6. Wiener and Adaptive Filters

J. Benesty, Y. Huang, J. Chen

The Wiener filter, named after its inventor, has been an extremely useful tool since its invention in the early 1930s. This optimal filter is not only popular in different aspects of speech processing but also in many other applications. This chapter presents the most fundamental results of the Wiener theory with an emphasis on the Wiener–Hopf equations, which are not convenient to solve in practice. An alternative approach to solving these equations directly is the use of an adaptive filter, which is why this work also describes the most classical adaptive algorithms that are able to converge, in a reasonable amount of time, to the optimal Wiener filter.

6.1 Overview

In his landmark manuscript on extrapolation, interpolation, and smoothing of stationary time series [6.1], Norbert Wiener was one of the first researchers to treat the filtering problem of estimating a process corrupted by additive noise. The optimum estimate that he derived, required the solution of an integral equation known as the Wiener–Hopf equation [6.2]. Soon after he published his work, Levinson formulated the same problem in discrete time [6.3]. Levinson's contribution has had a great impact on the field. Indeed, thanks to him, Wiener's ideas have become more accessible to many engineers. A very nice overview of linear filtering theory and the history of the different discoveries in this area can be found in [6.4].

In this chapter, we will show that the Wiener theory plays a fundamental role in system identification. For example, in many speech applications, impulse responses between loudspeakers (or speech sources) and microphones need to be identified. Thanks to many (adaptive) algorithms directly derived from the Wiener–Hopf equations, this task is now rather easy.

This chapter is organized as follows. Section 6.2 presents the four basic signal models used in this work. In Sect. 6.3, we derive the optimal Wiener filter for a single-input single-output (SISO) system. Section 6.4 explains what happens if the length of the modeling filter is shorter than the length of the true impulse response (this case always occurs in practice). It is extremely useful in many applications to be able to say how the input signal correlation matrix, which appears in the Wiener–Hopf equations, is conditioned. So we dedicate Sect. 6.5 to a detailed discussion on the condition number of this matrix. In Sect. 6.6, we present a collection of basic adaptive

filters. We insist on the classical normalized least-mean-square (NLMS) algorithm. Section 6.7 generalizes the Wiener filter to the multiple-input multiple-output (MIMO) system case. While this generalization is straightforward, the optimal solution does not always exist and identification problems may be possible only in some situations. Finally, we give our conclusions in Sect. 6.8.

6.2 Signal Models

In many speech applications, a system with a number of inputs and outputs needs to be identified. In this section, we explain the four basic signal models. This classification is now well accepted and is the basis of many interesting studies in different areas of control and signal processing.

6.2.1 SISO Model

The first model we consider is the single-input single-output (SISO) system, as shown in Fig. 6.1a. The output signal is given by

$$x(k) = h * s(k) + b(k) \,, \tag{6.1}$$

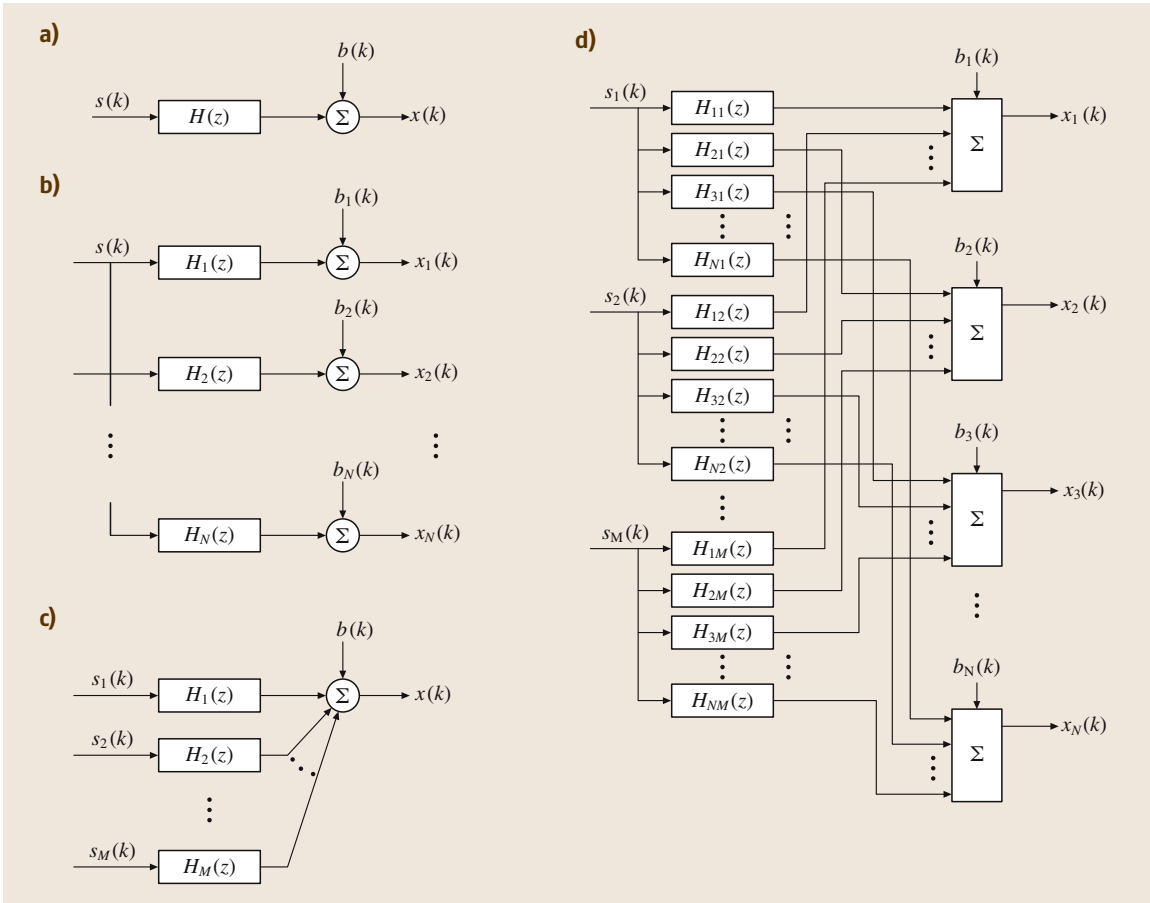

Fig. 6.1a–d Illustration of four distinct types of systems. **(a)** A single-input single-output (SISO) system. **(b)** A single-input multiple-output (SIMO) system. **(c)** A multiple-input single-output (MISO) system. **(d)** A multiple-input multiple-output (MIMO) system.

where h is the channel impulse response, the symbol $*$ denotes the linear convolution operator, $s(k)$ is the source signal at time k, and $b(k)$ is the additive noise at the output. Here we assume that the system is linear and shift-invariant. The channel impulse response is delineated usually with a finite impulse response (FIR) filter rather than an infinite impulse response (IIR) filter. In vector/matrix form, the SISO signal model (6.1) is written as:

$$x(k) = \boldsymbol{h}^{\mathrm{T}}\boldsymbol{s}(k) + b(k)\,, \tag{6.2}$$

where

$$\boldsymbol{h} = [h_0 \;\; h_1 \;\; \cdots \;\; h_{L-1}]^{\mathrm{T}}\,,$$

$$\boldsymbol{s}(k) = [s(k) \;\; s(k-1) \;\; \cdots \;\; s(k-L+1)]^{\mathrm{T}}\,,$$

where $[\cdot]^{\mathrm{T}}$ denotes the transpose of a matrix or a vector, and L is the channel length.

Using the z transform, the SISO signal model (6.2) is described as follows:

$$X(z) = H(z)S(z) + B(z)\,, \tag{6.3}$$

where $X(z)$, $S(z)$, and $B(z)$ are the z-transforms of $x(k)$, $s(k)$, and $b(k)$, respectively, and $H(z) = \sum_{l=0}^{L-1} h_l z^{-l}$.

The SISO model is simple and is probably the most widely used and studied model in communication, signal processing, and control theories.

6.2.2 SIMO Model

The diagram of a single-input multiple-output (SIMO) system is illustrated in Fig. 6.1b, in which there are N outputs from the same source as input and the n-th output is expressed as:

$$x_n(k) = \boldsymbol{h}_n^{\mathrm{T}}\boldsymbol{s}(k) + b_n(k)\,, \quad n = 1, 2, \cdots, N\,, \tag{6.4}$$

where $x_n(k)$, $\boldsymbol{h}_n$, and $b_n(k)$ are defined in a similar way to those in (6.2), and L is the length of the longest channel impulse response in this SIMO system. A more-comprehensive expression of the SIMO model is given by

$$\boldsymbol{x}(k) = \mathbf{H}\boldsymbol{s}(k) + \boldsymbol{b}(k)\,, \tag{6.5}$$

where

$$\boldsymbol{x}(k) = [x_1(k) \;\; x_2(k) \;\; \cdots \;\; x_N(k)]^{\mathrm{T}}\,,$$

$$\mathbf{H} = \begin{pmatrix} h_{1,0} & h_{1,1} & \cdots & h_{1,L-1} \\ h_{2,0} & h_{2,1} & \cdots & h_{2,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{N,0} & h_{N,1} & \cdots & h_{N,L-1} \end{pmatrix}_{N\times L}\,,$$

$$\boldsymbol{b}(k) = [b_1(k) \;\; b_2(k) \;\; \cdots \;\; b_N(k)]^{\mathrm{T}}\,.$$

The SIMO model (6.5) is described in the z-transform domain as:

$$\boldsymbol{X}(z) = \boldsymbol{H}(z)S(z) + \boldsymbol{B}(z)\,, \tag{6.6}$$

where

$$\boldsymbol{X}(z) = [X_1(z) \;\; X_2(z) \;\; \cdots \;\; X_N(z)]^{\mathrm{T}}\,,$$

$$\boldsymbol{H}(z) = [H_1(z) \;\; H_2(z) \;\; \cdots \;\; H_N(z)]^{\mathrm{T}}\,,$$

$$H_n(z) = \sum_{l=0}^{L-1} h_{n,l} z^{-l}, \quad n = 1, 2, \cdots, N\,,$$

$$\boldsymbol{B}(z) = [B_1(z) \;\; B_2(z) \;\; \cdots \;\; B_N(z)]^{\mathrm{T}}\,.$$

6.2.3 MISO Model

In the third type of systems as drawn in Fig. 6.1c, we suppose that there are M sources but only one output, whose signal is then expressed as:

$$\begin{aligned} x(k) &= \sum_{m=1}^{M} \boldsymbol{h}_m^{\mathrm{T}}\boldsymbol{s}_m(k) + b(k)\,, \\ &= \mathbf{h}^{\mathrm{T}}\mathbf{s}(k) + b(k)\,, \end{aligned} \tag{6.7}$$

where

$$\mathbf{h} = \left[\boldsymbol{h}_1^{\mathrm{T}} \;\; \boldsymbol{h}_2^{\mathrm{T}} \;\; \cdots \;\; \boldsymbol{h}_M^{\mathrm{T}}\right]^{\mathrm{T}}\,,$$

$$\boldsymbol{h}_m = [h_{m,0} \;\; h_{m,1} \;\; \cdots \;\; h_{m,L-1}]^{\mathrm{T}}\,,$$

$$\mathbf{s}(k) = \left[\boldsymbol{s}_1^{\mathrm{T}}(k) \;\; \boldsymbol{s}_2^{\mathrm{T}}(k) \;\; \cdots \;\; \boldsymbol{s}_M^{\mathrm{T}}(k)\right]^{\mathrm{T}}\,,$$

$$\boldsymbol{s}_m(k) = [s_m(k) \;\; s_m(k-1) \;\; \cdots \;\; s_m(k-L+1)]^{\mathrm{T}}\,.$$

In the z-transform domain, the MISO model is given by

$$X(z) = \boldsymbol{H}^{\mathrm{T}}(z)\boldsymbol{S}(z) + B(z)\,, \tag{6.8}$$

where

$$\boldsymbol{H}(z) = [H_1(z) \;\; H_2(z) \;\; \cdots \;\; H_M(z)]^{\mathrm{T}}\,,$$

$$H_m(z) = \sum_{l=0}^{L-1} h_{m,l} z^{-l}, \quad m = 1, 2, \cdots, M\,,$$

$$\boldsymbol{S}(z) = [S_1(z) \;\; S_2(z) \;\; \cdots \;\; S_M(z)]^{\mathrm{T}}\,.$$

Note that $\boldsymbol{H}(z)$ defined here is slightly different from that in (6.6). We do not deliberately distinguish them since their dimension can be easily deduced from the context if slight attention is paid.

6.2.4 MIMO Model

Figure 6.1d depicts a multiple-input multiple-output (MIMO) system. A MIMO system with M inputs and N outputs is referred to as an $M \times N$ system. At time k, we have

$$\boldsymbol{x}(k) = \mathbf{H}\boldsymbol{s}(k) + \boldsymbol{b}(k)\,, \tag{6.9}$$

where

$$\boldsymbol{x}(k) = [x_1(k)\ \ x_2(k)\ \ \cdots\ \ x_N(k)]^{\mathrm{T}}\,,$$

$$\mathbf{H} = (\boldsymbol{H}_1\ \ \boldsymbol{H}_2\ \ \cdots\ \ \boldsymbol{H}_M)\,,$$

$$\mathbf{H}_m = \begin{pmatrix} h_{1m,0} & h_{1m,1} & \cdots & h_{1m,L-1} \\ h_{2m,0} & h_{2m,1} & \cdots & h_{2m,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{Nm,0} & h_{Nm,1} & \cdots & h_{Nm,L-1} \end{pmatrix}_{N\times L}\,,$$

$$m = 1, 2, \cdots, M,$$

$$\boldsymbol{b}(k) = [b_1(k)\ \ b_2(k)\ \ \cdots\ \ b_N(k)]^{\mathrm{T}}\,,$$

where h_{nm} $(n = 1, 2, \cdots, N,\ m = 1, 2, \cdots, M)$ is the impulse response of the channel from input m to output n, and $\mathbf{s}(k)$ is defined similarly to that in (6.7). Again, we have the model presented in the z-transform domain as

$$\boldsymbol{X}(z) = \mathbf{H}(z)\boldsymbol{S}(z) + \boldsymbol{B}(z)\,, \tag{6.10}$$

where

$$\mathbf{H}(z) = \begin{pmatrix} H_{11}(z) & H_{12}(z) & \cdots & H_{1M}(z) \\ H_{21}(z) & H_{22}(z) & \cdots & H_{2M}(z) \\ \vdots & \vdots & \ddots & \vdots \\ H_{N1}(z) & H_{N2}(z) & \cdots & H_{NM}(z) \end{pmatrix}\,,$$

$$H_{nm}(z) = \sum_{l=0}^{L-1} h_{nm,l} z^{-l}\,, \quad n = 1, 2, \cdots, N\,,$$

$$m = 1, 2, \cdots, M\,.$$

Clearly the MIMO system is the most general model and the other three systems can be treated as special examples of a MIMO system.

6.3 Derivation of the Wiener Filter

In this section, we are interested in the SISO system represented by (6.2). We assume that $x(k)$ and the random noise signal $b(k)$ (independent of $s(k)$) are zero-mean and stationary.

With the Wiener theory, it is possible to identify the impulse response $\boldsymbol{h}$, given $s(k)$ and $x(k)$. Define the error signal,

$$\begin{aligned} e(k) &= x(k) - \hat{x}(k) \\ &= x(k) - \hat{\boldsymbol{h}}_{\mathrm{f}}^{\mathrm{T}} \boldsymbol{s}_{\mathrm{f}}(k)\,, \end{aligned} \tag{6.11}$$

where

$$\hat{\boldsymbol{h}}_{\mathrm{f}} = [\hat{h}_0\ \ \hat{h}_1\ \ \cdots\ \ \hat{h}_{L_{\mathrm{f}}-1}]^{\mathrm{T}}$$

is an estimate of $\boldsymbol{h}$ of length $L_{\mathrm{f}} \le L$ and

$$\boldsymbol{s}_{\mathrm{f}}(k) = [s(k)\ \ s(k-1)\ \ \cdots\ \ s(k-L_{\mathrm{f}}+1)]^{\mathrm{T}}\,.$$

To find the optimal filter, we need to minimize a cost function which is always built around the error signal (6.11). The usual choice for this criterion is the mean-square error (MSE) [6.5],

$$J(\hat{\boldsymbol{h}}_{\mathrm{f}}) = E\{e^2(k)\}\,, \tag{6.12}$$

where $E\{\cdot\}$ denotes mathematical expectation.

The optimal Wiener filter, $\hat{\boldsymbol{h}}_{\mathrm{f,o}}$, is the one that cancels the gradient of $J(\hat{\boldsymbol{h}}_{\mathrm{f}})$, i. e.,

$$\frac{\partial J(\hat{\boldsymbol{h}}_{\mathrm{f}})}{\partial \hat{\boldsymbol{h}}_{\mathrm{f}}} = \mathbf{0}_{L_{\mathrm{f}}\times 1}\,. \tag{6.13}$$

We have:

$$\begin{aligned} \frac{\partial J(\hat{\boldsymbol{h}}_{\mathrm{f}})}{\partial \hat{\boldsymbol{h}}_{\mathrm{f}}} &= 2E\left\{e(k)\frac{\partial e(k)}{\partial \hat{\boldsymbol{h}}_{\mathrm{f}}}\right\} \\ &= -2E\{e(k)\boldsymbol{s}_{\mathrm{f}}(k)\}\,. \end{aligned} \tag{6.14}$$

Therefore, at the optimum, we have:

$$E\{e_{\mathrm{o}}(k)\boldsymbol{s}_{\mathrm{f}}(k)\} = \mathbf{0}_{L_{\mathrm{f}}\times 1}\,, \tag{6.15}$$

where

$$e_{\mathrm{o}}(k) = x(k) - \hat{\boldsymbol{h}}_{\mathrm{f,o}}^{\mathrm{T}} \boldsymbol{s}_{\mathrm{f}}(k) \tag{6.16}$$

is the error signal for which $J(\hat{\boldsymbol{h}}_{\mathrm{f}})$ is minimized (i. e., the optimal filter). Expression (6.15) is called the *principle of orthogonality*.

The optimal estimate of $x(k)$ is:

$$\hat{x}_{\mathrm{o}}(k) = \hat{\boldsymbol{h}}_{\mathrm{f,o}}^{\mathrm{T}} \boldsymbol{s}_{\mathrm{f}}(k)\,. \tag{6.17}$$

It is then easy to check, with the help of the principle of orthogonality, that we also have:

$$E\left\{e_{\mathrm{o}}(k)\hat{x}_{\mathrm{o}}(k)\right\} = 0\,. \tag{6.18}$$

The previous expression is called the *corollary to the principle of orthogonality*.

If we substitute (6.16) into (6.15), we find the *Wiener–Hopf equations*,

$$\mathbf{R}_\mathrm{f}\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \boldsymbol{p}_\mathrm{f}\,, \tag{6.19}$$

where

$$\mathbf{R}_\mathrm{f} = E\left\{\boldsymbol{s}_\mathrm{f}(k)\boldsymbol{s}_\mathrm{f}^\mathrm{T}(k)\right\}$$

is the correlation matrix of the signal $s(k)$ and

$$\boldsymbol{p}_\mathrm{f} = E\{\boldsymbol{s}_\mathrm{f}(k)x(k)\}$$

is the cross-correlation vector between $\boldsymbol{s}_\mathrm{f}(k)$ and $x(k)$.

The correlation matrix is symmetric and positive semidefinite. It is also Toeplitz, i. e., a matrix which has constant values along diagonals,

$$\mathbf{R}_\mathrm{f} = \begin{pmatrix} r(0) & r(1) & \cdots & r(L_\mathrm{f}-1) \\ r(1) & r(0) & \cdots & r(L_\mathrm{f}-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(L_\mathrm{f}-1) & r(L_\mathrm{f}-2) & \cdots & r(0) \end{pmatrix},$$

with $r(l) = E\{s(k)s(k-l)\}$, $l = 0, 1, \cdots, L_\mathrm{f}-1$. In the SISO system case, this matrix is usually positive definite even for quasistationary signals like speech; however, it can be very ill-conditioned.

Assuming that $\mathbf{R}_\mathrm{f}$ is nonsingular, the optimal Wiener filter is:

$$\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \mathbf{R}_\mathrm{f}^{-1}\boldsymbol{p}_\mathrm{f}\,. \tag{6.20}$$

The MSE can be rewritten as:

$$J(\hat{\boldsymbol{h}}_\mathrm{f}) = \sigma_x^2 - 2\boldsymbol{p}_\mathrm{f}^\mathrm{T}\hat{\boldsymbol{h}}_\mathrm{f} + \hat{\boldsymbol{h}}_\mathrm{f}^\mathrm{T}\mathbf{R}_\mathrm{f}\hat{\boldsymbol{h}}_\mathrm{f}\,, \tag{6.21}$$

where $\sigma_x^2 = E\{x^2(k)\}$ is the variance of the input signal $x(k)$. The criterion $J(\hat{\boldsymbol{h}}_\mathrm{f})$ is a quadratic function of the filter coefficient vector $\hat{\boldsymbol{h}}_\mathrm{f}$ and has a single minimum point. This point combines the optimal Wiener filter, as shown above, and a value called the minimum MSE (MMSE), which is obtained by substituting (6.20) into (6.21):

$$\begin{aligned} J_\mathrm{min} &= J(\hat{\boldsymbol{h}}_{\mathrm{f,o}}) \\ &= \sigma_x^2 - \boldsymbol{p}_\mathrm{f}^\mathrm{T}\mathbf{R}_\mathrm{f}^{-1}\boldsymbol{p}_\mathrm{f} \\ &= \sigma_x^2 - \sigma_{\hat{x}_\mathrm{o}}^2\,, \end{aligned} \tag{6.22}$$

where $\sigma_{\hat{x}_\mathrm{o}}^2 = E\{\hat{x}_\mathrm{o}^2(k)\}$ is the variance of the optimal filter output signal $\hat{x}_\mathrm{o}(k)$. This MMSE can be rewritten as:

$$J_\mathrm{min} = \sigma_b^2 + \boldsymbol{h}^\mathrm{T}\mathbf{R}\boldsymbol{h} - \hat{\boldsymbol{h}}_{\mathrm{f,o}}^\mathrm{T}\mathbf{R}_\mathrm{f}\hat{\boldsymbol{h}}_{\mathrm{f,o}}\,, \tag{6.23}$$

where $\sigma_b^2 = E\{b^2(k)\}$ is the variance of the noise and $\mathbf{R} = E\{\boldsymbol{s}(k)\boldsymbol{s}^\mathrm{T}(k)\}$. The value J_min is bounded,

$$\sigma_b^2 \le J_\mathrm{min} \le \sigma_b^2 + \boldsymbol{h}^\mathrm{T}\mathbf{R}\boldsymbol{h}\,, \quad \forall L_\mathrm{f}\,. \tag{6.24}$$

We can easily check that for $L_\mathrm{f} = L$, $J_\mathrm{min} = \sigma_b^2$, and as L_f decreases compared to L, J_min gets closer to its maximum value $\sigma_b^2 + \boldsymbol{h}^\mathrm{T}\mathbf{R}\boldsymbol{h}$.

We define the normalized MMSE as:

$$\tilde{J}_\mathrm{min} = \frac{J_\mathrm{min}}{\sigma_x^2} = 1 - \frac{\sigma_{\hat{x}_\mathrm{o}}^2}{\sigma_x^2}\,. \tag{6.25}$$

According to (6.24), the normalized MMSE always satisfies,

$$\frac{\sigma_b^2}{\sigma_x^2} \le \tilde{J}_\mathrm{min} \le 1\,. \tag{6.26}$$

6.4 Impulse Response Tail Effect

In many scenarios, the impulse response that we try to estimate is either very long or its length is not known so that the length (L_f) of any FIR modeling filter $\hat{\boldsymbol{h}}_\mathrm{f}$ will usually be shorter than the length (L) of the actual impulse response. Let us split this impulse response into two parts:

$$\mathbf{h} = \begin{pmatrix} \boldsymbol{h}_\mathrm{f} \\ \boldsymbol{h}_\mathrm{t} \end{pmatrix}$$

where $\boldsymbol{h}_\mathrm{f}$ is a vector of size L_f and $\boldsymbol{h}_\mathrm{t}$ is the *tail* of the impulse response that is not modeled by $\hat{\boldsymbol{h}}_\mathrm{f}$. Equation (6.2), which represents the SISO system, is now:

$$x(k) = \boldsymbol{h}_\mathrm{f}^\mathrm{T}\boldsymbol{s}_\mathrm{f}(k) + \boldsymbol{h}_\mathrm{t}^\mathrm{T}\boldsymbol{s}_\mathrm{t}(k-L_\mathrm{f}) + b(k)\,, \tag{6.27}$$

where

$$\boldsymbol{s}_\mathrm{t}(k-L_\mathrm{f}) = [s(k-L_\mathrm{f})\;\; s(k-L_\mathrm{f}-1) \;\cdots\; s(k-L+1)]^\mathrm{T}\,.$$

Substituting (6.27) into the cross-correlation vector, we obtain,

$$\boldsymbol{p}_\mathrm{f} = E\{\boldsymbol{s}_\mathrm{f}(k)x(k)\} = \mathbf{R}_\mathrm{f}\boldsymbol{h}_\mathrm{f} + \mathbf{R}_\mathrm{t}(L_\mathrm{f})\boldsymbol{h}_\mathrm{t}\,, \tag{6.28}$$

with $\mathbf{R}_\mathrm{t}(L_\mathrm{f}) = E\{\boldsymbol{s}_\mathrm{f}(k)\boldsymbol{s}_\mathrm{t}^\mathrm{T}(k-L_\mathrm{f})\}$. Finally, inserting the previous expression into the Wiener–Hopf equations (6.20), we obtain:

$$\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \boldsymbol{h}_\mathrm{f} + \mathbf{R}_\mathrm{f}^{-1}\mathbf{R}_\mathrm{t}(L_\mathrm{f})\boldsymbol{h}_\mathrm{t}\,. \tag{6.29}$$

It is clear from (6.29) that the underestimation of the length of the impulse response in the Wiener method

will introduce a bias [equal to $\mathbf{R}_{\mathrm{f}}^{-1}\mathbf{R}_{\mathrm{t}}(L_{\mathrm{f}})\boldsymbol{h}_{\mathrm{t}}$] in the coefficients of the optimal filter. This bias depends on two things: the energy of the tail impulse response and the correlation of the input signal $s(k)$. If $s(k)$ is white, there is no bias since in this particular case the matrix $\mathbf{R}_{\mathrm{t}}(L_{\mathrm{f}})$ is zero. But for highly correlated signals like speech, $\mathbf{R}_{\mathrm{t}}(L_{\mathrm{f}})$ may not be negligible and the second term on the right-hand side of (6.29) may therefore be amplified if the energy of the tail is significative. As a consequence, it is important in practice to have a rough idea of the physics of the system, in order to choose an appropriate length for the modeling filter for good identification. As we can see, increasing the length of the filter will improve the accuracy of the solution. On the other hand, the complexity for solving the linear system will increase and the conditioning of $\mathbf{R}_{\mathrm{f}}$ will be worsened. Therefore, depending on the application, a reasonable balance has to be found.

For simplification, in the rest of this chapter, we will assume that $L_{\mathrm{f}} = L$ so that we can drop the subscript 'f' in all variables. In this scenario: $\mathbf{R}_{\mathrm{f}} = \mathbf{R}$, $\boldsymbol{s}_{\mathrm{f}}(k) = \boldsymbol{s}(k)$, $\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \hat{\boldsymbol{h}}_{\mathrm{o}}$, etc.

6.5 Condition Number

The correlation matrix that appears in the Wiener–Hopf equations needs to be inverted to find the optimal filter. If this matrix is ill-conditioned and the data is perturbed, the accuracy of the solution will suffer a lot if the linear system is solved directly. One way to improve the accuracy is to regularize the covariance matrix. However, this regularization depends on the condition number: the higher the condition number, the larger the regularization. So it is important to be able to estimate this condition number in an efficient way, in order to use this information to improve the quality of the solution. Many other problems require the knowledge of this condition number for different reasons. For example, the performance of many adaptive algorithms depends on this number. Therefore, it is of great interest to have a detailed discussion of this topic here and to develop a practical algorithm to determine this condition number.

6.5.1 Decomposition of the Correlation Matrix

For a vector of length $L+1$,

$$\boldsymbol{s}_{L+1}(k) = [s(k)\ \ s(k-1)\ \ \cdots\ \ s(k-L)]^{\mathrm{T}}\,,$$

the covariance matrix of size $(L+1)\times(L+1)$ is:

$$\begin{aligned}\mathbf{R}_{L+1} &= E\left\{\boldsymbol{s}_{L+1}(k)\boldsymbol{s}_{L+1}^{\mathrm{T}}(k)\right\}\\ &= \begin{pmatrix} r(0) & \boldsymbol{r}_L^{\mathrm{T}} \\ \boldsymbol{r}_L & \mathbf{R}_L \end{pmatrix} = \begin{pmatrix} \mathbf{R}_L & \boldsymbol{r}_{\mathrm{b},L} \\ \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}} & r(0) \end{pmatrix}\,, \end{aligned} \tag{6.30}$$

where $\mathbf{R}_L = E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}$ and

$$\begin{aligned}\boldsymbol{r}_L &= [r(1)\ \ r(2)\ \ \cdots\ \ r(L)]^{\mathrm{T}}\,,\\ \boldsymbol{r}_{\mathrm{b},L} &= [r(L)\ \ r(L-1)\ \ \cdots\ \ r(1)]^{\mathrm{T}}\,.\end{aligned}$$

By using the Schur complements, it is easy to invert $\mathbf{R}_{L+1}$:

$$\mathbf{R}_{L+1}^{-1} = \begin{pmatrix} \mathbf{R}_L^{-1} + \varrho_L^{-1}\boldsymbol{b}_L\boldsymbol{b}_L^{\mathrm{T}} & -\varrho_L^{-1}\boldsymbol{b}_L \\ -\varrho_L^{-1}\boldsymbol{b}_L^{\mathrm{T}} & \varrho_L^{-1} \end{pmatrix}\,, \tag{6.31}$$

where

$$\begin{aligned}\boldsymbol{b}_L &= \mathbf{R}_L^{-1}\boldsymbol{r}_{\mathrm{b},L}\\ &= [b_{L,1}\ \ b_{L,2}\ \ \cdots\ \ b_{L,L}]^{\mathrm{T}}\end{aligned} \tag{6.32}$$

is the backward predictor of length L,

$$\begin{aligned}\varrho_L &= r(0) - \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}}\boldsymbol{b}_L\\ &= r(0) - \boldsymbol{r}_L^{\mathrm{T}}\boldsymbol{a}_L\end{aligned} \tag{6.33}$$

is the prediction error energy, and $\boldsymbol{a}_L = \mathbf{J}_L\boldsymbol{b}_L$ is the forward predictor with $\mathbf{J}_L$ being the co-identity matrix. Equation (6.31) is important and will be used later for a fast computation of the condition number.

6.5.2 Condition Number with the Frobenius Norm

Usually, the condition number is computed by using the 2-norm matrix. However, in the context of Toeplitz matrices, it is more convenient to use the Frobenius norm as explained below and in [6.6, 7].

To simplify the notation, in this subsection we take $\mathbf{R}_{L+1} = \mathbf{R}$. This matrix is symmetric, positive, and assumed to be nonsingular. It can be diagonalized as follows:

$$\mathbf{Q}^{\mathrm{T}}\mathbf{R}\mathbf{Q} = \boldsymbol{\Lambda}\,, \tag{6.34}$$

where

$$\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{Q}\mathbf{Q}^{\mathrm{T}} = \mathbf{I}\,, \tag{6.35}$$

$$\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_{L+1}\}\,, \tag{6.36}$$

and $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{L+1}$. By definition, the square root of $\mathbf{R}$ is:

$$\mathbf{R}^{1/2} = \mathbf{Q}\boldsymbol{\Lambda}^{1/2}\mathbf{Q}^{\mathrm{T}} . \tag{6.37}$$

The condition number of a matrix $\mathbf{R}$ is [6.8]:

$$\chi(\mathbf{R}) = \|\mathbf{R}\|\|\mathbf{R}^{-1}\| , \tag{6.38}$$

where $\|\cdot\|$ can be any matrix norm. Note that $\chi(\mathbf{R})$ depends on the underlying norm and subscripts will be used to distinguish the different condition numbers.

Consider the Frobenius norm:

$$\|\mathbf{R}\|_{\mathrm{F}} = \{\mathrm{tr}(\mathbf{R}^{\mathrm{T}}\mathbf{R})\}^{1/2} . \tag{6.39}$$

We can easily check that, indeed, $\|\cdot\|_{\mathrm{F}}$ is a matrix norm since for any real matrices $\mathbf{A}$ and $\mathbf{B}$ and a real scalar c, the following three conditions are satisfied:

- $\|\mathbf{A}\|_{\mathrm{F}} \geq 0$ and $\|\mathbf{A}\|_{\mathrm{F}} = 0$ if $\mathbf{A} = \mathbf{0}_{(L+1)\times(L+1)}$,
- $\|\mathbf{A}+\boldsymbol{B}\|_{\mathrm{F}} \leq \|\mathbf{A}\|_{\mathrm{F}} + \|\boldsymbol{B}\|_{\mathrm{F}}$,
- $\|c\mathbf{A}\|_{\mathrm{F}} = |c|\|\mathbf{A}\|_{\mathrm{F}}$.

We have:

$$\|\mathbf{R}^{1/2}\|_{\mathrm{F}} = \{\mathrm{tr}(\mathbf{R})\}^{1/2} = \left\{\sum_{l=1}^{L+1}\lambda_l\right\}^{1/2} \tag{6.40}$$

and

$$\|\mathbf{R}^{-1/2}\|_{\mathrm{F}} = \{\mathrm{tr}(\mathbf{R}^{-1})\}^{1/2} = \left\{\sum_{l=1}^{L+1}\frac{1}{\lambda_l}\right\}^{1/2} . \tag{6.41}$$

Hence, the condition number of $\mathbf{R}^{1/2}$ associated with $\|\cdot\|_{\mathrm{F}}$ is:

$$\chi_{\mathrm{F}}(\mathbf{R}^{1/2}) = \|\mathbf{R}^{1/2}\|_{\mathrm{F}}\|\mathbf{R}^{-1/2}\|_{\mathrm{F}} \geq L+1 . \tag{6.42}$$

(The inequality in the previous expression is easy to show by using the Cauchy–Schwartz inequality.) In this section, we choose to work on $\chi_{\mathrm{F}}(\mathbf{R}^{1/2})$ [rather than $\chi_{\mathrm{F}}(\mathbf{R})$], because efficient algorithms can be derived to estimate its value, as will be shown in the next subsection. As far as we know, it does not seem obvious how to estimate $\chi_{\mathrm{F}}(\mathbf{R})$ efficiently.

If $\chi(\mathbf{R}^{1/2})$ is large, then $\mathbf{R}^{1/2}$ is said to be an ill-conditioned matrix. Note that this is a norm-dependent property. However, according to [6.8], any two condition numbers $\chi_\alpha(\mathbf{R}^{1/2})$ and $\chi_\beta(\mathbf{R}^{1/2})$ are equivalent in that constants c_1 and c_2 can be found for which:

$$c_1\chi_\alpha(\mathbf{R}^{1/2}) \leq \chi_\beta(\mathbf{R}^{1/2}) \leq c_2\chi_\alpha(\mathbf{R}^{1/2}) . \tag{6.43}$$

For example, for the 1- and 2-norm matrices and for $\mathbf{R}$, we can show [6.8]

$$\frac{1}{(L+1)^2}\chi_2(\mathbf{R}) \leq \frac{1}{L+1}\chi_1(\mathbf{R}) \leq \chi_2(\mathbf{R}) . \tag{6.44}$$

We now show the same principle for the F- and 2-norm matrices and for $\mathbf{R}^{1/2}$. We recall that:

$$\chi_2(\mathbf{R}^{1/2}) = \sqrt{\frac{\lambda_{L+1}}{\lambda_1}} . \tag{6.45}$$

Since $\mathrm{tr}(\mathbf{R}^{-1}) \geq 1/\lambda_1$ and $\mathrm{tr}(\mathbf{R}) \geq \lambda_{L+1}$, we have

$$\mathrm{tr}(\mathbf{R})\mathrm{tr}(\mathbf{R}^{-1}) \geq \frac{\mathrm{tr}(\mathbf{R})}{\lambda_1} \geq \frac{\lambda_{L+1}}{\lambda_1} , \tag{6.46}$$

hence,

$$\chi_{\mathrm{F}}(\mathbf{R}^{1/2}) \geq \chi_2(\mathbf{R}^{1/2}) . \tag{6.47}$$

Also, since $\mathrm{tr}(\mathbf{R}) \leq (L+1)\lambda_{L+1}$ and $\mathrm{tr}(\mathbf{R}^{-1}) \leq (L+1)/\lambda_1$, we obtain:

$$\mathrm{tr}(\mathbf{R})\mathrm{tr}(\mathbf{R}^{-1}) \leq (L+1)\frac{\mathrm{tr}(\mathbf{R})}{\lambda_1} \leq (L+1)^2\frac{\lambda_{L+1}}{\lambda_1} , \tag{6.48}$$

thus,

$$\chi_{\mathrm{F}}(\mathbf{R}^{1/2}) \leq (L+1)\chi_2(\mathbf{R}^{1/2}) . \tag{6.49}$$

Therefore, we deduce that

$$\chi_2(\mathbf{R}^{1/2}) \leq \chi_{\mathrm{F}}(\mathbf{R}^{1/2}) \leq (L+1)\chi_2(\mathbf{R}^{1/2}) . \tag{6.50}$$

Moreover, by using the two inequalities,

$$\left(\sum_{l=1}^{L+1}\beta_l\right)^2 \geq \sum_{l=1}^{L+1}\beta_l^2 , \tag{6.51}$$

$$\left(\sum_{l=1}^{L+1}\beta_l\right)^2 \leq (L+1)\sum_{l=1}^{L+1}\beta_l^2 , \tag{6.52}$$

where $\beta_l > 0, \forall l$, it is easy to show that

$$\frac{1}{L+1}\chi_{\mathrm{F}}^2(\mathbf{R}^{1/2}) \leq \chi_{\mathrm{F}}(\mathbf{R}) \leq \chi_{\mathrm{F}}^2(\mathbf{R}^{1/2}) \leq (L+1)\chi_{\mathrm{F}}(\mathbf{R}) . \tag{6.53}$$

Note that $\chi_2(\mathbf{R}) = \chi_2^2(\mathbf{R}^{1/2})$ but $\chi_{\mathrm{F}}(\mathbf{R}) \neq \chi_{\mathrm{F}}^2(\mathbf{R}^{1/2})$. According to expressions (6.50) and (6.53), $\chi_{\mathrm{F}}(\mathbf{R}^{1/2})$ and $\chi_{\mathrm{F}}^2(\mathbf{R}^{1/2})$ are a good measure of the condition number of matrices $\mathbf{R}^{1/2}$ and $\mathbf{R}$, respectively. Basically, there is no difference in the trend of the condition numbers of $\mathbf{R}$ and $\mathbf{R}^{1/2}$. In other words, if $\mathbf{R}^{1/2}$ is ill-conditioned (resp. well-conditioned) so is $\mathbf{R}$. In the next subsection, we will show how to compute $\chi_{\mathrm{F}}^2(\mathbf{R}^{1/2})$ by using the Levinson–Durbin algorithm.

Table 6.1 Computation of the condition number with the Levinson–Durbin algorithm

Initialization:	$\varrho_0 = r(0)$
Levinson-Durbin algorithm:	$k_l = \frac{1}{\varrho_{l-1}}\left[r(l) - \boldsymbol{a}_{l-1}^{\mathrm{T}} \boldsymbol{J}_{l-1} \boldsymbol{r}_{l-1}\right]$ $\boldsymbol{a}_l = \begin{pmatrix} \boldsymbol{a}_{l-1} \\ 0 \end{pmatrix} - k_l \boldsymbol{J}_l \begin{pmatrix} -1 \\ \boldsymbol{a}_{l-1} \end{pmatrix}$ $\varrho_l = \varrho_{l-1}(1-k_l^2)$ $l = 1, 2, \cdots, L$
Condition number:	$\chi_{\mathrm{F}}^2\left(\mathbf{R}_{L+1}^{1/2}\right) = (L+1)r(0)\sum_{l=0}^{L} \varrho_l^{-1}\left[1 + \boldsymbol{a}_l^{\mathrm{T}}\boldsymbol{a}_l\right]$

6.5.3 Fast Computation of the Condition Number

In this subsection, we need to compute the two norms $\left\|\mathbf{R}_{L+1}^{1/2}\right\|_{\mathrm{F}}^2$ and $\left\|\mathbf{R}_{L+1}^{-1/2}\right\|_{\mathrm{F}}^2$ efficiently. The calculation of the first is straightforward. Indeed:

$$\left\|\mathbf{R}_{L+1}^{1/2}\right\|_{\mathrm{F}}^2 = \mathrm{tr}(\mathbf{R}_{L+1}) = (L+1)r(0)\,. \tag{6.54}$$

Expression (6.54) requires one multiplication only. Consider the matrix $\mathbf{G}_{L+1} = \mathbf{R}_{L+1}^{-1}$ where its diagonal elements are $g_{L+1,ii}$, $i = 1, 2, \cdots, L+1$. It is clear from (6.31) that the last diagonal component of $\mathbf{G}_{L+1}$ is $g_{L+1,(L+1)(L+1)} = \varrho_L^{-1}$. The L-th diagonal element of $\mathbf{G}_{L+1}$ is $g_{L+1,\mathrm{LL}} = \varrho_{L-1}^{-1} + \varrho_L^{-1} b_{L,L}^2$. Continuing the same process, we easily find:

$$g_{L+1,ii} = \varrho_{i-1}^{-1} + \sum_{l=i}^{L} \varrho_l^{-1} b_{l,i}^2\,, \tag{6.55}$$

with $\varrho_0 = r(0)$. Therefore, from (6.55) we deduce that:

$$\begin{aligned}\left\|\mathbf{R}_{L+1}^{-1/2}\right\|_{\mathrm{F}}^2 &= \mathrm{tr}(\mathbf{G}_{L+1}) \\ &= \sum_{l=0}^{L} \varrho_l^{-1}\left(1 + \boldsymbol{b}_l^{\mathrm{T}}\boldsymbol{b}_l\right) \\ &= \sum_{l=0}^{L} \varrho_l^{-1}\left(1 + \boldsymbol{a}_l^{\mathrm{T}}\boldsymbol{a}_l\right)\,, \end{aligned} \tag{6.56}$$

with $\boldsymbol{a}_0^{\mathrm{T}}\boldsymbol{a}_0 = \boldsymbol{b}_0^{\mathrm{T}}\boldsymbol{b}_0 = 0$.

Finally, the condition number is:

$$\chi_{\mathrm{F}}^2\left(\mathbf{R}_{L+1}^{1/2}\right) = (L+1)r(0)\sum_{l=0}^{L} \varrho_l^{-1}\left(1 + \boldsymbol{a}_l^{\mathrm{T}}\boldsymbol{a}_l\right)\,. \tag{6.57}$$

By using the Toeplitz structure, the Levinson–Durbin algorithm solves the linear prediction equation, $\boldsymbol{a}_L = \mathbf{R}_L^{-1}\boldsymbol{r}_L$, in $O(L^2)$ operations instead of $O(L^3)$. This algorithm computes all predictors $\boldsymbol{a}_l$, $l = 1, 2, \cdots, L$, and this is exactly what we need to compute (6.57). Expression (6.57) also shows a very nice link between the condition number and the predictors of all orders. This algorithm, which has roughly the same complexity as the Levinson–Durbin algorithm, is summarized in Table 6.1. Note that a very efficient algorithm was recently proposed by *Dias* and *Leitão* [6.9] to compute $\mathrm{tr}\{\mathbf{T}\mathbf{R}^{-1}\}$ (where $\mathbf{T}$ is a Toeplitz matrix, this form is a much more-general form than the one used in this section) with the Trench algorithm. Using these techniques here, we can further reduce the complexity [to $\mathcal{O}(L \ln L)$] for the estimation of the overall algorithm.

6.6 Adaptive Algorithms

Solving the Wiener–Hopf equations directly is not very practical, so adaptive algorithms are usually preferred to find the optimal Wiener filter. The aim of this section is to present a couple of basic adaptive algorithms that converge to the actual impulse response $\boldsymbol{h}$ and where the inversion of the correlation matrix $\mathbf{R}$ is avoided.

6.6.1 Deterministic Algorithm

The deterministic or steepest-descent algorithm is actually an iterative algorithm. It is summarized by the simple recursion,

$$\begin{aligned}\hat{\boldsymbol{h}}(k) &= \hat{\boldsymbol{h}}(k-1) + \mu[\boldsymbol{p} - \mathbf{R}\hat{\boldsymbol{h}}(k-1)]\,, \\ & \qquad k = 0, 1, 2, \cdots\,, \end{aligned} \tag{6.58}$$

where μ is a positive constant called the step-size parameter. In this algorithm, $\boldsymbol{p}$ and $\mathbf{R}$ are supposed to be known. The deterministic algorithm can be reformulated with the error signal:

$$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\,, \tag{6.59}$$

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu E\{\boldsymbol{s}(k)e(k)\}\,. \tag{6.60}$$

Now the important question is: what are the conditions on μ to make the algorithm converge to the true impulse response $\boldsymbol{h}$? To answer this question, we will examine the *natural modes* of the algorithm [6.10].

We define the *misalignment vector* as,

$$\boldsymbol{m}(k) = \boldsymbol{h} - \hat{\boldsymbol{h}}(k)\,, \tag{6.61}$$

which is the difference between the true impulse response and the estimated one at time k. The positive quantity $\|\boldsymbol{m}(k)\|_2^2/\|\boldsymbol{h}\|_2^2$ is called the *normalized misalignment*. If we substitute (6.2) into the cross-correlation vector, we get,

$$\boldsymbol{p} = E\{\boldsymbol{s}(k)x(k)\} = \mathbf{R}\boldsymbol{h}\,. \tag{6.62}$$

Inserting (6.62) into (6.58) and subtracting $\boldsymbol{h}$ on both sides of the equation, we obtain:

$$\boldsymbol{m}(k) = (\mathbf{I} - \mu\mathbf{R})\boldsymbol{m}(k-1)\,, \tag{6.63}$$

where $\mathbf{I}$ is the identity matrix. Using the eigendecomposition of $\mathbf{R} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{\mathrm{T}}$ in the previous expression, we get the equivalent form,

$$\boldsymbol{v}(k) = (\mathbf{I} - \mu\boldsymbol{\Lambda})\boldsymbol{v}(k-1)\,, \tag{6.64}$$

where

$$\boldsymbol{v}(k) = \mathbf{Q}^{\mathrm{T}}\boldsymbol{m}(k) = \mathbf{Q}^{\mathrm{T}}[\boldsymbol{h} - \hat{\boldsymbol{h}}(k)]\,. \tag{6.65}$$

Thus, for the l-th natural mode of the steepest-descent algorithm, we have [6.5]

$$v_l(k) = (1 - \mu\lambda_l)v_l(k-1)\,, \quad l = 1, 2, \cdots, L \tag{6.66}$$

or, equivalently,

$$v_l(k) = (1 - \mu\lambda_l)^k v_l(0)\,, \quad l = 1, 2, \cdots, L\,. \tag{6.67}$$

The algorithm converges if,

$$\lim_{k\to\infty} v_l(k) = 0\,, \quad \forall l\,. \tag{6.68}$$

In this case,

$$\lim_{k\to\infty} \hat{\boldsymbol{h}}(k) = \boldsymbol{h}\,. \tag{6.69}$$

It is straightforward to see from (6.67) that a necessary and sufficient condition for the stability of the deterministic algorithm is that,

$$-1 < 1 - \mu\lambda_l < 1\,, \quad \forall l\,, \tag{6.70}$$

which implies,

$$0 < \mu < \frac{2}{\lambda_l}\,, \quad \forall l\,, \tag{6.71}$$

or

$$0 < \mu < \frac{2}{\lambda_{\max}}\,, \tag{6.72}$$

where $\lambda_{\max}$ is the largest eigenvalue of the correlation matrix $\mathbf{R}$.

Let us evaluate the time needed for each natural mode to converge to a given value. Expression (6.67) gives:

$$\ln\frac{|v_l(k)|}{|v_l(0)|} = k\ln|1 - \mu\lambda_l|\,, \tag{6.73}$$

hence,

$$k = \frac{1}{\ln|1 - \mu\lambda_l|}\ln\frac{|v_l(k)|}{|v_l(0)|}\,. \tag{6.74}$$

The *time constant*, τ_l, for the l-th natural mode is defined by taking $|v_l(k)|/|v_l(0)| = 1/e$ (where e is the base of the natural logarithm) in (6.74). Therefore,

$$\tau_l = \frac{-1}{\ln|1 - \mu\lambda_l|}\,. \tag{6.75}$$

We can link the time constant with the condition number of the correlation matrix $\mathbf{R}$. First, let

$$\mu = \frac{\alpha}{\lambda_{\max}}\,, \tag{6.76}$$

where

$$0 < \alpha < 2\,, \tag{6.77}$$

to guaranty the convergence of the algorithm. α is called the normalized step-size parameter. Suppose that the smallest eigenvalue is $\lambda_1 = \lambda_{\min}$; in this case,

$$\tau_1 = \frac{-1}{\ln|1 - \alpha\lambda_{\min}/\lambda_{\max}|} = \frac{-1}{\ln|1 - \alpha/\chi_2(\mathbf{R})|}\,, \tag{6.78}$$

where $\chi_2(\mathbf{R}) = \lambda_{\max}/\lambda_{\min}$. We see that the convergence time of the slowest natural mode depends on the conditioning of $\mathbf{R}$.

From (6.65), we deduce that,

$$\begin{aligned}\boldsymbol{m}^{\mathrm{T}}(k)\boldsymbol{m}(k) &= \boldsymbol{v}^{\mathrm{T}}(k)\boldsymbol{v}(k)\\ &= \left\|\boldsymbol{h}-\hat{\boldsymbol{h}}(k)\right\|_2^2\\ &= \sum_{l=1}^{L}\lambda_l(1-\mu\lambda_l)^k v_l(0)\,. \end{aligned} \tag{6.79}$$

This value gives an idea on the global convergence of the filter to the true impulse response. This convergence is clearly governed by the smallest eigenvalues of $\mathbf{R}$.

We now examine the transient behavior of the MSE. Using (6.2), the error signal (6.59) can be rewritten as

$$\begin{aligned}e(k) &= x(k)-\hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\\ &= b(k)+\boldsymbol{m}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\,, \end{aligned} \tag{6.80}$$

so that the MSE is:

$$\begin{aligned}J(k) &= E\{e^2(k)\}\\ &= \sigma_b^2+\boldsymbol{m}^{\mathrm{T}}(k-1)\mathbf{R}\boldsymbol{m}(k-1)\\ &= \sigma_b^2+\boldsymbol{v}^{\mathrm{T}}(k-1)\boldsymbol{\Lambda}\,\boldsymbol{v}(k-1)\\ &= \sigma_b^2+\sum_{l=1}^{L}\lambda_l(1-\mu\lambda_l)^{2k-2}v_l^2(0)\,. \end{aligned} \tag{6.81}$$

A plot of $J(k)$ versus k is called the *learning curve*. Note that the MSE decays exponentially. When the algorithm is convergent, we see that,

$$\lim_{k\to\infty} J(k) = \sigma_b^2\,. \tag{6.82}$$

This value corresponds to the MMSE, $J_{\min}$, obtained with the optimal Wiener filter when $L_{\mathrm{f}} = L$, which is what we assume in this section.

6.6.2 Stochastic Algorithm

The stochastic gradient or least-mean-square (LMS) algorithm, invented by *Widrow* and *Hoff* in the late 1950s [6.11], is certainly the most popular algorithm that we can find in the literature of adaptive filters. The popularity of the LMS is probably due to the fact that it is easy to understand, easy to implement, and robust in many respects.

One easy way to derive the stochastic gradient algorithm is by approximating the deterministic algorithm. Indeed, in practice, the two quantities $\boldsymbol{p} = E\{\boldsymbol{s}(k)x(k)\}$ and $\mathbf{R} = E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}$ are in general not known. If we take their instantaneous estimates:

$$\hat{\boldsymbol{p}}(k) = \boldsymbol{s}(k)x(k)\,, \tag{6.83}$$

$$\hat{\mathbf{R}}(k) = \boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\,, \tag{6.84}$$

and replace them in the steepest-descent algorithm (6.58), we get:

$$\begin{aligned}\hat{\boldsymbol{h}}(k) &= \hat{\boldsymbol{h}}(k-1)+\mu[\hat{\boldsymbol{p}}(k)-\hat{\mathbf{R}}(k)\hat{\boldsymbol{h}}(k-1)]\\ &= \hat{\boldsymbol{h}}(k-1)+\mu\boldsymbol{s}(k)[x(k)-\boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k-1)]\,. \end{aligned} \tag{6.85}$$

This simple recursion is the LMS algorithm. Contrary to the deterministic algorithm, the LMS weight vector $\hat{\boldsymbol{h}}(k)$ is now a random vector. The three following equations summarize this algorithm [6.5],

$$\hat{x}(k) = \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k-1)\,, \quad \text{filter output,} \tag{6.86}$$

$$e(k) = x(k)-\hat{x}(k)\,, \quad \text{error signal,} \tag{6.87}$$

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1)+\mu\boldsymbol{s}(k)e(k)\,, \quad \text{adaptation,} \tag{6.88}$$

which requires $2L$ additions and $2L+1$ multiplications at each iteration.

The stochastic gradient algorithm has been extensively studied and many theoretical results on its performance have been obtained [6.5, 10, 12]. In particular, we can show the convergence in the mean and mean square (see for example [6.13]), where under the independence assumption, the condition is remarkably the same as the one obtained for the deterministic algorithm, i. e.,

$$0 < \mu < \frac{2}{\lambda_{\max}}\,. \tag{6.89}$$

We can show that the asymptotic MSE for the LMS is:

$$\lim_{k\to\infty} J(k) = \sigma_b^2\left(1+\frac{\mu}{2}L\sigma_s^2\right)\,, \tag{6.90}$$

where $\sigma_s^2 = E\{s^2(k)\}$ is the variance of the input signal $s(k)$. If we compare (6.90) with the asymptotic MSE for the steepest-descent algorithm (6.82), we notice that a positive term,

$$J_{\mathrm{ex}}(\infty) = \frac{\mu}{2}L\sigma_s^2\sigma_b^2\,, \tag{6.91}$$

is added, called the *excess mean-square error*. This term, of course, has a negative effect on the final MSE and its appearance is due to the approximation discussed at the beginning of this subsection. We can reduce its effect by taking a very small μ, but this will increase the convergence time of the LMS. This tradeoff between fast convergence and increased MSE is a well-known effect and is something to consider in any practical implementation.

A simple condition for the stability of LMS is that,

$$|\varepsilon(k)| < |e(k)|\,, \tag{6.92}$$

Table 6.2 The normalized LMS (NLMS) algorithm

Initialization:	$\hat{\boldsymbol{h}}(0) = \mathbf{0}_{L\times 1}$
Parameters:	$0 < \alpha < 2$ $\delta > 0$
Error:	$e(k) = x(k) - \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k-1)$
Update:	$\mu(k) = \dfrac{\alpha}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)+\delta}$ $\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu(k)\boldsymbol{s}(k)e(k)$

where

$$\varepsilon(k) = x(k) - \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k) \tag{6.93}$$

is the a posteriori error signal, computed after the filter is updated. This makes sense intuitively since $\varepsilon(k)$ contains more meaningful information than $e(k)$.

This condition is necessary for the LMS to converge to the true impulse response but not sufficient. However, it is very useful to use here and in many other algorithms to find the bounds for the step size μ.

Inserting (6.88) into (6.93) and using the condition (6.92), we find:

$$0 < \mu < \frac{2}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.94}$$

For L large, $\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k) = L\sigma_s^2 = \mathrm{tr}(\mathbf{R})$. On the other hand, $\mathrm{tr}(\mathbf{R}) = \sum_{l=1}^{L}\lambda_l$ and this implies that $\mathrm{tr}(\mathbf{R}) \geq \lambda_{\max}$. Hence,

$$0 < \mu < \frac{2}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \leq \frac{2}{\lambda_{\max}} \,. \tag{6.95}$$

If we now introduce the normalized step size α ($0 < \alpha < 2$), as we did in the previous subsection, the step size of the LMS will vary with time as follows,

$$\mu(k) = \frac{\alpha}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,, \tag{6.96}$$

and the LMS becomes the normalized LMS (NLMS):

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \frac{\alpha\boldsymbol{s}(k)e(k)}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.97}$$

This algorithm is extremely helpful in practice, especially with nonstationary signals, since $\mu(k)$ can adjust itself at each new iteration. In order to avoid numerical difficulties when the energy of the input signal is small, we regularize the algorithm,

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \frac{\alpha\boldsymbol{s}(k)e(k)}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)+\delta} \,, \tag{6.98}$$

where $\delta > 0$ is the regularization factor. Table 6.2 summarizes this very important algorithm. (Note that the definition of $\mu(k)$ in this table is slightly modified in order to include the regularization parameter δ.)

6.6.3 Variable-Step-Size NLMS Algorithm

The stability of the NLMS algorithm is governed by a step-size parameter. As already discussed, the choice of this parameter, within the stability conditions, reflects a tradeoff between fast convergence and good tracking ability on the one hand, and low misadjustment on the other hand. To meet these conflicting requirements, the step size needs to be controlled. While the formulation of this problem is straightforward, a good and reliable solution is not that easy to find. Many different schemes have been proposed in the last two decades [6.14–21]. In this subsection, we show how to derive in a very simple and elegant way a nonparametric variable-step-size NLMS algorithm.

We define the a priori and a posteriori error signals as, respectively,

$$\begin{aligned} e(k) &= x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\mathbf{s}(k) \\ &= \boldsymbol{s}^{\mathrm{T}}(k)[\boldsymbol{h} - \hat{\boldsymbol{h}}(k-1)] + b(k) \,, \end{aligned} \tag{6.99}$$

$$\begin{aligned} \varepsilon(k) &= x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k)\mathbf{s}(k) \\ &= \boldsymbol{s}^{\mathrm{T}}(k)[\boldsymbol{h} - \hat{\boldsymbol{h}}(k)] + b(k) \,. \end{aligned} \tag{6.100}$$

Consider the linear update equation:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu(k)\boldsymbol{s}(k)e(k) \,. \tag{6.101}$$

One reasonable way to derive a $\mu(k)$ that makes (6.101) stable is to cancel the a posteriori error signal ([6.22] and references therein). Replacing (6.101) in (6.100) with the requirement $\varepsilon(k) = 0$ we easily find, assuming $e(k) \neq 0, \forall k$, that,

$$\mu_{\mathrm{NLMS}}(k) = \frac{1}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.102}$$

Therefore, the obtained algorithm is the classical NLMS.

While this procedure makes sense in the absence of noise, finding the $\mu(k)$, in the presence of noise, that cancels (6.100) will introduce noise in $\hat{\boldsymbol{h}}(k)$ since

Table 6.3 The nonparametric VSS-NLMS (NPVSS-NLMS) algorithm

Initialization:	$\hat{\boldsymbol{h}}(0) = \boldsymbol{0}$
	$\hat{\sigma}_e^2(0) = 0$
Parameters:	$\lambda = 1 - \frac{1}{KL}$, exponential window with $K \geq 2$ σ_b^2, noise power known or estimated $\delta = \text{cst} \cdot \sigma_s^2$, regularization $\epsilon > 0$, very small number to avoid division by zero
Error:	$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)$
Update:	$\hat{\sigma}_e^2(k) = \lambda\hat{\sigma}_e^2(k-1) + (1-\lambda)e^2(k)$ $\varsigma(k) = \left[\delta + \boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)\right]^{-1}\left[1 - \frac{\sigma_b}{\epsilon + \hat{\sigma}_e(k)}\right]$ $\mu_{\text{NPVSS}}(k) = \begin{cases} \varsigma(k) & \text{if } \hat{\sigma}_e(k) \geq \sigma_b \\ 0 & \text{otherwise} \end{cases}$ $\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu_{\text{NPVSS}}(k)\boldsymbol{s}(k)e(k)$

$\boldsymbol{s}^{\mathrm{T}}(k)(\boldsymbol{h} - \hat{\boldsymbol{h}}(k)) = -b(k) \neq 0$, $\forall k$. What we would like, in fact, is to have $\boldsymbol{s}^{\mathrm{T}}(k)(\boldsymbol{h} - \hat{\boldsymbol{h}}(k)) = 0$, $\forall k$, which implies that $\varepsilon(k) = b(k)$. Hence, in this procedure we wish to find the step-size parameter $\mu(k)$ in such a way that

$$E\{\varepsilon^2(k)\} = \sigma_b^2 \,, \quad \forall k \,. \tag{6.103}$$

Using the approximation $\mathbf{s}^{\mathrm{T}}(k)\mathbf{s}(k) = L\sigma_s^2$ for $L \gg 1$, knowing that $\mu(k)$ is deterministic in nature, substituting (6.101) into (6.100), using (6.99) to eliminate $\hat{\mathbf{h}}(k-1)$, and equating to (6.103), we find:

$$\begin{aligned} E\{\varepsilon^2(k)\} &= [1 - \mu(k)L\sigma_s^2]^2\sigma_e^2(k) \\ &= \sigma_b^2 \,, \end{aligned} \tag{6.104}$$

where $\sigma_e^2(k) = E\{e^2(k)\}$ is the power of the error signal. Developing (6.104), we obtain a quadratic equation,

$$\mu^2(k) - \frac{2}{L\sigma_s^2}\mu(k) + \frac{1}{\left(L\sigma_s^2\right)^2}\left[1 - \frac{\sigma_b^2}{\sigma_e^2(k)}\right] = 0 \,, \tag{6.105}$$

for which the obvious solution is,

$$\begin{aligned} \mu_{\text{NPVSS}}(k) &= \frac{1}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)}\left[1 - \frac{\sigma_b}{\sigma_e(k)}\right] \\ &= \mu_{\text{NLMS}}(k)\alpha(k) \,, \end{aligned} \tag{6.106}$$

where $\alpha(k)$ $[0 \leq \alpha(k) \leq 1]$ is the normalized step size. Therefore, the nonparametric VSS-NLMS (NPVSS-NLMS) algorithm is [6.23],

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu_{\text{NPVSS}}(k)\boldsymbol{s}(k)e(k) \,, \tag{6.107}$$

where $\mu_{\text{NPVSS}}(k)$ is defined in (6.106).

We see from (6.106) that, before the algorithm converges, $\sigma_e(k)$ is large compared to σ_b, thus $\mu_{\text{NPVSS}}(k) \approx \mu_{\text{NLMS}}(k)$. On the other hand, when the algorithm starts to converge to the true solution, $\sigma_e(k) \approx \sigma_b$ and $\mu_{\text{NPVSS}}(k) \approx 0$. This is exactly what we desire in order to have both good convergence and low misadjustment. As we can notice, this approach was derived with almost no assumptions compared to all other algorithms belonging to the same family. Table 6.3 summarizes a practical version of the NPVSS-NLMS algorithm.

6.6.4 Proportionate NLMS Algorithms

In this subsection, we explain two very useful algorithms: the proportionate NLMS (PNLMS) and improved PNLMS (IPNLMS) algorithms.

It is well known that the NLMS algorithm converges and tracks slowly, especially for long impulse responses. In many situations where an adaptive algorithm is required, convergence and tracking are critical for a good performance of the entire system. While in the NLMS, the adaptation step is the same for all components of the filter, in the PNLMS [6.24], an adaptive individual step size is assigned to each filter coefficient. The step sizes are calculated from the last estimate of the filter coefficients in such a way that a larger coefficient receives a larger increment, thus increasing the convergence rate of that coefficient. This has the effect that active coefficients are adjusted faster than inactive coefficients (i. e., small or zero coefficients). Hence, PNLMS converges much faster than NLMS for sparse impulse responses. Unfortunately, PNLMS behaves much worse than NLMS when the impulse response is not sparse. This problem is due to the fact that the proportionate update is not very well refined. In [6.25], an IPNLMS was proposed where the adaptive individual step size has a better balance between the fixed step size of NLMS and

the large amount of proportionality in PNLMS. As a result, IPNLMS always converges and tracks better than NLMS and PNLMS, no matter how sparse the impulse response.

The error signal and the coefficient update equation of the two previously discussed algorithms can be written as:

$$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\,, \tag{6.108}$$

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \frac{\alpha \mathbf{G}(k-1)\boldsymbol{s}(k)e(k)}{\boldsymbol{s}^{\mathrm{T}}(k)\mathbf{G}(k-1)\boldsymbol{s}(k)+\delta}\,, \tag{6.109}$$

where

$$\mathbf{G}(k-1) = \mathrm{diag}\{g_0(k-1)\ \ g_1(k-1) \cdots\ \ g_{L-1}(k-1)\} \tag{6.110}$$

is a diagonal matrix that adjusts the step sizes of the individual taps of the filter, α ($0 < \alpha < 2$) is the overall step-size factor, and δ is the regularization parameter.

The NLMS algorithm is obtained by taking:

$$\mathbf{G}(k) = \mathbf{I}\,, \tag{6.111}$$

$$\delta = \delta_{\mathrm{NLMS}} = \mathrm{cst}\sigma_s^2\,, \tag{6.112}$$

where $\mathbf{I}$, σ_s^2, and cst are the identity matrix, the power of the signal $s(k)$, and a small positive constant, respectively.

In the PNLMS, the diagonal elements of $\mathbf{G}(k) = \mathbf{G}_{\mathrm{p}}(k)$ are calculated as follows [6.24]:

$$\gamma_{\mathrm{p},l}(k) = \max\left\{\rho \max\left[\delta_{\mathrm{p}}, \left|\hat{h}_0(k)\right|, \cdots, \left|\hat{h}_{L-1}(k)\right|\right], \left|\hat{h}_l(k)\right|\right\}\,, \tag{6.113}$$

$$g_{\mathrm{p},l}(k) = \frac{\gamma_{\mathrm{p},l}(k)}{\|\boldsymbol{\gamma}_{\mathrm{p}}(k)\|_1}, \quad 0 \le l \le L-1\,, \tag{6.114}$$

where

$$\boldsymbol{\gamma}_{\mathrm{p}}(k) = [\gamma_{\mathrm{p},0}(k)\ \ \gamma_{\mathrm{p},1}(k)\ \cdots\ \gamma_{\mathrm{p},L-1}(k)]^{\mathrm{T}}\,.$$

The parameters δ_{p} and ρ are positive numbers with typical values $\delta_{\mathrm{p}} = 0.01$, $\rho = 0.01$. The first term in (6.113), ρ, prevents $\hat{h}_l(k)$ from stalling when its magnitude is much smaller than the magnitude of the largest coefficient and δ_{p} regularizes the updating when all coefficients are zero at initialization. For the regularization parameter, we usually choose:

$$\delta_{\mathrm{PNLMS}} = \delta_{\mathrm{NLMS}}/L\,. \tag{6.115}$$

For the IPNLMS algorithm, the diagonal matrix, $\mathbf{G}(k) = \mathbf{G}_{\mathrm{ip}}(k)$, is computed in a more-elegant way [6.25]:

$$\gamma_{\mathrm{ip},l}(k) = (1-\beta)\frac{\|\hat{\boldsymbol{h}}(k)\|_1}{L} + (1+\beta)\left|\hat{h}_l(k)\right|\,, \tag{6.116}$$

$$g_{\mathrm{ip},l}(k) = \frac{\gamma_{\mathrm{ip},l}(k)}{\|\boldsymbol{\gamma}_{\mathrm{ip}}(k)\|_1} = \frac{1-\beta}{2L} + (1+\beta)\frac{\left|\hat{h}_l(k)\right|}{2\|\hat{\boldsymbol{h}}(k)\|_1}\,, \quad 0 \le l \le L-1\,, \tag{6.117}$$

where β ($-1 \le \beta < 1$) is a parameter that controls the amount of proportionality in the IPNLMS. For $\beta = -1$, it can easily be checked that the IPNLMS and NLMS algorithms are identical. For β close to 1, IPNLMS behaves like PNLMS. In practice, a good choice for β is -0.5 or 0. With this choice and in simulations, IPNLMS always performs better than NLMS and PNLMS. As for the regularization parameter, it should be taken as:

$$\delta_{\mathrm{IPNLMS}} = \frac{1-\beta}{2L}\delta_{\mathrm{NLMS}}\,. \tag{6.118}$$

The IPNLMS algorithm is summarized in Table 6.4.

Table 6.4 The improved PNLMS (IPNLMS) algorithm

Initialization:	$\hat{h}_l(0) = 0, l = 0, 1, \cdots, L-1$
Parameters:	$-1 \le \beta < 1$ $0 < \alpha < 2$ $\delta_{\mathrm{IPNLMS}} = \mathrm{cst}\cdot\sigma_s^2\frac{1-\beta}{2L}$ $\epsilon > 0$, very small number to avoid division by zero
Error:	$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)$
Update:	$g_{\mathrm{ip},l}(k-1) = \frac{1-\beta}{2L} + (1+\beta)\frac{\left\|\hat{h}_l(k-1)\right\|}{2\left\|\hat{\boldsymbol{h}}(k-1)\right\|_1+\epsilon}$ $\mu(k) = \frac{\alpha}{\sum_{j=0}^{L-1} s^2(k-j)g_{\mathrm{ip},j}(k-1)+\delta_{\mathrm{IPNLMS}}}$ $\hat{h}_l(k) = \hat{h}_l(k-1) + \mu(k)g_{\mathrm{ip},l}(k-1)s(k-l)e(k)$ $l = 0, 1, \cdots, L-1$

Before finishing this subsection, it is worth mentioning another variant of PNLMS, called PNLMS++ [6.26]. In this algorithm, the adaptation of the filter coefficients alternates between NLMS and PNLMS; as a result, PNLMS++ seems slightly less sensitive to the assumption of a sparse impulse response than PNLMS.

6.6.5 Sign Algorithms

Up to now, the only cost function that we have used has been the MSE. What makes this criterion so interesting is that an optimal solution (Wiener) can easily be derived as well as very powerful adaptive algorithms. An alternative to the MSE is the mean absolute error (MAE) [6.27],

$$J_{\mathrm{a}}(\hat{\boldsymbol{h}}) = E\{|e(k)|\} = E\{|x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}\boldsymbol{s}(k)|\} . \tag{6.119}$$

The gradient of this cost function is:

$$\frac{\partial J_{\mathrm{a}}(\hat{\boldsymbol{h}})}{\partial \hat{\boldsymbol{h}}} = -E\{\boldsymbol{s}(k)\mathrm{sgn}[e(k)]\} , \tag{6.120}$$

where

$$\mathrm{sgn}[e(k)] = \frac{e(k)}{|e(k)|} . \tag{6.121}$$

From the instantaneous value of the gradient of $J_{\mathrm{a}}(\hat{\boldsymbol{h}})$, we can derive the sign-error adaptive filter:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu_{\mathrm{a}}\boldsymbol{s}(k)\mathrm{sgn}[e(k)] , \tag{6.122}$$

where μ_{a} is the adaptation step of the algorithm. This algorithm is simplified compared to the LMS since the L multiplications in the update equation are replaced by a sign change of the components of the signal vector $\boldsymbol{s}(k)$. Using the stability condition, $|\varepsilon(k)| < |e(k)|$, we deduce that:

$$0 < \mu_{\mathrm{a}} < \frac{2|e(k)|}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} . \tag{6.123}$$

Another way to simplify the LMS filter is to replace $s(k)$ with its sign. We get the sign–data algorithm:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu'_{\mathrm{a}}\mathrm{sgn}[\boldsymbol{s}(k)]e(k) , \tag{6.124}$$

where μ'_{a} is the adaptation step of the algorithm and the stability condition is:

$$0 < \mu'_{\mathrm{a}} < \frac{2}{\boldsymbol{s}^{\mathrm{T}}(k)\mathrm{sgn}[\boldsymbol{s}(k)]} . \tag{6.125}$$

Combining the two previous approaches, we derive the sign–sign algorithm:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu''_{\mathrm{a}}\mathrm{sgn}[\boldsymbol{s}(k)]\mathrm{sgn}[e(k)] , \tag{6.126}$$

for which the stability condition is:

$$0 < \mu''_{\mathrm{a}} < \frac{2|e(k)|}{\boldsymbol{s}^{\mathrm{T}}(k)\mathrm{sgn}[\boldsymbol{s}(k)]} . \tag{6.127}$$

The algorithms derived in this subsection are very simple to implement and can be very useful in some applications. However, their convergence rate is usually slower than the LMS and their excess MSE is higher [6.28–30].

6.7 MIMO Wiener Filter

In this section, we consider a MIMO system with M inputs and N outputs (see Sect. 6.2 for more details):

$$\boldsymbol{x}(k) = \mathbf{H}\boldsymbol{s}(k) + \boldsymbol{b}(k) , \tag{6.128}$$

where

$$x_n(k) = \sum_{m=1}^{M} \boldsymbol{h}_{nm}^{\mathrm{T}}\boldsymbol{s}_m(k) + b_n(k) = \boldsymbol{h}_{n:}^{\mathrm{T}}\boldsymbol{s}(k) + b_n(k), \quad n = 1, 2, \cdots, N , \tag{6.129}$$

and

$$\mathbf{H} = \begin{pmatrix} \boldsymbol{h}_{11}^{\mathrm{T}} & \boldsymbol{h}_{12}^{\mathrm{T}} & \cdots & \boldsymbol{h}_{1M}^{\mathrm{T}} \\ \boldsymbol{h}_{21}^{\mathrm{T}} & \boldsymbol{h}_{22}^{\mathrm{T}} & \cdots & \boldsymbol{h}_{2M}^{\mathrm{T}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{h}_{N1}^{\mathrm{T}} & \boldsymbol{h}_{N2}^{\mathrm{T}} & \cdots & \boldsymbol{h}_{NM}^{\mathrm{T}} \end{pmatrix}_{N\times ML} = \begin{pmatrix} \boldsymbol{h}_{1:}^{\mathrm{T}} \\ \boldsymbol{h}_{2:}^{\mathrm{T}} \\ \vdots \\ \boldsymbol{h}_{N:}^{\mathrm{T}} \end{pmatrix} . \tag{6.130}$$

We define the error signal at time k at the n-th output as:

$$e_n(k) = x_n(k) - \hat{x}_n(k) = x_n(k) - \hat{\boldsymbol{h}}_{n:}^{\mathrm{T}}\boldsymbol{s}(k) , \quad n = 1, 2, \cdots, N , \tag{6.131}$$

Table 6.5 The MISO NLMS algorithm

Initialization:	$\hat{\boldsymbol{h}}_{n:}(0) = \mathbf{0}_{ML\times 1}$
Parameters:	$0 < \alpha < 2$
	$\delta > 0$
Error:	$e_n(k) = x_n(k) - \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}_{n:}(k-1)$
Update:	$\mu(k) = \dfrac{\alpha}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)+\delta}$
	$\hat{\boldsymbol{h}}_{n:}(k) = \hat{\boldsymbol{h}}_{n:}(k-1) + \mu(k)\boldsymbol{s}(k)e_n(k)$

where $\hat{\boldsymbol{h}}_{n:}$ is an estimate of $\boldsymbol{h}_{n:}$. It is more convenient to define an error signal vector for all outputs:

$$\begin{aligned}\boldsymbol{e}(k) &= \boldsymbol{x}(k) - \hat{\boldsymbol{x}}(k)\\ &= \boldsymbol{x}(k) - \hat{\mathbf{H}}\boldsymbol{s}(k)\,,\end{aligned} \tag{6.132}$$

where $\hat{\boldsymbol{H}}$ is an estimate of $\boldsymbol{H}$ and

$$\boldsymbol{e}(k) = [e_1(k)\;\; e_2(k)\;\; \cdots\;\; e_N(k)]^{\mathrm{T}}\,.$$

Having written the error signal, we now define the MIMO MSE with respect to the modeling filters as:

$$\begin{aligned}J(\hat{\mathbf{H}}) &= E\{\boldsymbol{e}^{\mathrm{T}}(k)\boldsymbol{e}(k)\}\\ &= \sum_{n=1}^{N} E\left\{e_n^2(k)\right\} = \sum_{n=1}^{N} J_n(\hat{\boldsymbol{h}}_{n:})\,.\end{aligned} \tag{6.133}$$

The minimization of (6.133) leads to the MIMO Wiener–Hopf equations:

$$\mathbf{R}_{ss}\hat{\mathbf{H}}_{\mathrm{o}}^{\mathrm{T}} = \mathbf{P}_{sx}\,, \tag{6.134}$$

where

$$\begin{aligned}\mathbf{R}_{ss} &= E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}\\ &= \begin{pmatrix}\mathbf{R}_{s_1s_1} & \mathbf{R}_{s_1s_2} & \cdots & \mathbf{R}_{s_1s_M}\\ \mathbf{R}_{s_2s_1} & \mathbf{R}_{s_2s_2} & \cdots & \mathbf{R}_{s_2s_M}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{R}_{s_Ms_1} & \mathbf{R}_{s_Ms_2} & \cdots & \mathbf{R}_{s_Ms_M}\end{pmatrix}\end{aligned} \tag{6.135}$$

is the input signal covariance matrix (which has a block-Toeplitz structure) with $\mathbf{R}_{s_ms_i} = E\{\boldsymbol{s}_m(k)\boldsymbol{s}_i^{\mathrm{T}}(k)\}$, and

$$\begin{aligned}\mathbf{P}_{sx} &= E\{\boldsymbol{s}(k)\boldsymbol{x}^{\mathrm{T}}(k)\}\\ &= (\boldsymbol{p}_{sx_1}\;\; \boldsymbol{p}_{sx_2}\;\; \cdots\;\; \boldsymbol{p}_{sx_N})\end{aligned} \tag{6.136}$$

is the cross-correlation matrix between the input and output signals, with $\mathbf{p}_{sx_n} = E\{\boldsymbol{s}(k)x_n(k)\}$.

It can easily be seen that the MIMO Wiener–Hopf equations (6.134) can be decomposed into N independent MISO Wiener–Hopf equations,

$$\mathbf{R}_{ss}\hat{\boldsymbol{h}}_{n:,\mathrm{o}} = \boldsymbol{p}_{sx_n}\,,\quad n = 1, 2, \cdots, N\,, \tag{6.137}$$

each one corresponding to an output signal of the system. In other words, minimizing $J(\hat{\mathbf{H}})$ or minimizing each $J_n(\hat{\boldsymbol{h}}_{n:})$ independently gives exactly the same results from an identification point of view. This observation is very important from a practical point of view when adaptive algorithms need to be designed. Indeed, any MIMO adaptive filter is simplified to N MISO adaptive filters. As an example, we give the MISO NLMS algorithm in Table 6.5. We deduce from this discussion that, obviously, the identification of a SIMO system is equivalent to the identification of N independent SISO systems. As a result, with a reference signal, the identification of any acoustic system simplifies to the identification of SISO or MISO systems.

6.7.1 Conditioning of the Covariance Matrix

The best possible case for the identification of a MISO system is when the input signals $s_m(k)$, $m = 1, 2, \cdots, M$, are uncorrelated. In this scenario, we have:

$$\mathbf{R}_{s_ms_i} = \mathbf{0}_{L\times L}\,,\quad \forall m, i = 1, 2, \cdots, M,\; m \neq i\,, \tag{6.138}$$

and the input signal covariance matrix $\mathbf{R}_{ss}$ is block-diagonal. Therefore, if $\mathbf{R}_{s_ms_m}$, $m = 1, 2, \cdots, M$, are nonsingular and well conditioned, the impulse responses of the MISO system are easy to estimate. This case, however, does not often occur in practice so it is of little interest.

The worst possible case, from an identification point of view, is when the signals $s_m(k)$ are generated from a unique source $s_{\mathrm{s}}(k)$, i. e.,

$$s_m(k) = \boldsymbol{g}_m^{\mathrm{T}}s_{\mathrm{s}}(k)\,,\quad m = 1, 2, \cdots, M\,, \tag{6.139}$$

where

$$\boldsymbol{g}_m = [g_{m,0}\;\; g_{m,1}\;\; \cdots\;\; g_{m,L-1}]^{\mathrm{T}}$$

is the impulse response between the source $s_{\mathrm{s}}(k)$ and the signal $s_m(k)$. In this scenario, it can be shown that matrix $\mathbf{R}_{ss}$ is rank-deficient by, at least, $(M-2)L+1$.

As a matter of fact, from (6.139), the input signal vector $\boldsymbol{s}_{\mathrm{s}}(k)$ can be written as

$$
\begin{aligned}
\mathbf{s}(k) &= \left[\boldsymbol{s}_1^{\mathrm{T}}(k)\ \boldsymbol{s}_2^{\mathrm{T}}(k)\ \cdots\ \boldsymbol{s}_M^{\mathrm{T}}(k)\right]^{\mathrm{T}} \\
&= \mathbf{G}\boldsymbol{s}_{\mathrm{s}}(k) \\
&= \begin{pmatrix}
g_{1,0} & g_{1,1} & \cdots & g_{1,L-1} & 0 & 0 & \cdots & 0 \\
0 & g_{1,0} & g_{1,1} & \cdots & g_{1,L-1} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & g_{1,0} & g_{1,1} & \cdots & & g_{1,L-1} \\
g_{2,0} & g_{2,1} & \cdots & g_{2,L-1} & 0 & 0 & \cdots & 0 \\
0 & g_{2,0} & g_{2,1} & \cdots & g_{2,L-1} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & g_{2,0} & g_{2,1} & \cdots & & g_{2,L-1} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\
g_{M,0} & g_{M,1} & \cdots & g_{M,L-1} & 0 & 0 & \cdots & 0 \\
0 & g_{M,0} & g_{M,1} & \cdots & g_{M,L-1} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & g_{M,0} & g_{M,1} & \cdots & & g_{M,L-1}
\end{pmatrix} \\
&\quad \cdot \begin{pmatrix}
s_{\mathrm{s}}(k) \\
s_{\mathrm{s}}(k-1) \\
s_{\mathrm{s}}(k-2) \\
\vdots \\
s_{\mathrm{s}}(k-2L+3) \\
s_{\mathrm{s}}(k-2L+2)
\end{pmatrix},
\end{aligned}
$$

where $\mathbf{G}$ is an $ML\times(2L-1)$ matrix, containing the impulse responses g_m and is assumed to be of full column rank, and $\boldsymbol{s}_{\mathrm{s}}(k)$ is a $(2L-1)\times 1$ vector. We then have:

$$
\begin{aligned}
\mathbf{R}_{ss} &= E\{\mathbf{s}(k)\mathbf{s}^{\mathrm{T}}(k)\} \\
&= \mathbf{G}E\left\{\boldsymbol{s}_{\mathrm{s}}(k)\boldsymbol{s}_{\mathrm{s}}^{\mathrm{T}}(k)\right\}\mathbf{G}^{\mathrm{T}} \\
&= \mathbf{G}\mathbf{R}_{s_{\mathrm{s}}s_{\mathrm{s}}}\mathbf{G}^{\mathrm{T}}\,, \qquad (6.140)
\end{aligned}
$$

where $\mathbf{R}_{s_{\mathrm{s}}s_{\mathrm{s}}}$ is the source signal covariance matrix of size $(2L-1)\times(2L-1)$, assumed to be full rank. We immediately see from (6.140), that:

$$
\begin{aligned}
\mathrm{Rank}[\mathbf{R}_{ss}] &= \min\{\mathrm{Rank}[\mathbf{G}],\ \mathrm{Rank}[\mathbf{R}_{s_{\mathrm{s}}s_{\mathrm{s}}}]\} \\
&= 2L-1\,, \qquad (6.141)
\end{aligned}
$$

and

$$
\begin{aligned}
\mathrm{Null}[\mathbf{R}_{ss}] &= ML - \mathrm{Rank}[\mathbf{R}_{ss}] \\
&= (M-2)L+1\,, \qquad (6.142)
\end{aligned}
$$

where Null[] and Rank[] denote the dimension of the null space and the rank of a matrix, respectively.

From this analysis, one can see that a MISO system is rank deficient if its inputs are the filtered version of the same source signal. Thus, the Wiener–Hopf equations do not have a unique solution.

In most practical situations, the signals $s_m(k)$, $m=1,2,\cdots,M$, are somehow related. If they are highly coherent, adaptive algorithms will be very slow to converge to the true solution and in some situations, they will converge to a solution that is far from the desired one.

We are now going to show in the particular case of a MISO system with two inputs how a high coherence between these signals affects the condition number of the covariance matrix:

$$
\mathbf{R}_{ss} = \begin{pmatrix}
\mathbf{R}_{s_1s_1} & \mathbf{R}_{s_1s_2} \\
\mathbf{R}_{s_2s_1} & \mathbf{R}_{s_2s_2}
\end{pmatrix}.
$$

For $L\to\infty$, a Toeplitz matrix is asymptotically equivalent to a circulant matrix if its elements are absolutely summable [6.31], which is the case for speech signals. In this situation, we can decompose

$$
\mathbf{R}_{s_ms_i} = \mathbf{F}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}\mathbf{F}\,, \quad m,i=1,2\,, \qquad (6.143)
$$

where $\mathbf{F}$ is the Fourier matrix and the diagonal matrix

$$
\begin{aligned}
\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i} &= \mathrm{diag}\left\{\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(0),\ \underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(1),\right. \\
&\qquad \left.\cdots,\ \underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(L-1)\right\} \qquad (6.144)
\end{aligned}
$$

contains elements corresponding to the L frequency bins that are formed from the discrete Fourier transform (DFT) of the first column of $\mathbf{R}_{s_ms_i}$. Letting $r_{s_ms_i}(l)$ be the auto- and cross-correlation for $m=i$ and $m\neq i$, respectively, we see that the spectral content between two signals is related to the correlation function by

$$
\begin{aligned}
\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(f) &= \sum_{l=-\infty}^{\infty} r_{s_ms_i}(l)e^{-\mathrm{i}2\pi fl}\,, \\
&\quad f=0,1,\cdots,L-1\,. \qquad (6.145)
\end{aligned}
$$

Using (6.143), $\mathbf{R}_{ss}$ can be expressed in terms of its spectra as:

$$\begin{aligned}\mathbf{R}_{ss} &= \mathbf{F}_{\mathrm{d}}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{ss}\mathbf{F}_{\mathrm{d}} \\ &= \begin{pmatrix}\mathbf{F}^{-1} & \mathbf{0}_{L\times L}\\ \mathbf{0}_{L\times L} & \mathbf{F}^{-1}\end{pmatrix}\begin{pmatrix}\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1} & \underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}\\ \underset{\Rightarrow}{\mathbf{R}}_{s_2s_1} & \underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}\end{pmatrix} \\ &\quad\cdot\begin{pmatrix}\mathbf{F} & \mathbf{0}_{L\times L}\\ \mathbf{0}_{L\times L} & \mathbf{F}\end{pmatrix}. \end{aligned} \tag{6.146}$$

To compute the condition number $\chi_{\mathrm{F}}^2(\mathbf{R}_{ss}^{1/2})$ (Sect. 6.5), we need to compute $\mathrm{tr}(\mathbf{R}_{ss})$ and $\mathrm{tr}(\mathbf{R}_{ss}^{-1})$. The first trace is easy to compute. Indeed, using (6.146), we easily find:

$$\begin{aligned}\mathrm{tr}(\mathbf{R}_{ss}) &= \mathrm{tr}\left(\mathbf{F}_{\mathrm{d}}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{ss}\mathbf{F}_{\mathrm{d}}\right) = \mathrm{tr}\left(\underset{\Rightarrow}{\mathbf{R}}_{ss}\right) \\ &= \sum_{l=0}^{L-1}\left(\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}(l)+\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}(l)\right). \end{aligned} \tag{6.147}$$

For the second trace, we have:

$$\mathrm{tr}\left(\mathbf{R}_{ss}^{-1}\right) = \mathrm{tr}\left(\mathbf{F}_{\mathrm{d}}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{ss}^{-1}\mathbf{F}_{\mathrm{d}}\right) = \mathrm{tr}\left(\underset{\Rightarrow}{\mathbf{R}}_{ss}^{-1}\right). \tag{6.148}$$

Furthermore, it is easy to show that:

$$\begin{aligned}\underset{\Rightarrow}{\mathbf{R}}_{ss}^{-1} &= \begin{pmatrix}\underset{\Rightarrow}{\mathbf{R}}_1^{-1} & \mathbf{0}_{L\times L}\\ \mathbf{0}_{L\times L} & \underset{\Rightarrow}{\mathbf{R}}_2^{-1}\end{pmatrix} \\ &\quad\cdot\begin{pmatrix}\mathbf{I}_{L\times L} & -\underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}\\ -\underset{\Rightarrow}{\mathbf{R}}_{s_2s_1}\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1} & \mathbf{I}_{L\times L}\end{pmatrix}, \end{aligned} \tag{6.149}$$

where

$$\underset{\Rightarrow}{\mathbf{R}}_1 = \left[\mathbf{I}_{L\times L} - \underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}^2\left(\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}\right)\right]\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}, \tag{6.150}$$

$$\underset{\Rightarrow}{\mathbf{R}}_2 = \left[\mathbf{I}_{L\times L} - \underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}^2\left(\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}\right)\right]\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}. \tag{6.151}$$

Hence,

$$\begin{aligned}\mathrm{tr}\left(\mathbf{R}_{ss}^{-1}\right) &= \mathrm{tr}\left(\underset{\Rightarrow}{\mathbf{R}}_1^{-1}+\underset{\Rightarrow}{\mathbf{R}}_2^{-1}\right) \\ &= \sum_{l=0}^{L-1}(1-|\gamma(l)|^2)^{-1} \\ &\quad\times\left[\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}(l)+\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}(l)\right], \end{aligned} \tag{6.152}$$

where

$$\begin{aligned}|\gamma(f)|^2 &= \frac{\left|\underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}(f)\right|^2}{\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}(f)\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}(f)}, \\ f &= 0, 1, \cdots, L-1, \end{aligned} \tag{6.153}$$

is the squared interchannel coherence function of the f-th frequency bin.

We finally obtain the relationship between the interchannel coherence and the condition number based on the Frobenius norm:

$$\begin{aligned}\chi_{\mathrm{F}}^2\left(\mathbf{R}_{ss}^{1/2}\right) &= \left\{\sum_{l=0}^{L-1}\left[\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}(l)+\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}(l)\right]\right\} \\ &\quad\times\left\{\sum_{l=0}^{L-1}[1-|\gamma(l)|^2]^{-1}\left[\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}(l)\right.\right. \\ &\quad\left.\left.+\,\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}(l)\right]\right\}. \end{aligned} \tag{6.154}$$

It is now evident from the previous expression that $\chi_{\mathrm{F}}^2(\mathbf{R}_{ss}^{1/2})$ increases with the squared interchannel coherence function, hence degrading the condition of $\mathbf{R}_{ss}$; as $\gamma \to 1$, $\chi_{\mathrm{F}}^2(\mathbf{R}_{ss}^{1/2}) \to \infty$ and the identification of the system is increasingly difficult, if not impossible.

6.8 Conclusions

In this chapter, we have explained the most important results of the Wiener theory in the context of system identification.

After discussing the four basic signal models, we derived the optimal Wiener filter for a SISO system and showed that this filter can be a very good approximation of the desired impulse response.

We discussed in details the condition number of the input signal correlation matrix. This matrix appears explicitly in the Wiener–Hopf equations and implicitly in all adaptive filters. A high condition number will perturb the accuracy of the solution of the Wiener–Hopf equations and will slow the rate of convergence of most adaptive algorithms. A fast, efficient algorithm to compute the conditional number was also developed.

We also discussed several important adaptive filters. In particular, the NLMS algorithm, which is extremely popular and useful in practice, was derived. Other emerging algorithms, such as the IPNLMS, were presented.

We generalized the Wiener principle to the MIMO system case. We showed that the MIMO Wiener–Hopf

equations can be decomposed into N independent MISO Wiener–Hopf equations. As a result, adaptive filters for SISO and MISO systems, with a reference signal, cover all possible cases. A deep analysis of the conditioning of the input signal covariance matrix was given, showing that identification is not always obvious and depends on the interchannel coherence between the input signals.

References

6.1 N. Wiener: *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (Wiley, New York 1949)

6.2 N. Wiener, E. Hopf: On a class of singular integral equations, Proc. Prussian Acad. Math.-Phys. Ser. (1931) p. 696

6.3 N. Levinson: The Wiener rms (root-mean-square) error criterion in filter design and prediction, J. Math. Phys. **25**, 261–278 (1947)

6.4 T. Kailath: A view of three decades of linear filtering theory, IEEE Trans. Inf. Theory **IT-20**, 146–181 (1974)

6.5 S. Haykin: *Adaptive Filter Theory*, 4th edn. (Prentice Hall, Upper Saddle River 2002)

6.6 J. Benesty, T. Gänsler: Computation of the condition number of a non-singular symmetric Toeplitz matrix with the Levinson–Durbin algorithm, IEEE Trans. Signal Process. **54**, 2362–2364 (2006)

6.7 J. Benesty, T. Gänsler: New insights into the RLS algorithm, EURASIP J. Appl. Signal Process. **2004**, 331–339 (2004)

6.8 G.H. Golub, C.F. Van Loan: *Matrix Computations* (The Johns Hopkins Univ. Press, Baltimore 1996)

6.9 J.M.B. Dias, J.M.N. Leitão: Efficient computation of $\mathrm{tr}\mathbf{TR}^{-1}$ for Toeplitz matrices, IEEE Signal Process. Let. **9**, 54–56 (2002)

6.10 B. Widrow: Adaptive filters. In: *Aspects of Network and System Theory*, ed. by R.E. Kalman, N. DeClaris (Holt Rinehart and Winston, New York 1970)

6.11 B. Widrow, M.E. Hoff Jr.: Adaptive switching circuits, IRE WESCON Conv. Rec. **4**, 96–104 (1960)

6.12 A. Feuer, E. Weinstein: Convergence analysis of LMS filters with uncorrelated Gaussian data, IEEE Trans. Acoust. Speech ASSP **33**, 222–230 (1985)

6.13 B. Widrow, S.D. Stearns: *Adaptive Signal Processing* (Prentice Hall, Englewood Cliffs 1985)

6.14 R.W. Harris, D.M. Chabries, F.A. Bishop: A variable step (VS) adaptive filter algorithm, IEEE Trans. Acoust. Speech ASSP **34**, 309–316 (1986)

6.15 R.H. Kwong, E.W. Johnston: A variable step size LMS algorithm, IEEE Trans. Signal Process. **40**, 1633–1642 (1992)

6.16 V.J. Mathews, Z. Xie: A stochastic gradient adaptive filter with gradient adaptive step size, IEEE Trans. Signal Process. **41**, 2075–2087 (1993)

6.17 J.B. Evans, P. Xue, B. Liu: Analysis and implementation of variable step size adaptive algorithms, IEEE Trans. Signal Process. **41**, 2517–2535 (1993)

6.18 T. Aboulnasr, K. Mayyas: A robust variable step-size LMS-type algorithm: analysis and simulations, IEEE Trans. Signal Process. **45**, 631–639 (1997)

6.19 D.I. Pazaitis, A.G. Constantinides: A novel kurtosis driven variable step-size adaptive algorithm, IEEE Trans. Signal Process. **47**, 864–872 (1999)

6.20 A. Mader, H. Puder, G.U. Schmidt: Step-size control for acoustic echo cancellation filters – An overview, Signal Process. **80**, 1697–1719 (2000)

6.21 H.-C. Shin, A.H. Sayed, W.-J. Song: Variable step-size NLMS and affine projection algorithms, IEEE Signal Process. Lett. **11**, 132–135 (2004)

6.22 D.R. Morgan, S.G. Kratzer: On a class of computationally efficient, rapidly converging, generalized NLMS algorithms, IEEE Signal Process. Lett. **3**, 245–247 (1996)

6.23 J. Benesty, H. Rey, L.R. Vega, S. Tressens: A nonparametric VSS-NLMS algorithm, IEEE Signal Process. Lett. **13**, 581–584 (2006), .

6.24 D.L. Duttweiler: Proportionate normalized least-mean-square adaptation in echo cancelers, IEEE Trans. Audio Speech **8**, 508–518 (2000)

6.25 J. Benesty, S.L. Gay: An improved PNLMS algorithm, Proc. IEEE ICASSP (2002) pp. 1881–1884

6.26 S.L. Gay: An efficient fast converging adaptive filter for network echo cancellation, Proc. Assilomar Conf. **1**, 394–398 (1998)

6.27 A. Gersho: Adaptive filtering with binary reinforcement, IEEE Trans. Inf. Theory **IT-30**, 191–199 (1984)

6.28 M.G. Bellanger: *Adaptive Digital Filters and Signal Analysis* (Marcel Dekker, New York 1987)

6.29 T. Claasen, W. Mecklenbrauker: Comparison of the convergence of two algorithms for adaptive FIR digital filters, IEEE Trans. Acoust. Speech ASSP **29**, 670–678 (1981)

6.30 N.J. Bershad: On the optimum data non-linearity in LMS adaptation, IEEE Trans. Acoust. Speech ASSP **34**, 69–76 (1986)

6.31 R. Gray: On the asymptotic eigenvalue distribution of Toeplitz matrices, IEEE Trans. Inform. Theory **IT-18**, 725–730 (1972)