7 ADAPTIVE LATTICE-BASED RLS ALGORITHMS

7.1 INTRODUCTION

There are a large number of algorithms that solve the least-squares problem in a recursive form. In particular, the algorithms based on the lattice realization are very attractive because they allow modular implementation and require a reduced number of arithmetic operations (of order N) [1]-[7]. As a consequence, the lattice recursive least-squares (LRLS) algorithms are considered fast implementations of the RLS problem.

The LRLS algorithms are derived by solving the forward and backward linear prediction problems simultaneously. The lattice-based formulation provides the prediction and the general adaptive filter (joint-process estimation) solutions of all intermediate orders from 1 to N simultaneously. Consequently, the order of the adaptive filter can be increased or decreased without affecting the lower order solutions. This property allows the user to activate or deactivate sections of the lattice realization in real time according to performance requirements.

Unlike the RLS algorithm previously discussed, which requires only time-recursive equations, the lattice RLS algorithms use time-update and order-update equations. A key feature of the LRLS algorithms is that the prediction process discloses the properties (or the model) of the input signal. The internal signals of the prediction part retain in a sense nonredundant information of the input signal that can be utilized in a decoupled form in the following processing. This mechanism is inherently built in the lattice algorithm derivations.

The performance of the LRLS algorithms when implemented with infinite-precision arithmetic is identical to that of any other RLS algorithm. However, in finite-precision implementation each algorithm will perform differently.

In this chapter, several forms of the LRLS algorithm are presented. First, the standard LRLS algorithm based on *a posteriori* errors is presented, followed by the normalized version. The algorithms with error feedback are also derived. Finally, the LRLS algorithm based on *a priori* errors is described.

P.S.R. Diniz, *Adaptive Filtering*, DOI: 10.1007/978-0-387-68606-6_7,

7.2 RECURSIVE LEAST-SQUARES PREDICTION

The solutions of the RLS forward and backward prediction problems are essential to derive the order-updating equations inherent to the LRLS algorithms. In both cases, the results are derived following the same derivation procedure as in the conventional RLS algorithm, since the only distinct feature of the prediction problems is the definition of the reference signal $d(k)$. For example, in the forward prediction case we have $d(k) = x(k)$ whereas the input signal vector has the sample $x(k-1)$ as the most recent data. For the backward prediction case $d(k) = x(k-i-1)$, where the index i defines the sample in the past which we wish to predict, and the input signal vector has $x(k)$ as the most recent data. In this section, these solutions are studied and the results demonstrate how information can be exchanged between the forward and backward predictor solutions.

7.2.1 Forward Prediction Problem

The objective of the forward prediction is to predict a future sample of a given input sequence using the currently available information of the sequence. For example, one can try to predict the value of $x(k)$ using past samples $x(k-1), x(k-2)\ldots$, through an FIR prediction filter with $i+1$ coefficients as

$$y_f(k,i+1) = \mathbf{w}_f^T(k,i+1)\mathbf{x}(k-1,i+1) \tag{7.1}$$

where $y_f(k,i+1)$ is the predictor output signal,

$$\mathbf{w}_f(k,i+1) = [w_{f0}(k)\; w_{f1}(k)\ldots w_{fi}(k)]^T$$

is the FIR forward prediction coefficient vector, and

$$\mathbf{x}(k-1,i+1) = [x(k-1)\; x(k-2)\ldots x(k-i-1)]^T$$

is the available input signal vector. The second variable included in the vectors of equation (7.1) is to indicate the vector dimension, since it is required in the order-updating equations of the LRLS algorithm. This second variable will be included where needed in the present chapter.

The instantaneous *a posteriori* forward prediction error is given by

$$\varepsilon_f(k,i+1) = x(k) - \mathbf{w}_f^T(k,i+1)\mathbf{x}(k-1,i+1) \tag{7.2}$$

For the RLS formulation of the forward prediction problem, define the weighted forward prediction error vector as

$$\boldsymbol{\varepsilon}_f(k,i+1) = \hat{\mathbf{x}}(k) - \mathbf{X}^T(k-1,i+1)\mathbf{w}_f(k,i+1) \tag{7.3}$$

where

$$\hat{\mathbf{x}}(k) = [x(k)\;\; \lambda^{1/2}x(k-1)\;\; \lambda x(k-2)\ldots\lambda^{k/2}x(0)]^T$$

$$\boldsymbol{\varepsilon}_f(k,i+1) = [\varepsilon_f(k,i+1)\ \lambda^{1/2}\varepsilon_f(k-1,i+1)\ \lambda\varepsilon_f(k-2,i+1)\ldots\lambda^{k/2}\varepsilon_f(0,i+1)]^T$$

and

$$\mathbf{X}(k-1,i+1) = \begin{bmatrix} x(k-1) & \lambda^{1/2}x(k-2) & \cdots & \lambda^{(k-2)/2}x(1) & \lambda^{(k-1)/2}x(0) & 0 \\ x(k-2) & \lambda^{1/2}x(k-3) & \cdots & \lambda^{(k-2)/2}x(0) & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ x(k-i-1) & \lambda^{1/2}x(k-i-2) & \cdots & 0 & 0 & 0 \end{bmatrix}$$

It is straightforward to show that $\boldsymbol{\varepsilon}_f(k,i+1)$ can be rewritten as

$$\boldsymbol{\varepsilon}_f(k,i+1) = \mathbf{X}^T(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i+1) \end{bmatrix} \tag{7.4}$$

The objective function that we want to minimize in the least-squares sense is the forward prediction error given by

$$\begin{aligned}\xi_f^d(k,i+1) &= \boldsymbol{\varepsilon}_f^T(k,i+1)\boldsymbol{\varepsilon}_f(k,i+1) \\ &= \sum_{l=0}^{k}\lambda^{k-l}\varepsilon_f^2(l,i+1) \\ &= \sum_{l=0}^{k}\lambda^{k-l}[x(l)-\mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1)]^2 \end{aligned} \tag{7.5}$$

By differentiating $\xi_f^d(k,i+1)$ with respect to $\mathbf{w}_f(k,i+1)$ and equating the result to zero, we can find the optimum coefficient vector that minimizes the objective function, namely,

$$\begin{aligned}\mathbf{w}_f(k,i+1) &= \left[\sum_{l=0}^{k}\lambda^{k-l}\mathbf{x}(l-1,i+1)\mathbf{x}^T(l-1,i+1)\right]^{-1}\sum_{l=0}^{k}\lambda^{k-l}\mathbf{x}(l-1,i+1)x(l) \\ &= [\mathbf{X}(k-1,i+1)\mathbf{X}^T(k-1,i+1)]^{-1}\mathbf{X}(k-1,i+1)\hat{\mathbf{x}}(k) \\ &= \mathbf{R}_{Df}^{-1}(k-1,i+1)\mathbf{p}_{Df}(k,i+1) \end{aligned} \tag{7.6}$$

where $\mathbf{R}_{Df}(k-1,i+1)$ is equal to the deterministic correlation matrix $\mathbf{R}_D(k-1)$ of order $i+1$ and $\mathbf{p}_{Df}(k,i+1)$ is the deterministic cross-correlation vector between $x(l)$ and $\mathbf{x}(l-1,i+1)$.

The exponentially weighted sum of squared errors can be written as (see equation (7.5)):

$$\begin{aligned}
\xi_f^d(k,i+1) &= \sum_{l=0}^{k} \lambda^{k-l} \left\{ x^2(l) - 2x(l)\mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1) \right. \\
&\quad \left. + \left[\mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1)\right]^2 \right\} \\
&= \sum_{l=0}^{k} \lambda^{k-l} \left[x^2(l) - x(l)\mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1) \right] \\
&\quad + \sum_{l=0}^{k} \lambda^{k-l} \left[-x(l) + \mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1) \right] \mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1) \\
&= \sum_{l=0}^{k} \lambda^{k-l} x(l) \left[x(l) - \mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1) \right] \\
&\quad + \left[\textstyle\sum_{l=0}^{k} - \lambda^{k-l} x(l)\mathbf{x}^T(l-1,i+1) \right. \\
&\quad \left. + \mathbf{w}_f^T(k,i+1) \textstyle\sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}(l-1,i+1)\mathbf{x}^T(l-1,i+1) \right] \mathbf{w}_f(k,i+1) \qquad (7.7)
\end{aligned}$$

If we replace equation (7.6) in the second term of the last relation above, it can be shown by using the fact that $\mathbf{R}_D(k-1)$ is symmetric that this term is zero. Therefore, the minimum value of $\xi_f^d(k,i+1)$[1] is given by

$$\begin{aligned}
\xi_{f_{\min}}^d(k,i+1) &= \sum_{l=0}^{k} \lambda^{k-l} x(l)[x(l) - \mathbf{x}^T(l-1,i+1)\mathbf{w}_f(k,i+1)] \\
&= \sum_{l=0}^{k} \lambda^{k-l} x^2(l) - \mathbf{p}_{Df}^T(k,i+1)\mathbf{w}_f(k,i+1) \\
&= \sigma_f^2(k) - \mathbf{w}_f^T(k,i+1)\mathbf{p}_{Df}(k,i+1) \qquad (7.8)
\end{aligned}$$

By combining equation (7.6) for $\mathbf{w}_f(k,i)$ and equation (7.8) for $\xi_{f_{\min}}^d(k,i+1)$ the following matrix equation can be obtained

$$\begin{bmatrix} \sigma_f^2(k) & \mathbf{p}_{Df}^T(k,i+1) \\ \mathbf{p}_{Df}(k,i+1) & \mathbf{R}_{Df}(k-1,i+1) \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i+1) \end{bmatrix} = \begin{bmatrix} \xi_{f_{\min}}^d(k,i+1) \\ \mathbf{0} \end{bmatrix} \qquad (7.9)$$

[1]Notice that no special notation was previously used for the minimum value of the RLS objective function. However, when deriving the lattice algorithms, this definition is necessary.

Since $\sigma_f^2(k) = \sum_{l=0}^{k} \lambda^{k-l} x^2(l)$ and $\mathbf{p}_{Df}(k, i+1) = \sum_{l=0}^{k} \lambda^{k-l} \mathbf{x}(l-1, i+1) x(l)$, it is possible to conclude that the leftmost term of equation (7.9) can be rewritten as

$$\begin{aligned}
&\begin{bmatrix} \sum_{l=0}^{k} \lambda^{k-l} x^2(l) & \sum_{l=0}^{k} \lambda^{k-l} \mathbf{x}^T(l-1, i+1) x(l) \\ \sum_{l=0}^{k} \lambda^{k-l} \mathbf{x}(l-1, i+1) x(l) & \sum_{l=0}^{k} \lambda^{k-l} \mathbf{x}(l-1, i+1) \mathbf{x}^T(l-1, i+1) \end{bmatrix} \\
&= \sum_{l=0}^{k} \lambda^{k-l} \begin{bmatrix} x(l) \\ \mathbf{x}(l-1, i+1) \end{bmatrix} [x(l) \;\; \mathbf{x}^T(l-1, i+1)] \\
&= \mathbf{R}_D(k, i+2)
\end{aligned} \tag{7.10}$$

Therefore,

$$\mathbf{R}_D(k, i+2) \begin{bmatrix} 1 \\ -\mathbf{w}_f(k, i+1) \end{bmatrix} = \begin{bmatrix} \xi^d_{f_{\min}}(k, i+1) \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{R}_D(k, i+2)$ corresponds to $\mathbf{R}_D(k)$ used in the previous chapter with dimension $i+2$. The above equation relates the deterministic correlation matrix of order $i+2$ to the minimum least-squares forward prediction error. The appropriate partitioning of matrix $\mathbf{R}_D(k, i+2)$ enables the derivation of the order-updating equation for the predictor tap coefficients, as will be discussed later.

7.2.2 Backward Prediction Problem

The objective of the backward predictor is to generate an estimate of a past sample of a given input sequence using the currently available information of the sequence. For example, sample $x(k-i-1)$ can be estimated from $\mathbf{x}(k, i+1)$, through an FIR backward prediction filter with $i+1$ coefficients as

$$y_b(k, i+1) = \mathbf{w}_b^T(k, i+1) \mathbf{x}(k, i+1) \tag{7.11}$$

where $y_b(k, i+1)$ is the backward predictor output signal, and

$$\mathbf{w}_b^T(k, i+1) = [w_{b0}(k) \; w_{b1}(k) \ldots w_{bi}(k)]^T$$

is the FIR backward prediction coefficient vector.

The instantaneous *a posteriori* backward prediction error is given by

$$\varepsilon_b(k, i+1) = x(k-i-1) - \mathbf{w}_b^T(k, i+1) \mathbf{x}(k, i+1) \tag{7.12}$$

The weighted backward prediction error vector is defined as

$$\boldsymbol{\varepsilon}_b(k, i+1) = \hat{\mathbf{x}}(k-i-1) - \mathbf{X}^T(k, i+1) \mathbf{w}_b(k, i+1) \tag{7.13}$$

where

$$\hat{\mathbf{x}}(k-i-1) = [x(k-i-1) \;\; \lambda^{1/2} x(k-i-2) \ldots \lambda^{(k-i-1)/2} x(0) \;\; 0 \ldots 0]^T$$

$$\boldsymbol{\varepsilon}_b(k, i+1) = [\varepsilon_b(k, i+1) \;\; \lambda^{1/2} \varepsilon_b(k-1, i+1) \ldots \lambda^{k/2} \varepsilon_b(0, i+1)]^T$$

and

$$\mathbf{X}(k, i+1) = \begin{bmatrix} x(k) & \lambda^{1/2}x(k-1) & \cdots & \lambda^{(k-1)/2}x(1) & \lambda^{(k)/2}x(0) \\ x(k-1) & \lambda^{1/2}x(k-2) & \cdots & \lambda^{(k-2)/2}x(0) & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ x(k-i) & \lambda^{1/2}x(k-i-1) & \cdots & 0 \cdots & 0 \end{bmatrix}$$

The error vector can be rewritten as

$$\boldsymbol{\varepsilon}_b(k, i+1) = \mathbf{X}^T(k, i+2) \begin{bmatrix} -\mathbf{w}_b(k, i+1) \\ 1 \end{bmatrix} \tag{7.14}$$

The objective function to be minimized in the backward prediction problem is given by

$$\begin{aligned} \xi_b^d(k, i+1) &= \boldsymbol{\varepsilon}_b^T(k, i+1)\boldsymbol{\varepsilon}_b(k, i+1) \\ &= \sum_{l=0}^{k} \lambda^{k-l} \varepsilon_b^2(l, i+1) \\ &= \sum_{l=0}^{k} \lambda^{k-l} [x(l-i-1) - \mathbf{x}^T(l, i+1)\mathbf{w}_b(k, i+1)]^2 \end{aligned} \tag{7.15}$$

The optimal solution for the coefficient vector is

$$\begin{aligned} \mathbf{w}_b(k, i+1) &= \left[\sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}(l, i+1)\mathbf{x}^T(l, i+1)\right]^{-1} \sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}(l, i+1)x(l-i-1) \\ &= [\mathbf{X}(k, i+1)\mathbf{X}^T(k, i+1)]^{-1}\mathbf{X}(k, i+1)\hat{\mathbf{x}}(k-i-1) \\ &= \mathbf{R}_{Db}^{-1}(k, i+1)\mathbf{p}_{Db}(k, i+1) \end{aligned} \tag{7.16}$$

where $\mathbf{R}_{Db}(k, i+1)$ is equal to the deterministic correlation matrix $\mathbf{R}_D(k)$ of order $i+1$, and $\mathbf{p}_{Db}(k, i+1)$ is the deterministic cross-correlation vector between $x(l-i-1)$ and $\mathbf{x}(l, i+1)$.

Using the same procedure to derive the minimum least-squares solution in the RLS problem, it can be shown that the minimum value of $\xi_b^d(k)$ is given by

$$\begin{aligned} \xi_{b_{\min}}^d(k, i+1) &= \sum_{l=0}^{k} \lambda^{k-l} x(l-i-1)[x(l-i-1) - \mathbf{x}^T(l, i+1)\mathbf{w}_b(k, i+1)] \\ &= \sum_{l=0}^{k} \lambda^{k-l} x^2(l-i-1) - \mathbf{p}_{Db}^T(k, i+1)\mathbf{w}_b(k, i+1) \\ &= \sigma_b^2(k) - \mathbf{w}_b^T(k, i+1)\mathbf{p}_{Db}(k, i+1) \end{aligned} \tag{7.17}$$

By combining equations (7.16) and (7.17), we obtain

$$\begin{aligned}
&\begin{bmatrix} \mathbf{R}_{Db}(k,i+1) & \mathbf{p}_{Db}(k,i+1) \\ \mathbf{p}_{Db}^T(k,i+1) & \sigma_b^2(k) \end{bmatrix} \begin{bmatrix} -\mathbf{w}_b(k,i+1) \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} \sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}(l,i+1)\mathbf{x}^T(l,i+1) & \sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}(l,i+1)x(l-i-1) \\ \sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}^T(l,i+1)x(l-i-1) & \sum_{l=0}^{k} \lambda^{k-l}x^2(l-i-1) \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} -\mathbf{w}_b(k,i+1) \\ 1 \end{bmatrix} \\
&= \mathbf{R}_D(k,i+2) \begin{bmatrix} -\mathbf{w}_b(k,i+1) \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{0} \\ \xi_{b_{\min}}^d(k,i+1) \end{bmatrix} \qquad (7.18)
\end{aligned}$$

where $\mathbf{R}_D(k,i+2)$ is equal to $\mathbf{R}_D(k)$ of dimension $i+2$. The above equation relates the deterministic correlation matrix of order $i+1$ to the minimum least-squares backward prediction error. This equation is important in the derivation of the order-updating equation for the backward predictor tap coefficients. This issue is discussed in the following section.

7.3 ORDER-UPDATING EQUATIONS

The objective of this section is to derive the order-updating equations for the forward and backward prediction errors. These equations are the starting point to generate the lattice realization.

7.3.1 A New Parameter $\delta(k,i)$

Using the results of equations (7.9) and (7.10), and the decomposition of $\mathbf{R}_D(k,i+2)$ given in equation (7.18), we can show that

$$\begin{aligned}
\mathbf{R}_D(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \\ 0 \end{bmatrix} &= \begin{bmatrix} \mathbf{R}_D(k,i+1) & \mathbf{p}_{Db}(k,i+1) \\ \mathbf{p}_{Db}^T(k,i+1) & \sigma_b^2(k) \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} \xi_{f_{\min}}^d(k,i) \\ \mathbf{0} \\ \mathbf{p}_{Db}^T(k,i+1)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \end{bmatrix} \end{bmatrix} \\
&= \begin{bmatrix} \xi_{f_{\min}}^d(k,i) \\ \mathbf{0} \\ \delta_f(k,i) \end{bmatrix} \qquad (7.19)
\end{aligned}$$

where relation (7.9) was employed in the second equality. From the last element relation of the above vector and the definition of $\mathbf{p}_{Db}(k,i+1)$, we obtain

$$\begin{aligned}\delta_f(k,i) &= \sum_{l=0}^{k}\lambda^{k-l}x(l)x(l-i-1) - \sum_{l=0}^{k}\lambda^{k-l}x(l-i-1)\mathbf{x}^T(l-1,i)\mathbf{w}_f(k,i)\\ &= \sum_{l=0}^{k}\lambda^{k-l}x(l)x(l-i-1) - \sum_{l=0}^{k}\lambda^{k-l}x(l-i-1)y_f(l,i)\\ &= \sum_{l=0}^{k}\lambda^{k-l}\varepsilon_f(l,i)x(l-i-1)\end{aligned}$$

and $y_f(l,i) = \mathbf{x}^T(l-1,i)\mathbf{w}_f(k,i)$ is the output of a forward prediction filter of order $i-1$. Note that the parameter $\delta_f(k,i)$ can be interpreted as the deterministic cross-correlation between the forward prediction error $\varepsilon_f(l,i)$ with the coefficients fixed at $\mathbf{w}_f(k,i)$ and the desired signal of the backward predictor filter $x(l-i-1)$.

Similarly, using the results of equations (7.17) and (7.18) it can be shown that

$$\begin{aligned}\mathbf{R}_D(k,i+2)\begin{bmatrix}0\\ -\mathbf{w}_b(k-1,i)\\ 1\end{bmatrix} &= \begin{bmatrix}\sigma_f^2(k) & \mathbf{p}_{Df}^T(k,i+1)\\ \mathbf{p}_{Df}(k,i+1) & \mathbf{R}_D(k-1,i+1)\end{bmatrix}\begin{bmatrix}0\\ -\mathbf{w}_b(k-1,i)\\ 1\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{p}_{Df}^T(k,i+1)\begin{bmatrix}-\mathbf{w}_b(k-1,i)\\ 1\end{bmatrix}\\ \mathbf{0}\\ \xi^d_{b_{\min}}(k-1,i)\end{bmatrix}\\ &= \begin{bmatrix}\delta_b(k,i)\\ \mathbf{0}\\ \xi^d_{b_{\min}}(k-1,i)\end{bmatrix}\end{aligned} \tag{7.20}$$

where in the second equality we applied the result of equation (7.18), and

$$\begin{aligned}\delta_b(k,i) &= \sum_{l=0}^{k}\lambda^{k-l}x(l-i-1)x(l) - \sum_{l=0}^{k}\lambda^{k-l}x(l)\mathbf{x}^T(l-1,i)\mathbf{w}_b(k-1,i)\\ &= \sum_{l=0}^{k}\lambda^{k-l}x(l-i-1)x(l) - \sum_{l=0}^{k}\lambda^{k-l}x(l)y_b(l-1,i)\\ &= \sum_{l=0}^{k}\lambda^{k-l}\varepsilon_b(l-1,i)x(l)\end{aligned}$$

where $y_b(l-1,i) = \mathbf{x}^T(l-1,i)\mathbf{w}_b(k-1,i)$ is the output of a backward prediction filter of order $i-1$ with the input data of instant $l-1$, when the coefficients of the predictor are $\mathbf{w}_b(k-1,i)$. The parameter $\delta_b(k,i)$ can be interpreted as the deterministic cross-correlation between the backward prediction error $\varepsilon_b(l-1,i)$ and the desired signal of the forward predictor filter $x(l)$.

In equations (7.19) and (7.20) two new parameters were defined, namely $\delta_f(k,i)$ and $\delta_b(k,i)$. In the following derivations we will show that these parameters are equal. If $\mathbf{R}_D(k,i+2)$ is premultiplied by $[0 \quad -\mathbf{w}_b^T(k-1,i) \quad 1]$ and postmultiplied by $[1 \quad -\mathbf{w}_f(k,i) \quad 0]^T$, it can be shown that

$$[0 \quad -\mathbf{w}_b^T(k-1,i) \quad 1]\ \mathbf{R}_D(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \\ 0 \end{bmatrix} = \delta_f(k,i) \tag{7.21}$$

By transposing the first and last terms of equation (7.20) the following relation is obtained

$$[0 \quad -\mathbf{w}_b^T(k-1,i) \quad 1]\ \mathbf{R}_D(k,i+2) = [\delta_b(k,i) \quad \mathbf{0}^T \quad \xi^d_{b_{\min}}(k-1,i)] \tag{7.22}$$

By substituting this result in equation (7.21), we obtain

$$[\delta_b(k,i) \quad \mathbf{0}^T \quad \xi^d_{b_{\min}}(k-1,i)]\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \\ 0 \end{bmatrix} = \delta_b(k,i) \tag{7.23}$$

Therefore, from equations (7.21) and (7.23) we conclude that

$$\delta_f(k,i) = \delta_b(k,i) = \delta(k,i) \tag{7.24}$$

In effect, the deterministic cross-correlations between $\varepsilon_f(l,i)$ and $x(l-i-1)$ and between $\varepsilon_b(l-1,i)$ and $x(l)$ are equal.

7.3.2 Order Updating of $\xi^d_{b_{\min}}(k,i)$ and $\mathbf{w}_b(k,i)$

The order updating of the minimum LS error and the tap coefficients for the backward predictor can be deduced by multiplying equation (7.19) by the scalar $\delta(k,i)/\xi^d_{f_{\min}}(k,i)$, i.e.,

$$\frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)}\mathbf{R}_D(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \\ 0 \end{bmatrix} = \begin{bmatrix} \delta(k,i) \\ \mathbf{0} \\ \frac{\delta^2(k,i)}{\xi^d_{f_{\min}}(k,i)} \end{bmatrix} \tag{7.25}$$

Subtracting equation (7.20) from this result yields

$$\mathbf{R}_D(k,i+2)\begin{bmatrix} \frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)} \\ -\mathbf{w}_f(k,i)\frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)} + \mathbf{w}_b(k-1,i) \\ -1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -\xi^d_{b_{\min}}(k-1,i) + \frac{\delta^2(k,i)}{\xi^d_{f_{\min}}(k,i)} \end{bmatrix} \tag{7.26}$$

Comparing equations (7.18) and (7.26), we conclude that

$$\xi^d_{b_{\min}}(k,i+1) = \xi^d_{b_{\min}}(k-1,i) - \frac{\delta^2(k,i)}{\xi^d_{f_{\min}}(k,i)} \tag{7.27}$$

and

$$\mathbf{w}_b(k,i+1) = \begin{bmatrix} 0 \\ \mathbf{w}_b(k-1,i) \end{bmatrix} - \frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)} \begin{bmatrix} -1 \\ \mathbf{w}_f(k,i) \end{bmatrix} \tag{7.28}$$

7.3.3 Order Updating of $\xi^d_{f_{\min}}(k,i)$ and $\mathbf{w}_f(k,i)$

Similarly, by multiplying equation (7.20) by $\delta(k,i)/\xi^d_{b_{\min}}(k-1,i)$, we get

$$\frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}\mathbf{R}_D(k,i+2)\begin{bmatrix} 0 \\ -\mathbf{w}_b(k-1,i) \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\delta^2(k,i)}{\xi^d_{b_{\min}}(k-1,i)} \\ \mathbf{0} \\ \delta(k,i) \end{bmatrix} \tag{7.29}$$

Subtracting equation (7.29) from equation (7.19), it follows that

$$\mathbf{R}_D(k,i+2)\begin{bmatrix} 1 \\ \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}\mathbf{w}_b(k-1,i) - \mathbf{w}_f(k,i) \\ -\frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)} \end{bmatrix} = \begin{bmatrix} \xi^d_{f_{\min}}(k,i) - \frac{\delta^2(k,i)}{\xi^d_{b_{\min}}(k-1,i)} \\ \mathbf{0} \end{bmatrix} \tag{7.30}$$

Comparing this equation with equation (7.9), we conclude that

$$\xi^d_{f_{\min}}(k,i+1) = \xi^d_{f_{\min}}(k,i) - \frac{\delta^2(k,i)}{\xi^d_{b_{\min}}(k-1,i)} \tag{7.31}$$

and

$$\mathbf{w}_f(k,i+1) = \begin{bmatrix} \mathbf{w}_f(k,i) \\ 0 \end{bmatrix} - \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)} \begin{bmatrix} \mathbf{w}_b(k-1,i) \\ -1 \end{bmatrix} \tag{7.32}$$

7.3.4 Order Updating of Prediction Errors

The order updating of the *a posteriori* forward and backward prediction errors can be derived as described below. From the definition of *a posteriori* forward error, we have

$$\begin{aligned} \varepsilon_f(k,i+1) &= \mathbf{x}^T(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i+1) \end{bmatrix} \\ &= \mathbf{x}^T(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \\ 0 \end{bmatrix} + \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}\mathbf{x}^T(k,i+2)\begin{bmatrix} 0 \\ \mathbf{w}_b(k-1,i) \\ -1 \end{bmatrix} \\ &= \varepsilon_f(k,i) - \kappa_f(k,i)\varepsilon_b(k-1,i) \end{aligned} \tag{7.33}$$

where in the second equality we employed the order-updating equation (7.32) for the forward prediction coefficients. The coefficient $\kappa_f(k,i) = \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}$ is the so-called forward reflection coefficient.

The order updating of the *a posteriori* backward prediction error is obtained by using equation (7.28) as

$$\begin{aligned}
\varepsilon_b(k,i+1) &= \mathbf{x}^T(k,i+2)\begin{bmatrix} -\mathbf{w}_b(k,i+1) \\ 1 \end{bmatrix} \\
&= \mathbf{x}^T(k,i+2)\begin{bmatrix} 0 \\ -\mathbf{w}_b(k-1,i) \\ 1 \end{bmatrix} + \frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)}\mathbf{x}^T(k,i+2)\begin{bmatrix} -1 \\ \mathbf{w}_f(k,i) \\ 0 \end{bmatrix} \\
&= \varepsilon_b(k-1,i) - \kappa_b(k,i)\varepsilon_f(k,i)
\end{aligned} \tag{7.34}$$

where we employed the order-updating equation for the backward prediction coefficients (7.28) in the second equality. The coefficient $\kappa_b(k,i) = \frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)}$ is the backward reflection coefficient.

Equations (7.33) and (7.34) above can be implemented with a lattice section as illustrated in Fig. 7.1.a. An order-increasing lattice-based forward and backward predictor can be constructed as illustrated in Fig. 7.1.b. The coefficients $\kappa_b(k,i)$ and $\kappa_f(k,i)$ are often called reflection coefficients of the lattice realization.

In the first section of the lattice, the forward and backward prediction errors are equal to the input signal itself since no prediction is performed before the first lattice section; therefore

$$\varepsilon_b(k,0) = \varepsilon_f(k,0) = x(k) \tag{7.35}$$

and

$$\xi^d_{f_{\min}}(k,0) = \xi^d_{b_{\min}}(k,0) = \sum_{l=0}^{k} \lambda^{k-l}x^2(l) = x^2(k) + \lambda\xi^d_{f_{\min}}(k-1,0) \tag{7.36}$$

A closer look at equations (7.9) and (7.18) leads to the conclusion that the backward and forward predictors utilize the same information matrix $\mathbf{R}_D(k,i+2)$. This result was key in deriving the expressions for the *a posteriori* forward and backward prediction errors of equations (7.33) and (7.34). Of particular note, these expressions can be shown to be independent of the predictor tap coefficients. This result will be proved in the following section, which will present an updating formula for $\delta(k,i)$ that is not directly dependent on $\mathbf{w}_f(k,i)$ and $\mathbf{w}_b(k-1,i)$.

Now that all order-updating equations are available, it is necessary to derive the time-updating equations to allow the adaptation of the lattice predictor coefficients.

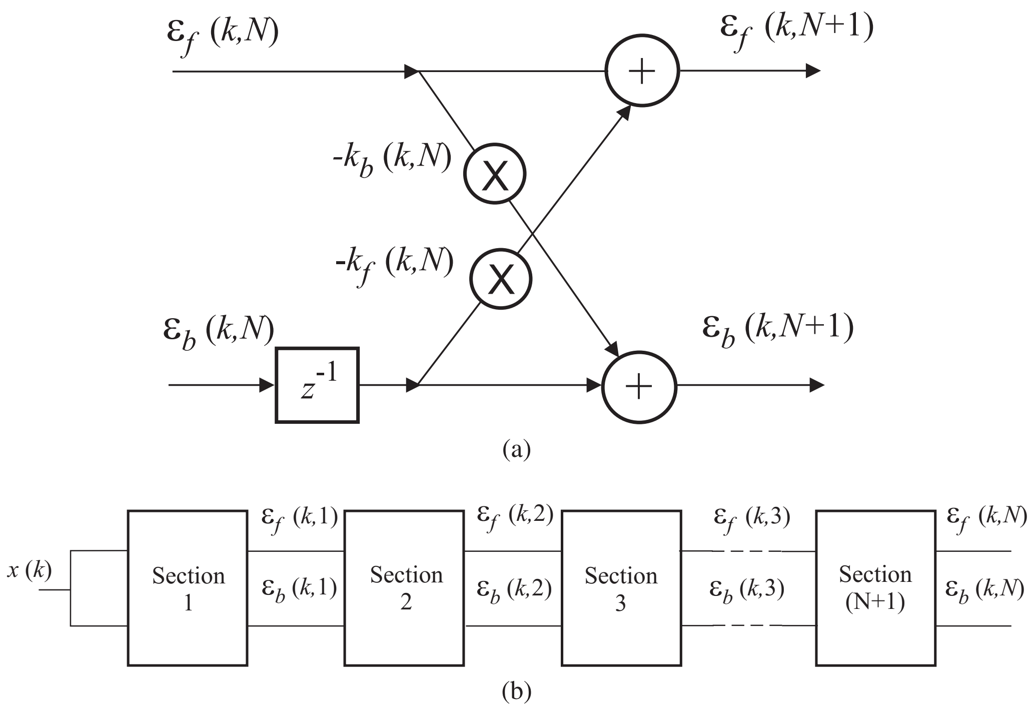

Figure 7.1 Least-squares lattice-based predictor.

7.4 TIME-UPDATING EQUATIONS

The time-updating equations are required to deal with the new incoming data that becomes available. Recall that up to this point in this text we have studied adaptive-filtering algorithms utilizing the new incoming data as soon as it becomes available. In this section, the time-updating equations for the internal quantities of the lattice algorithm are derived.

7.4.1 Time Updating for Prediction Coefficients

From equation (7.6), the time updating of the forward prediction filter coefficients is given by

$$\begin{aligned}\mathbf{w}_f(k,i) &= \mathbf{S}_D(k-1,i)\mathbf{p}_{Df}(k,i)\\ &= \mathbf{R}_D^{-1}(k-1,i)\mathbf{p}_{Df}(k,i)\end{aligned} \tag{7.37}$$

This is the standard expression for the computation of the optimal coefficient vector leading to the minimization of the LS objective function and adapted to the forward prediction case.

The updating formula of $\mathbf{S}_D(k,i)$ based on the matrix inversion lemma derived in the previous chapter (see Algorithm 5.2) for the conventional RLS algorithm can be used in equation (7.37). The resulting equation is given by

$$\begin{aligned}
\mathbf{w}_f(k,i) &= \frac{1}{\lambda}\left[\mathbf{S}_D(k-2,i) - \frac{\boldsymbol{\psi}(k-1,i)\boldsymbol{\psi}^T(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)}\right]\mathbf{p}_{Df}(k,i)\\
&= \frac{1}{\lambda}\left[\mathbf{S}_D(k-2,i) - \frac{\boldsymbol{\psi}(k-1,i)\mathbf{x}^T(k-1,i)\mathbf{S}_D(k-2,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)}\right]\\
&\quad \cdot\left[\lambda\mathbf{p}_{Df}(k-1,i)+x(k)\mathbf{x}(k-1,i)\right]\\
&= \mathbf{w}_f(k-1,i) - \frac{\boldsymbol{\psi}(k-1,i)\mathbf{x}^T(k-1,i)\mathbf{w}_f(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} + \frac{x(k)}{\lambda}\mathbf{c} \qquad (7.38)
\end{aligned}$$

where in the we have applied the time-recursive updating formula of $\mathbf{p}_{Df}(k,i)$ in the second equality, and we have replaced $\mathbf{S}_D(k-2,i)\mathbf{p}_{Df}(k-1,i)$ by $\mathbf{w}_f(k-1,i)$ in the second term of the final expression. Vector **c** is given by

$$\begin{aligned}
\mathbf{c} &= \mathbf{S}_D(k-2,i)\mathbf{x}(k-1,i) - \frac{\boldsymbol{\psi}(k-1,i)\mathbf{x}^T(k-1,i)\mathbf{S}_D(k-2,i)\mathbf{x}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)}\\
&= \frac{\lambda\mathbf{S}_D(k-2,i)\mathbf{x}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)}
\end{aligned}$$

It is convenient at this point to recall that $\boldsymbol{\psi}(k-1,i) = \mathbf{S}_D(k-2,i)\mathbf{x}(k-1,i)$ (see equation (5.10)).

The last term in equation (7.38) can be simplified if we apply the refined definition based on equation (5.11)

$$\boldsymbol{\phi}(k-1,i) = \frac{\boldsymbol{\psi}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \qquad (7.39)$$

where $\boldsymbol{\phi}(k-1,i)$ now includes the order index i. Using this definition in the second and third terms of the last expression of equation (7.38), it can be shown that

$$\begin{aligned}
\mathbf{w}_f(k,i) &= \mathbf{w}_f(k-1,i) + \boldsymbol{\phi}(k-1,i)[x(k)-\mathbf{w}_f^T(k-1,i)\mathbf{x}(k-1,i)]\\
&= \mathbf{w}_f(k-1,i) + \boldsymbol{\phi}(k-1,i)e_f(k,i) \qquad (7.40)
\end{aligned}$$

where $e_f(k,i)$ is the *a priori* forward prediction error of a predictor of order $i-1$[2], so-called because it utilizes the tap coefficients of the previous instant $k-1$.

Following similar steps to those used to derive equation (7.40), we can show that the time updating for the backward predictor filter is given by

$$\begin{aligned}
\mathbf{w}_b(k,i) &= \frac{1}{\lambda}\left[\mathbf{S}_D(k-1,i) - \frac{\boldsymbol{\psi}(k,i)\boldsymbol{\psi}^T(k,i)}{\lambda+\boldsymbol{\psi}^T(k,i)\mathbf{x}(k,i)}\right][\lambda\mathbf{p}_{Db}(k-1,i)+\mathbf{x}(k,i)x(k-i)]\\
&= \mathbf{w}_b(k-1,i) - \boldsymbol{\phi}(k,i)\mathbf{x}^T(k,i)\mathbf{w}_b(k-1,i) + \boldsymbol{\phi}(k,i)x(k-i)\\
&= \mathbf{w}_b(k-1,i) + \boldsymbol{\phi}(k,i)e_b(k,i) \qquad (7.41)
\end{aligned}$$

where $e_b(k,i)$ is the *a priori* backward prediction error of a predictor filter of order $i-1$.

[2]The predictor filter is of order $i-1$ whereas the predictor including the desired signal is of order i.

7.4.2 Time Updating for $\delta(k, i)$

From the computational point of view, it would be interesting to compute the prediction errors without explicitly using the predictor's tap coefficients, because working with these coefficients requires the use of inner products. In order to achieve this, a time-updating expression for $\delta(k, i)$ is derived. A byproduct of this derivation is the introduction of a new parameter, namely $\gamma(k, i)$, that is shown to be a conversion factor between *a priori* and *a posteriori* errors.

From the definition in equation (7.19), we have

$$\delta(k,i) = \mathbf{p}_{Db}^T(k,i+1)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k,i) \end{bmatrix} \tag{7.42}$$

where $\mathbf{p}_{Db}(k, i+1)$ can be expressed in recursive form as

$$\begin{aligned}\mathbf{p}_{Db}(k,i+1) &= \sum_{l=0}^{k}\lambda^{k-l}\mathbf{x}(l,i+1)x(l-i-1) \\ &= \mathbf{x}(k,i+1)x(k-i-1) + \lambda\mathbf{p}_{Db}(k-1,i+1)\end{aligned} \tag{7.43}$$

Substituting equations (7.40) and (7.43) in equation (7.42), we get

$$\begin{aligned}\delta(k,i) &= [x(k-i-1)\mathbf{x}^T(k,i+1) + \lambda\mathbf{p}_{Db}^T(k-1,i+1)] \\ &\quad \cdot \begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,i) - \boldsymbol{\phi}(k-1,i)e_f(k,i) \end{bmatrix} \\ &= \lambda\delta(k-1,i) + \lambda\mathbf{p}_{Db}^T(k-1,i+1)\begin{bmatrix} 0 \\ -\boldsymbol{\phi}(k-1,i)e_f(k,i) \end{bmatrix} \\ &\quad + x(k-i-1)\mathbf{x}^T(k,i+1)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,i) \end{bmatrix} \\ &\quad + x(k-i-1)\mathbf{x}^T(k,i+1)\begin{bmatrix} 0 \\ -\boldsymbol{\phi}(k-1,i)e_f(k,i) \end{bmatrix}\end{aligned} \tag{7.44}$$

where the equality of equation (7.42) for the order index $i-1$ was used to obtain the first term of the last equality.

We now derive two relations which are essential to obtain a time-updating equation for $\delta(k, i)$. The resulting equation is efficient from the computational point of view. From the definitions of $\boldsymbol{\phi}(k-1, i)$ and $\boldsymbol{\psi}(k-1, i)$, (see equation (7.39) and the comments after equation (7.38) respectively) it can be shown that

$$\begin{aligned}
\mathbf{p}_{Db}^T(k-1,i+1)\begin{bmatrix} 0 \\ \boldsymbol{\phi}(k-1,i) \end{bmatrix} &= \mathbf{p}_{Db}^T(k-2,i)\boldsymbol{\phi}(k-1,i) \\
&= \frac{\mathbf{p}_{Db}^T(k-2,i)\boldsymbol{\psi}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \\
&= \frac{\mathbf{p}_{Db}^T(k-2,i)\mathbf{S}_D(k-2,i)\mathbf{x}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \\
&= \frac{\mathbf{w}_b^T(k-2,i)\mathbf{x}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \\
&= -\frac{e_b(k-1,i)-x(k-i-1)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \qquad (7.45)
\end{aligned}$$

Now using equation (7.39) it is possible to obtain the relation

$$\begin{aligned}
\mathbf{x}^T(k,i+1)\begin{bmatrix} 0 \\ \boldsymbol{\phi}(k-1,i) \end{bmatrix} &= \frac{\mathbf{x}^T(k-1,i)\mathbf{S}_D(k-2,i)\mathbf{x}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \\
&= \frac{\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \qquad (7.46)
\end{aligned}$$

If we recall that the *a priori* forward prediction error can be computed in the form

$$\mathbf{x}^T(k,i+1)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,i) \end{bmatrix} = e_f(k,i)$$

and by substituting equations (7.45) and (7.46) into equation (7.44), after some straightforward manipulations, we obtain the following time-updating equation for $\delta(k,i)$

$$\begin{aligned}
\delta(k,i) &= \lambda\delta(k-1,i) + \frac{\lambda e_b(k-1,i)e_f(k,i)}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \\
&= \lambda\delta(k-1,i) + \gamma(k-1,i)e_b(k-1,i)e_f(k,i) \qquad (7.47)
\end{aligned}$$

where

$$\begin{aligned}
\gamma(k-1,i) &= \frac{\lambda}{\lambda+\boldsymbol{\psi}^T(k-1,i)\mathbf{x}(k-1,i)} \\
&= 1-\boldsymbol{\phi}^T(k-1,i)\mathbf{x}(k-1,i) \qquad (7.48)
\end{aligned}$$

The last relation follows from the definition of $\boldsymbol{\phi}(k-1,i)$ in equation (7.39). Parameter $\gamma(k-1,i)$ plays a key role in the relation between the *a posteriori* and *a priori* prediction errors, as will be demonstrated below.

In order to allow the derivation of a lattice-based algorithm utilizing only *a posteriori* errors, the relationship between the *a priori* and *a posteriori* errors is now derived. The *a posteriori* forward prediction error is related to the *a priori* forward prediction error as

$$\begin{aligned}\varepsilon_f(k,i) &= x(k) - \mathbf{w}_f^T(k,i)\mathbf{x}(k-1,i)\\ &= x(k) - \mathbf{w}_f^T(k-1,i)\mathbf{x}(k-1,i) - \boldsymbol{\phi}^T(k-1,i)\mathbf{x}(k-1,i)e_f(k,i)\\ &= e_f(k,i)[1-\boldsymbol{\phi}^T(k-1,i)\mathbf{x}(k-1,i)]\\ &= e_f(k,i)\gamma(k-1,i) \end{aligned} \tag{7.49}$$

Similarly, the relationship between *a posteriori* and *a priori* backward prediction errors can be expressed as

$$\begin{aligned}\varepsilon_b(k,i) &= x(k-i) - \mathbf{w}_b^T(k,i)\mathbf{x}(k,i)\\ &= x(k-i) - \mathbf{w}_b^T(k-1,i)\mathbf{x}(k,i) - \boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)e_b(k,i)\\ &= e_b(k,i)[1-\boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)]\\ &= e_b(k,i)\gamma(k,i) \end{aligned} \tag{7.50}$$

Parameter $\gamma(k,i)$ is often called a conversion factor between *a priori* and *a posteriori* errors.

Using equations (7.49) and (7.50), equation (7.47) can be expressed as

$$\delta(k,i) = \lambda\delta(k-1,i) + \frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)} \tag{7.51}$$

As a general rule each variable of the lattice-based algorithms requires an order-updating equation. Therefore, an order-updating equation for $\gamma(k,i)$ is necessary. This is the objective of the derivations in the following subsection.

7.4.3 Order Updating for $\gamma(k,i)$

Variable $\gamma(k-1,i)$ is defined by

$$\gamma(k-1,i) = 1 - \boldsymbol{\phi}^T(k-1,i)\mathbf{x}(k-1,i)$$

where $\boldsymbol{\phi}(k-1,i) = \mathbf{S}_D(k-1,i)\mathbf{x}(k-1,i)$. The relation for $\boldsymbol{\phi}(k-1,i)$ can be obtained by replacing $\mathbf{S}_D(k-1,i)$ by the expression derived by the matrix inversion lemma of equation (5.5) and verifying that the resulting simplified expression leads to equation (7.39). By multiplying the expression $\boldsymbol{\phi}(k-1,i) = \mathbf{S}_D(k-1,i)\mathbf{x}(k-1,i)$ by $\mathbf{R}_D(k-1,i)$ on both sides, we obtain the following relation

$$\mathbf{R}_D(k-1,i)\boldsymbol{\phi}(k-1,i) = \mathbf{x}(k-1,i) \tag{7.52}$$

With this equation, we will be able to derive an order-updating equation for $\boldsymbol{\phi}(k-1,i)$ with the aid of an appropriate partitioning of $\mathbf{R}_D(k-1,i)$.

By partitioning matrix $\mathbf{R}_D(k-1,i)$ as in equation (7.19), we get

$$\begin{aligned}\mathbf{R}_D(k-1,i)\begin{bmatrix}\boldsymbol{\phi}(k-1,i-1)\\ 0\end{bmatrix} &= \begin{bmatrix}\mathbf{R}_D(k-1,i-1) & \mathbf{p}_{Db}(k-1,i-1)\\ \mathbf{p}_{Db}^T(k-1,i-1) & \sigma_b^2(k-1)\end{bmatrix}\\ &\quad\cdot\begin{bmatrix}\boldsymbol{\phi}(k-1,i-1)\\ 0\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{R}_{Db}(k-1,i-1)\boldsymbol{\phi}(k-1,i-1)\\ \mathbf{p}_{Db}^T(k-1,i-1)\boldsymbol{\phi}(k-1,i-1)\end{bmatrix}\end{aligned}$$

We can proceed by replacing $\boldsymbol{\phi}(k-1,i-1)$ using equation (7.52) in the last element of the above vector, that is,

$$\begin{aligned}\mathbf{R}_D(k-1,i)\begin{bmatrix}\boldsymbol{\phi}(k-1,i-1)\\ 0\end{bmatrix} &= \begin{bmatrix}\mathbf{R}_{Db}(k-1,i-1)\boldsymbol{\phi}(k-1,i-1)\\ \mathbf{p}_{Db}^T(k-1,i-1)\mathbf{S}_{Db}(k-1,i-1)\mathbf{x}(k-1,i-1)\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{R}_{Db}(k-1,i-1)\boldsymbol{\phi}(k-1,i-1)\\ \mathbf{w}_b^T(k-1,i-1)\mathbf{x}(k-1,i-1)\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{x}(k-1,i-1)\\ x(k-i)-\varepsilon_b(k-1,i-1)\end{bmatrix}\\ &= \mathbf{x}(k-1,i)-\begin{bmatrix}\mathbf{0}\\ \varepsilon_b(k-1,i-1)\end{bmatrix}\end{aligned} \tag{7.53}$$

By multiplying the above equation by $\mathbf{S}_D(k-1,i)$, we have

$$\begin{bmatrix}\boldsymbol{\phi}(k-1,i-1)\\ 0\end{bmatrix} = \boldsymbol{\phi}(k-1,i)-\mathbf{S}_D(k-1,i)\begin{bmatrix}\mathbf{0}\\ \varepsilon_b(k-1,i-1)\end{bmatrix} \tag{7.54}$$

If we replace the above relation in the definition of the conversion factor, we deduce

$$\begin{aligned}\gamma(k-1,i) &= 1-\boldsymbol{\phi}^T(k-1,i)\mathbf{x}(k-1,i)\\ &= \gamma(k-1,i-1)-[\mathbf{0}^T\ \varepsilon_b(k-1,i)]^T\mathbf{S}_D(k-1,i)\mathbf{x}(k-1,i)\end{aligned} \tag{7.55}$$

This equation can be expressed into a more useful form by using a partitioned version of $\mathbf{S}_D(k-1,i)$ given by

$$\begin{aligned}\mathbf{S}_D(k-1,i) &= \begin{bmatrix}0 & \mathbf{0}^T\\ \mathbf{0} & \mathbf{S}_D(k-2,i-1)\end{bmatrix}\\ &\quad+\frac{1}{\xi^d_{f_{\min}}(k-1,i-1)}\begin{bmatrix}1\\ -\mathbf{w}_f(k-1,i-1)\end{bmatrix}\left[1\ \ -\mathbf{w}_f^T(k-1,i-1)\right]\end{aligned} \tag{7.56}$$

The proof of validity of the above expression follows.

Proof:

The partitioned expression of $\mathbf{R}_D(k-1,i)$ is

$$\mathbf{R}_D(k-1,i) = \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{R}_D(k-2,i-1) \end{bmatrix} + \begin{bmatrix} \sigma_f^2(k-1) & \mathbf{p}_{Df}^T(k-1,i-1) \\ \mathbf{p}_{Df}(k-1,i-1) & \mathbf{0}_{i-1,i-1} \end{bmatrix} \tag{7.57}$$

By assuming equation (7.56) is valid and premultiplying it by $\mathbf{R}_D(k-1,i)$ as in equation (7.57), it follows that

$$\begin{aligned}
\mathbf{R}_D(k-1,i)\mathbf{S}_D(k-1,i) &= \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{I}_{i-1,i-1} \end{bmatrix} + \begin{bmatrix} 0 & \mathbf{p}_{Df}^T(k-1,i-1)\mathbf{S}_D(k-2,i-1) \\ \mathbf{0} & \mathbf{0}^T \end{bmatrix} \\
&\quad + \frac{1}{\xi_{f_{\min}}^d(k-1,i-1)}\mathbf{R}_D(k-1,i) \\
&\quad \cdot \begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,i-1) \end{bmatrix} [1 \quad -\mathbf{w}_f^T(k-1,i-1)] \\
&= \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{I}_{i-1,i-1} \end{bmatrix} + \begin{bmatrix} 0 & \mathbf{w}_f^T(k-1,i-1) \\ \mathbf{0} & \mathbf{0}_{i-2,i-2} \end{bmatrix} \\
&\quad + \frac{1}{\xi_{f_{\min}}^d(k-1,i-1)} \begin{bmatrix} \xi_{f_{\min}}^d(k-1,i-1) \\ \mathbf{0} \end{bmatrix} \\
&\quad \cdot [1 \quad -\mathbf{w}_f^T(k-1,i-1)] \\
&= \begin{bmatrix} 0 & \mathbf{w}_f^T(k-1,i-1) \\ \mathbf{0} & \mathbf{I}_{i-1,i-1} \end{bmatrix} + \begin{bmatrix} 1 & -\mathbf{w}_f^T(k-1,i-1) \\ \mathbf{0} & \mathbf{0}_{i-1,i} \end{bmatrix} = \mathbf{I}_{i,i}
\end{aligned}$$

proving the validity of equation (7.56).

□

By applying equation (7.56) in equation (7.55), we obtain

$$\begin{aligned}
\gamma(k,i+1) &= 1 - \boldsymbol{\phi}^T(k,i+1)\mathbf{x}(k,i+1) \\
&= \gamma(k-1,i) - \frac{\varepsilon_f^2(k,i)}{\xi_{f_{\min}}^d(k,i)}
\end{aligned} \tag{7.58}$$

Following a similar method to that used in deriving equation (7.56), it can be shown that

$$\begin{aligned}
\mathbf{S}_D(k-1,i) &= \begin{bmatrix} \mathbf{S}_D(k-1,i-1) & \mathbf{0}_{i-1} \\ \mathbf{0}_{i-1}^T & 0 \end{bmatrix} \\
&\quad + \frac{1}{\xi_{b_{\min}}^d(k-1,i-1)} \begin{bmatrix} -\mathbf{w}_b(k-1,i-1) \\ 1 \end{bmatrix} [-\mathbf{w}_b^T(k-1,i-1) \; 1]
\end{aligned} \tag{7.59}$$

Now by replacing the above equation in equation (7.55), we can show that

$$\begin{aligned}\gamma(k-1,i) &= \gamma(k-1,i-1) - \frac{\varepsilon_b(k-1,i-1)}{\xi^d_{b_{\min}}(k-1,i-1)}\left[-\mathbf{w}_b^T(k-1,i-1)\;\;1\right]\mathbf{x}(k-1,i)\\ &= \gamma(k-1,i-1) - \frac{\varepsilon_b^2(k-1,i-1)}{\xi^d_{b_{\min}}(k-1,i-1)} \qquad (7.60)\end{aligned}$$

The last equation completes the set of relations required to solve the backward and forward prediction problems. In the following section, the modeling of a reference signal (joint-processor estimation) is discussed.

7.5 JOINT-PROCESS ESTIMATION

In the previous sections, we considered only the forward and backward prediction problems and explored some common features in their solutions. In a more general situation, the goal is to predict the behavior of one process represented by $d(k)$ through measurements of a related process contained in $\mathbf{x}(k,i+1)$. Therefore, it is important to derive an adaptive lattice-based realization to match a desired signal $d(k)$ through the minimization of the weighted squared error function given by

$$\begin{aligned}\xi^d(k,i+1) &= \sum_{l=0}^{k}\lambda^{k-l}\varepsilon^2(l,i+1)\\ &= \sum_{l=0}^{k}\lambda^{k-l}[d(l)-\mathbf{w}^T(k,i+1)\mathbf{x}(l,i+1)]^2 \qquad (7.61)\end{aligned}$$

where $y(k,i+1) = \mathbf{w}^T(k,i+1)\mathbf{x}(k,i+1)$ is the adaptive-filter output signal and $\varepsilon(l,i+1)$ is the *a posteriori* error at a given instant l if the adaptive-filter coefficients were fixed at $\mathbf{w}(k,i+1)$. The minimization procedure of $\xi^d(k,i+1)$ is often called joint-process estimation.

The prediction lattice realization generates the forward and backward prediction errors and requires some feedforward coefficients to allow the minimization of $\xi^d(k,i+1)$. In fact, the lattice predictor in this case works as a signal processing building block which improves the quality of the signals (in the sense of reducing the eigenvalue spread of the autocorrelation matrix) that are inputs to the output taps. The question is where should the taps be placed. We give some statistical arguments for this choice here. First, we repeat, for convenience, the expression of the backward prediction error:

$$\varepsilon_b(k,i+1) = \mathbf{x}^T(k,i+2)\begin{bmatrix}-\mathbf{w}_b(k,i+1)\\ 1\end{bmatrix}$$

From the orthogonality property of the RLS algorithm, for $k\rightarrow\infty$, we can infer that

$$E[\varepsilon_b(k,i+1)x(k-l)] = 0$$

for $l = 0,1,\ldots,i$. From this equation, it is possible to show that

$$E[\varepsilon_b(k,i+1)\mathbf{x}^T(k,i+1)] = \mathbf{0}^T$$

If we postmultiply the above equation by $[-\mathbf{w}_b(k,i)\ \ 1]^T$, we obtain

$$E\left\{\varepsilon_b(k,i+1)\mathbf{x}^T(k,i+1)\left[\begin{array}{c}-\mathbf{w}_b(k,i)\\1\end{array}\right]\right\}=E[\varepsilon_b(k,i+1)\varepsilon_b(k,i)]=0$$

This result shows that backward prediction errors of consecutive orders are uncorrelated. Using similar arguments one can show that $E[\varepsilon_b(k,i+1)\varepsilon_b(k,l)]=0$, for $l=0,1,\ldots,i$.

In problem 4, it is shown that backward prediction errors are uncorrelated with each other in the sense of time averaging and, as a consequence, should be naturally chosen as inputs to the output taps. The objective function can now be written as

$$\begin{aligned}\xi^d(k,i+1)&=\sum_{l=0}^{k}\lambda^{k-l}\varepsilon^2(l,i+1)\\&=\sum_{l=0}^{k}\lambda^{k-l}[d(l)-\hat{\boldsymbol{\varepsilon}}_b^T(k,i+1)\mathbf{v}(l,i+1)]^2\end{aligned}\tag{7.62}$$

where $\hat{\boldsymbol{\varepsilon}}_b^T(k,i+1)=[\varepsilon_b(k,0)\ \varepsilon_b(k,1)\ldots\varepsilon_b(k,i)]$ is the backward prediction error vector and $\mathbf{v}^T(k,i+1)=[v_0(k)\ v_1(k)\ldots v_i(k)]$ is the feedforward coefficient vector.

The main objective of the present section is to derive a time-updating formula for the output tap coefficients. From equations (7.61) and (7.62), it is obvious that the lattice realization generates the optimal estimation by using a parameterization different from that related to the direct-form realization. We can derive the updating equations for the elements of the forward coefficient vector using the order-updating equation for the tap coefficients of the direct-form realization. Employing equation (7.59), the equivalent optimal solution with the direct-form realization can be expressed as

$$\begin{aligned}\mathbf{w}(k,i+1)&=\mathbf{S}_D(k,i+1)\mathbf{p}_D(k,i+1)\\&=\left[\begin{array}{cc}\mathbf{S}_D(k,i)&\mathbf{0}_i\\\mathbf{0}_i^T&0\end{array}\right]\mathbf{p}_D(k,i+1)\\&\quad+\frac{1}{\xi^d_{b_{\min}}(k,i)}\left[\begin{array}{c}-\mathbf{w}_b(k,i)\\1\end{array}\right][-\mathbf{w}_b^T(k,i)\ 1]\mathbf{p}_D(k,i+1)\\&=\left[\begin{array}{c}\mathbf{w}(k,i)\\0\end{array}\right]+\frac{\delta_D(k,i)}{\xi^d_{b_{\min}}(k,i)}\left[\begin{array}{c}-\mathbf{w}_b(k,i)\\1\end{array}\right]\end{aligned}\tag{7.63}$$

where

$$\begin{aligned}\delta_D(k,i)&=[-\mathbf{w}_b^T(k,i)\ 1]\mathbf{p}_D(k,i+1)\\&=-\mathbf{w}_b^T(k,i)\sum_{l=0}^{k}\lambda^{k-l}\mathbf{x}(l,i)d(l)+\sum_{l=0}^{k}\lambda^{k-l}x(l-i)d(l)\\&=\sum_{l=0}^{k}\lambda^{k-l}\varepsilon_b(l,i)d(l)\end{aligned}$$

and

$$\mathbf{p}_D(k,i+1) = \sum_{l=0}^{k} \lambda^{k-l}\mathbf{x}(l,i+1)d(l)$$

Since

$$\mathbf{p}_D(k,i+1) = \lambda\mathbf{p}_D(k-1,i+1) + d(k)\mathbf{x}(k,i+1)$$

and

$$\mathbf{w}_b(k,i) = \mathbf{w}_b(k-1,i) + \boldsymbol{\phi}(k,i)e_b(k,i)$$

see equation (7.41), by following the same steps we used to deduce the time update of $\delta(k,i)$ in equation (7.47), we can show that

$$\delta_D(k,i) = \lambda\delta_D(k-1,i) + \frac{\varepsilon(k,i)\varepsilon_b(k,i)}{\gamma(k,i)} \tag{7.64}$$

By calculating the output signal of the joint-process estimator using the order-updating equation (7.63) for the direct-form realization, we can show that

$$\mathbf{w}^T(k,i+1)\mathbf{x}(k,i+1) = [\mathbf{w}^T(k,i)\ 0]\mathbf{x}(k,i+1) + \frac{\delta_D(k,i)}{\xi^d_{b_{\min}}(k,i)}[-\mathbf{w}_b^T(k,i)\ 1]\mathbf{x}(k,i+1) \tag{7.65}$$

This equation can be rewritten as

$$y(k,i+1) = y(k,i) + \frac{\delta_D(k,i)}{\xi^d_{b_{\min}}(k,i)}\varepsilon_b(k,i) \tag{7.66}$$

where it can now be noticed that the joint-predictor output $y(k,i+1)$ is a function of the backward prediction error $\varepsilon_b(k,i)$. This was the motivation for using the decomposition of $\mathbf{S}_D(k,i+1)$ given by equation (7.59) in equation (7.63).

The feedforward multiplier coefficients can be identified as

$$v_i(k) = \frac{\delta_D(k,i)}{\xi^d_{b_{\min}}(k,i)} \tag{7.67}$$

and the *a posteriori* output error of the adaptive filter of order i from 1 to N are obtained simultaneously, where

$$\varepsilon(k,i+1) = \varepsilon(k,i) - v_i(k)\varepsilon_b(k,i) \tag{7.68}$$

The above result was derived by subtracting $d(k)$ from both sides of equation (7.66). The resulting lattice realization is depicted in Fig. 7.2.

We now have available all the relations required to generate the lattice recursive least-squares adaptive-filtering algorithm based on *a posteriori* estimation errors. The algorithm is described in Algorithm 7.1, which highlights in boxes the terms that should be saved in order to avoid repeated computation.

Algorithm 7.1

Lattice RLS Algorithm
Based on *A Posteriori* Errors

```
Initialization

Do for $i = 0, 1 \ldots, N$
  $\delta(-1,i) = \delta_D(-1,i) = 0$ (assuming $x(k) = 0$ for $k < 0$)
  $\xi^d_{b_{\min}}(-1,i) = \xi^d_{f_{\min}}(-1,i) = \epsilon$ (a small positive constant)
  $\gamma(-1,i) = 1$
  $\varepsilon_b(-1,i) = 0$
End

Do for $k \geq 0$
  $\gamma(k,0) = 1$
  $\varepsilon_b(k,0) = \varepsilon_f(k,0) = x(k)$                                    (7.35)
  $\xi^d_{b_{\min}}(k,0) = \xi^d_{f_{\min}}(k,0) = x^2(k) + \lambda\xi^d_{f_{\min}}(k-1,0)$   (7.36)
  $\varepsilon(k,0) = d(k)$

  Do for $i = 0, 1 \ldots, N$

   $\delta(k,i) = \lambda\delta(k-1,i) + \boxed{\frac{\varepsilon_b(k-1,i)}{\gamma(k-1,i)}}\varepsilon_f(k,i)$   (7.51)
   $\gamma(k,i+1) = \gamma(k,i) - \frac{\varepsilon_b^2(k,i)}{\xi^d_{b_{\min}}(k,i)}$   (7.60)
   $\kappa_b(k,i) = \frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)}$
   $\kappa_f(k,i) = \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}$
   $\varepsilon_b(k,i+1) = \varepsilon_b(k-1,i) - \kappa_b(k,i)\varepsilon_f(k,i)$   (7.34)
   $\varepsilon_f(k,i+1) = \varepsilon_f(k,i) - \kappa_f(k,i)\varepsilon_b(k-1,i)$   (7.33)
   $\xi^d_{b_{\min}}(k,i+1) = \xi^d_{b_{\min}}(k-1,i) - \delta(k,i)\kappa_b(k,i)$   (7.27)
   $\xi^d_{f_{\min}}(k,i+1) = \xi^d_{f_{\min}}(k,i) - \delta(k,i)\kappa_f(k,i)$   (7.31)

   Feedforward Filtering

   $\delta_D(k,i) = \lambda\delta_D(k-1,i) + \boxed{\frac{\varepsilon_b(k,i)}{\gamma(k,i)}}\varepsilon(k,i)$   (7.64)
   $v_i(k) = \frac{\delta_D(k,i)}{\xi^d_{b_{\min}}(k,i)}$   (7.67)
   $\varepsilon(k,i+1) = \varepsilon(k,i) - v_i(k)\varepsilon_b(k,i)$   (7.68)
  End
End
```

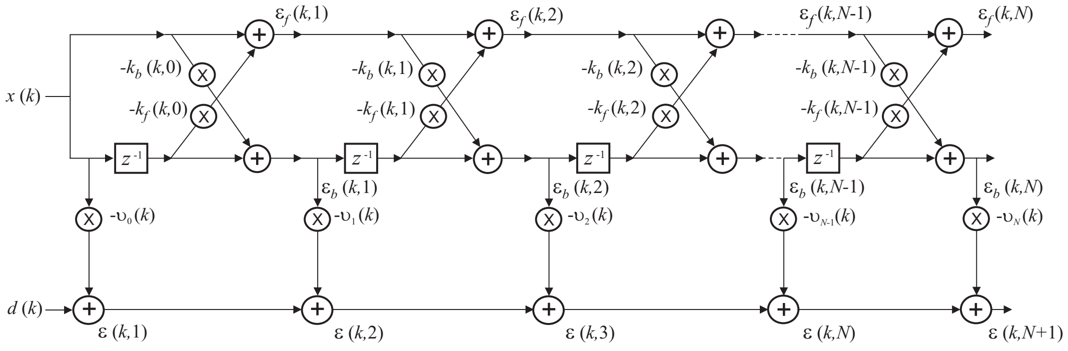

Figure 7.2 Joint-process estimation lattice realization.

7.6 TIME RECURSIONS OF THE LEAST-SQUARES ERROR

In this section, we provide a set of relations for the time updating of the minimum LS error of the prediction problems. These relations allow the derivation of two important equations involving the ratio of conversion factor of consecutive order prediction problems, namely $\frac{\gamma(k-1,i+1)}{\gamma(k-1,i)}$ and $\frac{\gamma(k,i+1)}{\gamma(k-1,i)}$. The results provided in this section are required for the derivation of some alternative lattice algorithms such as the error feedback, as well as for the fast RLS algorithms of Chapter 8.

By replacing each term in the definition of the minimum weighted least-squares error for the backward prediction problem by their time-updating equation, we have (see equations (7.16), (7.17))

$$\begin{aligned}
\xi^d_{b_{\min}}(k,i) &= \sigma_b^2(k) - \mathbf{w}_b^T(k,i)\mathbf{p}_{Db}(k,i) \\
&= \sigma_b^2(k) - \left[\mathbf{w}_b^T(k-1,i) + e_b(k,i)\boldsymbol{\phi}^T(k,i)\right][\lambda\mathbf{p}_{Db}(k-1,i) + x(k-i)\mathbf{x}(k,i)] \\
&= \sigma_b^2(k) - \lambda\mathbf{w}_b^T(k-1,i)\mathbf{p}_{Db}(k-1,i) - x(k-i)\mathbf{w}_b^T(k-1,i)\mathbf{x}(k,i) \\
&\quad - \lambda e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{p}_{Db}(k-1,i) - e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)x(k-i) \\
&= x^2(k-i) + \lambda\sigma_b^2(k-1) - \lambda\mathbf{w}_b^T(k-1,i)\mathbf{p}_{Db}(k-1,i) \\
&\quad -x(k-i)\mathbf{w}_b^T(k-1,i)\mathbf{x}(k,i) - \lambda e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{p}_{Db}(k-1,i) \\
&\quad -e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)x(k-i)
\end{aligned} \tag{7.69}$$

By combining the second and third terms, we get

$$\lambda[\sigma_b^2(k-1) - \mathbf{w}_b^T(k-1,i)\mathbf{p}_{Db}(k-1,i)] = \lambda\xi^d_{b_{\min}}(k-1,i)$$

Similarly, by combining the first, fourth and sixth terms, we obtain

$$\begin{aligned}
& x(k-i)[x(k-i) - \mathbf{w}_b^T(k-1,i)\mathbf{x}(k,i) - e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)] \\
=\ & x(k-i)[e_b(k,i) - e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)] \\
=\ & x(k-i)e_b(k,i)[1 - \boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)]
\end{aligned}$$

Now by applying these results in equation (7.69), we can show that

$$\begin{aligned}\xi^d_{b_{\min}}(k,i) &= \lambda\xi^d_{b_{\min}}(k-1,i) + x(k-i)e_b(k,i)[1-\boldsymbol{\phi}^T(k,i)\mathbf{x}(k,i)]\\ &\quad -\lambda e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{p}_{Db}(k-1,i)\\ &= \lambda\xi^d_{b_{\min}}(k-1,i) + x(k-i)e_b(k,i)\\ &\quad -e_b(k,i)\boldsymbol{\phi}^T(k,i)[x(k-i)\mathbf{x}(k,i)+\lambda\mathbf{p}_{Db}(k-1,i)]\end{aligned}$$

If we apply the definition of $\boldsymbol{\phi}(k,i)$ in equation (7.39) and the equation (7.16) for the backward prediction problem, we obtain

$$\begin{aligned}\xi^d_{b_{\min}}(k,i) &= \lambda\xi^d_{b_{\min}}(k-1,i) + x(k-i)e_b(k,i) - e_b(k,i)\boldsymbol{\phi}^T(k,i)\mathbf{p}_{Db}(k,i)\\ &= \lambda\xi^d_{b_{\min}}(k-1,i) + x(k-i)e_b(k,i) - e_b(k,i)\mathbf{x}^T(k,i)\mathbf{S}_D(k-1,i)\mathbf{p}_{Db}(k,i)\\ &= \lambda\xi^d_{b_{\min}}(k-1,i) + e_b(k,i)[x(k-i)-\mathbf{w}_b^T(k,i)\mathbf{x}(k,i)]\\ &= \lambda\xi^d_{b_{\min}}(k-1,i) + e_b(k,i)\varepsilon_b(k,i)\\ &= \lambda\xi^d_{b_{\min}}(k-1,i) + \frac{\varepsilon_b^2(k,i)}{\gamma(k,i)}\end{aligned}\tag{7.70}$$

Following similar steps to those used to obtain the above equation, we can show that

$$\xi^d_{f_{\min}}(k,i) = \lambda\xi^d_{f_{\min}}(k-1,i) + \frac{\varepsilon_f^2(k,i)}{\gamma(k-1,i)}\tag{7.71}$$

From the last two equations, we can easily infer the relations that are useful in deriving alternative lattice-based algorithms, namely the normalized and error-feedback algorithms. These relations are

$$\begin{aligned}\frac{\lambda\xi^d_{b_{\min}}(k-2,i)}{\xi^d_{b_{\min}}(k-1,i)} &= 1-\frac{\varepsilon_b^2(k-1,i)}{\gamma(k-1,i)\xi^d_{b_{\min}}(k-1,i)}\\ &= \frac{\gamma(k-1,i+1)}{\gamma(k-1,i)}\end{aligned}\tag{7.72}$$

and

$$\begin{aligned}\frac{\lambda\xi^d_{f_{\min}}(k-1,i)}{\xi^d_{f_{\min}}(k,i)} &= 1-\frac{\varepsilon_f^2(k,i)}{\gamma(k-1,i)\xi^d_{f_{\min}}(k,i)}\\ &= \frac{\gamma(k,i+1)}{\gamma(k-1,i)}\end{aligned}\tag{7.73}$$

where equations (7.60) and (7.58), respectively, were used in the derivation of the right-hand-side expressions of the above equations.

7.7 NORMALIZED LATTICE RLS ALGORITHM

An alternative form of the lattice RLS algorithm can be obtained by applying a judicious normalization to the internal variables of the algorithm, keeping their magnitude bounded by one. This normalized lattice is specially suitable for fixed-point arithmetic implementation. Also, this algorithm requires fewer recursions and variables than the unnormalized lattices, i.e., only three equations per prediction section per time sample.

7.7.1 Basic Order Recursions

A natural way to normalize the backward and forward prediction errors is to divide them by the square root of the corresponding weighted least-squares error. However, it will be shown that a wiser strategy leads to a reduction in the number of recursions. At the same time, we must think of a way to normalize variable $\delta(k,i)$. In the process of normalizing $\varepsilon_f(k,i)$, $\varepsilon_b(k,i)$, and $\delta(k,i)$, we can reduce the number of equations by eliminating the conversion variable $\gamma(k,i+1)$. Note that $\gamma(k,i+1)$ is originally normalized. These goals can be reached if the normalization of $\delta(k,i)$ is performed as

$$\overline{\delta}(k,i) = \frac{\delta(k,i)}{\sqrt{\xi^d_{f_{\min}}(k,i)\xi^d_{b_{\min}}(k-1,i)}} \tag{7.74}$$

By noting that the conversion variable $\gamma(k-1,i)$ divides the product $\varepsilon_f(k,i)\varepsilon_b(k-1,i)$ in the time-updating formula (7.51), we can devise a way to perform the normalization of the prediction errors leading to its elimination. The appropriate normalization of the forward and backward estimation errors are, respectively, performed as

$$\overline{\varepsilon}_f(k,i) = \frac{\varepsilon_f(k,i)}{\sqrt{\gamma(k-1,i)\xi^d_{f_{\min}}(k,i)}} \tag{7.75}$$

$$\overline{\varepsilon}_b(k,i) = \frac{\varepsilon_b(k,i)}{\sqrt{\gamma(k,i)\xi^d_{b_{\min}}(k,i)}} \tag{7.76}$$

where the terms $\sqrt{\xi^d_{f_{\min}}(k,i)}$ and $\sqrt{\xi^d_{b_{\min}}(k,i)}$ perform the power normalization whereas $\sqrt{\gamma(k-1,i)}$ and $\sqrt{\gamma(k,i)}$ perform the so-called angle normalization, since $\gamma(k,i)$ is related to the angle between the spaces spanned by $\mathbf{x}(k-1,i)$ and $\mathbf{x}(k,i)$.

From the above equations and equation (7.51), we can show that

$$\begin{aligned}\overline{\delta}(k,i)\sqrt{\xi^d_{f_{\min}}(k,i)\xi^d_{b_{\min}}(k-1,i)} &= \lambda\overline{\delta}(k-1,i)\sqrt{\xi^d_{f_{\min}}(k-1,i)\xi^d_{b_{\min}}(k-2,i)} \\ &\quad +\overline{\varepsilon}_b(k-1,i)\overline{\varepsilon}_f(k,i)\sqrt{\xi^d_{f_{\min}}(k,i)\xi^d_{b_{\min}}(k-1,i)}\end{aligned} \tag{7.77}$$

Therefore,

$$\overline{\delta}(k,i) = \lambda\overline{\delta}(k-1,i)\sqrt{\frac{\xi^d_{f_{\min}}(k-1,i)\xi^d_{b_{\min}}(k-2,i)}{\xi^d_{f_{\min}}(k,i)\xi^d_{b_{\min}}(k-1,i)}} + \overline{\varepsilon}_b(k-1,i)\overline{\varepsilon}_f(k,i) \tag{7.78}$$

We now show that the term under the square root in the above equation can be expressed in terms of the normalized errors by using equations (7.72), (7.73), (7.75), and (7.76), that is,

$$\begin{aligned}\frac{\lambda\xi^d_{b_{\min}}(k-2,i)}{\xi^d_{b_{\min}}(k-1,i)} &= \frac{\gamma(k-1,i+1)}{\gamma(k-1,i)}\\ &= 1 - \frac{\varepsilon^2_b(k-1,i)}{\gamma(k-1,i)\xi^d_{b_{\min}}(k-1,i)}\\ &= 1 - \overline{\varepsilon}^2_b(k-1,i)\end{aligned} \tag{7.79}$$

and

$$\begin{aligned}\frac{\lambda\xi^d_{f_{\min}}(k-1,i)}{\xi^d_{f_{\min}}(k,i)} &= \frac{\gamma(k,i+1)}{\gamma(k-1,i)}\\ &= 1 - \frac{\varepsilon^2_f(k,i)}{\gamma(k-1,i)\xi^d_{f_{\min}}(k,i)}\\ &= 1 - \overline{\varepsilon}^2_f(k,i)\end{aligned} \tag{7.80}$$

Substituting the last two equations into equation (7.78), we can show that

$$\overline{\delta}(k,i) = \overline{\delta}(k-1,i)\sqrt{(1-\overline{\varepsilon}^2_b(k-1,i))(1-\overline{\varepsilon}^2_f(k,i))} + \overline{\varepsilon}_b(k-1,i)\overline{\varepsilon}_f(k,i) \tag{7.81}$$

Following a similar procedure used to derive the time-updating equation for $\overline{\delta}(k,i)$, one can derive the order-updating equation of the normalized forward and backward prediction errors. In the case of the forward prediction error, the following order-updating relation results:

$$\overline{\varepsilon}_f(k,i+1) = \left[\overline{\varepsilon}_f(k,i) - \overline{\delta}(k,i)\overline{\varepsilon}_b(k-1,i)\right]\sqrt{\frac{\xi^d_{f_{\min}}(k,i)}{\xi^d_{f_{\min}}(k,i+1)}}\sqrt{\frac{\gamma(k-1,i)}{\gamma(k-1,i+1)}} \tag{7.82}$$

Here again, we can express the functions under the square roots in terms of normalized variables. Using equations (7.31), (7.74), and (7.77), it can be shown that

$$\overline{\varepsilon}_f(k,i+1) = \frac{\overline{\varepsilon}_f(k,i) - \overline{\delta}(k,i)\overline{\varepsilon}_b(k-1,i)}{\sqrt{1-\overline{\delta}^2(k,i)}\sqrt{1-\overline{\varepsilon}^2_b(k-1,i)}} \tag{7.83}$$

If the same steps to derive $\overline{\varepsilon}_f(k, i+1)$ are followed, we can derive the order-updating equation for the backward prediction error as

$$\begin{aligned}\overline{\varepsilon}_b(k, i+1) &= \left[\overline{\varepsilon}_b(k-1, i) - \overline{\delta}(k, i)\overline{\varepsilon}_f(k, i)\right]\sqrt{\frac{\xi^d_{b_{\min}}(k-1, i)}{\xi^d_{b_{\min}}(k, i+1)}}\sqrt{\frac{\gamma(k-1, i)}{\gamma(k, i+1)}} \\ &= \frac{\overline{\varepsilon}_b(k-1, i) - \overline{\delta}(k, i)\overline{\varepsilon}_f(k, i)}{\sqrt{1-\overline{\delta}^2(k, i)}\sqrt{1-\overline{\varepsilon}^2_f(k, i)}} \end{aligned} \quad (7.84)$$

7.7.2 Feedforward Filtering

The procedure to generate the joint-processor estimator is repeated here, using normalized variables. Define

$$\overline{\delta}_D(k, i) = \frac{\delta_D(k, i)}{\sqrt{\xi^d_{\min}(k, i)\xi^d_{b_{\min}}(k, i)}} \quad (7.85)$$

and

$$\overline{\varepsilon}(k, i) = \frac{\varepsilon(k, i)}{\sqrt{\gamma(k, i)\xi^d_{\min}(k, i)}} \quad (7.86)$$

Using a similar approach to that used to derive equation (7.31), one can show that

$$\xi^d_{\min}(k, i+1) = \xi^d_{\min}(k, i) - \frac{\delta^2_D(k, i)}{\xi^d_{b_{\min}}(k, i)} \quad (7.87)$$

The procedure used to derive the order-updating equations for the normalized prediction errors and the parameter $\overline{\delta}(k, i)$ can be followed to derive the equivalent parameters in the joint-process estimation case. For the *a posteriori* output error the following equation results

$$\begin{aligned}\overline{\varepsilon}(k, i+1) &= \sqrt{\frac{\gamma(k, i)}{\gamma(k, i+1)}}\sqrt{\frac{\xi^d_{\min}(k, i)}{\xi^d_{\min}(k, i+1)}}\left[\overline{\varepsilon}(k, i) - \overline{\delta}_D(k, i)\overline{\varepsilon}_b(k, i)\right] \\ &= \frac{1}{\sqrt{1-\overline{\varepsilon}^2_b(k, i)}}\frac{1}{\sqrt{1-\overline{\delta}^2_D(k, i)}}\left[\overline{\varepsilon}(k, i) - \overline{\delta}_D(k, i)\overline{\varepsilon}_b(k, i)\right] \end{aligned} \quad (7.88)$$

The order-updating equation of $\overline{\delta}_D(k, i)$ is (see equation (7.78))

$$\begin{aligned}\overline{\delta}_D(k, i) &= \sqrt{\frac{\lambda^2\xi^d_{\min}(k-1, i)\xi^d_{b_{\min}}(k-1, i)}{\xi^d_{\min}(k, i)\xi^d_{b_{\min}}(k, i)}}\overline{\delta}_D(k-1, i) + \overline{\varepsilon}(k, i)\overline{\varepsilon}_b(k, i) \\ &= \sqrt{(1-\overline{\varepsilon}^2_b(k, i))(1-\overline{\varepsilon}^2(k, i))}\overline{\delta}_D(k-1, i) + \overline{\varepsilon}(k, i)\overline{\varepsilon}_b(k, i) \end{aligned} \quad (7.89)$$

where we used the fact that

$$\frac{\lambda \xi^{d}_{\min}(k-1,i)}{\xi^{d}_{\min}(k,i)} = 1 - \bar{\varepsilon}^2(k,i) \tag{7.90}$$

The normalized lattice RLS algorithm based on *a posteriori* errors is described in Algorithm 7.2.

Notice that in the updating formulas of the normalized errors, the terms involving the square root operation could be conveniently implemented through separate multiplier coefficients, namely $\eta_f(k,i)$, $\eta_b(k,i)$, and $\eta_D(k,i)$. In this way, one can perform the order updating by calculating the numerator first and proceeding with a single multiplication. These coefficients are given by

$$\eta_f(k,i+1) = \frac{1}{\sqrt{1-\bar{\delta}^2(k,i)}\sqrt{1-\bar{\varepsilon}_b^2(k-1,i)}} \tag{7.91}$$

$$\eta_b(k,i+1) = \frac{1}{\sqrt{1-\bar{\delta}^2(k,i)}\sqrt{1-\bar{\varepsilon}_f^2(k,i)}} \tag{7.92}$$

$$\eta_D(k,i+1) = \frac{1}{\sqrt{1-\bar{\varepsilon}_b^2(k,i)}\sqrt{1-\bar{\delta}_D^2(k,i)}} \tag{7.93}$$

With these multipliers it is straightforward to obtain the structure for the joint-processor estimator depicted in Fig. 7.3.

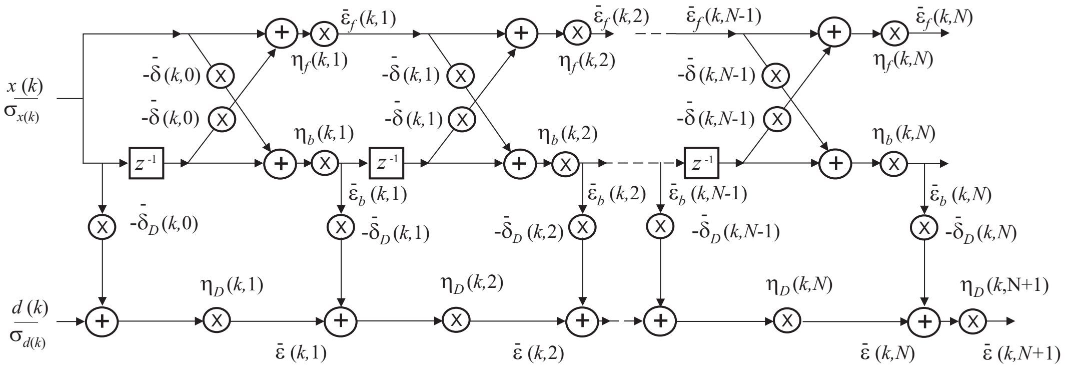
$\frac{x(k)}{\sigma_{x(k)}}$
$\frac{d(k)}{\sigma_{d(k)}}$
$\bar{\varepsilon}_f(k,1)$
$\bar{\varepsilon}_f(k,2)$
$\bar{\varepsilon}_f(k,N\text{-}1)$
$\bar{\varepsilon}_f(k,N)$
$\eta_f(k,1)$
$\eta_f(k,2)$
$\eta_f(k,N)$
$-\bar{\delta}(k,0)$
$-\bar{\delta}(k,1)$
$-\bar{\delta}(k,N\text{-}1)$
$\eta_b(k,1)$
$\eta_b(k,2)$
$\eta_b(k,N)$
z^{-1}
$\bar{\varepsilon}_b(k,1)$
$\bar{\varepsilon}_b(k,2)$
$\bar{\varepsilon}_b(k,N\text{-}1)$
$\bar{\varepsilon}_b(k,N)$
$-\bar{\delta}_D(k,0)$
$-\bar{\delta}_D(k,1)$
$-\bar{\delta}_D(k,2)$
$-\bar{\delta}_D(k,N\text{-}1)$
$-\bar{\delta}_D(k,N)$
$\eta_D(k,1)$
$\eta_D(k,2)$
$\eta_D(k,N)$
$\eta_D(k,N+1)$
$\bar{\varepsilon}(k,1)$
$\bar{\varepsilon}(k,2)$
$\bar{\varepsilon}(k,N)$
$\bar{\varepsilon}(k,N+1)$

Figure 7.3 Joint-process estimation normalized lattice realization.

The unique feature of the normalized lattice algorithm is the reduced number of equations and variables at the expense of employing a number of square root operations. These operations can be costly to implement in most types of hardware architectures. Another interesting feature of the

Algorithm 7.2

Normalized Lattice RLS Algorithm
Based on *A Posteriori* Error

```
Initialization

Do for $i = 0, 1 \ldots, N$
 $\overline{\delta}(-1, i) = 0$  (assuming $x(k) = d(k) = 0$ for $k < 0$)
 $\overline{\delta}_D(-1, i) = 0$
 $\overline{\varepsilon}_b(-1, i) = 0$
End

  $\sigma_x^2(-1) = \sigma_d^2(-1) = \epsilon$ ($\epsilon$ small positive constant)

Do for $k \geq 0$
  $\sigma_x^2(k) = \lambda\sigma_x^2(k-1) + x^2(k)$ (Input signal energy)
  $\sigma_d^2(k) = \lambda\sigma_d^2(k-1) + d^2(k)$ (Reference signal energy)
  $\overline{\varepsilon}_b(k, 0) = \overline{\varepsilon}_f(k, 0) = x(k)/\sigma_x(k)$
  $\overline{\varepsilon}(k, 0) = d(k)/\sigma_d(k)$

  Do for $i = 0, 1 \ldots, N$

   $\overline{\delta}(k, i) = \overline{\delta}(k-1, i)\sqrt{(1 - \overline{\varepsilon}_b^2(k-1, i))(1 - \overline{\varepsilon}_f^2(k, i))} + \overline{\varepsilon}_b(k-1, i)\overline{\varepsilon}_f(k, i)$   (7.81)
   $\overline{\varepsilon}_b(k, i+1) = \frac{\overline{\varepsilon}_b(k-1,i) - \overline{\delta}(k,i)\overline{\varepsilon}_f(k,i)}{\sqrt{(1-\overline{\delta}^2(k,i))(1-\overline{\varepsilon}_f^2(k,i))}}$   (7.84)
   $\overline{\varepsilon}_f(k, i+1) = \frac{\overline{\varepsilon}_f(k,i) - \overline{\delta}(k,i)\overline{\varepsilon}_b(k-1,i)}{\sqrt{(1-\overline{\delta}^2(k,i))(1-\overline{\varepsilon}_b^2(k-1,i))}}$   (7.83)

   Feedforward Filter

   $\overline{\delta}_D(k, i) = \overline{\delta}_D(k-1, i)\sqrt{(1 - \overline{\varepsilon}_b^2(k, i))(1 - \overline{\varepsilon}^2(k, i))} + \overline{\varepsilon}(k, i)\overline{\varepsilon}_b(k, i)$   (7.89)
   $\overline{\varepsilon}(k, i+1) = \frac{1}{\sqrt{(1-\overline{\varepsilon}_b^2(k,i))(1-\overline{\delta}_D^2(k,i))}}\left[\overline{\varepsilon}(k, i) - \overline{\delta}_D(k, i)\overline{\varepsilon}_b(k, i)\right]$   (7.88)
  End

End
```

normalized lattice algorithm is that the forgetting factor λ does not appear in the internal updating equations; it appears only in the calculation of the energy of the input and reference signals. This property may be advantageous from the computational point of view in situations where there is a need to vary the value of λ. On the other hand, since all internal variables are normalized, the actual amplitude of the error signals and other quantities do not match those in other lattice structures. In fact, from the normalized lattice structure one can only effectively extract the shape of the frequency model the structure identifies, since the mapping between the parameters of normalized and non normalized structures is computationally intensive.

7.8 ERROR-FEEDBACK LATTICE RLS ALGORITHM

The reflection coefficients of the lattice algorithm have so far been updated in an indirect way, without time recursions. This section describes an alternative method of updating the reflection coefficients using time updating. These updating equations are recursive in nature and are often called direct updating, since the updating equations used for $\kappa_b(k,i)$ and $\kappa_f(k,i)$ in Algorithm 7.1 are dependent exclusively on quantities other than past reflection coefficients. Algorithms employing the recursive time updating are called error-feedback lattice RLS algorithms. These algorithms have better numerical properties than their indirect updating counterparts [3].

7.8.1 Recursive Formulas for the Reflection Coefficients

The derivation of a direct updating equation for $\kappa_f(k,i)$ starts by replacing $\delta(k,i)$ by its time-updating equation (7.51)

$$\begin{aligned}\kappa_f(k,i) &= \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}\\ &= \frac{\lambda\delta(k-1,i)}{\xi^d_{b_{\min}}(k-1,i)} + \frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)\xi^d_{b_{\min}}(k-1,i)}\end{aligned}$$

By multiplying and dividing the first term by $\xi^d_{b_{\min}}(k-2,i)$ and next using equation (7.72) in the first and second terms, we obtain

$$\begin{aligned}\kappa_f(k,i) &= \frac{\delta(k-1,i)}{\xi^d_{b_{\min}}(k-2,i)}\frac{\lambda\xi^d_{b_{\min}}(k-2,i)}{\xi^d_{b_{\min}}(k-1,i)} + \frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)\xi^d_{b_{\min}}(k-1,i)}\\ &= \kappa_f(k-1,i)\frac{\gamma(k-1,i+1)}{\gamma(k-1,i)} + \frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)\gamma(k-1,i+1)}{\gamma^2(k-1,i)\lambda\xi^d_{b_{\min}}(k-2,i)}\\ &= \frac{\gamma(k-1,i+1)}{\gamma(k-1,i)}\left[\kappa_f(k-1,i) + \frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)\lambda\xi^d_{b_{\min}}(k-2,i)}\right] \qquad (7.94)\end{aligned}$$

Similarly, using equations (7.51) and (7.73), it is straightforward to show that

$$\kappa_b(k,i) = \frac{\gamma(k,i+1)}{\gamma(k-1,i)}\left[\kappa_b(k-1,i) + \frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)\lambda\xi^d_{f_{\min}}(k-1,i)}\right] \tag{7.95}$$

The feedforward coefficients can also be time updated in a recursive form, by appropriately combining equations (7.64), (7.67), and (7.72). The time-recursive updating equation for $v_i(k)$ is

$$v_i(k) = \frac{\gamma(k,i+1)}{\gamma(k,i)}\left[v_i(k-1) + \frac{\varepsilon(k,i)\varepsilon_b(k,i)}{\gamma(k,i)\lambda\xi^d_{b_{\min}}(k-1,i)}\right] \tag{7.96}$$

The error-feedback LRLS algorithm described in Algorithm 7.3 employs the equations (7.94), (7.95), and (7.96). This algorithm is directly derived from Algorithm 7.1.

Alternative *a posteriori* LRLS algorithms can be obtained if we replace equations (7.27) and (7.31) by (7.70) and (7.72) in Algorithms 7.1 and 7.3, respectively. These modifications as well as possible others do not change the behavior of the LRLS algorithm when implemented with infinite precision (long wordlength). However, differences exist in computational complexity and in the effects of quantization error propagation.

7.9 LATTICE RLS ALGORITHM BASED ON *A PRIORI* ERRORS

The lattice algorithms presented so far are based on *a posteriori* errors; however alternative algorithms based on *a priori* errors exist and one of them is derived in this section.

The time updating of the quantity $\delta(k,i)$ as a function of the *a priori* errors was previously derived (see equation (7.47)) and is repeated here for convenience.

$$\delta(k,i) = \lambda\delta(k-1,i) + \gamma(k-1,i)e_b(k-1,i)e_f(k,i) \tag{7.97}$$

The time updating of the forward prediction *a priori* error can be obtained by using equation (7.32) as

$$\begin{aligned}
e_f(k,i+1) &= \mathbf{x}^T(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,i+1) \end{bmatrix} \\
&= \mathbf{x}^T(k,i+2)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,i) \\ 0 \end{bmatrix} + \frac{\delta(k-1,i)}{\xi^d_{b_{\min}}(k-2,i)}\mathbf{x}^T(k,i+2)\begin{bmatrix} 0 \\ \mathbf{w}_b(k-2,i) \\ -1 \end{bmatrix} \\
&= e_f(k,i) - \frac{\delta(k-1,i)}{\xi^d_{b_{\min}}(k-2,i)}e_b(k-1,i) \\
&= e_f(k,i) - \kappa_f(k-1,i)e_b(k-1,i)
\end{aligned} \tag{7.98}$$

Algorithm 7.3

Error-Feedback LRLS Algorithm
Based on *A Posteriori* Errors

```
Initialization

Do for $i = 0, 1 \ldots, N$
  $\kappa_b(-1,i) = \kappa_f(-1,i) = v_i(-1) = \delta(-1,i) = 0, \gamma(-1,i) = 1$
  $\xi^d_{b_{\min}}(-2,i) = \xi^d_{b_{\min}}(-1,i) = \xi^d_{f_{\min}}(-1,i) = \epsilon$ (a small positive constant)
  $\varepsilon_b(-1,i) = 0$
End

Do for $k \geq 0$
  $\gamma(k,0) = 1$
  $\varepsilon_b(k,0) = \varepsilon_f(k,0) = x(k)$                                                    (7.35)
  $\xi^d_{f_{\min}}(k,0) = \xi^d_{b_{\min}}(k,0) = x^2(k) + \lambda\xi^d_{f_{\min}}(k-1,0)$         (7.36)
  $\varepsilon(k,0) = d(k)$

  Do for $i = 0, 1 \ldots, N$

    $\delta(k,i) = \lambda\delta(k-1,i) + \boxed{\frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)}}$     (7.51)
    $\gamma(k,i+1) = \gamma(k,i) - \frac{\varepsilon_b^2(k,i)}{\xi^d_{b_{\min}}(k,i)}$     (7.60)
    $\kappa_f(k,i) = \boxed{\frac{\gamma(k-1,i+1)}{\gamma(k-1,i)}}\left[\kappa_f(k-1,i) + \boxed{\frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)}}\frac{1}{\lambda\xi^d_{b_{\min}}(k-2,i)}\right]$     (7.94)
    $\kappa_b(k,i) = \frac{\gamma(k,i+1)}{\gamma(k-1,i)}\left[\kappa_b(k-1,i) + \boxed{\frac{\varepsilon_b(k-1,i)\varepsilon_f(k,i)}{\gamma(k-1,i)}}\frac{1}{\lambda\xi^d_{f_{\min}}(k-1,i)}\right]$     (7.95)
    $\varepsilon_b(k,i+1) = \varepsilon_b(k-1,i) - \kappa_b(k,i)\varepsilon_f(k,i)$     (7.34)
    $\varepsilon_f(k,i+1) = \varepsilon_f(k,i) - \kappa_f(k,i)\varepsilon_b(k-1,i)$     (7.33)
    $\xi^d_{f_{\min}}(k,i+1) = \xi^d_{f_{\min}}(k,i) - \frac{\delta^2(k,i)}{\xi^d_{b_{\min}}(k-1,i)}$     (7.31)
    $\xi^d_{b_{\min}}(k,i+1) = \xi^d_{b_{\min}}(k-1,i) - \frac{\delta^2(k,i)}{\xi^d_{f_{\min}}(k,i)}$     (7.27)

   Feedforward Filtering

    $v_i(k) = \boxed{\frac{\gamma(k,i+1)}{\gamma(k,i)}}\left[v_i(k-1) + \frac{\varepsilon(k,i)\varepsilon_b(k,i)}{\gamma(k,i)\lambda\xi^d_{b_{\min}}(k-1,i)}\right]$     (7.96)
    $\varepsilon(k,i+1) = \varepsilon(k,i) - v_i(k)\varepsilon_b(k,i)$     (7.68)
  End

End
```

With equation (7.28), we can generate the time-updating equation of the backward prediction *a priori* error as

$$\begin{aligned}
e_b(k,i+1) &= \mathbf{x}^T(k,i+2)\begin{bmatrix} 0 \\ -\mathbf{w}_b(k-2,i) \\ 1 \end{bmatrix} - \frac{\delta(k-1,i)}{\xi^d_{f_{\min}}(k-1,i)}\mathbf{x}^T(k,i+2)\begin{bmatrix} -1 \\ \mathbf{w}_f(k-1,i) \\ 0 \end{bmatrix} \\
&= e_b(k-1,i) - \frac{\delta(k-1,i)}{\xi^d_{f_{\min}}(k-1,i)}e_f(k,i) \\
&= e_b(k-1,i) - \kappa_b(k-1,i)e_f(k,i) \qquad (7.99)
\end{aligned}$$

The order updating of $\gamma(k-1,i)$ can be derived by employing the relations of equations (7.50) and (7.60). The result is

$$\gamma(k-1,i+1) = \gamma(k-1,i) - \frac{\gamma^2(k-1,i)e_b^2(k-1,i)}{\xi^d_{b_{\min}}(k-1,i)} \qquad (7.100)$$

The updating of the feedforward coefficients of the lattice realization based on *a priori* errors is performed by the following equations

$$\begin{aligned}
\delta_D(k,i) &= \lambda\delta_D(k-1,i) + \gamma(k,i)e_b(k,i)e(k,i) \qquad (7.101) \\
e(k,i+1) &= e(k,i) - v_i(k-1)e_b(k,i) \qquad (7.102) \\
v_i(k-1) &= \frac{\delta_D(k-1,i)}{\xi^d_{b_{\min}}(k-1,i)} \qquad (7.103)
\end{aligned}$$

The derivations are omitted since they follow the same steps of the predictor equations.

An LRLS algorithm based on *a priori* errors is described in Algorithm 7.4. The normalized and error-feedback versions of the LRLS algorithm based on *a priori* errors also exist and their derivations are left as problems.

7.10 QUANTIZATION EFFECTS

A major issue related to the implementation of adaptive filters is their behavior when implemented with finite-precision arithmetic. In particular, the roundoff errors arising from the quantization of the internal quantities of an algorithm propagate internally and can even cause instability. The numerical stability and accuracy are algorithm dependent. In this section, we summarize some of the results obtained in the literature related to the LRLS algorithms [3], [7]-[8].

One of the first attempts to study the numerical accuracy of the lattice algorithms was reported in [7]. Special attention was given to the normalized lattice RLS algorithm, since this algorithm is suitable for fixed-point arithmetic implementation, due to its internal normalization. In this study, it was

Algorithm 7.4

LRLS Algorithm
Based on *A Priori* Errors

```
Initialization
Do for $i = 0, 1 \ldots, N$
  $\delta(-1,i) = \delta_D(-1,i) = 0$   (assuming $x(k) = 0$ for $k < 0$)
  $\gamma(-1,i) = 1$
  $\xi^d_{b_{\min}}(-1,i) = \xi^d_{f_{\min}}(-1,i) = \epsilon$ (a small positive constant)
  $e_b(-1,i) = 0$
  $\kappa_f(-1,i) = \kappa_b(-1,i) = 0$
End

Do for $k \geq 0$
  $\gamma(k,0) = 1$
  $e_b(k,0) = e_f(k,0) = x(k)$
  $\xi^d_{f_{\min}}(k,0) = \xi^d_{b_{\min}}(k,0) = x^2(k) + \lambda\xi^d_{f_{\min}}(k-1,0)$
  $e(k,0) = d(k)$
  Do for $i = 0, 1 \ldots, N$
    $\delta(k,i) = \lambda\delta(k-1,i) + \gamma(k-1,i)e_b(k-1,i)e_f(k,i)$        (7.47)
    $\gamma(k,i+1) = \gamma(k,i) - \frac{\boxed{\gamma^2(k,i)e_b^2(k,i)}}{\xi^d_{b_{\min}}(k,i)}$        (7.100)
    $e_b(k,i+1) = e_b(k-1,i) - \kappa_b(k-1,i)e_f(k,i)$        (7.99)
    $e_f(k,i+1) = e_f(k,i) - \kappa_f(k-1,i)e_b(k-1,i)$        (7.98)
    $\kappa_f(k,i) = \frac{\delta(k,i)}{\xi^d_{b_{\min}}(k-1,i)}$
    $\kappa_b(k,i) = \frac{\delta(k,i)}{\xi^d_{f_{\min}}(k,i)}$
    $\xi^d_{f_{\min}}(k,i+1) = \xi^d_{f_{\min}}(k,i) - \delta(k,i)\kappa_f(k,i)$        (7.31)
    $\xi^d_{b_{\min}}(k,i+1) = \xi^d_{b_{\min}}(k-1,i) - \delta(k,i)\kappa_b(k,i)$        (7.27)

    Feedforward Filtering

    $\delta_D(k,i) = \lambda\delta_D(k-1,i) + \boxed{\gamma(k,i)e_b(k,i)}\,e(k,i)$        (7.101)
    $e(k,i+1) = e(k,i) - v_i(k-1)e_b(k,i)$        (7.102)
    $v_i(k) = \frac{\delta_D(k,i)}{\xi^d_{b_{\min}}(k,i)}$        (7.103)
  End
End
```

shown that the bias error in the reflection coefficients was more significant than the variance of the estimate error. The bias in the estimated reflection coefficients is mainly caused by the quantization error associated with the calculation of the square roots of $[1 - \bar{\varepsilon}_b^2(k-1,i)]$ and $[1 - \bar{\varepsilon}_f^2(k,i)]$, assuming they are calculated separately. An upper bound for this quantization error is given by

$$m_{sq} = 2^{-b} \tag{7.104}$$

assuming that b is the number of bits after the sign bit and that quantization is performed through rounding. In the analysis, the basic assumption that $1 - \lambda \gg 2^{-b+1}$ was used. The upper bound of the bias error in the reflection coefficients is given by [7]

$$\Delta\overline{\delta}(k,i) = \frac{2^{-b+1}\overline{\delta}(k,i)}{1-\lambda} \tag{7.105}$$

Obviously, the accuracy of this result depends on the validity of the assumptions used in the analysis [7]. However it is a good indication of how the bias is generated in the reflection coefficients. It should also be noted that the above result is valid as long as the updating of the related reflection coefficient does not stop. An analysis for the case in which the updating stops is also included in [7].

The bias error of a given stage of the lattice realization propagates to the succeeding stages and its accumulation in the prediction errors can be expressed as

$$\Delta\bar{\varepsilon}_b^2(k,i+1) = \Delta\bar{\varepsilon}_f^2(k,i+1) \approx 2^{-b+2}\sum_{l=0}^{i}\frac{\overline{\delta}^2(k,l)}{1-\overline{\delta}^2(k,l)} \tag{7.106}$$

for $i = 0, 1, \ldots, N$. This equation indicates that whenever the value of the parameter $\overline{\delta}^2(k,l)$ is small, the corresponding term in the summation is also small. On the other hand, if the value of this parameter tends to one, the corresponding term of the summation is large. Also note that the accumulated error tends to grow as the number of sections of the lattice is increased. In a finite-precision implementation, it is possible to determine the maximum order that the lattice can have such that the error signals at the end of the realization still represent actual signals and not only accumulated quantization noise.

The lattice algorithms remain stable even when using quite short wordlength in fixed- and floating-point implementations. In terms of accuracy the error-feedback algorithms are usually better than the conventional LRLS algorithms [3]. The reduction in the quantization effects of the error-feedback LRLS algorithms is verified in [3], where a number of examples show satisfactory performance for implementation with less than 10 bits in fixed-point arithmetic.

Another investigation examines the finite-wordlength implementation employing floating-point arithmetic of the unnormalized lattice with and without error feedback [8]. As expected, the variance of the accumulated error in the reflection coefficients of the error-feedback algorithms are smaller than that for the conventional LRLS algorithm. Another important issue relates to the so-called self-generated noise that originates in the internal stages of the lattice realization when the order of adaptive filter is greater than necessary. In the cases where the signal-to-noise ratio is high in the desired signal, the internal signals of the last stages of the lattice realization can reach the quantization level and start self-generated noise, leading to an excess mean-square error and possibly to instability. The stability problem can be avoided by turning off the stages after the one in which the weighted forward and backward squared errors are smaller than a given threshold.

Example 7.1

The system identification problem described in Chapter 3 (subsection 3.6.2) is solved using the lattice algorithms presented in this chapter. The main objective is to compare the performance of the algorithms when implemented in finite precision.

Solution:

We present here the results of using the unnormalized, the normalized and error-feedback *a posteriori* lattice RLS algorithms in the system identification example. All results presented are obtained by running 200 independent experiments and calculating the average of the quantities of interest. We consider the case of eigenvalue spread 20, and $\lambda = 0.99$. Parameter ϵ is 0.1, 0.01, and 0.1 for the unnormalized, the normalized, and the error-feedback lattice filters, respectively. The measured misadjustments of the lattice algorithms are given in Table 7.1. As expected, the results are close to those obtained by the conventional RLS algorithm, where in the latter the misadjustment is 0.0421. Not included is the result for the normalized lattice because the *a posteriori* error is not available, in this case the measured normalized MSE is 0.00974.

Table 7.2 summarizes the results obtained by the implementation of the lattice algorithms with finite precision. Parameter ϵ in the finite-precision implementation is 0.1, 0.04 and 0.5 for the unnormalized, normalized and error-feedback lattices, respectively. These values assure a good convergence behavior of the algorithms in this experiment. In short-wordlength implementation of the lattice algorithms, it is advisable to test if the denominator expressions of the algorithm steps involving division are not rounded to zero. In the case of the detection of a zero denominator, replace its value by the value of the least significant bit. Table 7.2 shows that for the unnormalized and error-feedback lattices, the mean-squared errors are comparable to the case of the conventional RLS previously shown in Table 5.2. The normalized lattice is more sensitive to quantization errors due to its higher computational complexity. The errors introduced by the calculations to obtain $\mathbf{w}(k)_Q$, starting with the lattice coefficients, is the main reason for the increased values of $E[||\Delta\mathbf{w}(k)_Q||^2]$ shown in Table 7.2. Therefore, this result should not be considered as an indication of poor performance of the normalized lattice implemented with finite precision.

□

Table 7.1 Evaluation of the Lattice RLS Algorithms

Algorithm	**Misadjustment**
Unnorm.	0.0416
Error Feed.	0.0407

Table 7.2 Results of the Finite-Precision Implementation of the Lattice RLS Algorithms

	$\xi(k)_Q$			$E[\|\|\Delta\mathbf{w}(k)_Q\|\|^2]$		
No. of bits	**Unnorm.**	**Norm.**	**Error Feed.**	**Unnorm.**	**Norm.**	**Error Feed.**
16	$1.563\,10^{-3}$	$8.081\,10^{-3}$	$1.555\,10^{-3}$	$9.236\,10^{-4}$	$2.043\,10^{-3}$	$9.539\,10^{-4}$
12	$1.545\,10^{-3}$	$8.096\,10^{-3}$	$1.567\,10^{-3}$	$9.317\,10^{-4}$	$2.201\,10^{-3}$	$9.271\,10^{-4}$
10	$1.587\,10^{-3}$	$10.095\,10^{-3}$	$1.603\,10^{-3}$	$9.347\,10^{-4}$	$4.550\,10^{-3}$	$9.872\,10^{-4}$

Example 7.2

The channel equalization example first described in subsection 3.6.3 is used in simulations using the lattice RLS algorithm with error feedback. The present example uses a 25th-order equalizer.

Solution:

Applying the error-feedback lattice RLS algorithm, using $\lambda = 0.99$ with a 25th-order equalizer, we obtain after 100 iterations the equalizer whose impulse response is shown in Fig. 7.4. The appropriate value of L for this case is 18. The algorithm is initialized with $\epsilon = 0.1$.

The convolution of this response with the channel impulse response is depicted in Fig. 7.5, which clearly approximates an impulse. In this case, the measured MSE was 0.3056, a value comparable with that obtained with the LMS algorithm in the example of subsection 3.6.3. Note that in the LMS case a 50th-order equalizer was used.

$\square$

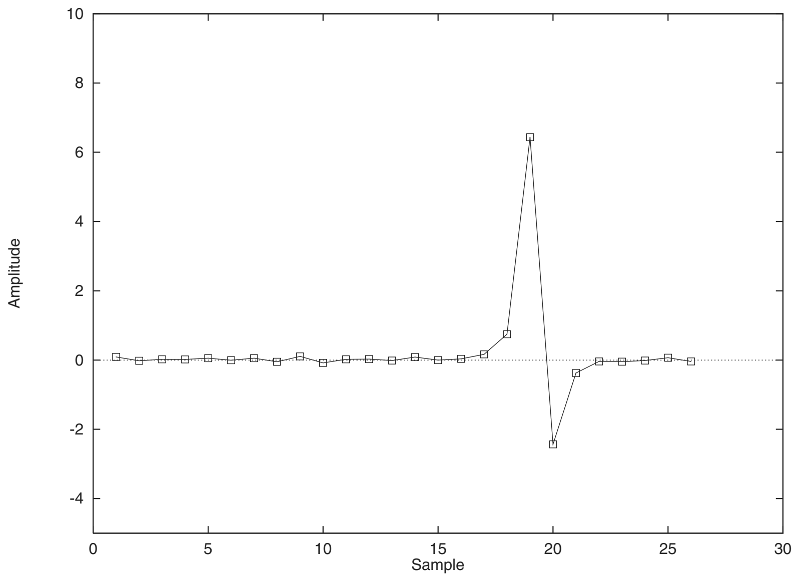

Figure 7.4 Equalizer impulse response, lattice RLS algorithm with error feedback.

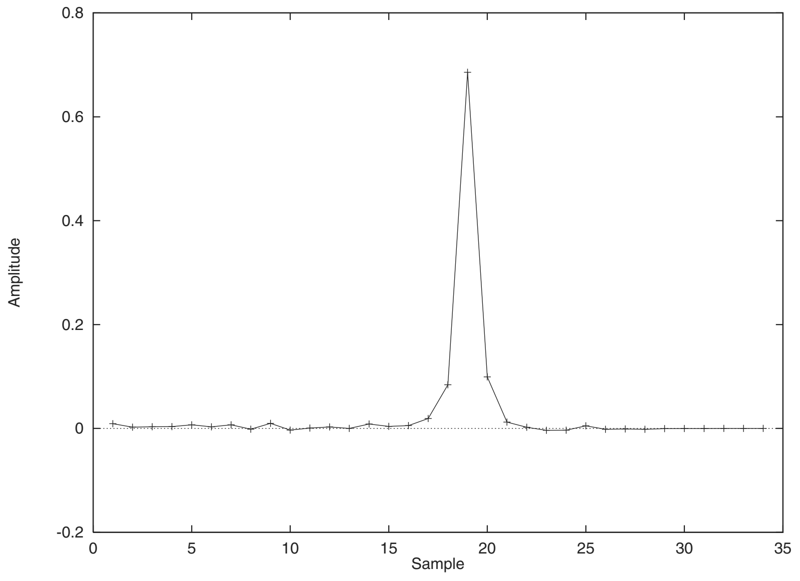

Figure 7.5 Convolution result, lattice RLS algorithm with error feedback.

7.11 CONCLUDING REMARKS

A number of alternative RLS algorithms based on the lattice realization were introduced. These algorithms consist of stages where growing-order forward and backward predictors of the input signal are built from stage to stage. This feature makes the lattice-based algorithms attractive in a number of applications where information about the statistics of the input signal, such as the order of the input signal model, is useful. Another important feature of the lattice-based algorithms is their robust performance when implemented in finite-precision arithmetic.

Also, their computational complexity of at least $16N$ multiplications per output sample is acceptable in a number of practical situations. However, by starting from the lattice formulation without making extensive use of order updating, it is possible to derive the fast transversal RLS algorithms, which can reduce the computational complexity to orders of $7N$ multiplications per output sample. The derivation of these algorithms is the subject of the Chapter 8.

Several interesting topics related to the lattice formulation of adaptive filters have been addressed in the open literature [9]-[13]. The geometric formulation of the least-squares estimation problem can be used to derive the lattice-based algorithms [9] in an elegant manner. Also, an important situation that we usually find in practice is the case where the input data cannot be considered zero before the first iteration of the adaptive algorithm. The derivation of the lattice algorithms that account for nonzero initial conditions for the input data is found in [10]. Another important problem is the characterization of the conditions under which the stability of the lattice algorithm is maintained when perturbations to the normal operation occur [11]. There is also a family of lattice-based algorithms employing gradient-type updating equations. These algorithms present reduced computational complexity and good behavior when implemented with finite-precision arithmetic [12]-[13].

A number of simulation examples involving the lattice algorithms were presented. In these examples the performance of the lattice algorithm was evaluated in different applications as well as in finite-precision implementations.

7.12 REFERENCES

1. D. L. Lee, M. Morf, and B. Friedlander, "Recursive least squares ladder estimation algorithms," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-29, pp. 627-641, June 1981.

2. B. Friedlander, "Lattice filters for adaptive processing," *Proceedings of the IEEE*, vol. 70, pp. 829-867, Aug. 1982.

3. F. Ling, D. Manolakis, and J. G. Proakis, "Numerically robust least-squares lattice-ladder algorithms with direct updating of the reflection coefficients," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-34, pp. 837-845, Aug. 1986.

4. M. Bellanger, *Adaptive Digital Filters and Signal Processing*, Marcel Dekker, Inc., New York, NY, 2nd edition, 2001.

5. S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Englewood Cliffs, NJ, 4th edition, 2002.

6. J. G. Proakis, C. M. Rader, F. Ling, and C. L. Nikias, *Advanced Digital Signal Processing*, MacMillan, New York, NY, 1992.

7. C. G. Samson and V. U. Reddy, "Fixed point error analysis of the normalized ladder algorithm," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-31, pp. 1177-1191, Oct. 1983.

8. R. C. North, J. R. Zeidler, W. H. Ku, and T. R. Albert, "A floating-point arithmetic error analysis of direct and indirect coefficient updating techniques for adaptive lattice filters," *IEEE Trans. on Signal Processing*, vol. 41, pp. 1809-1823, May 1993.

9. H. Lev-Ari, T. Kailath, and J. M. Cioffi, "Least-squares adaptive lattice and transversal filters: A unified geometric theory," *IEEE Trans. on Information Theory*, vol. IT-30, pp. 222-236, March 1984.

10. J. M. Cioffi, "An unwindowed RLS adaptive lattice algorithm," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. 36, pp. 365-371, March 1988.

11. H. Lev-Ari, K.-F. Chiang, and T. Kailath, "Constrained-input/constrained-output stability for adaptive RLS lattice filters," *IEEE Trans. on Circuits and Systems*, vol. 38, pp. 1478-1483, Dec. 1991.

12. V. J. Mathews and Z. Xie, "Fixed-point error analysis of stochastic gradient adaptive lattice filters," *IEEE Trans. on Signal Processing*, vol. 31, pp. 70-80, Jan. 1990.

13. M. Reed and B. Liu, "Analysis of simplified gradient adaptive lattice algorithms using power-of-two quantization," *Proc. IEEE Intern. Symp. on Circuits and Systems*, New Orleans, LA, pp. 792-795, May 1990.

7.13 PROBLEMS

1. Deduce the time-updating formula for the backward predictor coefficients.
2. Given a square matrix
$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}$$
where **A** and **D** are also square matrices, the inverse of **P** can be expressed as
$$\begin{aligned} \mathbf{P}^{-1} &= \begin{bmatrix} \mathbf{A}^{-1}[\mathbf{I} + \mathbf{B}(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1}] & -\mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1} \\ -(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1} & (\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1} \end{bmatrix} \\ &= \begin{bmatrix} (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} & -(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1}\mathbf{B}\mathbf{D}^{-1} \\ -\mathbf{D}^{-1}\mathbf{C}(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} & \mathbf{D}^{-1}[\mathbf{I} + \mathbf{C}(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1}\mathbf{B}\mathbf{D}^{-1}] \end{bmatrix} \end{aligned}$$
(a) Show the validity of this result.

(b) Use the appropriate partitioned forms of $\mathbf{R}_D(k-1,i)$ to derive the partitioned forms of $\mathbf{S}_D(k-1,i)$ of equations (7.56) and (7.59).
3. Derive the time-updating formula of $\delta_D(k,i)$.
4. Demonstrate that the backward *a posteriori* prediction errors $\varepsilon_b(k,i)$ and $\varepsilon_b(k,j)$ for $i \neq j$ are uncorrelated when the average is calculated over time.
5. Justify the initialization of $\xi^d_{b_{\min}}(0)$ and $\xi^d_{f_{\min}}(0)$ in the lattice RLS algorithm.
6. Derive the *a posteriori* lattice RLS algorithm for complex input signals.
7. Derive equation (7.71).
8. Derive the order-updating equation of the normalized forward and backward errors.
9. Demonstrate the validity of the order-updating formula of the weighted least-squares error of the joint-process estimation described in equation (7.88).
10. Derive equation (7.89).
11. Derive the error-feedback LRLS algorithm based on *a priori* errors.
12. Derive the normalized LRLS algorithm based on *a priori* errors.
13. The lattice RLS algorithm based on *a posteriori* errors is used to predict the signal $x(k) = \sin\frac{\pi k}{4}$. Given $\lambda = 0.99$, calculate the error and the tap coefficients for the first 10 iterations.
14. The normalized lattice RLS algorithm based on *a posteriori* errors is used to predict the signal $x(k) = \sin\frac{\pi k}{4}$. Given $\lambda = 0.99$, calculate the error and the multiplier coefficients for the first 10 iterations.

15. The error-feedback LRLS algorithm is applied to identify a 7th-order time-varying unknown system whose coefficients are first-order Markov processes with $\lambda_{\mathbf{w}} = 0.999$ and $\sigma_{\mathbf{w}}^2 = 0.033$. The initial time-varying system multiplier coefficients are

    $\mathbf{w}_o^T = [0.03490 \;\; -0.01100 \;\; -0.06864 \;\; 0.22391 \;\; 0.55686 \;\; 0.35798 \;\; -0.02390 \;\; -0.07594]$

    The input signal is Gaussian white noise with variance $\sigma_x^2 = 1$ and the measurement noise is also Gaussian white noise independent of the input signal and of the elements of $\mathbf{n}_{\mathbf{w}}(k)$ with variance $\sigma_n^2 = 0.01$.

    Simulate the experiment above described and measure the excess MSE for $\lambda = 0.97$ and $\lambda = 0.99$.

16. Repeat the experiment described in problem 15 using the normalized lattice algorithm.

17. Suppose that a 15th-order FIR digital filter with the multiplier coefficients given below is identified through an adaptive FIR filter of the same order using the unnormalized LRLS algorithm. Considering that fixed-point arithmetic is used, simulate the identification problem described using the following specifications:

Additional noise : white noise with variance	$\sigma_n^2 = 0.0015$
Coefficients wordlength:	$b_c = 16$ bits
Signal wordlength:	$b_d = 16$ bits
Input signal: Gaussian white noise with variance	$\sigma_x^2 = 0.7$
	$\lambda = 0.98$

    $\mathbf{w}_o^T = [0.0219360 \;\; 0.0015786 \;\; -0.0602449 \;\; -0.0118907 \;\; 0.1375379$
    $0.0574545 \;\; -0.3216703 \;\; -0.5287203 \;\; -0.2957797 \;\; 0.0002043 \;\; 0.290670$
    $-0.0353349 \;\; -0.0068210 \;\; 0.0026067 \;\; 0.0010333 \;\; -0.0143593]$

    Plot the learning curves for the finite- and infinite-precision implementations. Also plot $E[||\Delta\kappa_f(k,0)||^2]$ and $E[||\Delta\kappa_b(k,0)||^2]$ versus k in both cases.

18. Repeat the above problem for the following cases:

    (a) $\sigma_n^2 = 0.01, b_c = 9$ bits, $b_d = 9$ bits, $\sigma_x^2 = 0.7, \lambda = 0.98$.

    (b) $\sigma_n^2 = 0.1, b_c = 10$ bits, $b_d = 10$ bits, $\sigma_x^2 = 0.8, \lambda = 0.98$.

    (c) $\sigma_n^2 = 0.05, b_c = 8$ bits, $b_d = 16$ bits, $\sigma_x^2 = 0.8, \lambda = 0.98$.

19. In problem 17, rerun the simulations for $\lambda = 1, \lambda = 0.940$. Comment on the results.

20. Repeat problem 18, using the normalized and error-feedback LRLS algorithms. Compare the results for the different algorithms.

21. Repeat problem 17 for the case where the input signal is a first-order Markov process with $\lambda_{\mathbf{x}} = 0.98$.

22. Given a channel with impulse response

    $$h(k) = 0.9^k + 0.4^k$$

    for $k = 0, 1, 2, \ldots, 25$, design an adaptive equalizer. The input signal is white noise with unit variance and the adaptive-filter input signal-to-noise ratio is 30 dB. Use the unnormalized lattice algorithm of order 35.

23. The unnormalized lattice algorithm is used to perform the forward prediction of a signal $x(k)$ generated by applying zero-mean Gaussian white noise signal with unit variance to the input of a linear filter with transfer function given by

$$H(z) = \frac{0.5}{(1 - 1.512z^{-1} + 0.827z^{-2})(1 - 1.8z^{-1} + 0.87z^{-2})}$$

Calculate the zeros of the resulting predictor error transfer function and compare with the poles of the linear filter.

24. Determine the computational complexity of the Algorithms 7.1, 7.2, 7.3, and 7.4.