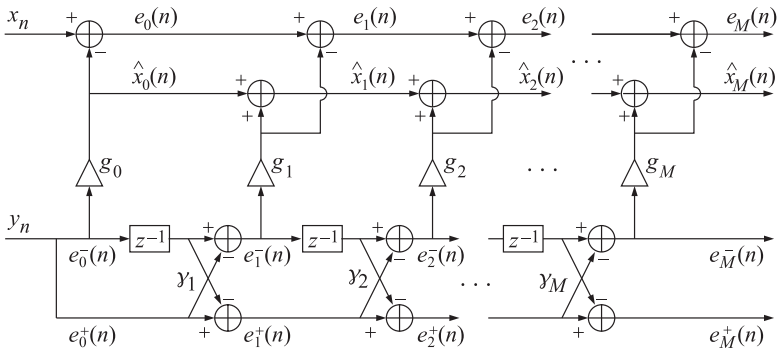
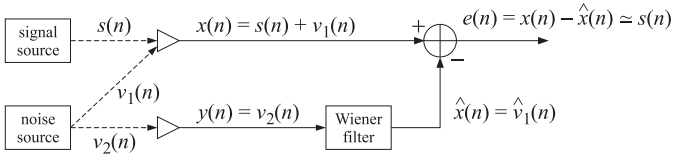


Fig. 12.11.3 Lattice realization of FIR Wiener filter.

where $s(n)$ is a desired signal corrupted by noise $v_1(n)$. The signal $v_2(n)$ is correlated with $v_1(n)$ but not with $s(n)$, and provides a reference noise signal. The noise canceler is to be implemented as a Wiener filter of order M, realized either in the direct or the lattice form. It is shown below:



Its basic operation is that of a correlation canceler; that is, the optimally designed filter $H(z)$ will transform the reference noise $v_2(n)$ into the best replica of $v_1(n)$, and then proceed to cancel it from the output, leaving a clean signal $s(n)$. For the purpose of the simulation, we took $s(n)$ to be a simple sinusoid

$$s(n) = \sin(\omega_0 n)\,, \quad \omega_0 = 0.075\pi \text{ [rads/sample]}$$

and $v_1(n)$ and $v_2(n)$ were generated by the difference equations

$$v_1(n) = -0.5v_1(n-1) + v(n)$$

$$v_2(n) = 0.8v_2(n-1) + v(n)$$

driven by a common, zero-mean, unit-variance, uncorrelated sequence $v(n)$. The difference equations establish a correlation between the two noise components v_1 and v_2, which is exploited by the canceler to effect the noise cancellation.

Figs. 12.11.4 and 12.11.5 show 100 samples of the signals $x(n)$, $s(n)$, and $y(n)$ generated by a particular realization of $v(n)$. For $M = 4$ and $M = 6$, the sample autocorrelation and cross-correlation lags, $R_{yy}(k)$, $R_{xy}(k)$, $k = 0, 1, \ldots, M$, were computed and sent through the function **firw** to get the filter weights **g** and **h**.

The reference signal y_n was filtered through $H(z)$ to get the estimate $\hat{x}_n$—which is really an estimate of $v_1(n)$—and the estimation error $e(n) = x(n) - \hat{x}(n)$, which is

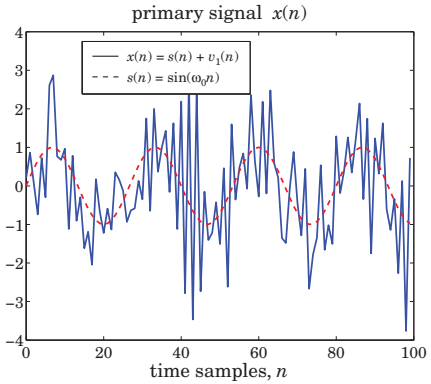


Fig. 12.11.4 Noise corrupted sinusoid.

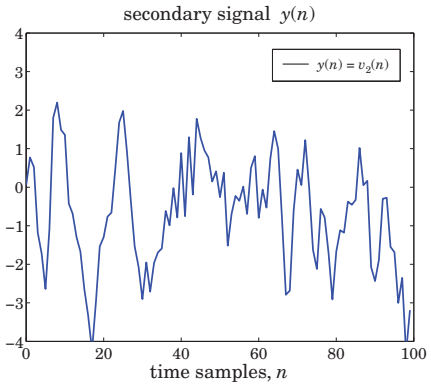


Fig. 12.11.5 Reference noise.

really an estimate of $s(n)$. This estimate of $s(n)$ is shown in Figs. (12.11.6) and 12.11.7, for the cases $M = 4$ and $M = 6$, respectively. The improvement afforded by a higher order filter is evident. For the particular realization of $x(n)$ and $y(n)$ that we used, the *sample* correlations $R_{yy}(k)$, $R_{xy}(k)$, $k = 0, 1, \ldots, M$, were:

$$R_{yy} = [2.5116,\ 1.8909,\ 1.2914,\ 0.6509,\ 0.3696,\ 0.2412,\ 0.1363]$$

$$R_{xy} = [0.7791,\ -0.3813,\ 0.0880,\ -0.3582,\ 0.0902,\ -0.0684,\ 0.0046]$$

and the resulting vector of lattice weights g_p, $p = 0, 1, \ldots, M$, reflection coefficients γ_p,

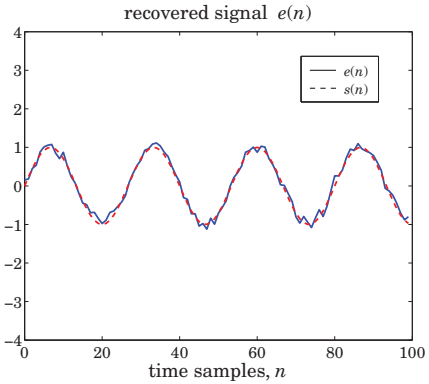


Fig. 12.11.6 Output of noise canceler ($M = 4$).

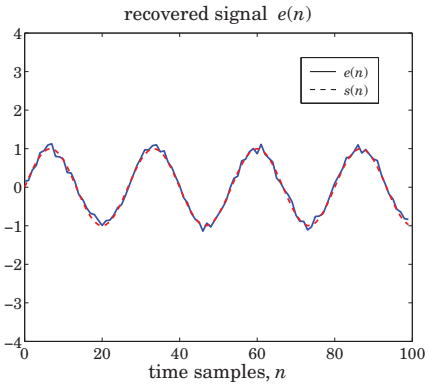


Fig. 12.11.7 Output of noise canceler ($M = 6$).

$p = 1, 2, \ldots, M$, and direct-form weights h_m, $m = 0, 1, \ldots, M$ were for $M = 6$,

$$\mathbf{g} = [0.3102,\ -0.8894,\ 0.4706,\ -0.2534,\ 0.1571,\ -0.0826,\ 0.0398]$$

$$\boldsymbol{\gamma} = [0.7528,\ -0.1214,\ -0.1957,\ 0.1444,\ 0.0354,\ -0.0937]$$

$$\mathbf{h} = [0.9713,\ -1.2213,\ 0.6418,\ -0.3691,\ 0.2245,\ -0.1163,\ 0.0398]$$

To get the $\mathbf{g}$ and $\boldsymbol{\gamma}$ of the case $M = 4$, simply ignore the last two entries in the above. The corresponding $\mathbf{h}$ is in this case:

$$\mathbf{h} = [0.9646,\ -1.2262,\ 0.6726,\ -0.3868,\ 0.1571]$$

Using the results of Problems 12.25 and 12.26, we may compute the theoretical filter weights for this example, and note that they compare fairly well with the estimated ones

that were based on the length-100 data blocks. For $M = 6$, we have:

$$\mathbf{g} = [0.2571,\ -0.9286,\ 0.4643,\ -0.2321,\ 0.1161,\ -0.0580,\ 0.0290]$$

$$\boldsymbol{\gamma} = [0.8,\ 0,\ 0,\ 0,\ 0,\ 0]$$

$$\mathbf{h} = [1,\ -1.3,\ 0.65,\ -0.325,\ 0.1625,\ -0.0812,\ 0.0290]$$

As we discussed in Sec. 1.8, the lattice realizations based on the backward orthogonal basis have three major advantages over the direct-form realizations: (a) the filter processes non-redundant information only, and hence adaptive implementations would adapt faster; (b) the design of the optimal filter weights $\mathbf{g}$ does not require any matrix inversion; and (c) the lower-order portions of $\mathbf{g}$ are already optimal. Moreover, it appears that adaptive versions of the lattice realizations have better numerical properties than the direct-form versions. In array processing problems, because the data vector $\mathbf{y}(n)$ does not have the tapped-delay line form (12.11.1), the Gram-Schmidt orthogonalization cannot be done by a simple a lattice filter. It requires a more complicated structure that basically amounts to carrying out the lower-triangular linear transformation (12.11.6). The benefits, however, are the same. We discuss adaptive versions of Gram-Schmidt preprocessors for arrays in Chap. 16.

12.12 Autocorrelation, Covariance, and Burg's Methods

As mentioned in Sec. 12.3, the finite order linear prediction problem may be thought of as an approximation to the infinite order prediction problem. For large enough order p of the predictor, the prediction-error filter $A_p(z)$ may be considered to be an adequate approximation to the whitening filter $A(z)$ of the process y_n. In this case, the prediction-error sequence $e_p^+(n)$ is approximately white, and the inverse synthesis filter $1/A_p(z)$ is an approximation to the signal model $B(z)$ of y_n. Thus, we have obtained an approximate solution to the signal modeling problem depicted below:

$$e_p^+(n) \longrightarrow \boxed{1/A_p(z)} \longrightarrow y_n$$

The variance of $e_p^+(n)$ is E_p. Depending on the realization one uses, the model parameters are either the set $\{a_{p1}, a_{p2}, \ldots, a_{pp}; E_p\}$, or, $\{\gamma_1, \gamma_2, \ldots, \gamma_p; E_p\}$. Because these can be determined by solving a simple linear system of equations—that is, the normal equations (12.3.7)—this approach to the modeling problem has become widespread.

In this section, we present three widely used methods of extracting the model parameters from a given block of measured signal values y_n [917,920,926,927,1008–1018]. These methods are:

1. The autocorrelation, or Yule-Walker, method
2. The covariance method.
3. Burg's method.

We have already discussed the Yule-Walker method, which consists simply of replacing the theoretical autocorrelations $R_{yy}(k)$ with the corresponding sample autocorrelations $\hat{R}_{yy}(k)$ computed from the given frame of data. This method, like the other

two, can be justified on the basis of an appropriate least-squares minimization criterion obtained by replacing the ensemble averages $E[e_p^+(n)^2]$ by appropriate time averages.

The theoretical minimization criteria for the optimal forward and backward predictors are

$$E[e_p^+(n)^2] = \min, \quad E[e_p^-(n)^2] = \min \tag{12.12.1}$$

where $e_p^+(n)$ and $e_p^-(n)$ are the result of filtering y_n through the prediction-error filter $\mathbf{a} = [1, a_{p1}, \ldots, a_{pp}]^T$ and its reverse $\mathbf{a}^R = [a_{pp}, a_{p,p-1}, \ldots, a_{p1}, 1]^T$, respectively; namely,

$$\begin{aligned} e_p^+(n) &= y_n + a_{p1}y_{n-1} + a_{p2}y_{n-2} + \cdots + a_{pp}y_{n-p} \\ e_p^-(n) &= y_{n-p} + a_{p1}y_{n-p+1} + a_{p2}y_{n-p+2} + \cdots + a_{pp}y_n \end{aligned} \tag{12.12.2}$$

Note that in both cases the mean-square value of $e_p^\pm(n)$ can be expressed in terms of the $(p+1)\times(p+1)$ autocorrelation matrix

$$R(i,j) = R(i-j) = E[y_{n+i-j}y_n] = E[y_{n-j}y_{n-i}], \quad 0 \le i,j \le p$$

as follows

$$E[e_p^+(n)^2] = E[e_p^-(n)^2] = \mathbf{a}^T R \mathbf{a} \tag{12.12.3}$$

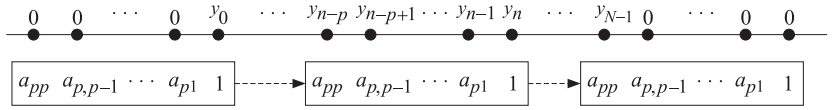
Consider a frame of length N of measured values of y_n

$$\boxed{y_0, y_1, \ldots, y_{N-1}}$$

1. The *Yule-Walker*, or *autocorrelation*, method replaces the ensemble average (12.12.1) by the least-squares time-average criterion

$$\mathcal{E} = \sum_{n=0}^{N+p-1} e_p^+(n)^2 = \min \tag{12.12.4}$$

where $e_p^+(n)$ is obtained by convolving the length-$(p+1)$ prediction-error filter $\mathbf{a} = [1, a_{p1}, \ldots, a_{pp}]^T$ with the length-N data sequence y_n. The length of the sequence $e_p^+(n)$ is, therefore, $N + (p+1) - 1 = N + p$, which justifies the upper-limit in the summation of Eq. (12.12.4). This convolution operation is equivalent to assuming that the block of data y_n has been extended both to the left and to the right by padding it with zeros and running the filter over this *extended* sequence. The last p output samples $e_p^+(n)$, $N \le n \le N+p-1$, correspond to running the filter off the ends of the data sequence to the right. These terms arise because the prediction-error filter has memory of p samples. This is depicted below:



Inserting Eq. (12.12.2) into (12.12.4), it is easily shown that $\mathcal{E}$ can be expressed in the equivalent form

$$\mathcal{E} = \sum_{n=0}^{N+p-1} e_p^+(n)^2 = \sum_{i,j=0}^{p} a_i \hat{R}(i-j) a_j = \mathbf{a}^T \hat{R} \mathbf{a} \tag{12.12.5}$$

where $\hat{R}(k)$ denotes the sample autocorrelation of the length-N data sequence y_n:

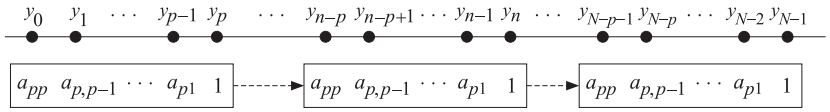
$$\hat{R}(k) = \hat{R}(-k) = \sum_{n=0}^{N-1-k} y_{n+k}y_n, \quad 0 \le k \le N-1$$

where the usual normalization factor $1/N$ has been ignored. This equation is identical to Eq. (12.12.3) with R replaced by $\hat{R}$. Thus, the minimization of the time-average index (12.12.5) with respect to the prediction coefficients will lead exactly to the *same set* of normal equations (12.3.7) with R replaced by $\hat{R}$. The *positive definiteness* of the sample autocorrelation matrix also guarantees that the resulting prediction-error filter will be *minimum phase*, and thus also that all reflection coefficients will have magnitude less than one.

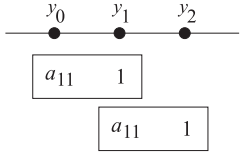
2. The *covariance method* replaces Eq. (12.12.1) by the time average

$$\mathcal{E} = \sum_{n=p}^{N-1} e_p^+(n)^2 = \min \tag{12.12.6}$$

where the summation in n is such that the filter *does not* run off the ends of the data block, as shown below:



To explain the method and to see its potential problems with stability, consider a simple example of a length-three sequence and a first-order predictor:



$$\mathcal{E} = \sum_{n=1}^{2} e_1^+(n)^2 = e_1^+(1)^2 + e_1^+(2)^2 = (y_1 + a_{11}y_0)^2 + (y_2 + a_{11}y_1)^2$$

Differentiating with respect to a_{11} and setting the derivative to zero gives

$$(y_1 + a_{11}y_0)y_0 + (y_2 + a_{11}y_1)y_1 = 0$$

$$a_{11} = -\frac{y_1y_0 + y_2y_1}{y_0^2 + y_1^2}$$

Note that the denominator does not depend on the variable y_2 and therefore it is possible, if y_2 is large enough, for a_{11} to have magnitude greater than one, making the prediction-error filter nonminimal phase. Although this potential stability problem exists, this method has been used with good success in speech processing, with few, if any, such stability problems. The autocorrelation method is sometimes preferred in

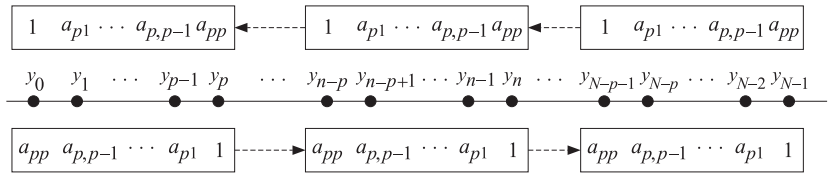
speech processing because the resulting normal equations have a Toeplitz structure and their solution can be obtained efficiently using Levinson's algorithm. However, similar ways of solving the covariance equations have been developed recently that are just as efficient [1013].

3. Although the autocorrelation method is implemented efficiently, and the resulting prediction-error filter is guaranteed to be minimum phase, it suffers from the effect of windowing the data sequence y_n, by padding it with zeros to the left and to the right. This reduces the accuracy of the method somewhat, especially when the data record N is short. In this case, the effect of windowing is felt more strongly. The proper way to extend the sequence y_n, if it must be extended, is a way compatible with the signal model generating this sequence. Since we are trying to determine this model, the fairest way of proceeding is to try to use the available data block in a way which is maximally noncommittal as to what the sequence is like beyond the ends of the block.

Burg's method, also known as the *maximum entropy* method (MEM), arose from the desire on the one hand not to run off the ends of the data, and, on the other, to always result in a minimum-phase filter. Burg's minimization criterion is to minimize the sum-squared of both the forward and the backward prediction errors:

$$\mathcal{E} = \sum_{n=p}^{N-1} \left[e_p^+(n)^2 + e_p^-(n)^2 \right] = \min \tag{12.12.7}$$

where the summation range is the same as in the covariance method, but with both the forward and the reversed filters running over the data, as shown:



If the minimization is performed with respect to the coefficients a_{pi}, it is still possible for the resulting prediction-error filter not to be minimum phase. Instead, Burg suggests an *iterative procedure*: Suppose that the prediction-error filter $[1, a_{p-1,1}, a_{p-1,2}, \ldots, a_{p-1,p-1}]$ of order $(p-1)$ has already been determined. Then, to determine the prediction-error filter of order p, one needs to know the reflection coefficient γ_p and to apply the Levinson recursion:

$$\begin{bmatrix} 1 \\ a_{p1} \\ a_{p2} \\ \vdots \\ a_{p,p-1} \\ a_{pp} \end{bmatrix} = \begin{bmatrix} 1 \\ a_{p-1,1} \\ a_{p-1,2} \\ \vdots \\ a_{p-1,p-1} \\ 0 \end{bmatrix} - \gamma_p \begin{bmatrix} 0 \\ a_{p-1,p-1} \\ a_{p-1,p-2} \\ \vdots \\ a_{p-1,1} \\ 1 \end{bmatrix} \tag{12.12.8}$$

To guarantee the minimum-phase property, the reflection coefficient γ_p must have magnitude less than one. The best choice for γ_p is that which minimizes the performance index (12.12.7). Differentiating with respect to γ_p and setting the derivative to zero we find

$$\frac{\partial \mathcal{E}}{\partial \gamma_p} = 2 \sum_{n=p}^{N-1} \left[e_p^+(n) \frac{\partial e_p^+(n)}{\partial \gamma_p} + e_p^-(n) \frac{\partial e_p^-(n)}{\partial \gamma_p} \right] = 0$$

Using the lattice relationships

$$\begin{aligned} e_p^+(n) &= e_{p-1}^+(n) - \gamma_p e_{p-1}^-(n-1) \\ e_p^-(n) &= e_{p-1}^-(n-1) - \gamma_p e_{p-1}^+(n) \end{aligned} \tag{12.12.9}$$

both valid for $p \le n \le N-1$ if the filter is not to run off the ends of the data, we find the condition

$$\sum_{n=p}^{N-1} \left[e_p^+(n) e_{p-1}^-(n-1) + e_p^-(n) e_{p-1}^+(n) \right] = 0, \quad \text{or,}$$

$$\sum_{n=p}^{N-1} \left[\left(e_{p-1}^+(n) - \gamma_p e_{p-1}^-(n-1) \right) e_{p-1}^-(n-1) + \left(e_{p-1}^-(n-1) - \gamma_p e_{p-1}^+(n) \right) e_{p-1}^+(n) \right] = 0$$

which can be solved for γ_p to give

$$\gamma_p = \frac{2 \sum_{n=p}^{N-1} e_{p-1}^+(n) e_{p-1}^-(n-1)}{\sum_{n=p}^{N-1} \left[e_{p-1}^+(n)^2 + e_{p-1}^-(n-1)^2 \right]} \tag{12.12.10}$$

This expression for γ_p is of the form

$$\gamma_p = \frac{2\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}|^2 + |\mathbf{b}|^2}$$

where $\mathbf{a}$ and $\mathbf{b}$ are vectors. Using the Schwarz inequality, it is easily verified that γ_p has magnitude less than one. Equations (12.12.8) through (12.12.10) define Burg's method. The computational steps are summarized below:

0. Initialize in order as follows:

$$e_0^+(n) = e_0^-(n) = y_n, \text{ for } 0 \le n \le N-1, \quad \text{and } A_0(z) = 1, \; E_0 = \frac{1}{N} \sum_{n=0}^{N-1} y_n^2$$

1. At stage $(p-1)$, we have available the quantities:

$$A_{p-1}(z), \; E_{p-1}, \text{ and } e_{p-1}^{\pm}(n), \text{ for } p-1 \le n \le N-1$$

2. Using Eq. (12.12.10), compute the reflection coefficient γ_p.
3. Using (12.12.8), compute $A_p(z)$.
4. Using (12.12.9), compute $e_p^{\pm}(n)$, for $p \le n \le N-1$.
5. Update the mean-square error by $E_p = (1 - \gamma_p^2) E_{p-1}$.
6. Go to stage p.

The function **burg** is an implementation of this method. The inputs to the function are the vector of data samples $\{y_0, y_1, \ldots, y_{N-1}\}$ and the desired final order M of the predictor. The outputs are all the prediction-error filters of order up to M, arranged as usual into the lower triangular matrix L, and the corresponding mean-square prediction errors $\{E_0, E_1, \ldots, E_M\}$.

Example 12.12.1: The length-six block of data

$$y_n = [4.684,\ 7.247,\ 8.423,\ 8.650,\ 8.640,\ 8.392]$$

for $n = 0, 1, 2, 3, 4, 5$, is known to have been generated by sending zero-mean, unit-variance, white-noise ϵ_n through the difference equation

$$y_n - 1.70y_{n-1} + 0.72y_{n-2} = \epsilon_n$$

Thus, the theoretical prediction-error filter and mean-square error are $A_2(z) = 1 - 1.70z^{-1} + 0.72z^{-2}$ and $E_2 = 1$. Using Burg's method, extract the model parameters for a second-order model. The reader is urged to go through the algorithm by hand. Sending the above six y_n samples through the function **burg**, we find the first- and second-order prediction-error filters and the corresponding errors:

$$A_1(z) = 1 - 0.987z^{-1}, \quad E_1 = 1.529$$
$$A_2(z) = 1 - 1.757z^{-1} + 0.779z^{-2}, \quad E_2 = 0.60$$

We note that the theoretical first-order filter obtained from $A_2(z) = 1 - 1.70z^{-1} + 0.72z^{-2}$ via the backward Levinson recursion is $A_1(z) = 1 - 0.9884\,z^{-1}$. □

The resulting set of LPC model parameters, from any of the above analysis methods, can be used in a number of ways as suggested in Sec. 1.13. One of the most successful applications has been to the analysis and synthesis of speech [920,1019–1027]. Each frame of speech, of duration of the order of 20 msec, is subjected to the Yule-Walker analysis method to extract the corresponding set of model parameters. The order M of the predictor is typically 10–15. Pitch and voiced/unvoiced information are also extracted. The resulting set of parameters represents that speech segment.

To synthesize the segment, the set of model parameters are recalled from memory and used in the synthesizer to drive the synthesis filter. The latter is commonly realized as a *lattice filter*. Lattice realizations are preferred because they are much better well-behaved under quantization of their coefficients (i.e., the reflection coefficients) than the direct-form realizations [920,1023,1024]. A typical speech analysis and synthesis system is shown in Fig. 12.12.1.

Linear predictive modeling techniques have also been applied to EEG signal processing in order to model EEG spectra, to *classify* EEGs automatically, to detect EEG *transients* that might have diagnostic significance, and to *predict* the onset of epileptic seizures [1028–1035].

LPC methods have been applied successfully to *signal classification* problems such as speech recognition [1022,1036–1041] or the automatic classification of EEGs [1032]. *Distance measures* between two sets of model parameters extracted from two signal frames can be used as measures of similarity between the frames. Itakura's *LPC distance*

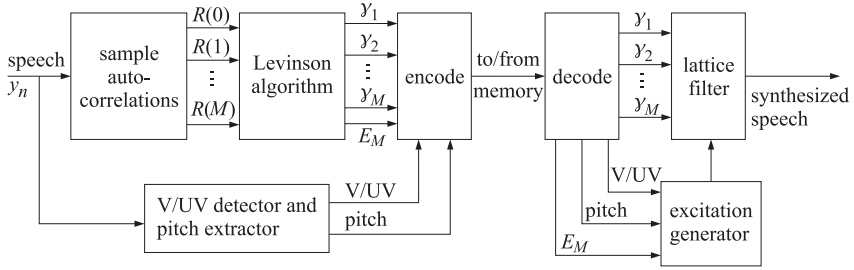


Fig. 12.12.1 LPC analysis and synthesis of speech.

measure can be introduced as follows: Consider two *autoregressive* signal sequences, the test sequence $y_T(n)$ to be compared against the reference sequence $y_R(n)$. Let $A_T(z)$ and $A_R(z)$ be the two whitening filters, both of order M. The two signal models are

$$\epsilon_T(n) \longrightarrow \boxed{1/A_T(z)} \longrightarrow y_T(n) \qquad \epsilon_R(n) \longrightarrow \boxed{1/A_R(z)} \longrightarrow y_R(n)$$

Now, suppose the sequence to be tested, $y_T(n)$, is filtered through the whitening filter of the reference signal

$$y_T(n) \longrightarrow \boxed{A_R(z)} \longrightarrow e_T(n)$$

resulting in the output signal $e_T(n)$. The mean output power is easily expressed as

$$E[e_T(n)^2] = \mathbf{a}_R^\dagger R_T \mathbf{a}_R = \int_{-\pi}^{\pi} S_{e_T e_T}(\omega)\,\frac{d\omega}{2\pi} = \int_{-\pi}^{\pi} |A_R(\omega)|^2 S_{y_T y_T}(\omega)\,\frac{d\omega}{2\pi}$$
$$= \int_{-\pi}^{\pi} |A_R(\omega)|^2 \frac{\sigma_{\epsilon_T}^2}{|A_T(\omega)|^2}\,\frac{d\omega}{2\pi}$$

where R_T is the autocorrelation matrix of $y_T(n)$. On the other hand, if $y_T(n)$ is filtered through its own whitening filter, it will produce $\epsilon_T(n)$. Thus, in this case

$$\sigma_{\epsilon_T}^2 = E[\epsilon_T(n)^2] = \mathbf{a}_T^\dagger R_T \mathbf{a}_T$$

It follows that

$$\frac{E[e_T(n)^2]}{E[\epsilon_T(n)^2]} = \frac{\mathbf{a}_R^\dagger R_T \mathbf{a}_R}{\mathbf{a}_T^\dagger R_T \mathbf{a}_T} = \int_{-\pi}^{\pi} \frac{|A_R(\omega)|^2}{|A_T(\omega)|^2}\,\frac{d\omega}{2\pi} \tag{12.12.11}$$

The log of this quantity is Itakura's LPC distance measure

$$d(\mathbf{a}_T, \mathbf{a}_R) = \log\left(\frac{E[e_T(n)^2]}{E[\epsilon_T(n)^2]}\right) = \log\left(\frac{\mathbf{a}_R^\dagger R_T \mathbf{a}_R}{\mathbf{a}_T^\dagger R_T \mathbf{a}_T}\right) = \log\left[\int_{-\pi}^{\pi} \frac{|A_R(\omega)|^2}{|A_T(\omega)|^2}\,\frac{d\omega}{2\pi}\right]$$

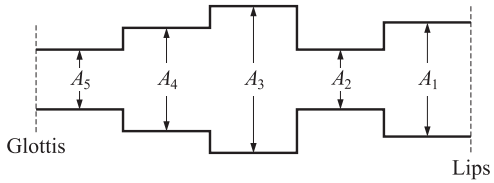
In practice, the quantities $\mathbf{a}_T$, R_T, and $\mathbf{a}_R$ are extracted from a frame of $y_T(n)$ and a frame of $y_R(n)$. If the model parameters are equal, the distance is zero. This distance

measure effectively provides a comparison between the two spectra of the processes y_T and y_R, but instead of comparing them directly, a prewhitening of $y_T(n)$ is carried out by sending it through the whitening filter of the other signal. If the two spectra are close, the filtered signal $e_T(n)$ will be close to white—that is, with a spectrum close to being flat; a measure of this flatness is precisely the above integrated spectrum of Eq. (12.12.11).

12.13 Dynamic Predictive Deconvolution—Waves in Layered Media

The analysis and synthesis lattice filters, implemented via the Levinson recursion, were obtained within the context of linear prediction. Here, we would like to point out the remarkable fact that the same analysis and synthesis lattice structures also occur naturally in the problem of wave propagation in layered media [920–925,974,976,1010,1019,1042–1059]. This is perhaps the reason behind the great success of linear prediction methods in speech and seismic signal processing. In fact, historically many linear prediction techniques were originally developed within the context of these two application areas.

In speech, the vocal tract is modeled as an acoustic tube of varying cross-sectional area. It can be approximated by the piece-wise constant area approximation shown below:

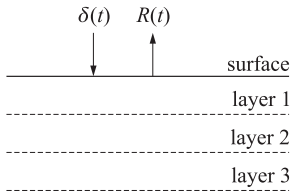


The acoustic impedance of a sound wave varies inversely with the tube area

$$Z = \frac{\rho c}{A}$$

where ρ, c, A are the air density, speed of sound, and tube area, respectively. Therefore, as the sound wave propagates from the glottis to the lips, it will suffer reflections every time it encounters an interface; that is, every time it enters a tube segment of different diameter. Multiple reflections will be set up within each segment and the tube will *reverberate* in a complicated manner depending on the number of segments and the diameter of each segment. By measuring the speech wave that eventually comes out of the lips, it is possible to remove, or deconvolve, the reverberatory effects of the tube and, in the process, extract the *tube parameters*, such as the areas of the segments or, equivalently, the reflection coefficients at the interfaces. During speech, the configuration of the vocal tract tube changes continuously. But being a mechanical system, it does so fairly slowly, and for short periods of time (of the order of 20–30 msec) it may be assumed to maintain a fixed configuration. From each such short segment of speech, a set of configuration parameters (e.g., reflection coefficients) may be extracted. This set may be used to synthesize the speech segment.

The seismic problem is somewhat different. Here it is not the transmitted wave that is experimentally accessible, but rather the overall reflected wave:

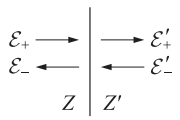


An impulsive input to the earth, such as a dynamite explosion near the surface, will set up seismic elastic waves propagating downwards. As the various earth layers are encountered, reflections will take place. Eventually each layer will be reverberating and an overall reflected wave will be measured at the surface. On the basis of this reflected wave, the layered structure (i.e., reflection coefficients, impedances, etc.) must be extracted by deconvolution techniques. These are essentially identical to the linear prediction methods.

In addition to geophysical and speech applications, this wave problem and the associated *inverse* problem of extracting the structure of the medium from the observed (reflected or transmitted) response have a number of other applications. Examples include the probing of dielectric materials by electromagnetic waves, the study of the optical properties of thin films, the probing of tissues by ultrasound, and the design of broadband terminations of transmission lines. The mathematical analysis of such wave propagation problems has been done more or less independently in each of these application areas, and is well known dating back to the time of Stokes.

In this type of wave propagation problem there are always *two* associated propagating field quantities, the ratio of which is constant and equal to the corresponding *characteristic impedance* of the propagation medium. Examples of these include the electric and magnetic fields in the case of EM waves, the air pressure and particle volume velocity for sound waves, the stress and particle displacement for seismic waves, and the voltage and current waves in the case of TEM transmission lines.

As a concrete example, we have chosen to present in some detail the case of EM waves propagating in lossless dielectrics. The simplest and most basic scattering problem arises when there is a single interface separating two semi-infinite dielectrics of characteristic impedances Z and Z', as shown



where $\mathcal{E}_+$ and $\mathcal{E}_-$ are the right and left moving electric fields in medium Z, and $\mathcal{E}'_+$ and $\mathcal{E}'_-$ are those in medium Z'. The arrows indicate the directions of propagation, the fields are perpendicular to these directions. Matching the boundary conditions (i.e., continuity

of the tangential fields at the interface), gives the two equations:

$$\mathcal{E}_+ + \mathcal{E}_- = \mathcal{E}'_+ + \mathcal{E}'_- \qquad \text{(continuity of electric field)}$$

$$\frac{1}{Z}(\mathcal{E}_+ - \mathcal{E}_-) = \frac{1}{Z'}(\mathcal{E}'_+ - \mathcal{E}'_-) \qquad \text{(continuity of magnetic field)}$$

Introducing the reflection and transmission coefficients,

$$\rho = \frac{Z' - Z}{Z' + Z}, \quad \tau = 1 + \rho, \quad \rho' = -\rho, \quad \tau' = 1 + \rho' = 1 - \rho \tag{12.13.1}$$

the above equations can be written in a *transmission matrix* form

$$\begin{bmatrix} \mathcal{E}_+ \\ \mathcal{E}_- \end{bmatrix} = \frac{1}{\tau}\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}\begin{bmatrix} \mathcal{E}'_+ \\ \mathcal{E}'_- \end{bmatrix} \tag{12.13.2}$$

The flow of energy carried by these waves is given by the Poynting vector

$$P = \frac{1}{2}\,\mathrm{Re}\left[(\mathcal{E}_+ + \mathcal{E}_-)^* \frac{1}{Z}(\mathcal{E}_+ - \mathcal{E}_-)\right] = \frac{1}{2Z}(\mathcal{E}_+^*\mathcal{E}_+ - \mathcal{E}_-^*\mathcal{E}_-) \tag{12.13.3}$$

One consequence of the above matching conditions is that the total energy flow to the right is preserved *across* the interface; that is,

$$\frac{1}{2Z}(\mathcal{E}_+^*\mathcal{E}_+ - \mathcal{E}_-^*\mathcal{E}_-) = \frac{1}{2Z'}(\mathcal{E}_+'^*\mathcal{E}'_+ - \mathcal{E}_-'^*\mathcal{E}'_-) \tag{12.13.4}$$

It proves convenient to absorb the factors $1/2Z$ and $1/2Z'$ into the definitions for the fields by renormalizing them as follows:

$$\begin{bmatrix} E_+ \\ E_- \end{bmatrix} = \frac{1}{\sqrt{2Z}}\begin{bmatrix} \mathcal{E}_+ \\ \mathcal{E}_- \end{bmatrix}, \quad \begin{bmatrix} E'_+ \\ E'_- \end{bmatrix} = \frac{1}{\sqrt{2Z'}}\begin{bmatrix} \mathcal{E}'_+ \\ \mathcal{E}'_- \end{bmatrix}$$

Then, Eq. (12.13.4) reads

$$E_+^* E_+ - E_-^* E_- = E_+'^* E'_+ - E_-'^* E'_- \tag{12.13.5}$$

and the matching equations (12.13.2) can be written in the normalized form

$$\begin{bmatrix} E_+ \\ E_- \end{bmatrix} = \frac{1}{t}\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}\begin{bmatrix} E'_+ \\ E'_- \end{bmatrix}, \quad t = \sqrt{1-\rho^2} = \sqrt{\tau\tau'} \tag{12.13.6}$$

They may also be written in a *scattering matrix* form that relates the *outgoing* fields to the *incoming* ones, as follows:

$$\begin{bmatrix} E'_+ \\ E_- \end{bmatrix} = \begin{bmatrix} t & \rho' \\ \rho & t \end{bmatrix}\begin{bmatrix} E_+ \\ E'_- \end{bmatrix} = S\begin{bmatrix} E_+ \\ E'_- \end{bmatrix} \qquad \begin{matrix} E_+ \longrightarrow \\ E_- \longleftarrow \end{matrix} \Bigg| \begin{matrix} \longrightarrow E'_+ \\ \longleftarrow E'_- \end{matrix} \tag{12.13.7}$$

This is the most elementary scattering matrix of all, and ρ and t are the most elementary reflection and transmission responses. From these, the reflection and transmission response of more complicated structures can be built up. In the more general

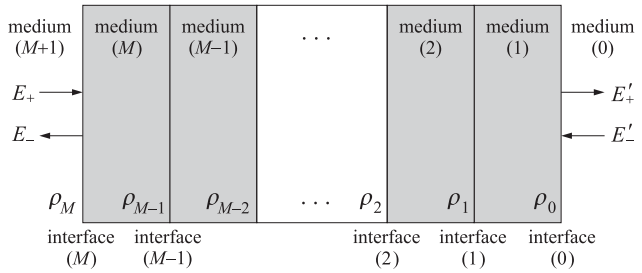


Fig. 12.13.1 Layered structure.

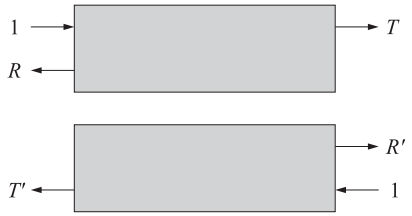


Fig. 12.13.2 Reflection and transmission responses.

case, we have a dielectric structure consisting of M slabs stacked together as shown in Fig. 12.13.1.

The media to the left and right in the figure are assumed to be semi-infinite. The reflection and transmission responses (from the left, or from the right) of the structure are defined as the responses of the structure to an impulse (incident from the left, or from the right) as shown in Fig. 12.13.2.

The corresponding scattering matrix is defined as

$$S = \begin{bmatrix} T & R' \\ R & T' \end{bmatrix}$$

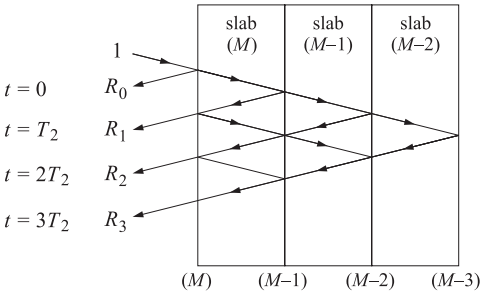
and by linear superposition, the relationship between arbitrary incoming and outgoing waves is

$$\begin{bmatrix} E'_+ \\ E_- \end{bmatrix} = \begin{bmatrix} T & R' \\ R & T' \end{bmatrix}\begin{bmatrix} E_+ \\ E'_- \end{bmatrix}$$

The *inverse scattering problem* that we pose is how to extract the detailed properties of the layered structure, such as the reflection coefficients $\rho_0, \rho_1, \ldots, \rho_M$ from the knowledge of the scattering matrix S; that is, from observations of the reflection response R or the transmission response T.

Without loss of generality, we may assume the M slabs have *equal travel time*. We denote the common one-way travel time by T_1 and the two-way travel time by $T_2 = 2T_1$.

As an impulse $\delta(t)$ is incident from the left on interface M, there will be immediately a reflected wave and a transmitted wave into medium M. When the latter reaches interface $M-1$, part of it will be transmitted into medium $M-1$, and part will be reflected back towards interface M where it will be partially rereflected towards $M-1$ and partially transmitted to the left into medium $M+1$, thus contributing towards the overall reflection response. Since the wave had to travel to interface $M-1$ and back, this latter contribution will occur at time T_2. Similarly, another wave will return back to interface M due to reflection from the second interface $M-2$; this wave will return $2T_2$ seconds later and will add to the contribution from the zig-zag path within medium M which is also returning at $2T_2$, and so on. The timing diagram below shows all the possible return paths up to time $t = 3T_2$, during which the original impulse can only travel as far as interface $M-3$:



When we add the contributions of all the returned waves we see that the reflection response will be a linear superposition of returned impulses

$$R(t) = \sum_{k=0}^{\infty} R_k \delta(t - kT_2)$$

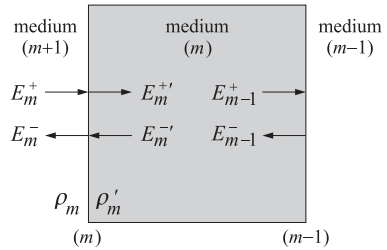
It has a Fourier transform expressible more conveniently as the z-transform

$$R(z) = \sum_{k=0}^{\infty} R_k z^{-k}, \quad z = e^{j\omega T_2}, \quad \text{(here, } \omega \text{ is in rads/sec)}$$

We observe that R is *periodic* in frequency ω with period $2\pi/T_2$, which plays a role analogous to the sampling frequency in a sample-data system. Therefore, it is enough to specify R within the *Nyquist interval* $[-\pi/T_2, \pi/T_2]$.

Next, we develop the *lattice recursions* that facilitate the solution of the direct and the inverse scattering problems. Consider the mth slab and let $E_m^{\pm}$ be the right/left moving waves incident on the left side of the mth interface. To relate them to the same quantities $E_{m-1}^{\pm}$ incident on the left side of the $(m-1)$st interface, first we use the *matching* equations to "pass" to the other side of the mth interface and into the mth slab, and then we *propagate* these quantities to reach the left side of the $(m-1)$st

interface. This is shown below.



The matching equations are:

$$\begin{bmatrix} E_m^+ \\ E_m^- \end{bmatrix} = \frac{1}{t_m} \begin{bmatrix} 1 & \rho_m \\ \rho_m & 1 \end{bmatrix} \begin{bmatrix} E_m^{+\prime} \\ E_m^{-\prime} \end{bmatrix}, \quad t_m = (1 - \rho_m^2)^{1/2} \tag{12.13.8}$$

Since the left-moving wave $E_m^{-\prime}$ is the delayed replica of E_{m-1}^- by T_1 seconds, and $E_m^{+\prime}$ is the advanced replica of E_{m-1}^+ by T_1 seconds, it follows that

$$E_m^{+\prime} = z^{1/2} E_{m-1}^+, \quad E_m^{-\prime} = z^{-1/2} E_{m-1}^-$$

or, in matrix form

$$\begin{bmatrix} E_m^{+\prime} \\ E_m^{-\prime} \end{bmatrix} = \begin{bmatrix} z^{1/2} & 0 \\ 0 & z^{-1/2} \end{bmatrix} \begin{bmatrix} E_{m-1}^+ \\ E_{m-1}^- \end{bmatrix} \tag{12.13.9}$$

where the variable z^{-1} was defined above and represents the two-way travel time delay, while $z^{-1/2}$ represents the one-way travel time delay. Combining the matching and propagation equations (12.13.8) and (12.13.9), we obtain the desired relationship between $E_m^{\pm}$ and $E_{m-1}^{\pm}$:

$$\begin{bmatrix} E_m^+ \\ E_m^- \end{bmatrix} = \frac{z^{1/2}}{t_m} \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \begin{bmatrix} E_{m-1}^+ \\ E_{m-1}^- \end{bmatrix} \tag{12.13.10}$$

Or, written in a convenient vector notation

$$E_m(z) = \psi_m(z) E_{m-1}(z) \tag{12.13.11}$$

where we defined

$$E_m(z) = \begin{bmatrix} E_m^+(z) \\ E_m^-(z) \end{bmatrix}, \quad \psi_m(z) = \frac{z^{1/2}}{t_m} \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \tag{12.13.12}$$

The "match-and-propagate" transition matrix $\psi_m(z)$ has two interesting properties; namely, defining $\bar{\psi}_m(z) = \psi_m(z^{-1})$

$$\bar{\psi}_m(z)^T J_3 \psi_m(z) = J_3, \quad J_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{12.13.13}$$

$$\bar{\psi}_m(z) = J_1 \psi_m(z) J_1, \quad J_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{12.13.14}$$

where J_1, J_3 are recognized as two of the three Pauli spin matrices. From Eq. (12.3.13), we have with $\bar{E}_m^{\pm}(z) = E_m^{\pm}(z^{-1})$:

$$\begin{aligned}\bar{E}_m^+ E_m^+ - \bar{E}_m^- E_m^- &= \bar{E}_m^T J_3 E_m = \bar{E}_{m-1}^T \bar{\psi}_m^T J_3 \psi_m E_{m-1} = \bar{E}_{m-1}^T J_3 \bar{E}_{m-1} \\ &= \bar{E}_{m-1}^+ E_{m-1}^+ - \bar{E}_{m-1}^- E_{m-1}^-\end{aligned} \tag{12.13.15}$$

which is equivalent to *energy conservation*, according to Eq. (12.13.5). The second property, Eq. (12.13.14), expresses *time-reversal invariance* and allows the construction of a second, linearly independent, solution of the recursive equations (12.13.11), Using the property $J_1^2 = I$, we have

$$\hat{E}_m = J_1 \bar{E}_m = \begin{bmatrix} \bar{E}_m^- \\ \bar{E}_m^+ \end{bmatrix} = J_1 \bar{\psi}_m \bar{E}_{m-1} = J_1 \bar{\psi}_m J_1 J_1 \bar{E}_{m-1} = \psi_m \hat{E}_{m-1} \tag{12.13.16}$$

The recursions (12.13.11) may be iterated now down to $m = 0$. By an additional boundary match, we may pass to the right side of interface $m = 0$:

$$E_m = \psi_m \psi_{m-1} \cdots \psi_1 E_0 = \psi_m \psi_{m-1} \cdots \psi_1 \psi_0 E_0'$$

where we defined ψ_0 by

$$\psi_0 = \frac{1}{t_0} \begin{bmatrix} 1 & \rho_0 \\ \rho_0 & 1 \end{bmatrix}$$

or, more explicitly

$$\begin{bmatrix} E_m^+ \\ E_m^- \end{bmatrix} = \frac{z^{m/2}}{t_m t_{m-1} \cdots t_1 t_0} \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \cdots \begin{bmatrix} 1 & \rho_1 z^{-1} \\ \rho_1 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & \rho_0 \\ \rho_0 & 1 \end{bmatrix} \begin{bmatrix} E_0^{+\prime} \\ E_0^{-\prime} \end{bmatrix} \tag{12.13.17}$$

To deal with this product of matrices, we define

$$\begin{bmatrix} A_m & C_m \\ B_m & D_m \end{bmatrix} = \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \cdots \begin{bmatrix} 1 & \rho_1 z^{-1} \\ \rho_1 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & \rho_0 \\ \rho_0 & 1 \end{bmatrix} \tag{12.13.18}$$

where A_m, C_m, B_m, D_m are *polynomials of degree* m in the variable z^{-1}. The energy conservation and time-reversal invariance properties of the ψ_m matrices imply similar properties for these polynomials. Writing Eq. (12.13.18) in terms of the ψ_ms, we have

$$\begin{bmatrix} A_m & C_m \\ B_m & D_m \end{bmatrix} = z^{-m/2} \sigma_m \psi_m \psi_{m-1} \cdots \psi_1 \psi_0$$

where we defined the quantity

$$\sigma_m = t_m t_{m-1} \cdots t_1 t_0 = \prod_{i=0}^{m} (1 - \rho_i^2)^{1/2} \tag{12.13.19}$$

Property (12.13.13) implies the same for the above product of matrices; that is, with $\bar{A}_m(z) = A_m(z^{-1})$, etc.,

$$\begin{bmatrix} \bar{A}_m & \bar{C}_m \\ \bar{B}_m & \bar{D}_m \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} A_m & C_m \\ B_m & D_m \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \sigma_m^2$$

which implies that the quantity $\bar{A}_m(z) A_m(z) - \bar{B}_m(z) B_m(z)$ is independent of z:

$$\bar{A}_m(z) A_m(z) - \bar{B}_m(z) B_m(z) = \sigma_m^2 \tag{12.13.20}$$

Property (12.13.14) implies that C_m and D_m are the reverse polynomials B_m^R and A_m^R, respectively; indeed

$$\begin{aligned}\begin{bmatrix} A_m^R & C_m^R \\ B_m^R & D_m^R \end{bmatrix} &= z^{-m} \begin{bmatrix} \bar{A}_m & \bar{C}_m \\ \bar{B}_m & \bar{D}_m \end{bmatrix} = z^{-m} z^{m/2} \sigma_m \bar{\psi}_m \cdots \bar{\psi}_1 \bar{\psi}_0 \\ &= z^{-m/2} \sigma_m J_1 (\psi_m \cdots \psi_0) J_1 = J_1 \begin{bmatrix} A_m & C_m \\ B_m & D_m \end{bmatrix} J_1 \\ &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} A_m & C_m \\ B_m & D_m \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} D_m & B_m \\ C_m & A_m \end{bmatrix}\end{aligned} \tag{12.13.21}$$

from which it follows that $C_m(z) = B_m^R(z)$ and $D_m(z) = A_m^R(z)$. The definition (12.13.18) implies also the recursion

$$\begin{bmatrix} A_m & B_m^R \\ B_m & A_m^R \end{bmatrix} = \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \begin{bmatrix} A_{m-1} & B_{m-1}^R \\ B_{m-1} & A_{m-1}^R \end{bmatrix}$$

Therefore each column of the $ABCD$ matrix satisfies the same recursion. To summarize, we have

$$\begin{bmatrix} A_m(z) & B_m^R(z) \\ B_m(z) & A_m^R(z) \end{bmatrix} = \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \cdots \begin{bmatrix} 1 & \rho_1 z^{-1} \\ \rho_1 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & \rho_0 \\ \rho_0 & 1 \end{bmatrix} \tag{12.13.22}$$

with the *lattice recursion*

$$\begin{bmatrix} A_m(z) \\ B_m(z) \end{bmatrix} = \begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix} \begin{bmatrix} A_{m-1}(z) \\ B_{m-1}(z) \end{bmatrix} \tag{12.13.23}$$

and the property (12.13.20). The lattice recursion is initialized at $m = 0$ by:

$$A_0(z) = 1\,, \quad B_0(z) = \rho_0\,, \quad \text{or,} \quad \begin{bmatrix} A_0(z) & B_0^R(z) \\ B_0(z) & A_0^R(z) \end{bmatrix} = \begin{bmatrix} 1 & \rho_0 \\ \rho_0 & 1 \end{bmatrix} \tag{12.13.24}$$

Furthermore, it follows from the lattice recursion (12.13.23) that the reflection coefficients ρ_m always appear in the *first* and *last* coefficients of the polynomials $A_m(z)$ and $B_m(z)$, as follows

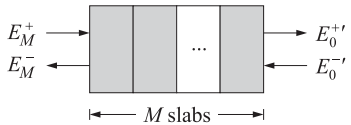
$$a_m(0) = 1\,, \quad a_m(m) = \rho_0 \rho_m\,, \quad b_m(0) = \rho_m\,, \quad b_m(m) = \rho_0 \tag{12.13.25}$$

Eq. (12.13.17) for the field components reads now

$$\begin{bmatrix} E_m^+ \\ E_m^- \end{bmatrix} = \frac{z^{m/2}}{\sigma_m} \begin{bmatrix} A_m & B_m^R \\ B_m & A_m^R \end{bmatrix} \begin{bmatrix} E_0^{+\prime} \\ E_0^{-\prime} \end{bmatrix}$$

Setting $m = M$, we find the relationship between the fields incident on the dielectric slab structure from the left to those incident from the right:

$$\begin{bmatrix} E_M^+ \\ E_M^- \end{bmatrix} = \frac{z^{M/2}}{\sigma_M} \begin{bmatrix} A_M & B_M^R \\ B_M & A_M^R \end{bmatrix} \begin{bmatrix} E_0^{+\prime} \\ E_0^{-\prime} \end{bmatrix} \tag{12.13.26}$$



All the multiple reflections and reverberatory effects of the structure are buried in the transition matrix

$$\begin{bmatrix} A_M & B_M^R \\ B_M & A_M^R \end{bmatrix}$$

In reference to Fig. 12.13.2, the reflection and transmission responses R, T, R', T' of the structure can be obtained from Eq. (12.13.26) by noting that

$$\begin{bmatrix} 1 \\ R \end{bmatrix} = \frac{z^{M/2}}{\sigma_M} \begin{bmatrix} A_M & B_M^R \\ B_M & A_M^R \end{bmatrix} \begin{bmatrix} T \\ 0 \end{bmatrix}, \qquad \begin{bmatrix} 0 \\ T' \end{bmatrix} = \frac{z^{M/2}}{\sigma_M} \begin{bmatrix} A_M & B_M^R \\ B_M & A_M^R \end{bmatrix} \begin{bmatrix} R' \\ 1 \end{bmatrix}$$

which may be combined into one equation:

$$\begin{bmatrix} 1 & 0 \\ R & T' \end{bmatrix} = \frac{z^{M/2}}{\sigma_M} \begin{bmatrix} A_M & B_M^R \\ B_M & A_M^R \end{bmatrix} \begin{bmatrix} T & R' \\ 0 & 1 \end{bmatrix}$$

that can be written as follows:

$$\frac{z^{M/2}}{\sigma_M} \begin{bmatrix} A_M & B_M^R \\ B_M & A_M^R \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ R & T' \end{bmatrix} \begin{bmatrix} T & R' \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 \\ R & 1 \end{bmatrix} \begin{bmatrix} T^{-1} & 0 \\ 0 & T' \end{bmatrix} \begin{bmatrix} 1 & -R' \\ 0 & 1 \end{bmatrix}$$

Solving these for the reflection and transmission responses, we find:

$$\begin{aligned} R(z) &= \frac{B_M(z)}{A_M(z)}, & T(z) &= \frac{\sigma_M z^{-M/2}}{A_M(z)} \\ R'(z) &= -\frac{B_M^R(z)}{A_M(z)}, & T'(z) &= \frac{\sigma_M z^{-M/2}}{A_M(z)} \end{aligned} \tag{12.13.27}$$

Note that $T(z) = T'(z)$. Since on physical grounds the transmission response $T(z)$ must be a *stable and causal* z-transform, it follows that necessarily the polynomial $A_M(z)$ must be a *minimum-phase polynomial*. The overall delay factor $z^{-M/2}$ in $T(z)$ is of no consequence. It just means that before anything can be transmitted through the structure, it must traverse all M slabs, each with a travel time delay of T_1 seconds; that is, with overall delay of MT_1 seconds.

Let $R_{m-1}(z)$ and $T_{m-1}(z)$ be the reflection and transmission responses based on $m-1$ layers. The addition of one more layer will change the responses to $R_m(z)$ and $T_m(z)$. Using the lattice recursions, we may derive a recursion for these responses:

$$R_m(z) = \frac{B_m(z)}{A_m(z)} = \frac{\rho_m A_{m-1}(z) + z^{-1} B_{m-1}(z)}{A_{m-1}(z) + \rho_m z^{-1} B_{m-1}(z)}$$

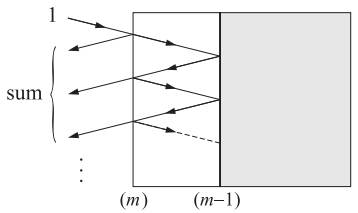
Dividing numerator and denominator by $A_{m-1}(z)$ we obtain

$$R_m(z) = \frac{\rho_m + z^{-1} R_{m-1}(z)}{1 + \rho_m z^{-1} R_{m-1}(z)} \tag{12.13.28}$$

It describes the effect of adding a layer. Expanding it in a power series, we have

$$R_m(z) = \rho_m + (1-\rho_m^2)\left[z^{-1}R_{m-1}(z)\right] - (1-\rho_m^2)\rho_m\left[z^{-1}R_{m-1}(z)\right]^2 + \cdots$$

It can be verified easily that the various terms in this sum correspond to the multiple reflections taking place within the mth layer, as shown below:



The first term in the expansion is always ρ_m; that is, $\rho_m = R_m(\infty)$. Thus, from the knowledge of $R_m(z)$ we may extract ρ_m. With ρ_m known, we may invert Eq. (12.13.28) to get $R_{m-1}(z)$ from which we can extract ρ_{m-1}; and so on, we may extract the series of reflection coefficients. The inverse of Eq. (12.13.28), which describes the effect of removing a layer, is

$$R_{m-1}(z) = z\,\frac{R_m(z) - \rho_m}{1 - \rho_m R_m(z)} \tag{12.13.29}$$

Up to a difference in the sign of ρ_m, this is recognized as the Schur recursion (12.10.25). It provides a nice physical interpretation of that recursion; namely, the Schur functions represent the overall reflection responses at the successive layer interfaces, which on physical grounds must be stable, causal, and bounded $|R_m(z)| \le 1$ for all z in their region of convergence that includes, at least, the unit circle and all the points outside it. We may also derive a recursion for the transmission responses, which requires the simultaneous recursion of $R_m(z)$:

$$T_m(z) = \frac{t_m z^{-1/2} T_{m-1}(z)}{1 + \rho_m z^{-1} R_{m-1}(z)}, \quad T_{m-1}(z) = z^{1/2}\,\frac{t_m T_m(z)}{1 - \rho_m R_m(z)} \tag{12.13.30}$$

The dynamic predictive deconvolution method is an alternative method of extracting the sequence of reflection coefficients and is discussed below.

The equations (12.13.27) for the scattering responses R, T, R', T' imply the *unitarity* of the scattering matrix S given by

$$S = \begin{bmatrix} T & R' \\ R & T' \end{bmatrix}$$

that is,

$$\bar{S}(z)^T S(z) = S(z^{-1})^T S(z) = I \tag{12.13.31}$$

where I is the 2×2 unit matrix. On the unit circle $z = e^{j\omega T_2}$ the scattering matrix becomes a unitary matrix: $S(\omega)^\dagger S(\omega) = I$. Component-wise, Eq. (12.13.31) becomes

$$\bar{T}T + \bar{R}R = \bar{T}'T' + \bar{R}'R' = 1, \quad \bar{T}R' + \bar{R}T' = 0 \tag{12.13.32}$$

Robinson and Treitel's *dynamic predictive deconvolution method* [974] of solving the inverse scattering problem is based on the above unitarity equation. In the inverse problem, it is required to extract the set of reflection coefficients from measurements of either the reflection response R or the transmission response T. In speech processing it is the transmission response that is available. In geophysical applications, or in studying the reflectivity properties of thin films, it is the reflection response that is available. The problem of designing terminations of transmission lines also falls in the latter category. In this case, an appropriate termination is desired that must have a specified reflection response $R(z)$; for example, to be reflectionless over a wide band of frequencies about some operating frequency.

The solution of both types of problems follows the same steps. First, from the knowledge of the reflection response $R(z)$, or the transmission response $T(z)$, the *spectral function* of the structure is defined:

$$\Phi(z) = 1 - R(z)\bar{R}(z) = T(z)\bar{T}(z) = \frac{\sigma_M^2}{A_M(z)\bar{A}_M(z)} \tag{12.13.33}$$

This is recognized as the *power spectrum* of the transmission response, and it is of the autoregressive type. Thus, linear prediction methods can be used in the solution.

In the time domain, the autocorrelation lags $\phi(k)$ of the spectral function are obtained from the sample autocorrelations of the reflection sequence, or the transmission sequence:

$$\phi(k) = \delta(k) - C(k) = D(k) \tag{12.13.34}$$

where $C(k)$ and $D(k)$ are the sample autocorrelations of the reflection and transmission time responses:

$$C(k) = \sum_n R(n+k)R(n)\,, \quad D(k) = \sum_n T(n+k)T(n) \tag{12.13.35}$$

In practice, only a finite record of the reflection (or transmission) sequence will be available, say $\{R(0), R(1), \ldots, R(N-1)\}$. Then, an approximation to $C(k)$ must be used, as follows:

$$C(k) = \sum_{n=0}^{N-1-k} R(n+k)R(n)\,, \quad k = 0, 1, \ldots, M \tag{12.13.36}$$

The polynomial $A_M(z)$ may be recovered from the knowledge of the first M lags of the spectral function; that is, $\{\phi(0), \phi(1), \ldots, \phi(M)\}$. The determining equations for the coefficients of $A_M(z)$ are precisely the normal equations of linear prediction. In the present context, they may be derived directly by noting that $\Phi(z)$ is a stable spectral density and is already factored into its minimum-phase factors in Eq. (12.13.33). Thus, writing

$$\Phi(z)A_M(z) = \frac{\sigma_M^2}{A_M(z^{-1})}$$

it follows that the right-hand side is expandable in positive powers of z; the negative powers of z in the left-hand side must be set equal to zero. This gives the normal

equations:

$$\begin{bmatrix} \phi(0) & \phi(1) & \phi(2) & \cdots & \phi(M) \\ \phi(1) & \phi(0) & \phi(1) & \cdots & \phi(M-1) \\ \phi(2) & \phi(1) & \phi(0) & \cdots & \phi(M-2) \\ \vdots & \vdots & \vdots & & \vdots \\ \phi(M) & \phi(M-1) & \phi(M-2) & \cdots & \phi(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_M(1) \\ a_M(2) \\ \vdots \\ a_M(M) \end{bmatrix} = \begin{bmatrix} \sigma_M^2 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{12.13.37}$$

which can be solved efficiently using Levinson's algorithm. Having obtained $A_M(z)$ and noting the $B_M(z) = A_M(z)R(z)$, the coefficients of the polynomial $B_M(z)$ may be recovered by convolution:

$$b_M(n) = \sum_{m=0}^{n} a_M(n-m)R(m)\,, \quad n = 0, 1, \ldots, M \tag{12.13.38}$$

Having obtained both $A_M(z)$ and $B_M(z)$ and noting that $\rho_M = b_M(0)$, the lattice recursion (12.13.23) may be inverted to recover the polynomials $A_{M-1}(z)$ and $B_{M-1}(z)$ as well as the next reflection coefficient $\rho_{M-1} = b_{M-1}(0)$, and so on. The inverse of the lattice recursion matrix is

$$\begin{bmatrix} 1 & \rho_m z^{-1} \\ \rho_m & z^{-1} \end{bmatrix}^{-1} = \frac{1}{1-\rho_m^2} \begin{bmatrix} 1 & -\rho_m \\ -\rho_m z & z \end{bmatrix}$$

Therefore, the *backward* recursion becomes:

$$\rho_m = b_m(0)\,, \quad \begin{bmatrix} A_{m-1}(z) \\ B_{m-1}(z) \end{bmatrix} = \frac{1}{1-\rho_m^2} \begin{bmatrix} 1 & -\rho_m \\ -\rho_m z & z \end{bmatrix} \begin{bmatrix} A_m(z) \\ B_m(z) \end{bmatrix} \tag{12.13.39}$$

In this manner, all the reflection coefficients $\{\rho_0, \rho_1, \ldots, \rho_M\}$ can be extracted. The computational algorithm is summarized as follows:

1. Measure $R(0), R(1), \ldots, R(N-1)$.
2. Select a reasonable value for the number of slabs M.
3. Compute the $M+1$ sample autocorrelation lags $C(0), C(1), \ldots, C(M)$ of the reflection response $R(n)$, using Eq. (12.13.36).
4. Compute $\phi(k) = \delta(k) - C(k)$, $k = 0, 1, \ldots, M$.
5. Using Levinson's algorithm, solve the normal equations (12.13.37) for the coefficients of $A_M(z)$.
6. Convolve $A_M(z)$ with $R(z)$ to find $B_M(z)$.
7. Compute $\rho_M = b_M(0)$ and iterate the backward recursion (12.13.39) from $m = M$ down to $m = 0$.

The function **dpd** is an implementation of the dynamic predictive deconvolution procedure. The inputs to the function are N samples of the reflection response $\{R(0), R(1), \ldots, R(N-1)\}$ and the number of layers M. The outputs are the lattice polynomials $A_i(z)$ and

$B_i(z)$, for $i = 0, 1, \ldots, M$, arranged in the two lower-triangular matrices A and B whose rows hold the coefficients of these polynomials; that is, $A(i,j) = a_i(j)$, or

$$A_i(z) = \sum_{j=0}^{i} A(i,j)\, z^{-j}$$

and similarly for $B_i(z)$. The function invokes the function **lev** to solve the normal equations (12.13.34). The *forward scattering problem* is implemented by the function **scatt**, whose inputs are the set of reflection coefficients $\{\rho_0, \rho_1, \ldots, \rho_M\}$ and whose outputs are the lattice polynomials $A_i(z)$ and $B_i(z)$, for $i = 0, 1, \ldots, M$, as well as a pre-specified number N of reflection response samples $\{R(0), R(1), \ldots, R(N-1)\}$. It utilizes the forward lattice recursion (12.13.23) to obtain the lattice polynomials, and then computes the reflection response samples by taking the inverse z-transform of Eq. (12.13.27).

Next, we present a number of deconvolution examples simulated by means of the functions **scatter** and **dpd**. In each case, we specified the five reflection coefficients of a structure consisting of four layers. Using **scatter** we generated the exact lattice polynomials whose coefficients are arranged in the matrices A and B, and also generated 16 samples of the reflection response $R(n)$, $n = 0, 1, \ldots, 15$. These 16 samples were sent through the **dpd** function to extract the lattice polynomials A and B.

The first figure of each example displays a table of the reflection response samples and the exact and extracted polynomials. Note that the first column of the matrix B is the vector of reflection coefficients, according to Eq. (12.13.25). The remaining two graphs of each example show the reflection response R in the time domain and in the frequency domain. Note that the frequency response is plotted only over one Nyquist interval $[0, 2\pi/T_2]$, and it is symmetric about the Nyquist frequency π/T_2.

Figs. 12.13.3 and 12.13.4 correspond to the case of equal reflection coefficients $\{\rho_0, \rho_1, \rho_2, \rho_3, \rho_4\} = \{0.5, 0.5, 0.5, 0.5, 0.5\}$.

In Figs. 12.13.5 and 12.13.6 the reflection coefficients have been tapered somewhat at the ends (windowed) and are $\{0.3, 0.4, 0.5, 0.4, 0.3\}$. Note the effect of tapering on the lobes of the reflection frequency response. Figs. 12.13.7 and 12.13.8 correspond to the set of reflection coefficients $\{0.1, 0.2, 0.3, 0.2, 0.1\}$. Note the broad band of frequencies about the Nyquist frequency for which there is very little reflection. In contrast, the example in Figs. 12.13.9 and 12.13.10 exhibits high reflectivity over a broad band of frequencies about the Nyquist frequency. Its set of reflection coefficients is $\{0.5, -0.5, 0.5, -0.5, 0.5\}$.

In this section we have discussed the inverse problem of unraveling the structure of a medium from the knowledge of its reflection response. The connection of the dynamic predictive deconvolution method to the conventional inverse scattering methods based on the *Gelfand-Levitan-Marchenko* approach [1054] has been discussed in [1043,1055,1056].'The lattice recursions characteristic of the wave propagation problem were derived as a direct consequence of the boundary conditions at the interfaces between media, whereas the lattice recursions of linear prediction were a direct consequence of the Gram-Schmidt orthogonalization process and the minimization of the prediction-error performance index. Is there a deeper connection between these two problems [1005–1007]? One notable result in this direction has been to show that the

$$A_{\text{exact}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & 0.2500 & 0 & 0 & 0 \\ 1.0000 & 0.5000 & 0.2500 & 0 & 0 \\ 1.0000 & 0.7500 & 0.5625 & 0.2500 & 0 \\ 1.0000 & 1.0000 & 0.9375 & 0.6250 & 0.2500 \end{bmatrix}$$

$$A_{\text{extract}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & 0.2509 & 0 & 0 & 0 \\ 1.0000 & 0.5009 & 0.2510 & 0 & 0 \\ 1.0000 & 0.7509 & 0.5638 & 0.2508 & 0 \\ 1.0000 & 1.0009 & 0.9390 & 0.6263 & 0.2504 \end{bmatrix}$$

$$B_{\text{exact}} = \begin{bmatrix} 0.5000 & 0 & 0 & 0 & 0 \\ 0.5000 & 0.5000 & 0 & 0 & 0 \\ 0.5000 & 0.6250 & 0.5000 & 0 & 0 \\ 0.5000 & 0.7500 & 0.7500 & 0.5000 & 0 \\ 0.5000 & 0.8750 & 1.0313 & 0.8750 & 0.5000 \end{bmatrix}$$

$$B_{\text{extract}} = \begin{bmatrix} 0.5010 & 0 & 0 & 0 & 0 \\ 0.5000 & 0.5010 & 0 & 0 & 0 \\ 0.5000 & 0.6255 & 0.5010 & 0 & 0 \\ 0.5000 & 0.7505 & 0.7510 & 0.5010 & 0 \\ 0.5000 & 0.8755 & 1.0323 & 0.8764 & 0.5010 \end{bmatrix}$$

k	$R(k)$
0	0.5000
1	0.3750
2	0.1875
3	0.0234
4	−0.0586
5	−0.1743
6	0.1677
7	0.0265
8	−0.0601
9	−0.0259
10	0.0238
11	0.0314
12	−0.0225
13	−0.0153
14	0.0109
15	0.0097

Fig. 12.13.3 Reflection response and lattice polynomials.

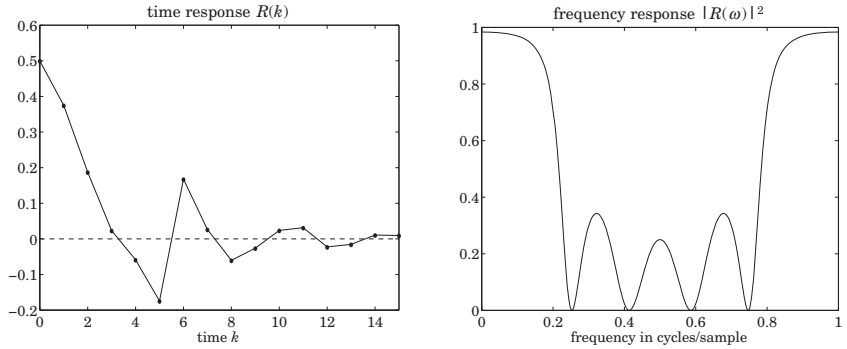


Fig. 12.13.4 Reflection responses in the time and frequency domains.

Cholesky factorization of Toeplitz or near-Toeplitz matrices via the Schur algorithm can be cast in a wave propagation model and derived as a simple consequence of energy conservation [1002].

$$A_{\text{exact}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & 0.1200 & 0 & 0 & 0 \\ 1.0000 & 0.3200 & 0.1500 & 0 & 0 \\ 1.0000 & 0.5200 & 0.3340 & 0.1200 & 0 \\ 1.0000 & 0.6400 & 0.5224 & 0.2760 & 0.0900 \end{bmatrix}$$

$$A_{\text{extract}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & 0.1200 & 0 & 0 & 0 \\ 1.0000 & 0.3200 & 0.1500 & 0 & 0 \\ 1.0000 & 0.5200 & 0.3340 & 0.1200 & 0 \\ 1.0000 & 0.6400 & 0.5224 & 0.2760 & 0.0900 \end{bmatrix}$$

$$B_{\text{exact}} = \begin{bmatrix} 0.3000 & 0 & 0 & 0 & 0 \\ 0.4000 & 0.3000 & 0 & 0 & 0 \\ 0.5000 & 0.4600 & 0.3000 & 0 & 0 \\ 0.4000 & 0.6280 & 0.5200 & 0.3000 & 0 \\ 0.3000 & 0.5560 & 0.7282 & 0.5560 & 0.3000 \end{bmatrix}$$

$$B_{\text{extract}} = \begin{bmatrix} 0.3000 & 0 & 0 & 0 & 0 \\ 0.4000 & 0.3000 & 0 & 0 & 0 \\ 0.5000 & 0.4600 & 0.3000 & 0 & 0 \\ 0.4000 & 0.6280 & 0.5200 & 0.3000 & 0 \\ 0.3000 & 0.5560 & 0.7282 & 0.5560 & 0.3000 \end{bmatrix}$$

k	$R(k)$
0	0.3000
1	0.3640
2	0.3385
3	0.0664
4	−0.0468
5	−0.1309
6	0.0594
7	0.0373
8	−0.0146
9	−0.0148
10	0.0014
11	0.0075
12	−0.0001
13	−0.0029
14	−0.0003
15	0.0010

Fig. 12.13.5 Reflection response and lattice polynomials.

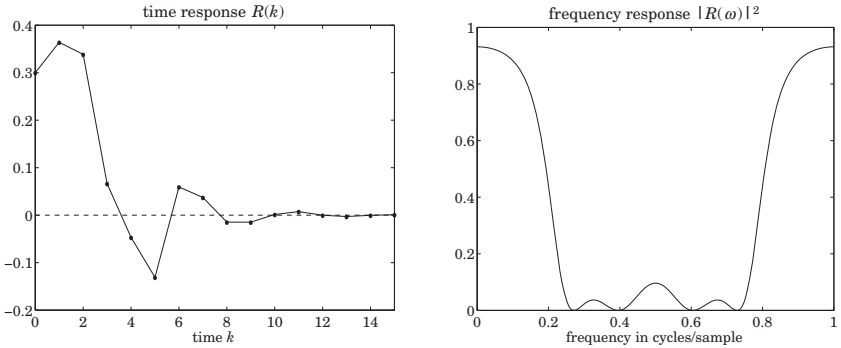


Fig. 12.13.6 Reflection responses in the time and frequency domains.

$$A_{\text{exact}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & 0.0200 & 0 & 0 & 0 \\ 1.0000 & 0.0800 & 0.0300 & 0 & 0 \\ 1.0000 & 0.1400 & 0.0712 & 0.0200 & 0 \\ 1.0000 & 0.1600 & 0.1028 & 0.0412 & 0.0100 \end{bmatrix}$$

$$A_{\text{extract}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & 0.0200 & 0 & 0 & 0 \\ 1.0000 & 0.0800 & 0.0300 & 0 & 0 \\ 1.0000 & 0.1400 & 0.0712 & 0.0200 & 0 \\ 1.0000 & 0.1600 & 0.1028 & 0.0412 & 0.0100 \end{bmatrix}$$

$$B_{\text{exact}} = \begin{bmatrix} 0.1000 & 0 & 0 & 0 & 0 \\ 0.2000 & 0.1000 & 0 & 0 & 0 \\ 0.3000 & 0.2060 & 0.1000 & 0 & 0 \\ 0.2000 & 0.3160 & 0.2120 & 0.1000 & 0 \\ 0.1000 & 0.2140 & 0.3231 & 0.2140 & 0.1000 \end{bmatrix}$$

$$B_{\text{extract}} = \begin{bmatrix} 0.1000 & 0 & 0 & 0 & 0 \\ 0.2000 & 0.1000 & 0 & 0 & 0 \\ 0.3000 & 0.2060 & 0.1000 & 0 & 0 \\ 0.2000 & 0.3160 & 0.2120 & 0.1000 & 0 \\ 0.1000 & 0.2140 & 0.3231 & 0.2140 & 0.1000 \end{bmatrix}$$

k	$R(k)$
0	0.1000
1	0.1980
2	0.2812
3	0.1445
4	0.0388
5	−0.0346
6	−0.0072
7	0.0017
8	0.0015
9	0.0002
10	−0.0002
11	−0.0001
12	0.0000
13	0.0000
14	0.0000
15	−0.0000

Fig. 12.13.7 Reflection response and lattice polynomials.

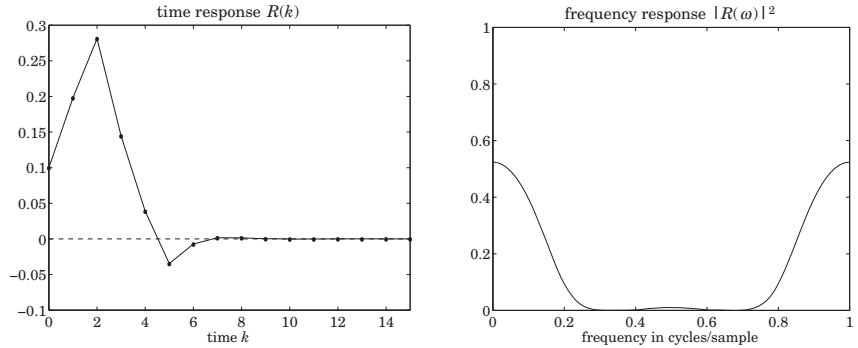


Fig. 12.13.8 Reflection responses in the time and frequency domains.

$$A_{\text{exact}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & -0.2500 & 0 & 0 & 0 \\ 1.0000 & -0.5000 & 0.2500 & 0 & 0 \\ 1.0000 & -0.7500 & 0.5625 & -0.2500 & 0 \\ 1.0000 & -1.0000 & 0.9375 & -0.6250 & 0.2500 \end{bmatrix}$$

$$A_{\text{extract}} = \begin{bmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 1.0000 & -0.2509 & 0 & 0 & 0 \\ 1.0000 & -0.5009 & 0.2510 & 0 & 0 \\ 1.0000 & -0.7509 & 0.5638 & -0.2508 & 0 \\ 1.0000 & -1.0009 & 0.9390 & -0.6263 & 0.2504 \end{bmatrix}$$

$$B_{\text{exact}} = \begin{bmatrix} 0.5000 & 0 & 0 & 0 & 0 \\ -0.5000 & 0.5000 & 0 & 0 & 0 \\ 0.5000 & -0.6250 & 0.5000 & 0 & 0 \\ -0.5000 & 0.7500 & -0.7500 & 0.5000 & 0 \\ 0.5000 & -0.8750 & 1.0313 & -0.8750 & 0.5000 \end{bmatrix}$$

$$B_{\text{extract}} = \begin{bmatrix} 0.5010 & 0 & 0 & 0 & 0 \\ -0.5000 & 0.5010 & 0 & 0 & 0 \\ 0.5000 & -0.6255 & 0.5010 & 0 & 0 \\ -0.5000 & 0.7505 & -0.7510 & 0.5010 & 0 \\ 0.5000 & -0.8755 & 1.0323 & -0.8764 & 0.5010 \end{bmatrix}$$

k	$R(k)$
0	0.5000
1	−0.3750
2	0.1875
3	−0.0234
4	−0.0586
5	0.1743
6	0.1677
7	−0.0265
8	−0.0601
9	0.0259
10	0.0238
11	−0.0314
12	−0.0225
13	0.0153
14	0.0109
15	−0.0097

Fig. 12.13.9 Reflection response and lattice polynomials.

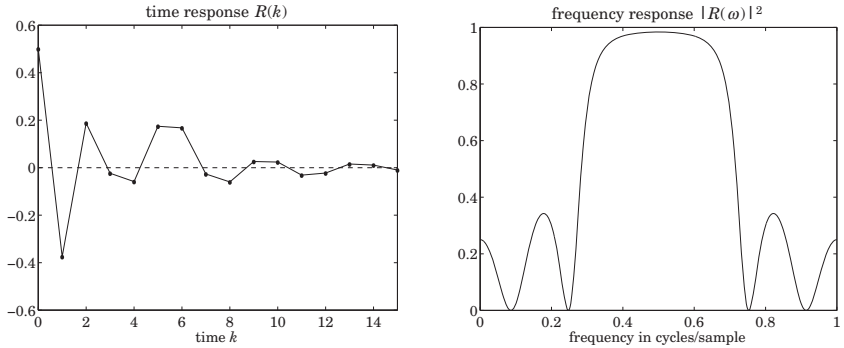


Fig. 12.13.10 Reflection responses in the time and frequency domains.

12.14 Least-Squares Waveshaping and Spiking Filters

In linear prediction, the three practical methods of estimating the prediction error filter coefficients were all based on replacing the ensemble mean-square minimization criterion by a least-squares criterion based on time averages. Similarly, the more general Wiener filtering problem may be recast in terms of such time averages. A practical formulation, which is analogous to the Yule-Walker or autocorrelation method, is as follows [974,975,1010,1059]. Given a record of available data

$$\boxed{y_0, y_1, \ldots, y_N}$$

find the best linear FIR filter of order M

$$\boxed{h_0, h_1, \ldots, h_M}$$

which reshapes y_n into a desired signal x_n, specified in terms of the samples:

$$\boxed{x_0, x_1, \ldots, x_{N+M}}$$

where for consistency of convolution, we assumed we know $N + M + 1$ samples of the desired signal. The actual convolution output of the waveshaping filter will be:

$$\hat{x}_n = \sum_{m=\max(0,n-N)}^{\min(n,M)} h_m x_{n-m}\,, \quad 0 \le n \le N+M \tag{12.14.1}$$

and the estimation error:

$$e_n = x_n - \hat{x}_n\,, \quad 0 \le n \le N+M \tag{12.14.2}$$

As the optimality criterion, we choose the *least-squares* criterion:

$$\mathcal{E} = \sum_{n=0}^{N+M} e_n^2 = \min \tag{12.14.3}$$

The optimal filter weights h_m are selected to minimize $\mathcal{E}$. It is convenient to recast the above in a compact matrix form. Define the $(N+M+1)\times(M+1)$ convolution data matrix Y, the $(M+1)\times 1$ vector of filter weights $\mathbf{h}$, the $(N+M+1)\times 1$ vector of desired samples $\mathbf{x}$, (and estimates $\hat{\mathbf{x}}$ and estimation errors $\mathbf{e}$), as follows:

$$Y = \begin{bmatrix} y_0 & 0 & 0 & \cdots & 0 \\ y_1 & y_0 & 0 & \cdots & 0 \\ y_2 & y_1 & y_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ y_N & y_{N-1} & y_{N-2} & \cdots & y_{N-M} \\ 0 & y_N & y_{N-1} & \cdots & y_{N-M+1} \\ 0 & 0 & y_N & \cdots & y_{N-M+2} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & y_N \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_M \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N+M} \end{bmatrix} \tag{12.14.4}$$

Equations (12.14.1) through (12.14.3) now become

$$\hat{\mathbf{x}} = Y\mathbf{h}\,, \quad \mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}\,, \quad \mathcal{E} = \mathbf{e}^T\mathbf{e} \tag{12.14.5}$$

Minimizing $\mathcal{E}$ with respect to the weight vector $\mathbf{h}$ results in the *orthogonality* equations:

$$Y^T\mathbf{e} = Y^T(\mathbf{x} - Y\mathbf{h}) = 0 \tag{12.14.6}$$

which are equivalent to the *normal* equations:

$$Y^TY\mathbf{h} = Y^T\mathbf{x} \tag{12.14.7}$$

Solving for $\mathbf{h}$, we find

$$\mathbf{h} = (Y^TY)^{-1}Y^T\mathbf{x} = R^{-1}\mathbf{r} \tag{12.14.8}$$

where the quantities

$$R = Y^TY\,, \quad \mathbf{r} = Y^T\mathbf{x} \tag{12.14.9}$$

may be recognized (see Sec. 1.11) as the $(M+1)\times(M+1)$ autocorrelation matrix formed by the sample autocorrelations $\hat{R}_{yy}(0), \hat{R}_{yy}(1), \ldots \hat{R}_{yy}(M)$ of y_n, and as the $(M+1)\times 1$ vector of sample cross-correlations $\hat{R}_{xy}(0), \hat{R}_{xy}(1), \ldots \hat{R}_{xy}(M)$ between the desired and the available vectors x_n and y_n. We have already used this expression for the weight vector $\mathbf{h}$ in the example of Sec. 12.11. Here we have justified it in terms of the least-squares criterion (12.14.3). The function **firw** may be used to solve for the weights (12.14.8) and, if so desired, to give the corresponding lattice realization. The actual filter output $\hat{\mathbf{x}}$ is expressed as

$$\hat{\mathbf{x}} = Y\mathbf{h} = YR^{-1}Y^T\mathbf{x} = P\mathbf{x} \tag{12.14.10}$$

where

$$P = YR^{-1}Y^T = Y(Y^TY)^{-1}Y^T \tag{12.14.11}$$

The error vector becomes $\mathbf{e} = (I - P)\mathbf{x}$. The "performance" matrix P is a projection matrix, and thus, so is $(I - P)$. Then, the error square becomes

$$\mathcal{E} = \mathbf{e}^T\mathbf{e} = \mathbf{x}^T(I - P)^2\mathbf{x} = \mathbf{x}^T(I - P)\mathbf{x} \tag{12.14.12}$$

The $(N + M + 1)\times(N + M + 1)$ matrix P has trace equal to $M + 1$, as can be checked easily. Since its eigenvalues as a projection matrix are either 0 or 1, it follows that in order for the sum of all the eigenvalues (the trace) to be equal to $M + 1$, there must necessarily be $M + 1$ eigenvalues that are equal to 1, and N eigenvalues equal to 0. Therefore, the matrix P has rank $M + 1$, and if the desired vector $\mathbf{x}$ is selected to be any of the $M + 1$ eigenvectors belonging to eigenvalue 1, the corresponding estimation error will be zero.

Among all possible waveshapes that may be chosen for the desired vector $\mathbf{x}$, of particular importance are the *spikes*, or impulses. In this case, $\mathbf{x}$ is a unit impulse, say at the origin; that is, $x_n = \delta_n$. The convolution $\hat{x}_n = h_n * y_n$ of the corresponding filter with y_n is the best least-squares approximation to the unit impulse. In other words, h_n is the best least-squares inverse filter to y_n that attempts to reshape, or compress, y_n into a unit impulse. Such least squares inverse filters are used extensively in deconvolution

applications. More generally. the vector $\mathbf{x}$ may be chosen to be any one of the unit vectors

$$\mathbf{x} = \mathbf{u}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \leftarrow i\text{th slot}\,, \quad i = 0, 1, \ldots, N + M \tag{12.14.13}$$

which corresponds to a unit impulse occurring at the ith time instant instead of at the origin; that is, $x_n = \delta(n - i)$. The actual output from the spiking filter is given by

$$\hat{\mathbf{x}} = P\mathbf{x} = P\mathbf{u}_i = i\text{th column of } P \tag{12.14.14}$$

Thus, the ith column of the matrix P is the output of the ith spiking filter which attempts to compress y_n into a spike with i delays. The corresponding ith filter is $\mathbf{h} = R^{-1}Y^T\mathbf{u}_i$. Therefore, the columns of the matrix

$$H = R^{-1}Y^T = (Y^TY)^{-1}Y^T \tag{12.14.15}$$

are all the optimal spiking filters. The estimation error of the ith filter is

$$\mathcal{E}_i = \mathbf{u}_i^T(I - P)\mathbf{u}_i = 1 - P_{ii} \tag{12.14.16}$$

where P_{ii}, is the ith diagonal element of P. Since the delay i may be positioned anywhere from $i = 0$ to $i = N + M$, there are $N + M + 1$ such spiking filters, each with error $\mathcal{E}_i$. Among these, there will be one that has the optimal delay i which corresponds to the smallest of the $\mathcal{E}_i$s; or, equivalently, to the maximum of the diagonal elements P_{ii}.

The design procedure for least-squares spiking filters for a given finite signal y_n, $n = 0, 1, \ldots, N - 1$ is summarized as follows:

1. Compute $R = Y^TY$.
2. Compute the inverse R^{-1} (preferably by the Levinson recursion).
3. Compute $H = R^{-1}Y^T$ = all the spiking filters.
4. Compute $P = YH = YR^{-1}Y^T$ = all spiking filter outputs.
5. Select that column i of P for which P_{ii} is the largest.

If the Levinson-Cholesky algorithm is used to compute the inverse R^{-1}, this design procedure becomes fairly efficient. An implementation of the procedure is given by the function **spike**. The inputs to the function are the $N + 1$ samples $\{y_0, y_1, \ldots, y_N\}$, the desired order M of the spiking filter, and a so-called "prewhitening" or Backus-Gilbert parameter ϵ, which will be explained below. The outputs of the function are the matrices P and H.

To explain the role of the parameter ϵ, let us go back to the waveshaping problem. When the data sequence y_n to be reshaped into x_n is inaccurately known—if, for example, it has been contaminated by white noise v_n—the least-squares minimization criterion

(12.14.3) can be extended slightly to accomplish the double task of (1) producing the best estimate of x_n and (2) reducing the noise at the output of the filter h_n as much as possible.

The input to the filter is the noisy sequence $y_n + v_n$ and its output is $h_n * y_n + h_n * v_n = \hat{x}_n + u_n$, where we set $u_n = h_n * v_n$. The term u_n represents the filtered noise. The minimization criterion (12.14.3) may be replaced by

$$\mathcal{E} = \sum_n e_n^2 + \lambda E[u_n^2] = \min \tag{12.14.17}$$

where λ is a positive parameter which can be chosen by the user. Large λ emphasizes large reduction of the output noise, but this is done at the expense of resolution; that is, at the expense of obtaining a very good estimate. On the other hand, small λ emphasizes higher resolution but with lesser noise reduction. This tradeoff between resolution and noise reduction is the basic property of this performance index. Assuming that v_n is white with variance σ_v^2, we have

$$E[u_n^2] = \sigma_v^2 \sum_{n=0}^{M} h_n^2 = \sigma_v^2 \mathbf{h}^T \mathbf{h}$$

Thus, Eq. (12.14.17) may be written as

$$\mathcal{E} = \mathbf{e}^T \mathbf{e} + \lambda \sigma_v^2 \mathbf{h}^T \mathbf{h} = \min \tag{12.14.18}$$

Its minimization with respect to $\mathbf{h}$ gives the normal equations:

$$(Y^T Y + \lambda \sigma_v^2 I)\mathbf{h} = Y^T \mathbf{x} \tag{12.14.19}$$

from which it is evident that the diagonal of $Y^T Y$ is shifted by an amount $\lambda \sigma_v^2$; that is,

$$\hat{R}_{yy}(0) \longrightarrow \hat{R}_{yy}(0) + \lambda \sigma_v^2 \equiv (1+\epsilon)\hat{R}_{yy}(0)\,, \quad \epsilon = \frac{\lambda \sigma_v^2}{\hat{R}_{yy}(0)}$$

In practice, ϵ may be taken to be a few percent or less. It is evident from Eq. (12.14.19) that one beneficial effect of the parameter ϵ is the stabilization of the inverse of the matrix $Y^T Y + \lambda \sigma_v^2 I$.

The main usage of spiking filters is in deconvolution problems [59,60,95,144–146], where the desired and the available signals x_n and y_n are related to each other by the convolutional relationship

$$y_n = f_n * x_n = \sum_m f_m x_{n-m} \tag{12.14.20}$$

where f_n is a "blurring" function which is assumed to be approximately known. The basic deconvolution problem is to recover x_n from y_n if f_n is known. For example, y_n may represent the image of an object x_n recorded through an optical system with a point-spread function f_n. Or, y_n might represent the recorded seismic trace arising from the excitation of the layered earth by an impulsive waveform f_n (the source wavelet) which is convolved with the reflection impulse response x_n of the earth (in the previous section x_n was denoted by R_n.) If the effect of the source wavelet f_n can be "deconvolved away," the resulting reflection sequence x_n may be subjected to the dynamic predictive deconvolution procedure to unravel the earth structure. Or, f_n may represent the impulse response of a channel, or a magnetic recording medium, which broadens and blurs (intersymbol interference) the desired message x_n.

The least-squares inverse spiking filters offer a way to solve the deconvolution problem: Simply design a least-squares spiking filter h_n corresponding to the blurring function f_n; that is, $h_n * f_n \simeq \delta_n$, in the least-squares sense. Then, filtering y_n through h_n will recover the desired signal x_n:

$$\hat{x}_n = h_n * y_n = (h_n * f_n) * x_n \simeq \delta_n * x_n = x_n \tag{12.14.21}$$

If the ith spiking filter is used, which compresses f_n into an impulse with i delays, $h_n * f_n \simeq \delta(n-i)$, then the desired signal x_n will be recovered with a delay of i units of time.

This and all other approaches to deconvolution work well when the data y_n are not noisy. In presence of noise, Eq. (12.14.20) becomes

$$y_n = f_n * x_n + v_n \tag{12.14.22}$$

where v_n may be assumed to be zero-mean white noise of variance σ_v^2. Even if the blurring function f_n is known exactly and a good least-squares inverse filter h_n can be designed, the presence of the noise term can distort the deconvolved signal beyond recognition. This may be explained as follows. Filtering y_n through the inverse filter h_n results in

$$h_n * y_n = (h_n * f_n) * x_n + h_n * v_n \simeq x_n + u_n$$

where $u_n = h_n * v_n$ is the filtered noise. Its variance is

$$E[u_n^2] = \sigma_v^2 \mathbf{h}^T \mathbf{h} = \sigma_v^2 \sum_{n=0}^{M} h_n^2$$

which, depending on the particular shape of h_n may be much larger than the original variance σ_v^2. This happens, for example, when f_n consists mainly of low frequencies. For h_n to compress f_n into a spike with a high frequency content, the impulse response h_n itself must be very spiky, which can result in values for $\mathbf{h}^T \mathbf{h}$ which are greater than one.

To combat the effects of noise, the least-squares design criterion for $\mathbf{h}$ must be changed by adding to it a term $\lambda E[u_n^2]$ as was done in Eq. (12.14.17). The modified design criterion is then

$$\mathcal{E} = \sum_n (\delta_n - h_n * f_n)^2 + \lambda \sigma_v^2 \sum_{n=0}^{M} h_n^2$$

which effectively amounts to changing the autocorrelation lag $\hat{R}_{ff}(0)$ into $(1+\epsilon)\hat{R}_{ff}(0)$. The first term in this performance index tries to produce a good inverse filter; the second term tries to minimize the output power of the noise after filtering by the deconvolution filter h_n. Note that conceptually this index is somewhat different from that of

Eq. (12.14.17), because now v_n represents the noise in the data y_n whereas there v_n represented inaccuracies in the knowledge of the wavelet f_n.

In this approach to deconvolution we are not attempting to determine the best least-squares estimate of the desired signal x_n, but rather the best least-squares inverse to the blurring function f_n. If the second order statistics of x_n were known, we could, of course, determine the optimal (Wiener) estimate $\hat{x}_n$ of x_n. This is also done in many applications.

The performance of the spiking filters and their usage in deconvolution are illustrated by the following example: The blurring function f_n to be spiked was chosen as

$$f_n = \begin{cases} g(n-25), & n = 0, 1, \ldots, 65 \\ 0, & \text{for other } n \end{cases}$$

where $g(k)$ was the "gaussian hat" function:

$$g(k) = \cos(0.15k)\exp(-0.004k^2)$$

The signal x_n to be recovered was taken to be the series of delayed spikes:

$$x_n = \sum_{i=0}^{9} a_i \delta(n - n_i)$$

where the amplitudes a_i and delays n_i were chosen as

$$a_i = 1,\ 0.8,\ 0.5,\ 0.95,\ 0.7,\ 0.5,\ 0.3,\ 0.9,\ 0.5,\ 0.85$$

$$n_i = 25,\ 50,\ 60,\ 70,\ 80,\ 90,\ 100,\ 120,\ 140,\ 160$$

for $i = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$.

Fig. 12.14.1 shows the signal f_n to be spiked. Since the gaussian hat is symmetric about the origin, we chose the spiking delay to be at $i = 25$. The order of the spiking filter h_n was $M = 50$. The right graph in Fig. 12.14.1 shows the impulse response h_n versus time. Note the spiky nature of h_n which is required here because f_n has a fairly low frequency content. Fig. 12.14.2 shows the results of the convolution $h_n * f_n$, which is the best least-squares approximation to the impulse $\delta(n - 25)$.

The "goodness" of the spiking filter is judged by the diagonal entries of the performance matrix P, according to Eq. (12.14.16). For the chosen delay $k = 25$, we find $P(25, 25) = 0.97$. To obtain a better picture of the overall performance of the spiking filters, on the right in Fig. 12.14.2 we have plotted the diagonal elements $P(k, k)$ versus k. It is seen that the chosen delay $k = 25$ is nearly optimal. Fig. 12.14.3 shows the composite signal y_n obtained by convolving f_n and x_n, according to Eq. (12.14.20).

Fig. 12.14.3 shows on the right the deconvolved signal x_n according to Eq. (12.14.21). The recovery of the amplitudes a_i and delays n_i of x_n is very accurate. These results represent the idealistic case of noise-free data y_n and perfect knowledge of the blurring function f_n. To study the sensitivity of the deconvolution technique to inaccuracies in the knowledge of the signal f_n we have added a small high frequency perturbation on f_n as follows:

$$f'_n = f_n + 0.05\sin\big(1.5(n - 25)\big)$$

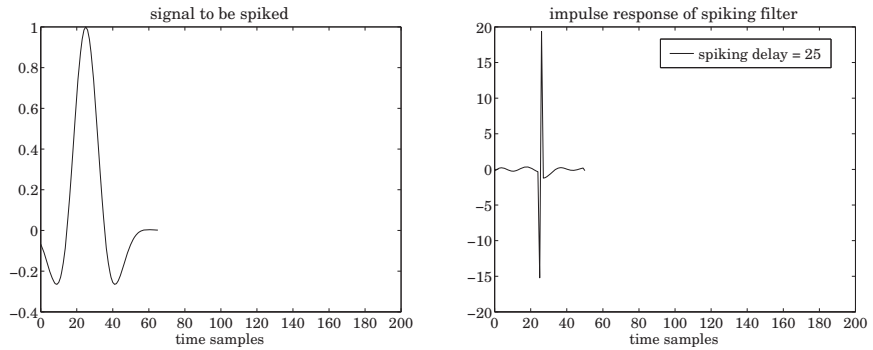


Fig. 12.14.1 Spiking filter and its inverse.

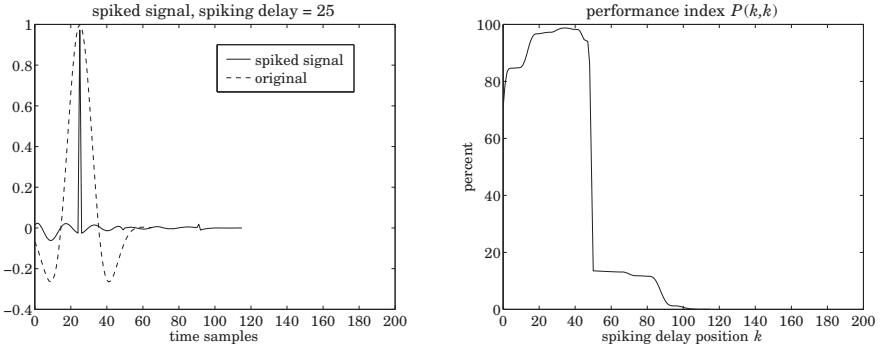


Fig. 12.14.2 Deconvolved signal and performance index.

The approximate signal f'_n is shown in Fig. 12.14.4. The spiking filter was designed on the basis of f'_n rather than f_n. The result of filtering the same composite signal y_n through the corresponding inverse filter is shown on the right in Fig. 12.14.4. The delays and amplitudes a_i and n_i are not well resolved, but the basic nature of x_n can still be seen. Inspecting Fig. 12.14.1 we note the large spikes that are present in the impulse response h_n of the inverse filter; these can cause the amplification of any additive noise component. Indeed, the noise reduction ratio of the filter h_n is $\mathbf{h}^T\mathbf{h} = 612$, thus it will tend to amplify even small amounts of noise.

To study the effect of noise, we have added a noise term v_n, as in Eq. (12.14.22), with variance equal to 10^{-4} (this corresponds to just 1% of the amplitude a_0); the composite signal y_n is shown on the left in Fig. 12.14.5. One can barely see the noise. Yet, after filtering with the inverse filter h_n of Fig. 12.14.1, the noise component is amplified to a great extent. The result of deconvolving the noisy y_n with h_n is shown on the right in Fig. 12.14.5. To reduce the effects of noise, the prewhitening parameter ϵ must be chosen to be nonzero. Even a small nonzero value of ϵ can have a beneficial effect. The

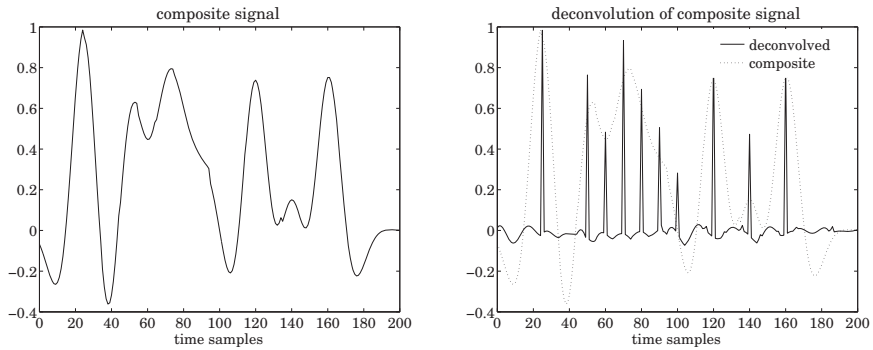


Fig. 12.14.3 Composite and deconvolved signal.

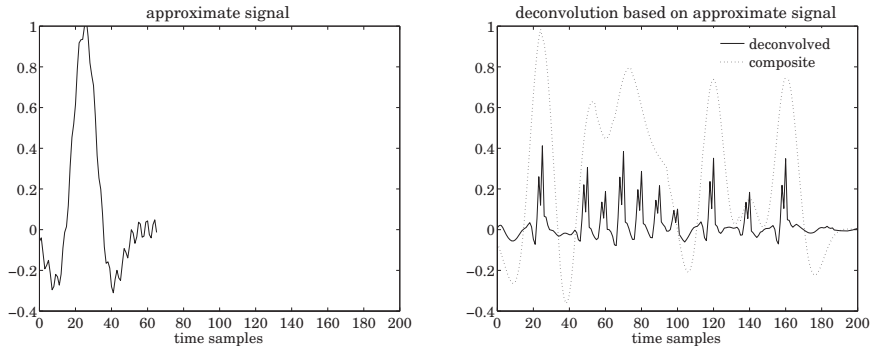


Fig. 12.14.4 Composite and deconvolved signal.

graphs in Fig. 12.14.6 show the deconvolved signal x_n when the filter h_n was designed with the choices $\epsilon = 0.0001$ and $\epsilon = 0.001$, respectively. Note the trade-off between the noise reduction and the loss of resolution in the recovered spikes of x_n.

Based on the studies of Robinson and Treitel [974], Oldenburg [1065], and others, the following summary of the use of the above deconvolution method may be made:

1. If the signal f_n to be spiked is a minimum-phase signal, the optimal spiking delay must be chosen at the origin $i = 0$. The optimality of this choice is not actually seen until the filter order M is sufficiently high. The reason for this choice has to do with the minimum-delay property of such signals which implies that most of their energy is concentrated at the beginning, therefore, they may be more easily compressed to spikes with zero delay.
2. If f_n is a mixed-delay signal, as in the above example, then the optimal spiking delay will have some intermediate value.
3. Even if the shape of f_n is not accurately known, the deconvolution procedure based on the approximate f_n might have some partial success in deconvolving the replicas of f_n.
4. In the presence of noise in the data y_n to deconvolved, some improvement may result by introducing a nonzero value for the prewhitening parameter ϵ, where effectively the sample autocorrelation $R_{ff}(0)$ is replaced by $(1+\epsilon)\hat{R}_{ff}(0)$. The trade-off is a resulting loss of resolution.

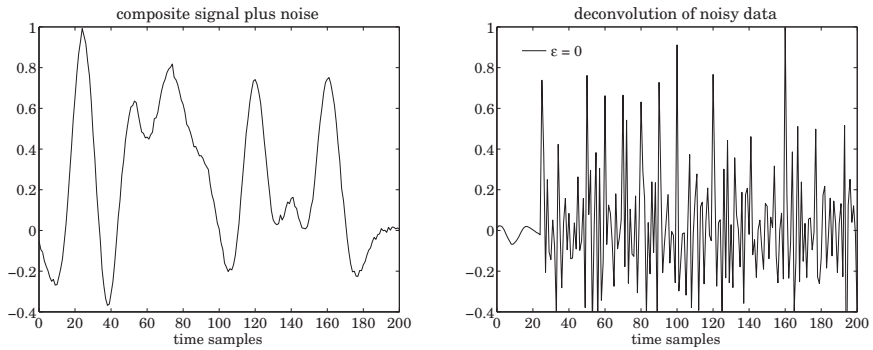


Fig. 12.14.5 Composite and deconvolved signal.

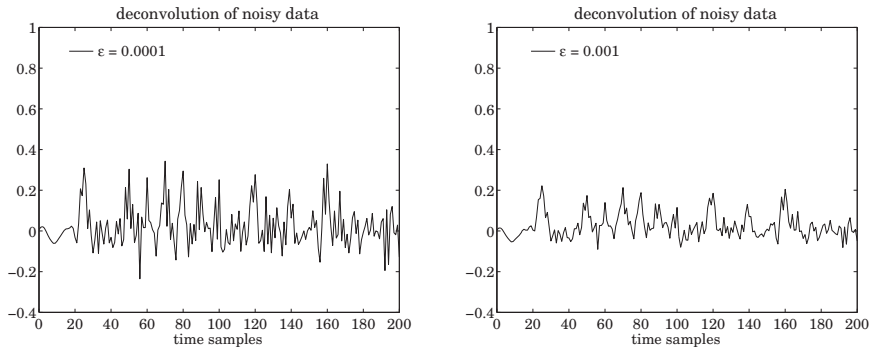


Fig. 12.14.6 Deconvolved signals.

The deconvolution problem of Eq. (12.14.20) and (12.14.22) has been approached by a wide variety of other methods. Typically, a finite number of samples y_n, $n = 0, 1, \ldots, N$ is available. Collecting these into a vector $\mathbf{y} = [y_0, y_1, \ldots, y_N]^T$, we write Eq. (12.14.22) in an obvious vectorial form

$$\mathbf{y} = F\mathbf{x} + \mathbf{v} \tag{12.14.23}$$

Instead of determining an approximate inverse filter for the blurring function F, an alternative method is to attempt to determine the best—in some sense—vector $\mathbf{x}$ which is compatible with these equations. A popular method is based on the least-squares

criterion [1060].

$$\mathcal{E} = \sum_{n=0}^{N} v_n^2 = \mathbf{v}^T\mathbf{v} = (\mathbf{y} - F\mathbf{x})^T(\mathbf{y} - F\mathbf{x}) = \min \tag{12.14.24}$$

That is, $\mathbf{x}$ is chosen so as to minimize $\mathcal{E}$. Setting the derivative with respect to $\mathbf{x}$ to zero gives the standard least-squares solution

$$\hat{\mathbf{x}} = (F^TF)^{-1}F^T\mathbf{y}$$

A prewhitening term can be added to the right of the performance index to stabilize the indicated inverse

$$\mathcal{E} = \mathbf{v}^T\mathbf{v} + \lambda\mathbf{x}^T\mathbf{x}$$

with solution $\hat{\mathbf{x}} = (F^TF + \lambda I)^{-1}F^T\mathbf{y}$. Another approach that has been used with success is based on the L_1-norm criterion

$$\mathcal{E} = \sum_{n=0}^{N} |v_n| = \min \tag{12.14.25}$$

This quantity is referred to as the L_1 norm of the vector $\mathbf{v}$. The minimization of this norm with respect to $\mathbf{x}$ may be formulated as a *linear programming* problem [1063–1073]. It has been observed that this method performs very well in the presence of noise, and it tends to ignore a few "bad" data points—that is, those for which the noise value v_n might be abnormally high—in favor of the good points, whereas the standard least-squares method based on the L_2-norm (12.14.24) will spend all its efforts trying to minimize the few large terms in the sum (12.14.24), and might not result in as good an estimate of $\mathbf{x}$ as it would if the few bad data points were to be ignored. We discuss this approach further in Sec. 15.11

Another class of deconvolution methods are *iterative* methods, reviewed in [1074]. Such methods, like the linear programming method mentioned above, offer the additional option of enforcing *priori constraints* that may be known to be satisfied by $\mathbf{x}$, for example, positivity, band-limiting, or time-limiting constraints. The imposition of such constraints can improve the restoration process dramatically.

12.15 Computer Project – ARIMA Modeling

The Box-Jenkins airline data set has served as a benchmark in testing seasonal ARIMA models. In particular, it has led to the popular "airline model", which, for monthly data with yearly periodicity, is defined by the following innovations signal model:

$$\boxed{(1 - Z^{-1})(1 - Z^{-12})y_n = (1 - bZ^{-1})(1 - BZ^{-12})\varepsilon_n} \tag{12.15.1}$$

where Z^{-1} denotes the delay operator and b, B are constants such that $|b| < 1$ and $|B| < 1$. In this experiment, we briefly consider this model, but then replace it with the following ARIMA model,

$$\boxed{(1 - Z^{-12})y_n = \frac{1}{A(Z)}\,\varepsilon_n}, \quad A(z) = 1 + a_1z^{-1} + a_2z^{-2} + \cdots + a_pz^{-p} \tag{12.15.2}$$

The airline data set can be loaded with the MATLAB commands:


```
Y = load('airline.dat');          % in OSP data file folder
Y = Y'; Y = Y(:);                 % concatenate rows
y = log(Y);                       % log data
```

The data represent monthly airline passengers for the period Jan. 1949 to Dec. 1960. There are $N = 144$ data points. In this experiment, we will work with a subset of the first $n_0 = 108$ points, that is, y_n, $0 \le n \le n_0 - 1$, and attempt to predict the future 36 months of data (108 + 36 = 144.)

a. Plot Y_n and the log-data $y_n = \ln Y_n$ versus n and note the yearly periodicity. Note how the log-transformation tends to equalize the apparent increasing amplitude of the original data.

b. Compute and plot the normalized sample ACF, $\rho_k = R(k)/R(0)$, of the *zero-mean* log-data for lags $0 \le k \le 40$ and note the peaks at multiples of 12 months.

c. Let $x_n = (1 - Z^{-1})(1 - Z^{-12})y_n$ in the model of Eq. (12.15.1). The signal x_n follows an MA model with spectral density:

$$S_{xx}(z) = \sigma_\varepsilon^2(1 - bz^{-1})(1 - bz)(1 - Bz^{-12})(1 - Bz^{12})$$

Multiply the factors out and perform an inverse z-transform to determine the autocorrelation lags $R_{xx}(k)$. Show in particular, that

$$\frac{R_{xx}(1)}{R_{xx}(0)} = -\frac{b}{1 + b^2}, \quad \frac{R_{xx}(12)}{R_{xx}(0)} = -\frac{B}{1 + B^2} \tag{12.15.3}$$

Filter the subblock $y_n, 0 \le n \le n_0 - 1$ through the filter $(1 - z^{-1})(1 - z^{-12})$ to determine x_n. You may discard the first 13 transient outputs of x_n. Use the rest of x_n to calculate its sample ACF and apply Eq. (12.15.3) to solve for the model parameters b, B.

This simple method gives values that are fairly close to the Box/Jenkins values determined by maximum likelihood methods, namely, $b = 0.4$ and $B = 0.6$, see Ref. [22].

d. Consider, next, the model of Eq. (12.15.2) with a second-order AR model i.e., $A(z)$ filter order of $p = 2$. Define $x_n = (1 - Z^{-12})y_n = y_n - y_{n-12}$ and denote its autocorrelation function by R_k. Since x_n follows an AR(2) model, its model parameters $a_1, a_2, \sigma_\varepsilon^2$ can be computed from:

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = -\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix}^{-1}\begin{bmatrix} R_1 \\ R_2 \end{bmatrix}, \quad \sigma_\varepsilon^2 = R_0 + a_1R_1 + a_2R_2 \tag{12.15.4}$$

Using the data subset $y_n, 0 \le n \le n_0 - 1$, calculate the signal x_n and discard the first 12 transient samples. Using the rest of x_n, compute its sample ACF, $\hat{R}_k$ for $0 \le k \le M$ with $M = 40$, and use the first 3 computed lags $\hat{R}_0, \hat{R}_1, \hat{R}_2$ in Eqs. (12.15.4) to estimate $a_1, a_2, \sigma_\varepsilon^2$ (in computing the ACF, the signal x_n need not be replaced by its zero-mean version).

Because of the assumed autoregressive model, it is possible to calculate all the autocorrelation lags R_k for $k \ge p + 1$ from the first $p + 1$ lags. This can accomplished by the MATLAB "autocorrelation sequence extension" function:

```
M=40; Rext = acext(R(1:p+1), zeros(1,M-p));
```

On the same graph, plot the sample and extended ACFs, $\hat{R}(k)$ and $R_{\text{ext}}(k)$ versus $0 \le k \le M$ normalized by their lag-0 values.

e. Let $\hat{x}_n$ denote the estimate/prediction of x_n based on the data subset $\{x_m, m \le n_0 - 1\}$. Clearly, $\hat{x}_n = x_n$, if $n \le n_0 - 1$. Writing Eq. (12.15.2) recursively, we obtain for the predicted values into the future beyond n_0:

$$\begin{aligned} x_n &= -(a_1 x_{n-1} + a_2 x_{n-2}) + \varepsilon_n \\ \hat{x}_n &= -(a_1 \hat{x}_{n-1} + a_2 \hat{x}_{n-2}), \quad \text{for } n \ge n_0 \end{aligned} \tag{12.15.5}$$

where we set $\hat{\varepsilon}_n = 0$ because all the observations are in the strict past of ε_n when $n \ge n_0$.

Calculate the predicted values 36 steps into the future, $\hat{x}_n$ for $n_0 \le n \le N-1$, using the fact that $\hat{x}_n = x_n$, if $n \le n_0 - 1$. Once you have the predicted x_n's, you can calculate the predicted y_n's by the recursive equation:

$$\hat{y}_n = \hat{y}_{n-12} + \hat{x}_n \tag{12.15.6}$$

where you must use $\hat{y}_n = y_n$, if $n \le n_0 - 1$. Compute $\hat{y}_n$ for $n_0 \le n \le N-1$, and plot it on the same graph with the original data y_n, $0 \le n \le N-1$. Indicate the start of the prediction horizon with a vertical line at $n = n_0$ (see example graph at end.)

f. Repeat parts (d,e) using an AR(4) model (i.e., $p = 4$), with signal model equations:

$$x_n = y_n - y_{n-12}, \quad x_n = -(a_1 x_{n-1} + a_2 x_{n-2} + a_3 x_{n-3} + a_4 x_{n-4}) + \varepsilon_n$$

and normal equations:

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = -\begin{bmatrix} R_0 & R_1 & R_2 & R_3 \\ R_1 & R_0 & R_1 & R_2 \\ R_2 & R_1 & R_0 & R_1 \\ R_3 & R_2 & R_1 & R_0 \end{bmatrix}^{-1} \begin{bmatrix} R_1 \\ R_2 \\ R_3 \\ R_4 \end{bmatrix}, \quad \sigma_\varepsilon^2 = R_0 + a_1 R_1 + a_2 R_2 + a_3 R_3 + a_4 R_4$$

with predictions:

$$\begin{aligned} \hat{x}_n &= -(a_1 \hat{x}_{n-1} + a_2 \hat{x}_{n-2} + a_3 \hat{x}_{n-3} + a_4 \hat{x}_{n-4}) \\ \hat{y}_n &= \hat{y}_{n-12} + \hat{x}_n \end{aligned}$$

taking into account the properties that $\hat{x}_n = x_n$ and $\hat{y}_n = y_n$, if $n \le n_0 - 1$.

g. Inspecting the log-data, it is evident that there is an almost linear trend as well as a twelve-month, or even, a six-month periodicity. In this part, we will attempt to fit the the data using a conventional basis-functions method and then use this model to predict the last 36 months.

Consider the following model for the first n_0 points of the log data y_n that assumes a quadratic trend and 12-month, 6-month, and 4-month periodicities, i.e., three harmonics of the fundamental frequency of 1/12 cycle/month, for $0 \le n \le n_0 - 1$,

$$\begin{aligned} \hat{y}_n = c_0 + c_1 n + c_2 n^2 &+ c_3 \sin\left(\frac{2\pi n}{12}\right) + c_4 \cos\left(\frac{2\pi n}{12}\right) + \\ &+ c_5 \sin\left(\frac{4\pi n}{12}\right) + c_6 \cos\left(\frac{4\pi n}{12}\right) \\ &+ c_7 \sin\left(\frac{6\pi n}{12}\right) + c_8 \cos\left(\frac{6\pi n}{12}\right) \end{aligned} \tag{12.15.7}$$

Carry out a least-squares fit to determine the nine coefficients c_i, that is, determine the c_i that minimize the quadratic performance index,

$$\mathcal{J} = \sum_{n=0}^{n_0-1} (y_n - \hat{y}_n)^2 = \min \tag{12.15.8}$$

In addition, carry out a fit based on the L_1 criterion,

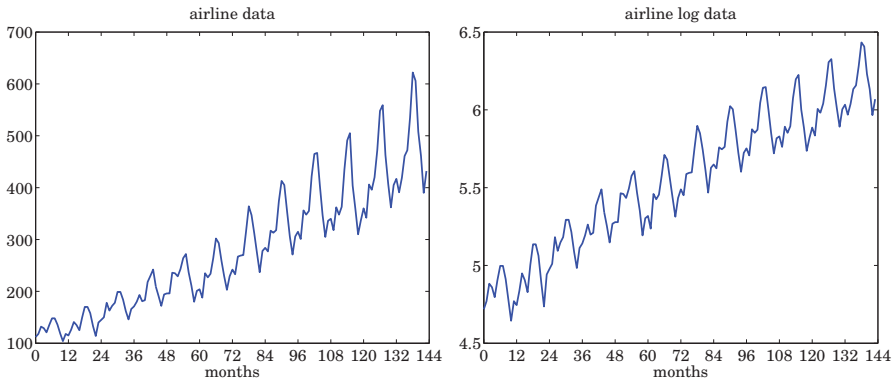
$$\mathcal{J} = \sum_{n=0}^{n_0-1} |y_n - \hat{y}_n| = \min \tag{12.15.9}$$

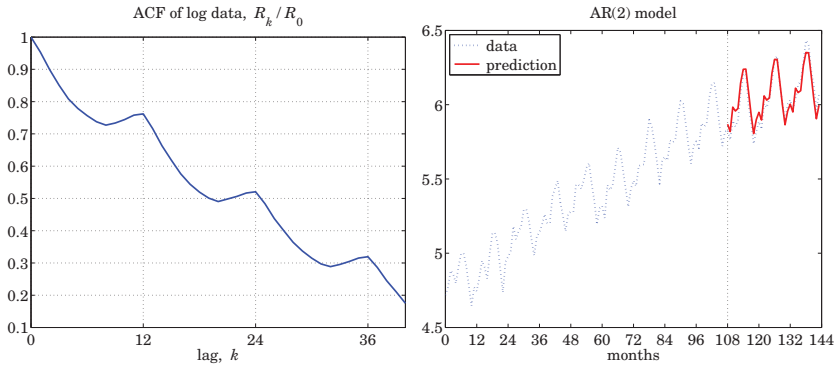
Show that the coefficients are, for the two criteria,

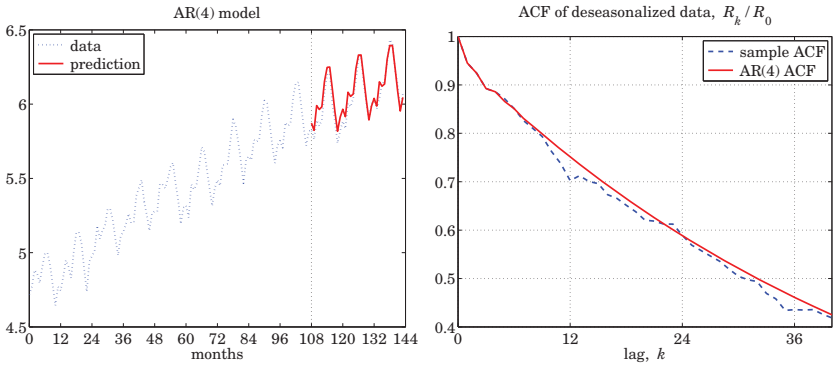
	L_2	L_1
c_0	4.763415	4.764905
c_1	0.012010	0.011912
c_2	−0.000008	−0.000006
c_3	0.036177	0.026981
c_4	−0.135907	−0.131071
c_5	0.063025	0.067964
c_6	0.051890	0.051611
c_7	−0.025511	−0.017619
c_8	−0.009168	−0.012753

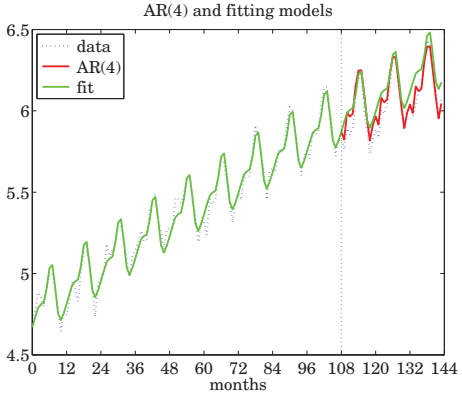
Using the found coefficients for each criterion, evaluate the fitted quantity $\hat{y}_n$ of Eq. (12.15.7) over the entire data record, $0 \le n \le N-1$, and plot it on the same graph together with the actual data, and with the AR(4) prediction. It appears that this approach also works well but over a shorter prediction interval, i.e., 24 months instead of 36 for the AR(4) method.

Some example graphs for this experiment are included below.









12.16 Problems

12.1 (a) Following the methods of Sec. 12.1, show that the optimal filter for predicting D steps into the future—i.e., estimating $y(n+D)$ on the basis of $\{y(m);\ m \le n\}$—is given by

$$H(z) = \frac{1}{B(z)}\left[z^D B(z)\right]_+$$

(b) Express $\left[z^D B(z)\right]_+$ in terms of $B(z)$ itself and the first $D-1$ impulse response coefficients b_m, $m = 1, 2, \ldots, D-1$ of $B(z)$.

(c) For the two random signals y_n defined in Examples 12.1.1 and 12.1.2, find the optimal prediction filters for $D = 2$ and $D = 3$, and write the corresponding I/O equations.

12.2 Consider the order-p autoregressive sequence y_n defined by the difference equation (12.2.3). Show that a direct consequence of this difference equation is that the projection of y_n onto the subspace spanned by the entire past $\{y_{n-i};\ 1 \le i < \infty\}$ is the same as the projection of y_n onto the subspace spanned only by the past p samples $\{y_{n-i};\ 1 \le i \le p\}$.

12.3 (a) Show that the performance index (12.3.2) may be written as

$$\mathcal{E} = E[e_n^2] = \mathbf{a}^T R \mathbf{a}$$

where $\mathbf{a} = [1, a_1, \ldots, a_p]^T$ is the order-p prediction-error filter, and R the autocorrelation matrix of y_n; that is, $R_{ij} = E[y_{n-i} y_{n-j}]$.

(b) Derive Eq. (12.3.7) by minimizing the index $\mathcal{E}$ with respect to the weights $\mathbf{a}$, subject to the linear constraint that $a_0 = 1$, and incorporating this constraint by means of a Lagrange multiplier.

12.4 Take the inverse z-transform of Eq. (12.3.17) and compare the resulting equation with Eq. (12.3.15).

12.5 Verify that Eq. (12.3.22) and (12.3.23) are inverses of each other.

12.6 A fourth order all-pole random signal process $y(n)$ is represented by the following set of signal model parameters (reflection coefficients and input variance):

$$\{\gamma_1, \gamma_2, \gamma_3, \gamma_4, \sigma_\epsilon^2\} = \{0.5,\ -0.5,\ 0.5,\ -0.5,\ 40.5\}$$

(a) Using the Levinson recursion, find the prediction error filter $A_4(z)$.

(b) Determine $\sigma_y^2 = R_{yy}(0)$. Using intermediate results from part (a), determine the autocorrelation lags $R_{yy}(k)$, $k = 1, 2, 3, 4$.

12.7 The first five lags of the autocorrelation function of a fourth-order autoregressive random sequence $y(n)$ are

$$\{R(0), R(1), R(2), R(3), R(4)\} = \{256,\ 128,\ -32,\ -16,\ 22\}$$

Determine the best prediction-error filters and the corresponding mean-square errors of orders $p = 1, 2, 3, 4$ by using Levinson's algorithm in matrix form.

12.8 The fourth-order prediction-error filter and mean-square prediction error of a random signal have been determined to be

$$A_4(z) = 1 - 1.25z^{-1} + 1.3125z^{-2} - z^{-3} + 0.5z^{-4}, \quad E_4 = 0.81$$

Using the function **rlev**, determine the autocorrelation lags $R(k)$, $0 \le k \le 4$, the four reflection coefficients, and all the lower order prediction-error filters.

12.9 Verify the results of Example 12.3.1 using the functions **lev**, **frwlev**, **bkwlev**, and **rlev**, as required.

12.10 (a) Given the five signal samples

$$\{y_0, y_1, y_2, y_3, y_4\} = \{1,\ -1,\ 1,\ -1,\ 1\}$$

compute the corresponding sample autocorrelation lags $\hat{R}(k)$, $k = 0, 1, 2, 3, 4$, and send them through the function **lev** to determine the fourth-order prediction error filter $A_4(z)$.

(b) Predict the sixth sample in this sequence.

(c) Repeat (a) and (b) for the sequence of samples $\{1, 2, 3, 4, 5\}$.

12.11 Find the infinite autoregressive or maximum-entropy extension of the two autocorrelation sequences

(a) $\{R(0), R(1)\} = \{1,\ 0.5\}$

(b) $\{R(0), R(1), R(2)\} = \{4,\ 0,\ 1\}$

In both cases, determine the corresponding power spectrum density $S_{yy}(z)$ and from it calculate the $R(k)$ for all lags k.

12.12 Write Eq. (12.3.24) for order $p + 1$. Derive Eq. (12.5.1) from Eq. (12.3.24) by replacing the filter $\mathbf{a}_{p+1}$ in terms of the filter $\mathbf{a}_p$ via the Levinson recursion.

12.13 Do Problem 12.7 using the split Levinson algorithm.

12.14 Draw the lattice realization of the analysis and synthesis filters $A_4(a)$ and $1/A_4(z)$ obtained in Problems 12.6, 12.7, and 12.8.

12.15 Test the minimum-phase property of the two polynomials

$$A(z) = 1 - 1.08z^{-1} + 0.13z^{-2} + 0.24z^{-3} - 0.5z^{-4}$$

$$A(z) = 1 + 0.18z^{-1} - 0.122z^{-2} - 0.39z^{-3} - 0.5z^{-4}$$

12.16 (a) The entropy of an M-dimensional random vector is defined by $S = -\int p(\mathbf{y}) \ln p(\mathbf{y})\, d^M\mathbf{y}$. Show that the entropy of a zero-mean gaussian $\mathbf{y}$ with covariance matrix R is given, up to an additive constant, by $S = \frac{1}{2}\ln(\det R)$.

(b) With the help of the LU factorization (12.9.1), show that ratio of the determinants of an order M autocorrelation matrix and its order p $(p < M)$ submatrix is

$$\frac{\det R_M}{\det R_p} = \prod_{i=p+1}^{M} E_i$$

(c) Consider all possible autocorrelation extensions of the set $\{R(0), R(1), \ldots, R(p)\}$ up to order M. For gaussian processes, use the results in parts (a) and (b) to show that the particular extension defined by the choice $y_i = 0$, $i = p+1, \ldots, M$ maximizes the entropy of the order-M process; hence, the name maximum entropy extension.

12.17 Consider the LU factorization $LRL^T = D$ of an order-M autocorrelation matrix R. Denote by $\mathbf{b}_p^T$, $p = 0, 1, \ldots, M$ the rows of L. They are the backward prediction filters with zeros padded to their ends to make them $(M+1)$-dimensional vectors.

(a) Show that the inverse factorization $R^{-1} = L^T D^{-1} L$ can be written as

$$R^{-1} = \sum_{p=0}^{M} \frac{1}{E_p} \mathbf{b}_p \mathbf{b}_p^T$$

(b) Define the "phasing" vectors $\mathbf{s}(z) = [1,\ z^{-1},\ z^{-2},\ \ldots,\ z^{-M}]^T$. Show that the z-transform of an order-M filter and its inverse can be expressed compactly as

$$A(z) = \mathbf{s}(z)^T \mathbf{a}, \quad \mathbf{a} = \oint_{\text{u.c}} A(z)\, \mathbf{s}(z^{-1})\, \frac{dz}{2\pi j z}$$

(c) Define the "kernel" vector $\mathbf{k}(w) = R^{-1}\mathbf{s}(w)$. The z-transform of this vector is called a *reproducing kernel* [972,973,981]. Show that it can be written in the alternative forms

$$K(z, w) = \mathbf{s}(z)^T \mathbf{k}(w) = \mathbf{k}(z)^T \mathbf{s}(w) = \mathbf{k}(z)^T R\, \mathbf{k}(w) = \mathbf{s}(z)^T R^{-1} \mathbf{s}(w)$$

(d) Let J denote the $(M+1)\times(M+1)$ reversing matrix. Show that $J\mathbf{s}(z) = z^{-M}\mathbf{s}(z^{-1})$. And that $K(z, w) = z^{-M} w^{-M} K(z^{-1}, w^{-1})$.

(e) Show that $K(z, w)$ admits the following representations in terms of the backward and forward prediction polynomials

$$K(z, w) = \sum_{p=0}^{M} \frac{1}{E_p} B_p(z) B_p(w) = \sum_{p=0}^{M} \frac{1}{E_p} A_p(z) A_p(w) z^{-(M-p)} w^{-(M-p)}$$

12.18 Let $S_{yy}(z)$ be the power spectral density of the autocorrelation function $R(k)$ from which we build the matrix R of the previous problem. Show that R and R^{-1} admit the following representations in terms of the phasing and kernel vectors:

$$R = \oint_{\text{u.c.}} S_{yy}(z)\, \mathbf{s}(z^{-1})\, \mathbf{s}(z)^T \frac{dz}{2\pi j z}, \quad R^{-1} = \oint_{\text{u.c.}} S_{yy}(z)\, \mathbf{k}(z^{-1})\, \mathbf{k}(z)^T \frac{dz}{2\pi j z}$$

Then, show the *reproducing kernel* property

$$K(z, w) = \oint_{\text{u.c.}} K(z, u^{-1}) K(w, u) S_{yy}(u)\, \frac{du}{2\pi j u}$$

12.19 (a) Let $\mathbf{s}_p(z) = [1, z^{-1}, z^{-2}, \ldots, z^{-p}]^T$. Using the order-updating formulas for R_p^{-1} show that the kernel vector $\mathbf{k}_p(w) = R_p^{-1}\mathbf{s}_p(w)$ satisfies the following order-recursive equations

$$\mathbf{k}_p(w) = \begin{bmatrix} \mathbf{k}_{p-1}(w) \\ 0 \end{bmatrix} + \frac{1}{E_p}\mathbf{b}_p B_p(w)\,, \quad \mathbf{k}_p(w) = \begin{bmatrix} 0 \\ w^{-1}\mathbf{k}_{p-1}(w) \end{bmatrix} + \frac{1}{E_p}\mathbf{a}_p A_p(w)$$

(b) Show that the corresponding reproducing kernels satisfy

$$K_p(z,w) = K_{p-1}(z,w) + \frac{1}{E_p}B_p(z)B_p(w)$$

$$K_p(z,w) = z^{-1}w^{-1}K_{p-1}(z,w) + \frac{1}{E_p}A_p(z)A_p(w)$$

(c) Using part (b), show the Christoffel-Darboux formula [972,973,981]

$$K_p(z,w) = \frac{1}{E_p}\,\frac{A_p(z)A_p(w) - z^{-1}w^{-1}B_p(z)B_p(w)}{1 - z^{-1}w^{-1}}$$

(d) Let z_i be the ith zero of the prediction polynomial $A_p(z)$. Using part (c), evaluate $K_p(z_i, z_i^*)$ and thereby show that necessarily $|z_i| \le 1$. This is yet another proof of the minimum-phase property of the prediction-error filters. Show further that if the prediction filter $\mathbf{a}_p$ is symmetric; i.e., $\mathbf{a}_p = \mathbf{a}_p^R$, then its zeros lie on the unit circle.

(e) Show the Christoffel-Darboux formula [972,973,981]

$$K_{p-1}(z,w) = \frac{1}{E_p}\,\frac{A_p(z)A_p(w) - B_p(z)B_p(w)}{1 - z^{-1}w^{-1}}$$

and use this expression to prove the result in (d) that $|z_i| \le 1$.

12.20 Do Problem 12.7 using the Schur algorithm, determine the Cholesky factor G, and verify $R = GD^{-1}G^T$ by explicit matrix multiplication.

12.21 For the Example 12.10.2, compute the entries of the output matrices $Y^{\pm}$ by directly convolving the forward/backward prediction filters with the input autocorrelation lags.

12.22 Do Problem 12.7 using the split Schur algorithm, and determine the Cholesky factor G by the recursion (12.10.21).

12.23 (a) Show the identity

$$\left|\frac{-a^* + z^{-1}}{1 - az^{-1}}\right|^2 = 1 - \frac{\left(1 - |z^{-1}|^2\right)\left(1 - |a|^2\right)}{|1 - az^{-1}|^2}$$

(b) Using part (a), show that the all-pass Schur function $S_p(z)$ defined by Eq. (12.10.22) satisfies the boundedness inequality (12.10.23), with equality attained on the unit circle. Show that it also satisfies $|S_p(z)| > 1$ for $|z| < 1$.

12.24 Define the Schur function

$$S_3(z) = \frac{0.125 - 0.875z^{-2} + z^{-3}}{1 - 0.875z^{-1} + 0.125z^{-3}}$$

Carry out the recursions (12.10.24) and (12.10.25) to construct the lower order Schur functions $S_p(z)$, $p = 2, 1, 0$, and, in the process, extract the corresponding reflection coefficients.

12.25 Consider a generalized version of the simulation example discussed in Sec. 12.11, defined by

$$x(n) = s(n) + v_1(n)\,, \quad y(n) = v_2(n)$$

where

$$\begin{aligned} s(n) &= \sin(\omega_0 n + \phi) \\ v_1(n) &= a_1 v_1(n-1) + v(n) \\ v_2(n) &= a_2 v_2(n-1) + v(n) \end{aligned}$$

where $v(n)$ is zero-mean, unit-variance, white noise, and ϕ is a random phase independent of $v(n)$. This ensures that the $s(n)$ component is uncorrelated with $v_1(n)$ and $v_2(n)$.

(a) Show that

$$R_{xy}(k) = \frac{a_1^k}{1 - a_1 a_2}\,, \quad R_{yy}(k) = \frac{a_2^k}{1 - a_2^2}\,, \quad k \ge 0$$

(b) Show that the infinite-order Wiener filter for estimating $x(n)$ on the basis of $y(n)$ has a (causal) impulse response

$$h_0 = 1\,, \quad h_k = (a_1 - a_2)a_1^{k-1}\,, \quad k \ge 1$$

(c) Next, consider the order-M FIR Wiener filter. Send the theoretical correlations of part (a) for $k = 0, 1, \ldots, M$ through the function **firw** to obtain the theoretical Mth order Wiener filter realized both in the direct and the lattice forms. Draw these realizations. Compare the theoretical values of the weights $\mathbf{h}$, $\mathbf{g}$, and $\boldsymbol{\gamma}$ with the simulated values presented in Sec. 12.11 that correspond to the choice of parameters $M = 4$, $a_1 = -0.5$, and $a_2 = 0.8$. Also compare the answer for $\mathbf{h}$ with the first $(M+1)$ samples of the infinite-order Wiener filter impulse response of part (b).

(d) Repeat (c) with $M = 6$.

12.26 A closed form solution of Problem 12.25 can be obtained as follows.

(a) Show that the inverse of the $(M+1)\times(M+1)$ autocorrelation matrix defined by the autocorrelation lags $R_{yy}(k)$, $k = 0, 1, \ldots, M$ of Problem 12.25(a) is given by the tridiagonal matrix:

$$R_{yy}^{-1} = \begin{bmatrix} 1 & -a_2 & 0 & \cdots & 0 & 0 \\ -a_2 & b & -a_2 & \cdots & 0 & 0 \\ 0 & -a_2 & b & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & b & -a_2 \\ 0 & 0 & 0 & \cdots & -a_2 & 1 \end{bmatrix}$$

where $b = 1 + a_2^2$.

(b) Using this inverse, show that the optimal Mth order Wiener filter has impulse response

$$h_0 = 1\,, \quad h_k = (a_1 - a_2)a_1^{k-1}\,, \quad \text{for } 1 \le k \le M-1\,, \quad \text{and } h_M = \frac{a_1 - a_2}{1 - a_1 a_2}\,a_1^{M-1}$$

(c) Show that the lattice weights $\mathbf{g}$ can be obtained from $\mathbf{h}$ by the backward substitution

$$g_M = h_M\,, \quad \text{and} \quad g_m = a_2 g_{m+1} + h_m\,, \quad m = M-1, M-2, \ldots, 1, 0$$

(d) For $M = 4$, $a_1 = -0.5$, $a_2 = 0.8$, compute the numerical values of **h** and **g** using the above expressions and compare them with those of Problem 12.25(c).

12.27 *Computer Experiment.* Consider the noise canceling example of Sec. 12.11 and Problem 12.25, defined by the choice of parameters

$$\omega_0 = 0.075\pi \text{ [radians/sample]}, \quad \phi = 0, \quad a_1 = -0.5, \quad a_2 = 0.8, \quad M = 4$$

(a) Generate 100 samples of the signals $x(n)$, $s(n)$, and $y(n)$. On the same graph, plot $x(n)$ and $s(n)$ versus n. Plot $y(n)$ versus n.

(b) Using these samples, compute the sample correlations $\hat{R}_{yy}(k)$, $\hat{R}_{xy}(k)$, for $k = 0, 1, \ldots, M$, and compare them with the theoretical values obtained in Problem 12.25(a).

(c) Send these lags through the function **firw** to get the optimal Wiener filter weights **h** and **g**, and the reflection coefficients $\boldsymbol{\gamma}$. Draw the lattice and direct-form realizations of the Wiener filter.

(d) Filter $y(n)$ through the Wiener filter realized in the lattice form, and plot the output $e(n) = x(n) - \hat{x}(n)$ versus n.

(e) Repeat (d) using the direct-form realization of the Wiener filter.

(f) Repeat (d) when $M = 6$.

12.28 The following six samples

$$\{y_0, y_1, y_2, y_3, y_4, y_5\} = \{4.684,\ 7.247,\ 8.423,\ 8.650,\ 8.640,\ 8.392\}$$

have been generated by sending zero-mean unit-variance white noise through the difference equation

$$y_n = a_1 y_{n-1} + a_2 y_{n-2} + \epsilon_n$$

where $a_1 = 1.70$ and $a_2 = -0.72$. Iterating Burg's method by hand, obtain estimates of the model parameters a_1, a_2, and σ_ϵ^2.

12.29 Derive Eq. (12.12.11).

12.30 *Computer Experiment.* Ten samples from a fourth-order autoregressive process $y(n)$ are given. It is desired to extract the model parameters $\{a_1, a_2, a_3, a_4, \sigma_\epsilon^2\}$ as well as the equivalent parameter set $\{\gamma_1, \gamma_2, \gamma_3, \gamma_4, \sigma_\epsilon^2\}$.

n	$y(n)$
0	4.503
1	−10.841
2	−24.183
3	−25.662
4	−14.390
5	1.453
6	10.980
7	13.679
8	15.517
9	15.037

(a) Determine these parameters using Burg's method.

(b) Repeat using the Yule-Walker method.

Note: The exact parameter values by which the above simulated samples were generated are

$$\{a_1, a_2, a_3, a_4, \sigma_\epsilon^2\} = \{-2.2137,\ 2.9403,\ -2.2697,\ 0.9606,\ 1\}$$

12.31 Using the continuity equations at an interface, derive the transmission matrix equation (12.13.2) and the energy conservation equation (12.13.4).

12.32 Show Eq. (12.13.6).

12.33 Fig. 12.13.2 defines the scattering matrix S. Explain how the principle of linear superposition may be used to show the general relationship

$$\begin{bmatrix} E'_+ \\ E_- \end{bmatrix} = S \begin{bmatrix} E_+ \\ E'_- \end{bmatrix}$$

between incoming and outgoing fields.

12.34 Show the two properties of the matrix $\psi_m(z)$ stated in Eqs. (12.13.13) and (12.13.14).

12.35 Show Eqs. (12.13.25).

12.36 The reflection response of a stack of four dielectrics has been found to be

$$R(z) = \frac{-0.25 + 0.0313z^{-1} + 0.2344z^{-2} - 0.2656z^{-3} + 0.25z^{-4}}{1 - 0.125z^{-1} + 0.0664z^{-3} - 0.0625z^{-4}}$$

Determine the reflection coefficients $\{\rho_0, \rho_1, \rho_2, \rho_3, \rho_4\}$.

12.37 *Computer Experiment.* It is desired to probe the structure of a stack of dielectrics from its reflection response. To this end, a unit impulse is sent incident on the stack and the reflection response is measured as a function of time.

It is known in advance (although this is not necessary) that the stack consists of four equal travel-time slabs stacked in front of a semi-infinite medium.

Thirteen samples of the reflection response are collected as shown here. Determine the reflection coefficients $\{\rho_0, \rho_1, \rho_2, \rho_3, \rho_4\}$ by means of the dynamic predictive deconvolution procedure.

k	$R(k)$
0	−0.2500
1	0.0000
2	0.2344
3	−0.2197
4	0.2069
5	0.0103
6	0.0305
7	−0.0237
8	0.0093
9	−0.0002
10	0.0035
11	−0.0017
12	0.0004

12.38 *Computer Experiment.* Generate the results of Figures 5.16–5.17 and 5.25–5.26.

12.39 *Computer Experiment.* This problem illustrates the use of the dynamic predictive deconvolution method in the design of broadband terminations of transmission lines. The termination is constructed by the cascade of M equal travel-time segments of transmission lines such that the overall reflection response of the structure approximates the desired reflection response. The characteristic impedances of the various segments are obtainable from the reflection coefficients $\{\rho_0, \rho_1, \ldots, \rho_M\}$. The reflection response $R(\omega)$ of the structure is a periodic function of ω with period $\omega_s = 2\pi/T_2$, where T_2 is the two-way travel time delay of each segment. The design procedure is illustrated by the following example: The desired frequency response $R(\omega)$ is defined over one Nyquist period, as shown in Fig. 5.39:

$$R(\omega) = \begin{cases} 0, & \text{for } 0.25\omega_2 \le \omega \le 0.75\omega_s \\ 0.9, & \text{for } 0 \le \omega < 0.25\omega_s \text{ and } 0.75\omega_s < \omega \le \omega_s \end{cases}$$

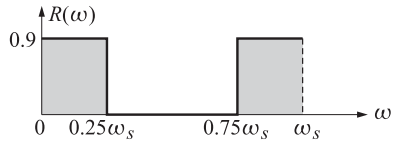


Fig. 12.16.1 Desired reflection frequency response.

(a) Using the Fourier series method of designing digital filters, design an $N = 21$-tap filter with impulse response $R(k)$, $k = 0, 1, \ldots, N-1$, whose frequency response

approximates the desired response defined above. Window the designed reflection impulse response $R(k)$ by a length-N Hamming window. Plot the magnitude frequency response of the windowed reflection series over one Nyquist interval $0 \le \omega \le \omega_s$.

(b) For $M = 6$, send the N samples of the windowed reflection series through the dynamic predictive deconvolution function **dpd** to obtain the polynomials $A_M(z)$ and $B_M(z)$ and the reflection coefficients $\{\rho_0, \rho_1, \ldots, \rho_M\}$. Plot the magnitude response of the structure; that is, plot

$$|R(\omega)| = \left|\frac{B_M(z)}{A_M(z)}\right|, \quad z = \exp(j\omega T_2) = \exp\left(2\pi j \frac{\omega}{\omega_s}\right)$$

and compare it with the windowed response of part (a). To facilitate the comparison, plot both responses of parts (a) and (b) on the same graph.

(c) Repeat part (b) for $M = 2$, $M = 3$, and $M = 10$.

(d) Repeat parts (a) through (c) for $N = 31$ reflection series samples.

(e) Repeat parts (a) through (c) for $N = 51$.

12.40 Show that the performance matrix P defined by Eq. (12.14.11) has trace equal to $M + 1$.

12.41 *Computer Experiment.* Reproduce the results of Figs. 12.14.1 through 12.14.6.

12.42 *Computer Experiment – ARIMA Models.* The file `GNPC96.dat` contains the data for the U.S. Real Gross National Product (in billions of chained 2005 dollars, measured quarterly and seasonally adjusted.) In this experiment, you will test whether an ARIMA$(p, 1, 0)$ model is appropriate for these data.

Extract the data from 1980 onwards and take their log. This results in $N = 119$ observations, $y_n = \ln(\text{GNP}_n)$, $n = 0, 1, \ldots, N-1$. Because of the upward trend of the data, define the length-$(N-1)$ differenced signal $z_n = y_n - y_{n-1}$, for $n = 1, 2, \ldots, N-1$. Then, subtract its sample mean, say μ, to get the signal $x_n = z_n - \mu$, for $n = 1, 2, \ldots, N-1$.

a. Calculate and plot the first $M = 24$ sample autocorrelation lags $R(k)$, $0 \le k \le M$, of the signal x_n. Send these into the Levinson algorithm to determine and plot the corresponding reflection coefficients γ_p, for $p = 1, 2 \ldots, M$. Add on that graph the 95% confidence bands, that is, the horizontal lines at $\pm 1.96/\sqrt{N}$. Based on this plot, determine a reasonable value for the order p of an autoregressive model of fitting the x_n data.

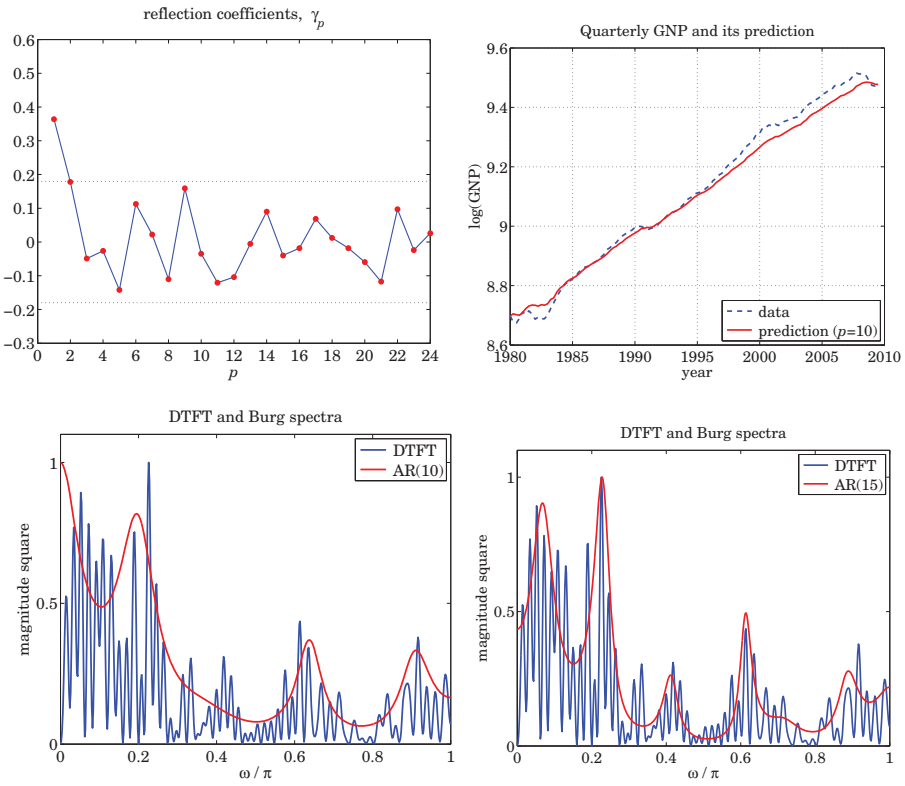
b. Check the chosen value of p against also the FPE, AIC, and MDL criteria, that is, by plotting them versus $p = 1, 2, \ldots, M$, and identifying their minimum:

$$\text{FPE}_p = E_p \frac{N+p+1}{N-p-1}, \quad \text{AIC}_p = N \ln E_p + 2(p+1), \quad \text{MDL}_p = N \ln E_p + (p+1) \ln N$$

c. For the chosen value of p, use the Yule-Walker method to determine the linear prediction error filter $\mathbf{a}_p$ of order p, then, calculate the one-step-ahead prediction $\hat{x}_{n/n-1}$, add the mean μ to get the prediction $\hat{z}_{n/n-1}$, and undo the differencing operation to compute the prediction $\hat{y}_{n/n-1}$ of y_n. There is a small subtlety here that has to do with the initial value of y_n. On the same graph plot y_n and its prediction.

d. Repeat part (c) for a couple of other values of p, say $p = 1$ and $p = 10$.

e. Calculate the DTFT $|X(\omega)|^2$ of the data x_n over $0 \le \omega \le \pi$. For the chosen order p, calculate the corresponding AR power spectrum but this time use the Burg method to determine the order-p prediction filter. Plot the DTFT and AR spectra on the same graph, but for convenience in comparing them, normalize both spectra to their maximum values. Investigate if higher values of p can model these spectra better, for example, try the orders $p = 10$ and $p = 15$. Some example graphs are included below.



12.43 *Computer Experiment – Wiener Filter Design.* It is desired to design a Wiener filter to enhance a sinusoidal signal buried in noise. The noisy sinusoidal signal is given by

$$x_n = s_n + v_n, \qquad \text{where } s_n = \sin(\omega_0 n)$$

with $\omega_0 = 0.075\pi$. The noise component v_n is related to the secondary signal y_n by

$$v_n = y_n + y_{n-1} + y_{n-2} + y_{n-3} + y_{n-4} + y_{n-5} + y_{n-6}$$

a. Generate $N = 200$ samples of the signal y_n by assuming that it is an AR(4) process with reflection coefficients:

$$\{\gamma_1, \gamma_2, \gamma_3, \gamma_4\} = \{0.5, -0.5, 0.5, -0.5\}$$

The variance σ_ϵ^2 of the driving white noise of the model must be chosen in such a way as to make the variance σ_v^2 of the noise component v_n approximately $\sigma_v^2 = 0.5$, such that the two terms s_n and v_n of x_n have approximately equal strengths, that is, 0 dB signal-to-noise ratio.

(This can be done by writing $v_n = \mathbf{c}^T \mathbf{y}(n)$ and therefore, $\sigma_v^2 = \mathbf{c}^T R \mathbf{c}$, where $\mathbf{c}$ is a 7-dimensional column vector of ones, and R is the order-6 autocorrelation matrix, you can then write $\sigma_v^2 = \sigma_y^2 \mathbf{c}^T R_{\text{norm}} \mathbf{c}$, where R_{norm} has all its entries normalized by

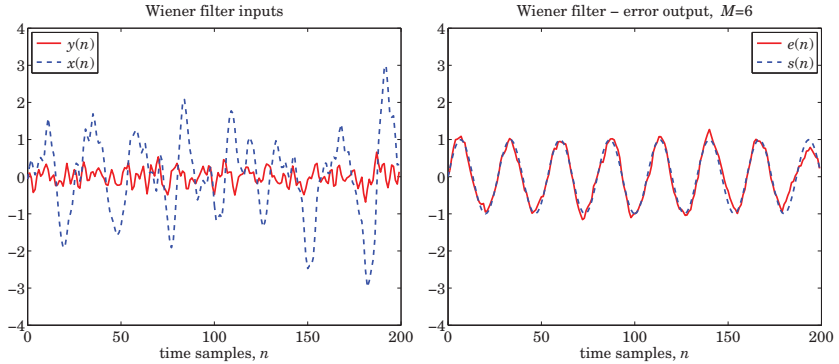
$R(0) = \sigma_y^2$. You can easily determine R_{norm} by doing a maximum entropy extension to order six, starting with the four reflection coefficients and setting $\gamma_5 = \gamma_6 = 0$.)

In generating y_n make sure that the transients introduced by the filter have died out. Then, generate the corresponding N samples of the signal x_n. On the same graph, plot x_n together with the desired signal s_n. On a separate graph (but using the same vertical scales as the previous one) plot the reference signal y_n versus n.

b. For $M = 4$, design a Wiener filter of order-M based on the generated signal blocks $\{x_n, y_n\}$, $n = 0, 1, \ldots, N-1$, and realize it in both the direct and lattice forms.

c. Using the *lattice* form, filter the signals x_n, y_n through the designed filter and generate the outputs $\hat{x}_n, e_n$. Explain why e_n should be an estimate of the desired signal s_n. On the same graph, plot e_n and s_n using the same vertical scales as in part (a).

d. Repeat parts (b) and (c) for filter orders $M = 5, 6, 7, 8$. Discuss the improvement obtained with increasing order. What is the smallest M that would, at least theoretically, result in $e_n = s_n$? Some example graphs are included below.



13

Kalman Filtering

13.1 State-Space Models

The Kalman filter is based on a state/measurement model of the form:

$$\begin{aligned} \mathbf{x}_{n+1} &= A_n \mathbf{x}_n + \mathbf{w}_n \qquad \text{(state model)} \\ \mathbf{y}_n &= C_n \mathbf{x}_n + \mathbf{v}_n \qquad \text{(measurement model)} \end{aligned} \tag{13.1.1}$$

where $\mathbf{x}_n$ is a p-dimensional state vector and $\mathbf{y}_n$, an r-dimensional vector of observations. The $p \times p$ state-transition matrix A_n and $r \times p$ measurement matrix C_n may depend on time n. The signals $\mathbf{w}_n, \mathbf{v}_n$ are assumed to be mutually-independent, zero-mean, white-noise signals with known covariance matrices Q_n and R_n:

$$\begin{aligned} E[\mathbf{w}_n \mathbf{w}_i^T] &= Q_n \delta_{ni} \\ E[\mathbf{v}_n \mathbf{v}_i^T] &= R_n \delta_{ni} \\ E[\mathbf{w}_n \mathbf{v}_i^T] &= 0 \end{aligned} \tag{13.1.2}$$

The model is iterated starting at $n = 0$. The initial state vector $\mathbf{x}_0$ is assumed to be random and independent of $\mathbf{w}_n, \mathbf{v}_n$, but with a known mean $\bar{\mathbf{x}}_0 = E[\mathbf{x}_0]$ and covariance matrix $\Sigma_0 = E[(\mathbf{x}_0 - \bar{\mathbf{x}}_0)(\mathbf{x}_0 - \bar{\mathbf{x}}_0)^T]$. We will assume, for now, that $\mathbf{x}_0, \mathbf{w}_n, \mathbf{v}_n$ are normally distributed, and therefore, their statistical description is completely determined by their means and covariances. A non-zero cross-covariance $E[\mathbf{w}_n \mathbf{v}_i^T] = S_n \delta_{ni}$ may also be assumed. A scalar version of the model was discussed in Chap. 11.

The covariance matrices Q_n, R_n have dimensions $p \times p$ and $r \times r$, but they need not have full rank (which would mean that some of the components of $\mathbf{x}_n$ or $\mathbf{y}_n$ would, in an appropriate basis, be noise-free.) For example, to allow the possibility of fewer state noise components, the model (13.1.1) is often written in the form:

$$\begin{aligned} \mathbf{x}_{n+1} &= A_n \mathbf{x}_n + G_n \mathbf{w}_n \qquad \text{(state model)} \\ \mathbf{y}_n &= C_n \mathbf{x}_n + \mathbf{v}_n \qquad \text{(measurement model)} \end{aligned} \tag{13.1.3}$$