4

LMS-BASED ALGORITHMS

4.1 INTRODUCTION

There are a number of algorithms for adaptive filters which are derived from the conventional LMS algorithm discussed in the previous chapter. The objective of the alternative LMS-based algorithms is either to reduce computational complexity or convergence time. In this chapter, several LMS-based algorithms are presented and analyzed, namely, the quantized-error algorithms [1]-[11], the frequency-domain (or transform-domain) LMS algorithm [12]-[14], the normalized LMS algorithm [15], the LMS-Newton algorithm [16]-[17], and the affine projection algorithm [19]-[25]. Several algorithms that are related to the main algorithms presented in this chapter are also briefly discussed.

The quantized-error algorithms reduce the computational complexity of the LMS algorithms by representing the error signal with short wordlength or by a simple power-of-two number.

The convergence speed in the LMS-Newton algorithm is independent of the eigenvalue spread of the input signal correlation matrix. This improvement is achieved by using an estimate of the inverse of the input signal correlation matrix, leading to a substantial increase in the computational complexity.

The normalized LMS algorithm utilizes a variable convergence factor that minimizes the instantaneous error. Such a convergence factor usually reduces the convergence time but increases the misadjustment.

In the frequency-domain algorithm, a transform is applied to the input signal in order to allow the reduction of the eigenvalue spread of the transformed signal correlation matrix as compared to the eigenvalue spread of the input signal correlation matrix. The LMS algorithm applied to the better conditioned transformed signal achieves faster convergence.

The affine projection algorithm reuses old data resulting in fast convergence when the input signal is highly correlated, leading to a family of algorithms that can trade-off computational complexity with convergence speed.

P.S.R. Diniz, *Adaptive Filtering*, DOI: 10.1007/978-0-387-68606-6_4,
© Springer Science+Business Media, LLC 2008

4.2 QUANTIZED-ERROR ALGORITHMS

The computational complexity of the LMS algorithm is mainly due to multiplications performed in the coefficient updating and in the calculation of the adaptive-filter output. In applications where the adaptive filters are required to operate in high speed, such as echo cancellation and channel equalization, it is important to minimize hardware complexity.

A first step to simplify the LMS algorithm is to apply quantization to the error signal, generating the quantized-error algorithm which updates the filter coefficients according to

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu Q[e(k)]\mathbf{x}(k) \tag{4.1}$$

where $Q[\cdot]$ represents a quantization operation. The quantization function is discrete valued, bounded, and nondecreasing. The type of quantization identifies the quantized-error algorithm.

If the convergence factor μ is a power-of-two number, the coefficient updating can be implemented with simple multiplications, basically consisting of bit shifts and additions. In a number of applications, such as the echo cancellation in full-duplex data transmission [2] and equalization of channels with binary data [3], the input signal $x(k)$ is a binary signal, i.e., assumes values $+1$ and -1. In this case, the adaptive filter can be implemented without any intricate multiplication.

The quantization of the error actually implies a modification in the objective function that is minimized, denoted by $F[e(k)]$. In a general gradient-type algorithm coefficient updating is performed by

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\frac{\partial F[e(k)]}{\partial \mathbf{w}(k)} = \mathbf{w}(k) - \mu\frac{\partial F[e(k)]}{\partial e(k)}\frac{\partial e(k)}{\partial \mathbf{w}(k)} \tag{4.2}$$

For a linear combiner the above equation can be rewritten as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\frac{\partial F[e(k)]}{\partial e(k)}\mathbf{x}(k) \tag{4.3}$$

Therefore, the objective function that is minimized in the quantized-error algorithms is such that

$$\frac{\partial F[e(k)]}{\partial e(k)} = 2Q[e(k)] \tag{4.4}$$

where $F[e(k)]$ is obtained by integrating $2Q[e(k)]$ with respect to $e(k)$. Note that the chain rule applied in equation (4.3) is not valid at the points of discontinuity of $Q[\cdot]$ where $F[e(k)]$ is not differentiable [6].

The performances of the quantized-error and LMS algorithms are obviously different. The analyses of some widely used quantized-error algorithms are presented in the following subsections.

Algorithm 4.1

Sign-Error Algorithm

```
Initialization
 $\mathbf{x}(0) = \mathbf{w}(0) = [0\,0 \ldots 0]^T$
Do for $k \geq 0$
  $e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k)$
  $\rho = \mathrm{sgn}[e(k)]$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu\rho\mathbf{x}(k)$
```

4.2.1 Sign-Error Algorithm

The simplest form for the quantization function is the sign (sgn) function defined by

$$\mathrm{sgn}[b] = \begin{cases} 1, & b > 0 \\ 0, & b = 0 \\ -1, & b < 0 \end{cases} \tag{4.5}$$

The sign-error algorithm utilizes the sign function as the error quantizer, where the coefficient vector updating is performed by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu\,\mathrm{sgn}[e(k)]\,\mathbf{x}(k) \tag{4.6}$$

Fig. 4.1 illustrates the realization of the sign-error algorithm for a delay line input $\mathbf{x}(k)$. If μ is a power-of-two number, one iteration of the sign-error algorithm requires $N+1$ multiplications for the error generation. The total number of additions is $2N+2$. The detailed description of the sign-error algorithm is shown in Algorithm 4.1. Obviously, the vectors $\mathbf{x}(0)$ and $\mathbf{w}(0)$ can be initialized in a different way from that described in the algorithm.

The objective function that is minimized by the sign-error algorithm is the modulus of the error multiplied by two, i.e.,

$$F[e(k)] = 2|e(k)| \tag{4.7}$$

Note that the factor two is included only to present the sign-error and LMS algorithms in a unified form. Obviously, in real implementation this factor can be merged with convergence factor μ.

Some of the properties related to the convergence behavior of the sign-error algorithm in a stationary environment are described, following the same procedure used in the previous chapter for the LMS algorithm.

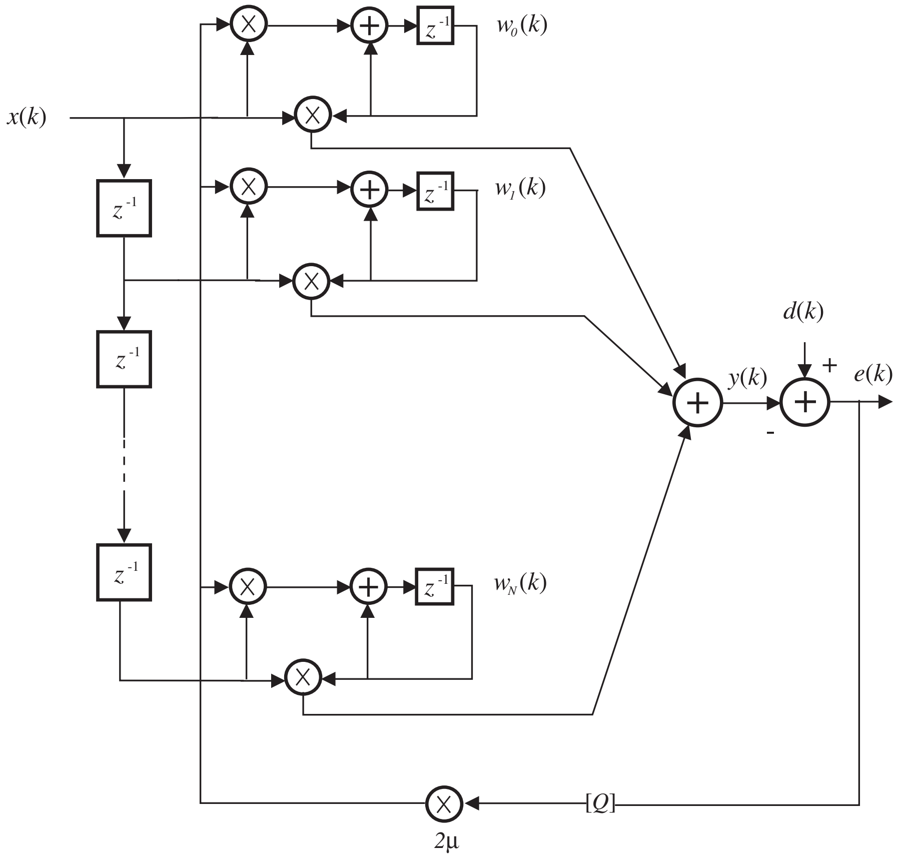


Figure 4.1 Sign-error adaptive FIR filter: $Q[e(k)] = sgn[e(k)]$.

4.2.1.1 *Steady-State Behavior of the Coefficient Vector*

The sign-error algorithm can be alternatively described by

$$\Delta\mathbf{w}(k+1) = \Delta\mathbf{w}(k) + 2\mu\,\mathrm{sgn}[e(k)]\,\mathbf{x}(k) \tag{4.8}$$

where $\Delta\mathbf{w}(k) = \mathbf{w}(k) - \mathbf{w}_o$. The expected value of the coefficient-error vector is then given by

$$E[\Delta\mathbf{w}(k+1)] = E[\Delta\mathbf{w}(k)] + 2\mu E\{\mathrm{sgn}[e(k)]\,\mathbf{x}(k)\} \tag{4.9}$$

The importance of the probability density function of the measurement noise $n(k)$ on the convergence of the sign-error algorithm is a noteworthy characteristic. This is due to the fact that $E\{\mathrm{sgn}[e(k)]\,\mathbf{x}(k)\} = E\{\mathrm{sgn}[-\Delta\mathbf{w}^T(k)\mathbf{x}(k) + n(k)]\mathbf{x}(k)\}$, where the result of the sign operation is highly dependent on the probability density function of $n(k)$. In [1], the authors present a convergence analysis of the output MSE, i.e., $E[e^2(k)]$, for different distributions of the additional noise, such as Gaussian, uniform, and binary distributions.

A closer examination of equation (4.8) indicates that even if the error signal becomes very small, the adaptive-filter coefficients will be continually updated due to the sign function applied to the error signal. Therefore, in a situation where the adaptive filter has a sufficient number of coefficients to model the desired signal, and there is no additional noise, $\Delta\mathbf{w}(k)$ will not converge to zero. In this case, $\mathbf{w}(k)$ will be convergent to a balloon centered at $\mathbf{w}_o$, when μ is appropriately chosen. The mean absolute value of $e(k)$ is also convergent to a balloon centered around zero, that means $|e(k)|$ remains smaller than the balloon radius r [6].

Recall that the desired signal without measurement noise is denoted as $d'(k)$. If it is considered that $d'(k)$ and the elements of $\mathbf{x}(k)$ are zero mean and jointly Gaussian and that the additional noise $n(k)$ is also zero mean, Gaussian, and independent of $\mathbf{x}(k)$ and $d'(k)$, the error signal will also be zero-mean Gaussian signal conditioned on $\Delta\mathbf{w}(k)$. In this case, using the results of the Price theorem described in [27] and in Papoulis [28], the following result is valid

$$E\{\text{sgn}[e(k)]\,\mathbf{x}(k)\} \approx \sqrt{\frac{2}{\pi\xi(k)}}E[\mathbf{x}(k)e(k)] \tag{4.10}$$

where $\xi(k)$ is the variance of $e(k)$ assuming the error has zero mean. The above approximation is valid for small values of μ. For large μ, $e(k)$ is dependent on $\Delta\mathbf{w}(k)$ and conditional expected value on $\Delta\mathbf{w}(k)$ should be used instead [3]-[5].

By applying equation (4.10) in equation (4.9) and by replacing $e(k)$ by $e_o(k) - \Delta\mathbf{w}^T(k)\mathbf{x}(k)$, it follows that

$$\begin{aligned} E[\Delta\mathbf{w}(k+1)] &= \left\{\mathbf{I} - 2\mu\sqrt{\frac{2}{\pi\xi(k)}}E[\mathbf{x}(k)\mathbf{x}^T(k)]\right\} E[\Delta\mathbf{w}(k)] \\ &\quad +2\mu\sqrt{\frac{2}{\pi\xi(k)}}\, E[e_o(k)\mathbf{x}(k)] \end{aligned} \tag{4.11}$$

From the orthogonality principle we know that $E[e_o(k)\mathbf{x}(k)] = \mathbf{0}$, so that the last element of the above equation is zero. Therefore,

$$E[\Delta\mathbf{w}(k+1)] = \left[\mathbf{I} - 2\mu\sqrt{\frac{2}{\pi\xi(k)}}\mathbf{R}\right] E[\Delta\mathbf{w}(k)] \tag{4.12}$$

Following the same steps for the analysis of $E[\Delta\mathbf{w}(k)]$ in the traditional LMS algorithm, it can be shown that the coefficients of the adaptive filter implemented with the sign-error algorithm converge in the mean if the convergence factor is chosen in the range

$$0 < \mu < \frac{1}{\lambda_{\max}}\sqrt{\frac{\pi\xi(k)}{2}} \tag{4.13}$$

where $\lambda_{\max}$ is the largest eigenvalue of $\mathbf{R}$. It should be mentioned that in case $\frac{\lambda_{\max}}{\lambda_{\min}}$ is large, the convergence speed of the coefficients depends on the value of $\lambda_{\min}$ which is related to the slowest mode in equation (4.12). This conclusion can be drawn by following the same steps of the convergence analysis of the LMS algorithm, where by applying a transformation to equation (4.12) we obtain an equation similar to equation (3.17).

A more practical range for μ, avoiding the use of eigenvalue, is given by

$$0 < \mu < \frac{1}{\mathrm{tr}[\mathbf{R}]}\sqrt{\frac{\pi\xi(k)}{2}} \tag{4.14}$$

Note that the upper bound for the value of μ requires the knowledge of the MSE, i.e., $\xi(k)$.

4.2.1.2 Coefficient-Error-Vector Covariance Matrix

The covariance of the coefficient-error vector defined as

$$\mathrm{cov}[\Delta\mathbf{w}(k)] = E\left[(\mathbf{w}(k)-\mathbf{w}_o)(\mathbf{w}(k)-\mathbf{w}_o)^T\right] \tag{4.15}$$

is calculated by replacing equation (4.8) in equation (4.15) following the same steps used in the LMS algorithm. The resulting difference equation for $\mathrm{cov}[\Delta\mathbf{w}(k)]$ is given by

$$\begin{aligned}\mathrm{cov}[\Delta\mathbf{w}(k+1)] &= \mathrm{cov}[\Delta\mathbf{w}(k)] + 2\mu E\{\mathrm{sgn}[e(k)]\mathbf{x}(k)\Delta\mathbf{w}^T(k)\}\\ &\quad +2\mu E\{\mathrm{sgn}[e(k)]\Delta\mathbf{w}(k)\mathbf{x}^T(k)\} + 4\mu^2\mathbf{R}\end{aligned} \tag{4.16}$$

The first term with expected value operation in the above equation can be expressed as

$$\begin{aligned}E\{\mathrm{sgn}[e(k)]\mathbf{x}(k)\Delta\mathbf{w}^T(k)\} &= E\{\mathrm{sgn}[e_o(k)-\Delta\mathbf{w}^T(k)\mathbf{x}(k)]\mathbf{x}(k)\Delta\mathbf{w}^T(k)\}\\ &= E\{E[\mathrm{sgn}[e_o(k)-\Delta\mathbf{w}^T(k)\mathbf{x}(k)]\mathbf{x}(k)|\Delta\mathbf{w}(k)]\Delta\mathbf{w}^T(k)\}\end{aligned}$$

where $E[a|\Delta\mathbf{w}(k)]$ is the expected value of a conditioned on the value of $\Delta\mathbf{w}(k)$. In the first equality, $e(k)$ was replaced by the relation $d(k) - \mathbf{w}^T(k)\mathbf{x}(k) - \mathbf{w}_o^T\mathbf{x}(k) + \mathbf{w}_o^T\mathbf{x}(k) = e_o(k) - \Delta\mathbf{w}^T(k)\mathbf{x}(k)$. In the second equality, the concept of conditioned expected value was applied.

Using the Price theorem and considering that the minimum output error $e_o(k)$ is zero-mean and uncorrelated with $\mathbf{x}(k)$, the following approximations result

$$\begin{aligned}&E\{E[\mathrm{sgn}[e_o(k)-\Delta\mathbf{w}^T(k)\mathbf{x}(k)]\mathbf{x}(k)|\Delta\mathbf{w}(k)]\Delta\mathbf{w}^T(k)\}\\ \approx\;& E\left\{\sqrt{\frac{2}{\pi\xi(k)}}E[e_o(k)\mathbf{x}(k)-\mathbf{x}(k)\mathbf{x}^T(k)\Delta\mathbf{w}(k)|\Delta\mathbf{w}(k)]\Delta\mathbf{w}^T(k)\right\}\\ \approx\;& -E\left\{\sqrt{\frac{2}{\pi\xi(k)}}\mathbf{R}\Delta\mathbf{w}(k)\Delta\mathbf{w}^T(k)\right\}\\ =\;& -\sqrt{\frac{2}{\pi\xi(k)}}\mathbf{R}\mathrm{cov}[\Delta\mathbf{w}(k)]\end{aligned} \tag{4.17}$$

Following similar steps to derive the above equation, the second term with the expected value operation in equation (4.16) can be approximated as

$$E\{\mathrm{sgn}[e(k)]\Delta\mathbf{w}(k)\mathbf{x}^T(k)\} \approx -\sqrt{\frac{2}{\pi\xi(k)}}\mathrm{cov}[\Delta\mathbf{w}(k)]\mathbf{R} \tag{4.18}$$

Substituting equations (4.17) and (4.18) in equation (4.16), we can calculate the vector $\mathbf{v}'(k)$ consisting of diagonal elements of $\text{cov}[\Delta\mathbf{w}'(k)]$, using the same steps employed in the LMS case (see equation (3.26)). The resulting dynamic equation for $\mathbf{v}'(k)$ is given by

$$\mathbf{v}'(k+1) = \left(\mathbf{I} - 4\mu\sqrt{\frac{2}{\pi\xi(k)}}\,\mathbf{\Lambda}\right)\,\mathbf{v}'(k) + 4\mu^2\boldsymbol{\lambda} \tag{4.19}$$

The value of μ must be chosen in a range that guarantees the convergence of $\mathbf{v}'(k)$, which is given by

$$0 < \mu < \frac{1}{2\lambda_{\max}}\sqrt{\frac{\pi\xi(k)}{2}} \tag{4.20}$$

A more severe and practical range for μ is

$$0 < \mu < \frac{1}{2\text{tr}[\mathbf{R}]}\sqrt{\frac{\pi\xi(k)}{2}} \tag{4.21}$$

For $k \rightarrow \infty$ each element of $\mathbf{v}'(k)$ tends to

$$v_i(\infty) = \mu\sqrt{\frac{\pi\xi(\infty)}{2}} \tag{4.22}$$

4.2.1.3 *Excess Mean-Square Error and Misadjustment*

The excess MSE can be expressed as a function of the elements of $\mathbf{v}'(k)$ by

$$\Delta\xi(k) = \sum_{i=0}^{N}\lambda_i v_i(k) = \boldsymbol{\lambda}^T\mathbf{v}'(k) \tag{4.23}$$

Substituting equation (4.22) in equation (4.23) yields

$$\begin{aligned}\xi_{\text{exc}} &= \mu\sum_{i=0}^{N}\lambda_i\sqrt{\frac{\pi\xi(k)}{2}}, k \rightarrow \infty \\ &= \mu\sum_{i=0}^{N}\lambda_i\sqrt{\pi\,\frac{\xi_{\min}+\xi_{\text{exc}}}{2}}\end{aligned} \tag{4.24}$$

since $\lim_{k\rightarrow\infty}\xi(k) = \xi_{\min} + \xi_{\text{exc}}$. Therefore,

$$\xi_{\text{exc}}^2 = \mu^2\left(\sum_{i=0}^{N}\lambda_i\right)^2\left(\frac{\pi\xi_{\min}}{2} + \frac{\pi\xi_{\text{exc}}}{2}\right) \tag{4.25}$$

There are two solutions for ξ_{exc}^2 in the above equation, where only the positive one is valid. The meaningful solution for ξ_{exc}, when μ is small, is approximately given by

$$\begin{aligned}\xi_{\text{exc}} &\approx \mu\sqrt{\frac{\pi\xi_{\min}}{2}}\sum_{i=0}^{N}\lambda_i \\ &= \mu\sqrt{\frac{\pi\xi_{\min}}{2}}\,\text{tr}[\mathbf{R}]\end{aligned} \tag{4.26}$$

By comparing the excess MSE predicted by the above equation with the corresponding equation (3.49) for the LMS algorithm, it can be concluded that both can generate the same excess MSE if μ in the sign-error algorithm is chosen such that

$$\mu = \mu_{\text{LMS}} \sqrt{\frac{2}{\pi} \xi_{\min}^{-1}} \tag{4.27}$$

The misadjustment in the sign-error algorithm is

$$M = \mu \sqrt{\frac{\pi}{2\xi_{\min}}}\ \text{tr}[\mathbf{R}] \tag{4.28}$$

Equation (4.26) would leave the impression that if there is no additional noise and there are sufficient parameters in the adaptive filter, the output MSE would converge to zero. However, when $\xi(k)$ becomes small, $||E[\Delta\mathbf{w}(k+1)]||$ in equation (4.11) can increase, since the condition of equation (4.13) will not be satisfied. This is the situation where the parameters reach the convergence balloon. In this case, from equation (4.8) we can conclude that

$$||\Delta\mathbf{w}(k+1)||^2 - ||\Delta\mathbf{w}(k)||^2 = -4\mu\ \text{sgn}[e(k)]\ e(k) + 4\mu^2||\mathbf{x}(k)||^2 \tag{4.29}$$

from where it is possible to show that a decrease in the norm of $\Delta\mathbf{w}(k)$ is obtained only when

$$|e(k)| > \mu||\mathbf{x}(k)||^2 \tag{4.30}$$

For no additional noise, first transpose the vectors in equation (4.8) and postmultiply each side by $\mathbf{x}(k)$. Next, squaring the resulting equation and applying the expected value operation on each side, the obtained result is

$$E[e^2(k+1)] = E[e^2(k)] - 4\mu E[|e(k)|\ ||\mathbf{x}(k)||^2] + 4\mu^2 E[||\mathbf{x}(k)||^4] \tag{4.31}$$

After convergence $E[e^2(k+1)] \approx E[e^2(k)]$. Also, considering that

$$E[|e(k)|\ ||\mathbf{x}(k)||^2] \approx E[|e(k)|]E[||\mathbf{x}(k)||^2]$$

and

$$\frac{E[||\mathbf{x}(k)||^4]}{E[||\mathbf{x}(k)||^2]} \approx E[||\mathbf{x}(k)||^2]$$

we conclude that

$$E[|e(k)|] \approx \mu E[||\mathbf{x}(k)||^2], k \to \infty \tag{4.32}$$

For zero-mean Gaussian $e(k)$, the following approximation is valid

$$E[|e(k)|] \approx \sqrt{\frac{2}{\pi}}\sigma_e(k), k \to \infty \tag{4.33}$$

therefore, the expected variance of $e(k)$ is

$$\sigma_e^2(k) \approx \frac{\pi}{2}\mu^2\ \text{tr}^2[\mathbf{R}], k \to \infty \tag{4.34}$$

where we used the relation $\text{tr}[\mathbf{R}] = E[||\mathbf{x}(k)||^2]$. This relation gives an estimate of the variance of the output error when no additional noise exists. As can be noted, unlike the LMS algorithm, there is an excess MSE in the sign-error algorithm caused by the nonlinear device, even when $\sigma_n^2 = 0$.

If $n(k)$ has frequently large absolute values as compared to $-\Delta\mathbf{w}^T(k)\mathbf{x}(k)$, then for most iterations $\text{sgn}[e(k)] = \text{sgn}[n(k)]$. As a result, the sign-error algorithm is fully controlled by the additional noise. In this case, the algorithm does not converge.

4.2.1.4 Transient Behavior

The ratios r_{w_i} of the geometric decaying convergence curves of the coefficients in the sign-error algorithm can be derived from equation (4.12) by employing an identical analysis of the transient behavior for the LMS algorithm. The ratios are given by

$$r_{w_i} = \left(1 - 2\mu\sqrt{\frac{2}{\pi\xi(k)}}\lambda_i\right) \tag{4.35}$$

for $i = 0, 1, \ldots, N$. If μ is chosen as suggested in equation (4.27), in order to reach the same excess MSE of the LMS algorithm, then

$$r_{w_i} = \left(1 - \frac{4}{\pi}\mu_{\text{LMS}}\sqrt{\frac{\xi_{\min}}{\xi(k)}}\,\lambda_i\right) \tag{4.36}$$

By recalling that r_{w_i} for the LMS algorithm is $(1 - 2\mu_{\text{LMS}}\lambda_i)$, since $\frac{2}{\pi}\sqrt{\frac{\xi_{\min}}{\xi(k)}} < 1$, it is concluded that the sign-error algorithm is slower than the LMS for the same excess MSE.

Example 4.1

Suppose in an adaptive-filtering environment that the input signal consists of

$$x(k) = \mathrm{e}^{\jmath\omega_0 k} + n(k)$$

and that the desired signal is given by

$$d(k) = \mathrm{e}^{\jmath\omega_0(k-1)}$$

where $n(k)$ is a uniformly distributed white noise with variance $\sigma_n^2 = 0.1$ and $\omega_0 = \frac{2\pi}{M}$. In this case $M = 8$.

Compute the input signal correlation matrix for a first-order adaptive filter. Calculate the value of $\mu_{\max}$ for the sign-error algorithm.

Solution:

The input signal correlation matrix for this example can be calculated as shown below:

$$\mathbf{R} = \begin{bmatrix} 1+\sigma_n^2 & \mathrm{e}^{\jmath\omega_0} \\ \mathrm{e}^{-\jmath\omega_0} & 1+\sigma_n^2 \end{bmatrix}$$

Since in this case $\text{tr}[\mathbf{R}] = 2.2$ and $\xi_{\min} = 0.1$, we have

$$\xi_{\text{exc}} \approx \mu\sqrt{\frac{\pi\xi_{\min}}{2}}\,\text{tr}[\mathbf{R}] = 0.87\mu$$

The range of values of the convergence factor is given by

$$0 < \mu < \frac{1}{2\text{tr}[\mathbf{R}]}\sqrt{\frac{\pi(\xi_{\min} + \xi_{\text{exc}})}{2}}$$

From the above expression, it is straightforward to calculate the upper bound for the convergence factor that is given by

$$\mu_{\max} \approx 0.132$$

□

4.2.2 Dual-Sign Algorithm

The dual-sign algorithm attempts to perform large corrections to the coefficient vector when the modulus of the error signal is larger than a prescribed level. The basic motivation to use the dual-sign algorithm is to avoid the slow convergence inherent to the sign-error algorithm that is caused by replacing $e(k)$ by $\text{sgn}[e(k)]$ when $|e(k)|$ is large.

The quantization function for the dual-sign algorithm is given by

$$\text{ds}[a] = \begin{cases} \gamma\,\text{sgn}[a], & |a| > \rho \\ \text{sgn}[a], & |a| \leq \rho \end{cases} \tag{4.37}$$

where $\gamma > 1$ is a power of two. The dual-sign algorithm utilizes the function above described as the error quantizer, and the coefficient updating is performed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu\,\text{ds}[e(k)]\mathbf{x}(k) \tag{4.38}$$

The objective function that is minimized by the dual-sign algorithm is given by

$$F[e(k)] = \begin{cases} 2\gamma|e(k)| - 2\rho(\gamma - 1), & |e(k)| > \rho \\ 2|e(k)|, & |e(k)| \leq \rho \end{cases} \tag{4.39}$$

where the constant $2\rho(\gamma-1)$ was included in the objective function to make it continuous. Obviously the gradient of $F[e(k)]$ with respect to the filter coefficients is $2\mu\,\text{ds}[e(k)]\mathbf{x}(k)$ except at points where $\text{ds}[e(k)]$ is nondifferentiable [6].

The same analysis procedure used for the sign-error algorithm can be applied to the dual-sign algorithm except for the fact that the quantization function is now different. The alternative quantization leads to particular expectations of nonlinear functions whose solutions are not presented here. The interested reader should refer to the work of Mathews [7]. The choice of γ and ρ determine the convergence behavior of the dual-sign algorithm [7], typically, a large γ tends to increase both convergence speed and excess MSE. A large ρ tends to reduce both the convergence speed and the excess MSE. If $\lim_{k\rightarrow\infty}\xi(k) \ll \rho^2$, the excess MSE of the dual-sign algorithm is approximately equal to the one given by equation (4.26) for the sign-error algorithm [7], since in this case $|e(k)|$ is usually much smaller than ρ. For a given MSE in steady state, the dual-sign algorithm is expected to converge faster than the sign-error algorithm.

4.2.3 Power-of-Two Error Algorithm

The power-of-two error algorithm applies to the error signal a quantization defined by

$$\text{pe}[b] = \begin{cases} \text{sgn}[b], & |b| \geq 1 \\ 2^{\text{floor}[log_2|b|]}\,\text{sgn}[b], & 2^{-b_d+1} \leq |b| < 1 \\ \tau\text{sgn}[b], & |b| < 2^{-b_d+1} \end{cases} \tag{4.40}$$

where floor$[\cdot]$ indicates integer smaller than $[\cdot]$, b_d is the data wordlength excluding the sign bit, and τ is usually 0 or 2^{-b_d}.

The coefficient updating for the power-of-two error algorithm is given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu\ \text{pe}[e(k)]\mathbf{x}(k) \tag{4.41}$$

For $\tau = 2^{-b_d}$, the additional noise and the convergence factor can be arbitrarily small and the algorithm will not stop updating. For $\tau = 0$, when $|e(k)| < 2^{-b_d+1}$ the algorithm reaches the so-called *dead zone*, where the algorithm stops updating if $|e(k)|$ is smaller than 2^{-b_d+1} most of the time [4], [8].

A simplified and somewhat accurate analysis of this algorithm can be performed by approximating the function $\text{pe}[e(k)]$ by a straight line passing through the center of each quantization step. In this case, the quantizer characteristics can be approximated by $\text{pe}[e(k)] \approx \frac{2}{3}e(k)$ as illustrated in Fig. 4.2. Using this approximation, the algorithm analysis can be performed exactly in the same way as the LMS algorithm. The results for the power-of-two error algorithm can be obtained from the results for the LMS algorithm, by replacing μ by $\frac{2}{3}\mu$. It should be mentioned that such results are only approximate, and more accurate ones can be found in [8].

4.2.4 Sign-Data Algorithm

The algorithms discussed in this subsection cannot be considered as quantized error algorithms, but since they were proposed with similar motivation we decided to introduce them here. An alternative way to simplify the computational burden of the LMS algorithm is to apply quantization to the data vector $\mathbf{x}(k)$. One possible quantization scheme is to apply the sign function to the input signals, giving rise to the sign-data algorithm whose coefficient updating is performed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu e(k)\ \text{sgn}[\mathbf{x}(k)] \tag{4.42}$$

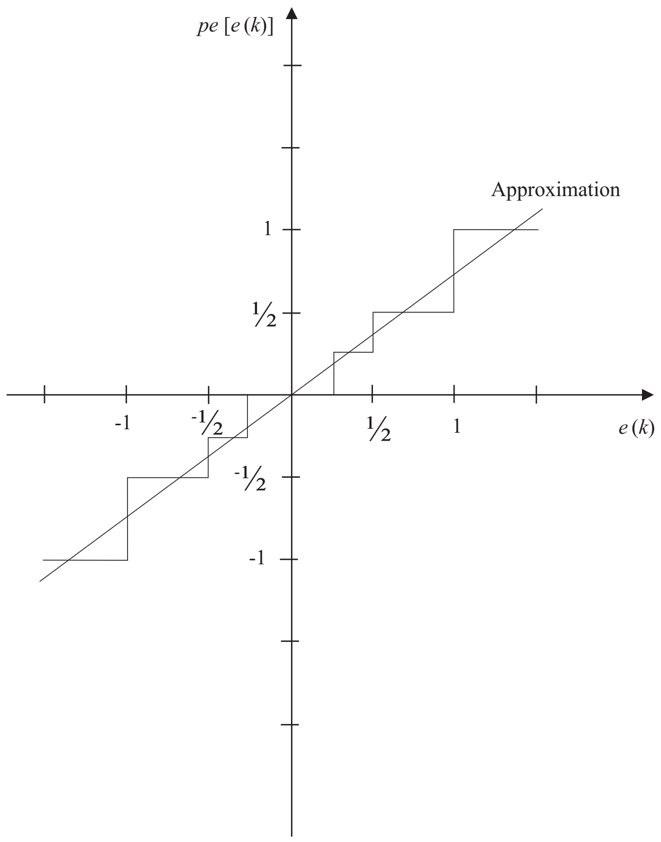


Figure 4.2 Transfer characteristic of a quantizer with 3 bits and $\tau = 0$.

where the sign operation is applied to each element of the input vector.

The quantization of the data vector can lead to a decrease in the convergence speed, and possible divergence. In the LMS algorithm, the average gradient direction follows the true gradient direction (or steepest-descent direction), whereas in the sign-data algorithm only a discrete set of directions can be followed. The limitation in the gradient direction followed by the sign-data algorithm may cause updates that result in frequent increase in the squared error, leading to instability. Therefore, it is relatively easy to find inputs that would lead to the convergence of the LMS algorithm and to the divergence of the sign-data algorithm [6], [9]. It should be mentioned, however, that the sign-data algorithm is stable for Gaussian inputs, and, as such, has been found useful in certain applications.

Another related algorithm is the sign-sign algorithm that has very simple implementation. The coefficient updating in this case is given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu \,\mathrm{sgn}[e(k)]\, \mathrm{sgn}[\mathbf{x}(k)] \tag{4.43}$$

The sign-sign algorithm also presents the limitations related to the quantized-data algorithm.

4.3 THE LMS-NEWTON ALGORITHM

In this section, the LMS-Newton algorithm incorporating estimates of the second-order statistics of the environment signals is introduced. The objective of the algorithm is to avoid the slow convergence of the LMS algorithm when the input signal is highly correlated. The improvement in the convergence rate is achieved at the expense of an increased computational complexity.

Nonrecursive realization of the adaptive filter leads to an MSE surface that is a quadratic function of the filter coefficients. For the direct-form FIR structure, the MSE can be described by

$$\xi(k+1) \;=\; \xi(k) + \mathbf{g_w}^T(k)\,[\mathbf{w}(k+1)-\mathbf{w}(k)] + [\mathbf{w}(k+1)-\mathbf{w}(k)]^T\,\mathbf{R}\,[\mathbf{w}(k+1)-\mathbf{w}(k)] \tag{4.44}$$

$\xi(k)$ represents the MSE when the adaptive-filter coefficients are fixed at $\mathbf{w}(k)$ and $\mathbf{g_w}(k) = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}(k)$ is the gradient vector of the MSE surface as related to the filter coefficients at $\mathbf{w}(k)$. The MSE is minimized at the instant $k+1$ if

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \frac{1}{2}\mathbf{R}^{-1}\mathbf{g_w}(k) \tag{4.45}$$

This equation is the updating formula of the Newton method. Note that in the ideal case, where matrix $\mathbf{R}$ and gradient vector $\mathbf{g_w}(k)$ are known precisely, $\mathbf{w}(k+1) = \mathbf{R}^{-1}\mathbf{p} = \mathbf{w}_o$. Therefore, the Newton method converges to the optimal solution in a single iteration, as expected for a quadratic objective function.

In practice, only estimates of the autocorrelation matrix $\mathbf{R}$ and of the gradient vector are available. These estimates can be applied to the Newton updating formula in order to derive a Newton-like method given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\hat{\mathbf{R}}^{-1}(k)\hat{\mathbf{g}}_{\mathbf{w}}(k) \tag{4.46}$$

The convergence factor μ is introduced so that the algorithm can be protected from divergence, originated by the use of noisy estimates of $\mathbf{R}$ and $\mathbf{g_w}(k)$.

For stationary input signals, an unbiased estimate of $\mathbf{R}$ is

$$\begin{aligned}\hat{\mathbf{R}}(k) &= \frac{1}{k+1}\sum_{i=0}^{k}\mathbf{x}(i)\mathbf{x}^T(i)\\ &= \frac{k}{k+1}\hat{\mathbf{R}}(k-1) + \frac{1}{k+1}\mathbf{x}(k)\mathbf{x}^T(k)\end{aligned} \tag{4.47}$$

since

$$\begin{aligned}E[\hat{\mathbf{R}}(k)] &= \frac{1}{k+1}\sum_{i=0}^{k}E[\mathbf{x}(i)\mathbf{x}^T(i)]\\ &= \mathbf{R}\end{aligned} \tag{4.48}$$

However, this is not a practical estimate for $\mathbf{R}$, since for large k any change on the input signal statistics would be disregarded due to the infinite memory of the estimation algorithm.

Another form to estimate the autocorrelation matrix can be generated by employing a weighted summation as follows:

$$\begin{aligned}\hat{\mathbf{R}}(k) &= \alpha\mathbf{x}(k)\mathbf{x}^T(k) + (1-\alpha)\hat{\mathbf{R}}(k-1)\\ &= \alpha\mathbf{x}(k)\mathbf{x}^T(k) + \alpha\sum_{i=0}^{k-1}(1-\alpha)^{k-i}\mathbf{x}(i)\mathbf{x}^T(i)\end{aligned} \tag{4.49}$$

where in practice, α is a small factor chosen in the range $0 < \alpha \leq 0.1$. This range of values of α allows a good balance between the present and past input signal information. By taking the expected value on both sides of the above equation and assuming that $k \rightarrow \infty$, it follows that

$$\begin{aligned}E[\hat{\mathbf{R}}(k)] &= \alpha\sum_{i=0}^{k}(1-\alpha)^{k-i}E[\mathbf{x}(i)\mathbf{x}^T(i)]\\ &= \mathbf{R} \qquad k \rightarrow \infty\end{aligned} \tag{4.50}$$

Therefore, the estimate of $\mathbf{R}$ of equation (4.49) is unbiased.

In order to avoid inverting $\hat{\mathbf{R}}(k)$, which is required by the Newton-like algorithm, we can use the so-called matrix inversion lemma given by

$$[\mathbf{A}+\mathbf{BCD}]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}[\mathbf{DA}^{-1}\mathbf{B}+\mathbf{C}^{-1}]^{-1}\mathbf{DA}^{-1} \tag{4.51}$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are matrices of appropriate dimensions, and $\mathbf{A}$ and $\mathbf{C}$ are nonsingular. The above relation can be proved by simply showing that the result of premultiplying the expression on the right-hand side by $\mathbf{A}+\mathbf{BCD}$ is the identity matrix (see problem 21). If we choose $\mathbf{A} = (1-\alpha)$ $\hat{\mathbf{R}}(k-1)$, $\mathbf{B} = \mathbf{D}^T = \mathbf{x}(k)$, and $\mathbf{C} = \alpha$, it can be shown that

$$\hat{\mathbf{R}}^{-1}(k) = \frac{1}{1-\alpha}\left[\hat{\mathbf{R}}^{-1}(k-1) - \frac{\hat{\mathbf{R}}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\hat{\mathbf{R}}^{-1}(k-1)}{\frac{1-\alpha}{\alpha} + \mathbf{x}^T(k)\hat{\mathbf{R}}^{-1}(k-1)\mathbf{x}(k)}\right] \tag{4.52}$$

The resulting equation to calculate $\hat{\mathbf{R}}^{-1}(k)$ is less complex to update (of order N^2 multiplications) than the direct inversion of $\hat{\mathbf{R}}(k)$ at every iteration (of order N^3 multiplications).

If the estimate for the gradient vector used in the LMS algorithm is applied in equation (4.46), the following coefficient updating formula for the LMS-Newton algorithm results

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\,\mu\,e(k)\,\hat{\mathbf{R}}^{-1}(k)\mathbf{x}(k) \tag{4.53}$$

The complete LMS-Newton algorithm is outlined in Algorithm 4.2. It should be noticed that alternative initialization procedures to the one presented in Algorithm 4.2 are possible.

As previously mentioned, the LMS gradient direction has the tendency to approach the ideal gradient direction. Similarly, the vector resulting from the multiplication of $\hat{\mathbf{R}}^{-1}(k)$ to the LMS gradient

Algorithm 4.2

LMS-Newton Algorithm

```
Initialization
  $\hat{\mathbf{R}}^{-1}(-1) = \delta\mathbf{I}$   ($\delta$ a small positive constant)
  $\mathbf{w}(0) = \mathbf{x}(-1) = [0\,0 \ldots 0]^T$
Do for $k \geq 0$
  $e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k)$
  $\hat{\mathbf{R}}^{-1}(k) = \frac{1}{1-\alpha}\left[\hat{\mathbf{R}}^{-1}(k-1) - \frac{\hat{\mathbf{R}}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\hat{\mathbf{R}}^{-1}(k-1)}{\frac{1-\alpha}{\alpha}+\mathbf{x}^T(k)\hat{\mathbf{R}}^{-1}(k-1)\mathbf{x}(k)}\right]$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + 2\,\mu\, e(k)\,\hat{\mathbf{R}}^{-1}(k)\mathbf{x}(k)$
```

direction tends to approach the Newton direction. Therefore, we can conclude that the LMS-Newton algorithm converges in a more straightforward path to the minimum of the MSE surface. It can also be shown that the convergence characteristics of the algorithm is independent of the eigenvalue spread of **R**.

The LMS-Newton algorithm is mathematically identical to the recursive least-squares (RLS) algorithm if the forgetting factor (λ) in the latter is chosen such that $2\mu = \alpha = 1 - \lambda$ [39]. Since a complete discussion of the RLS algorithm is given later, no further discussion of the LMS-Newton algorithm is included here.

4.4 THE NORMALIZED LMS ALGORITHM

If one wishes to increase the convergence speed of the LMS algorithm without using estimates of the input signal correlation matrix, a variable convergence factor is a natural solution. The normalized LMS algorithm usually converges faster than the LMS algorithm, since it utilizes a variable convergence factor aiming at the minimization of the instantaneous output error.

The updating equation of the LMS algorithm can employ a variable convergence factor μ_k in order to improve the convergence rate. In this case, the updating formula is expressed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu_k e(k)\mathbf{x}(k) = \mathbf{w}(k) + \Delta\tilde{\mathbf{w}}(k) \qquad (4.54)$$

where μ_k must be chosen with the objective of achieving a faster convergence. A possible strategy is to reduce the instantaneous squared error as much as possible. The motivation behind this strategy is that the instantaneous squared error is a good and simple estimate of the MSE.

The instantaneous squared error is given by

$$e^2(k) = d^2(k) + \mathbf{w}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{w}(k) - 2d(k)\mathbf{w}^T(k)\mathbf{x}(k) \tag{4.55}$$

If a change given by $\tilde{\mathbf{w}}(k) = \mathbf{w}(k) + \Delta\tilde{\mathbf{w}}(k)$ is performed in the weight vector, the corresponding squared error can be shown to be

$$\tilde{e}^2(k) \;=\; e^2(k) + 2\Delta\tilde{\mathbf{w}}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{w}(k) + \Delta\tilde{\mathbf{w}}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\Delta\tilde{\mathbf{w}}(k) - 2d(k)\Delta\tilde{\mathbf{w}}^T(k)\mathbf{x}(k) \tag{4.56}$$

It then follows that

$$\begin{aligned}\Delta e^2(k) &\overset{\triangle}{=} \tilde{e}^2(k) - e^2(k)\\ &= -2\Delta\tilde{\mathbf{w}}^T(k)\mathbf{x}(k)e(k) + \Delta\tilde{\mathbf{w}}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\Delta\tilde{\mathbf{w}}(k)\end{aligned} \tag{4.57}$$

In order to increase the convergence rate, the objective is to make $\Delta e^2(k)$ negative and minimum by appropriately choosing μ_k.

By replacing $\Delta\tilde{\mathbf{w}}(k) = 2\mu_k e(k)\mathbf{x}(k)$ in equation (4.57), it follows that

$$\Delta e^2(k) = -4\mu_k e^2(k)\mathbf{x}^T(k)\mathbf{x}(k) + 4\mu_k^2 e^2(k)[\mathbf{x}^T(k)\mathbf{x}(k)]^2 \tag{4.58}$$

The value of μ_k such that $\frac{\partial \Delta e^2(k)}{\partial \mu_k} = 0$ is given by

$$\mu_k = \frac{1}{2\mathbf{x}^T(k)\mathbf{x}(k)} \tag{4.59}$$

This value of μ_k leads to a negative value of $\Delta e^2(k)$, and, therefore, it corresponds to a minimum point of $\Delta e^2(k)$.

Using this variable convergence factor, the updating equation for the LMS algorithm is then given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{e(k)\mathbf{x}(k)}{\mathbf{x}^T(k)\mathbf{x}(k)} \tag{4.60}$$

Usually a fixed convergence factor μ_n is introduced in the updating formula in order to control the misadjustment, since all the derivations are based on instantaneous values of the squared errors and not on the MSE. Also a parameter γ should be included, in order to avoid large step sizes when $\mathbf{x}^T(k)\mathbf{x}(k)$ becomes small. The coefficient updating equation is then given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\mu_n}{\gamma + \mathbf{x}^T(k)\mathbf{x}(k)}\, e(k)\, \mathbf{x}(k) \tag{4.61}$$

The resulting algorithm is called the normalized LMS algorithm, and is summarized in Algorithm 4.3.

Algorithm 4.3

The Normalized LMS Algorithm

```
Initialization
  $\mathbf{x}(0) = \hat{\mathbf{w}}(0) = [0\,0 \ldots 0]^T$
  choose $\mu_n$ in the range $0 < \mu_n \leq 2$
  $\gamma =$ small constant
Do for $k \geq 0$
  $e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k)$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\mu_n}{\gamma + \mathbf{x}^T(k)\mathbf{x}(k)}\, e(k)\, \mathbf{x}(k)$
```

The range of values of μ_n to guarantee stability can be derived by first considering that $E[\mathbf{x}^T(k)\mathbf{x}(k)] = \mathrm{tr}[\mathbf{R}]$ and that

$$E\left[\frac{e(k)\mathbf{x}(k)}{\mathbf{x}^T(k)\mathbf{x}(k)}\right] \approx \frac{E[e(k)\mathbf{x}(k)]}{E[\mathbf{x}^T(k)\mathbf{x}(k)]}$$

Next, consider that the average value of the convergence factor actually applied to the LMS direction $2e(k)\mathbf{x}(k)$ is $\frac{\mu_n}{2\,\mathrm{tr}[\mathbf{R}]}$. Finally, by comparing the updating formula of the standard LMS algorithm with that of the normalized LMS algorithm, the desired upper bound result follows:

$$0 < \mu = \frac{\mu_n}{2\,\mathrm{tr}[\mathbf{R}]} < \frac{1}{\mathrm{tr}[\mathbf{R}]} \tag{4.62}$$

or $0 < \mu_n < 2$.

4.5 THE TRANSFORM-DOMAIN LMS ALGORITHM

The transform-domain LMS algorithm is another technique to increase the convergence speed of the LMS algorithm when the input signal is highly correlated. The basic idea behind this methodology is to modify the input signal to be applied to the adaptive filter such that the conditioning number of the corresponding correlation matrix is improved.

In the transform-domain LMS algorithm, the input signal vector $\mathbf{x}(k)$ is transformed in a more convenient vector $\mathbf{s}(k)$, by applying an orthonormal (or unitary) transform [10]-[12], i.e.,

$$\mathbf{s}(k) = \mathbf{T}\mathbf{x}(k) \tag{4.63}$$

where $\mathbf{T}\mathbf{T}^T = \mathbf{I}$. The MSE surface related to the direct-form implementation of the FIR adaptive filter can be described by

$$\xi(k) = \xi_{\min} + \Delta\mathbf{w}^T(k)\mathbf{R}\Delta\mathbf{w}(k) \tag{4.64}$$

where $\Delta\mathbf{w}(k) = \mathbf{w}(k) - \mathbf{w}_o$. In the transform-domain case, the MSE surface becomes

$$\begin{aligned}\xi(k) &= \xi_{\min} + \Delta\hat{\mathbf{w}}^T(k)E[\mathbf{s}(k)\mathbf{s}^T(k)]\Delta\hat{\mathbf{w}}(k)\\ &= \xi_{\min} + \Delta\hat{\mathbf{w}}^T(k)\mathbf{TRT}^T\Delta\hat{\mathbf{w}}(k) \qquad (4.65)\end{aligned}$$

where $\hat{\mathbf{w}}(k)$ represents the adaptive coefficients of the transform-domain filter. Fig. 4.3 depicts the transform-domain adaptive filter.

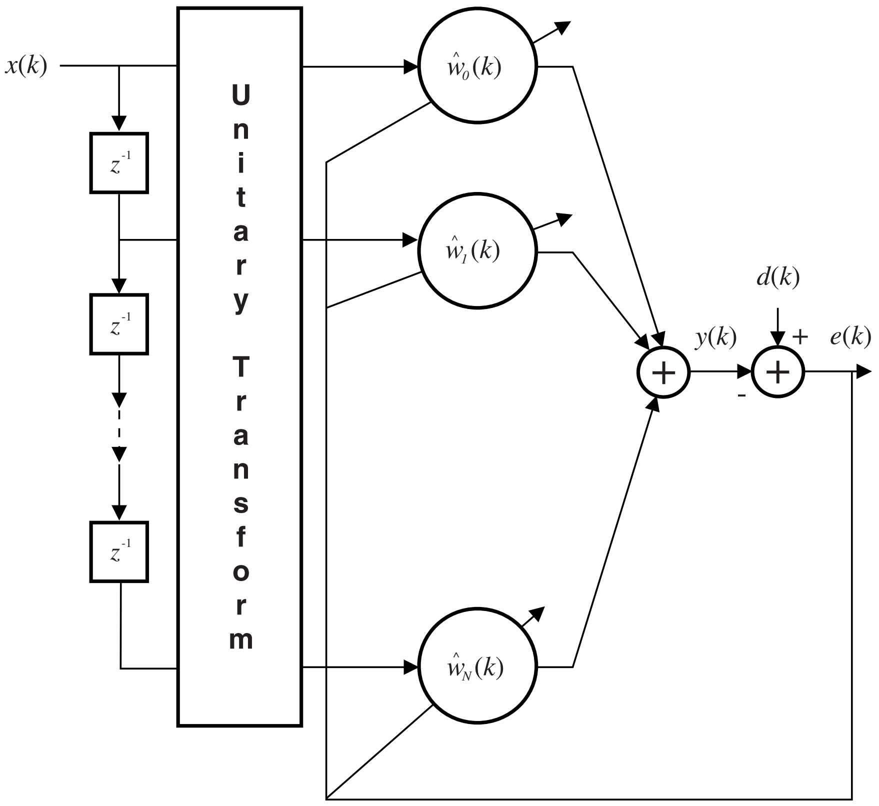
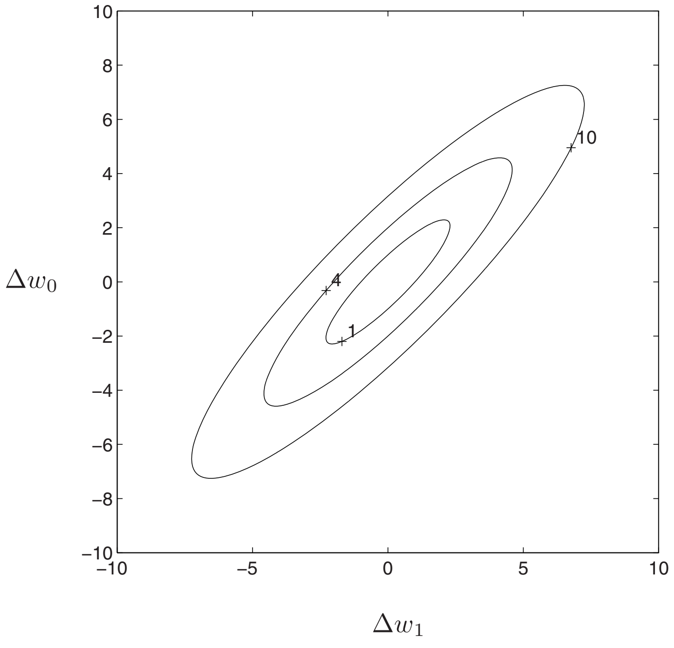


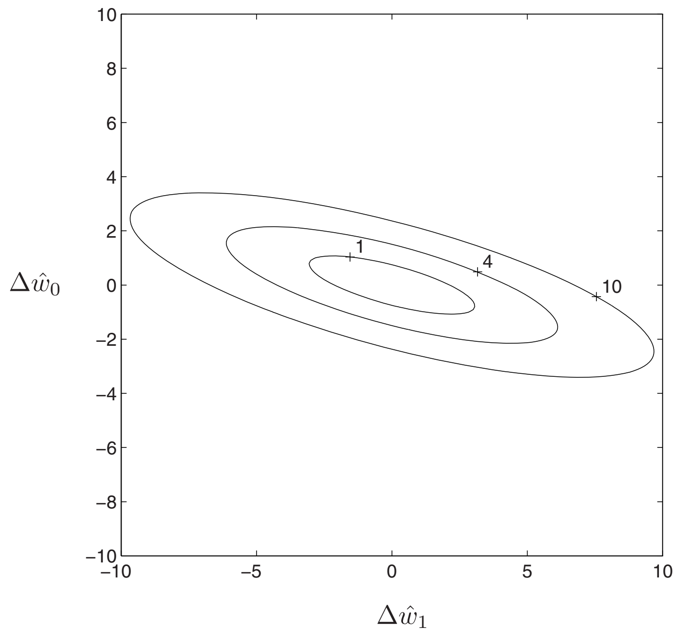
Figure 4.3 Transform-domain adaptive filter.

The effect of applying the transformation matrix $\mathbf{T}$ to the input signal is to rotate the error surface as illustrated in the numerical examples of Figs. 4.4 and 4.5. It can be noticed that the eccentricity of the MSE surface remains unchanged by the application of the transformation, and, therefore, the eigenvalue spread is unaffected by the transformation. As a consequence, no improvement in the convergence rate is expected to occur. However, if in addition each element of the transform output is power normalized, the distance between the points where the equal-error contours (given by the ellipses) meet the coefficient axes ($\Delta\hat{w}_0$ and $\Delta\hat{w}_1$) and the origin (point 0×0) are equalized. As a result, a reduction in the eigenvalue spread is expected, especially when the coefficient axes are almost aligned with the principal axes of the ellipses. Fig. 4.6 illustrates the effect of power normalization. The perfect alignment and power normalization means that the error surface will become a hyperparaboloid spheric, with the eigenvalue spread becoming equal to one. Alternatively, it means that the transform was able to turn the elements of the vector $\mathbf{s}(k)$ uncorrelated. Fig. 4.7 shows another error surface which after properly rotated and normalized is transformed into the error surface of Fig. 4.8.



$$\mathbf{R} = \begin{bmatrix} 1 & -0.9 \\ -0.9 & 1 \end{bmatrix}$$

Figure 4.4 Contours of the original MSE surface.



$$\mathbf{T} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \quad \theta = 60^\circ$$

Figure 4.5 Rotated contours of the MSE surface.

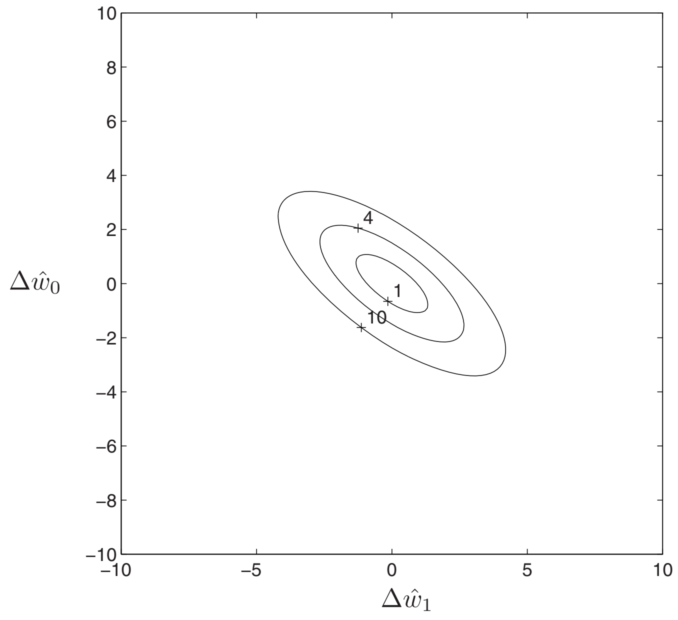
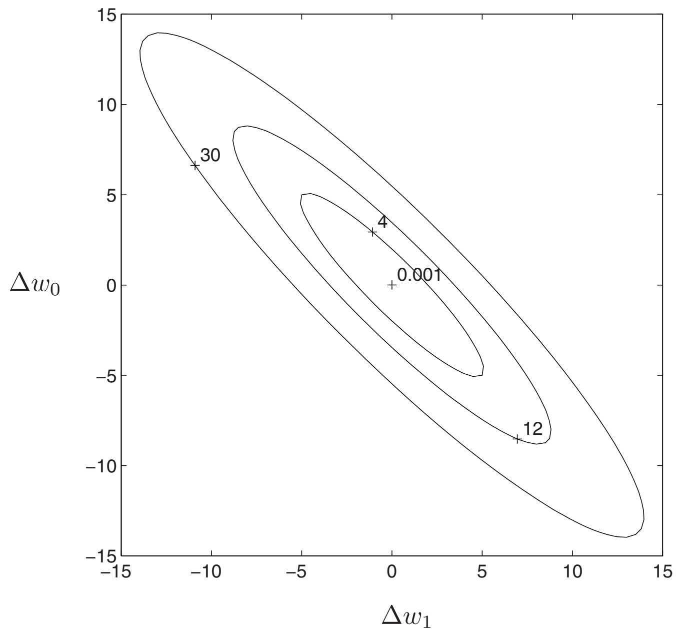


Figure 4.6 Contours of the power normalized MSE surface.



$$\mathbf{R} = \begin{bmatrix} 1 & 0.92 \\ 0.92 & 1 \end{bmatrix}$$

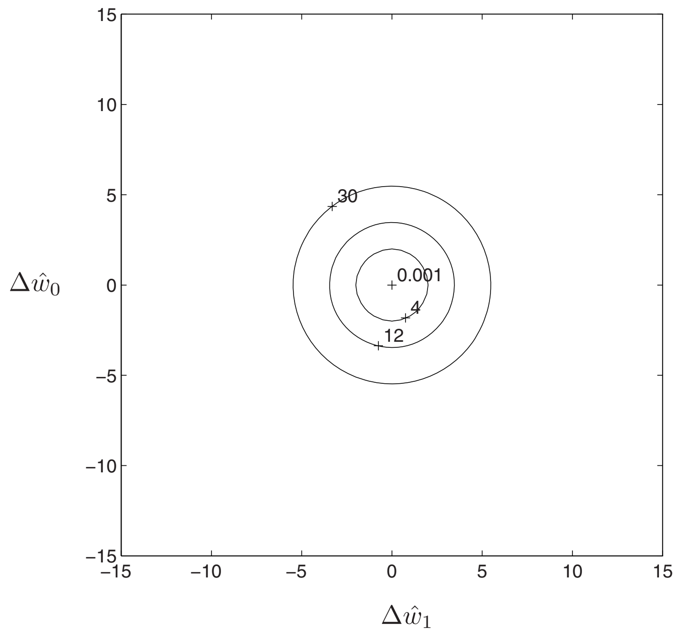
Figure 4.7 Contours of the original MSE surface.

The autocorrelation matrix related to the transform-domain filter is given by

$$\mathbf{R}_s = \mathbf{T}\mathbf{R}\mathbf{T}^T \tag{4.66}$$

therefore if the elements of $\mathbf{s}(k)$ are uncorrelated, matrix $\mathbf{R}_s$ is diagonal, meaning that the application of the transformation matrix was able to diagonalize the autocorrelation matrix $\mathbf{R}$. It can then be concluded that $\mathbf{T}^T$, in this case, corresponds to a matrix whose columns consist of the orthonormal eigenvectors of $\mathbf{R}$. The resulting transformation matrix corresponds to the Karhunen-Loève Transform (KLT)[26].

The normalization of $\mathbf{s}(k)$ and subsequent application of the LMS algorithm would lead to a transform-domain algorithm with the limitation that the solution would be independent of the input signal power. An alternative solution, without this limitation, is to apply the normalized LMS algorithm to update the coefficients of the transform-domain algorithm. We can give an interpretation for the good performance of this solution. Assuming the transform was efficient in the rotation of the MSE surface, the variable convergence factor is large in the update of the coefficients corresponding to low signal power. On the other hand, the convergence factor is small if the corresponding transform



$$\mathbf{TR} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Figure 4.8 Contours of the rotated and power normalized MSE surface.

output power is high. Specifically, the signals $s_i(k)$ are normalized by their power denoted by $\sigma_i^2(k)$ only when applied in the updating formula. The coefficient update equation in this case is

$$\hat{w}_i(k+1) = \hat{w}_i(k) + \frac{2\mu}{\gamma + \sigma_i^2(k)} e(k) s_i(k) \tag{4.67}$$

where $\sigma_i^2(k) = \alpha s_i^2(k) + (1-\alpha)\sigma_i^2(k-1)$, α is a small factor chosen in the range $0 < \alpha \leq 0.1$, and γ is also a small constant to avoid that the second term of the update equation becomes too large when $\sigma_i^2(k)$ is small.

In matrix form the above updating equation can be rewritten as

$$\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + 2\mu e(k) \mathbf{\Sigma}^{-2}(k) \mathbf{s}(k) \tag{4.68}$$

where $\mathbf{\Sigma}^{-2}(k)$ is a diagonal matrix containing as elements the inverse of the power estimates of the elements of $\mathbf{s}(k)$ added to γ.

It can be shown that if μ is chosen appropriately, the adaptive-filter coefficients converge to

$$\hat{\mathbf{w}}_o = {\mathbf{R}_s}^{-1}\mathbf{p}_s \tag{4.69}$$

where $\mathbf{R}_s = \mathbf{T}\mathbf{R}\mathbf{T}^T$ and $\mathbf{p}_s = \mathbf{T}\mathbf{p}$. As a consequence, the optimum coefficient vector is

$$\hat{\mathbf{w}}_o = \left(\mathbf{T}\mathbf{R}\mathbf{T}^T\right)^{-1}\mathbf{T}\mathbf{p} = \mathbf{T}\mathbf{R}^{-1}\mathbf{p} = \mathbf{T}\mathbf{w}_o \tag{4.70}$$

The convergence speed of the coefficient vector $\hat{\mathbf{w}}(k)$ is determined by the eigenvalue spread of $\mathbf{\Sigma}^{-2}(k)\mathbf{R}_s$.

The requirement on the transformation matrix is that it should be invertible. If the matrix $\mathbf{T}$ is not square (number of columns larger than rows), the space spanned by the polynomials formed with the rows of $\mathbf{T}$ will be of dimension $N+1$, but these polynomials are of order larger than N. This subspace does not contain the complete space of polynomials of order N. In general, except for very specific desired signals, the entire space of Nth-order polynomials would be required. For an invertible matrix $\mathbf{T}$ there is a one-to-one correspondence between the solutions obtained by the LMS and transform-domain LMS algorithms. Although the transformation matrix is not required to be unitary, it appears that no advantages are obtained by using nonunitary transforms [13].

The best unitary transform for the transform-domain adaptive filter is the KLT. However, since the KLT is a function of the input signal, it cannot be efficiently computed in real time. An alternative is to choose a unitary transform that is close to the KLT of the particular input signal. By close is meant that both transforms perform nearly the same rotation of the MSE surface. In any situation, the choice of an appropriate transform is not an easy task. Some guidelines can be given, such as: i) Since the KLT of a real signal is real, the chosen transform should be real for real input signals; ii) For speech signals the discrete-time cosine transform (DCT) is a good approximation for the KLT [26]; iii) Transforms with fast algorithms should be given special attention.

A number of real transforms such as DCT, discrete-time Hartley transform, and others, are available [26]. Most of them have fast algorithms or can be implemented in recursive frequency-domain format. In particular, the outputs of the DCT are given by

$$s_0(k) = \frac{1}{\sqrt{N+1}}\sum_{l=0}^{N} x(k-l) \tag{4.71}$$

and

$$s_i(k) = \sqrt{\frac{2}{N+1}}\sum_{l=0}^{N} x(k-l)\cos\left[\pi i\frac{(2l+1)}{2(N+1)}\right] \tag{4.72}$$

From Fig. 4.3, we observe that the delay line and the unitary transform form a single-input and multiple-output preprocessing filter. In case the unitary transform is the DCT, the transfer function from the input to the outputs of the DCT preprocessing filter can be described in a recursive format as follows:

$$T_i(z) = \frac{k_0}{N+1}\,\cos\tau_i\,\frac{[z^{N+1}-(-1)^i](z-1)}{z^N[z^2-(2\cos 2\tau_i)z+1]} \tag{4.73}$$

Algorithm 4.4

The Transform-Domain LMS Algorithm

```
Initialization
  $\mathbf{x}(0) = \hat{\mathbf{w}}(0) = [0\,0\ldots 0]^T$
  $\gamma$ = small constant
  $0 < \alpha \leq 0.1$
Do for each $x(k)$ and $d(k)$ given for $k \geq 0$
  $\mathbf{s}(k) = \mathbf{T}\mathbf{x}(k)$
  $e(k) = d(k) - \mathbf{s}^T(k)\hat{\mathbf{w}}(k)$
  $\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + 2\,\mu\,e(k)\,\mathbf{\Sigma}^{-2}(k)\mathbf{s}(k)$
```

where

$$k_0 = \begin{cases} \sqrt{2} & if \quad i = 0 \\ 2 & if \quad i = 1, ..., N \end{cases}$$

and $\tau_i = \frac{\pi i}{2(N+1)}$. The derivation details are not given here, since they are beyond the scope of this text.

For complex input signals, the discrete-time Fourier transform (DFT) is a natural choice due to its efficient implementations.

Although no general procedure is available to choose the best transform when the input signal is not known *a priori*, the decorrelation performed by the transform, followed by the power normalization, is sufficient to reduce the eigenvalue spread for a broad (not all) class of input signals. Therefore, the transform-domain LMS algorithms are expected to converge faster than the standard LMS algorithm in most applications [13].

The complete transform-domain LMS algorithm is outlined on Algorithm 4.4.

Example 4.2

Repeat the equalization problem of example 3.1 of the previous chapter using the transform-domain LMS algorithm.

(a) Compute the Wiener solution.

(b) Choose an appropriate value for μ and plot the convergence path for the transform-domain LMS algorithm on the MSE surface.

Solution:

(a) In this example, the correlation matrix of the adaptive-filter input signal is given by

$$\mathbf{R} = \begin{bmatrix} 1.6873 & -0.7937 \\ -0.7937 & 1.6873 \end{bmatrix}$$

and the cross-correlation vector **p** is

$$\mathbf{p} = \begin{bmatrix} 0.9524 \\ 0.4762 \end{bmatrix}$$

For square matrix **R** of dimension 2, the transformation matrix corresponding to the cosine transform is given by

$$\mathbf{T} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix}$$

For this filter order, the above transformation matrix coincides with the KLT.

The coefficients corresponding to the Wiener solution of the transform-domain filter are given by

$$\begin{aligned} \hat{\mathbf{w}}_o &= (\mathbf{T}\mathbf{R}\mathbf{T}^T)^{-1}\mathbf{T}\mathbf{p} \\ &= \begin{bmatrix} \frac{1}{0.8936} & 0 \\ 0 & \frac{1}{2.4810} \end{bmatrix} \begin{bmatrix} 1.0102 \\ 0.3367 \end{bmatrix} \\ &= \begin{bmatrix} 1.1305 \\ 0.1357 \end{bmatrix} \end{aligned}$$

(b) The transform-domain LMS algorithm is applied to minimize the MSE using a small convergence factor $\mu = 1/300$, in order to obtain a smoothly converging curve. The convergence path of the algorithm in the MSE surface is depicted in Fig. 4.9. As can be noted, the transformation aligned the coefficient axes with the main axes of the ellipses belonging to the error surface. The reader should notice that the algorithm follows an almost straight path to the minimum and that the effect of the eigenvalue spread is compensated by the power normalization. The convergence in this case is faster than for the LMS case.

□

From the transform-domain LMS algorithm point of view, we can consider that the LMS-Newton algorithm attempts to utilize an estimate of the KLT through $\hat{\mathbf{R}}^{-1}(k)$. On the other hand, the normalized LMS algorithm utilizes an identity transform with an instantaneous estimate of the input signal power given by $\mathbf{x}^T(k)\mathbf{x}(k)$.

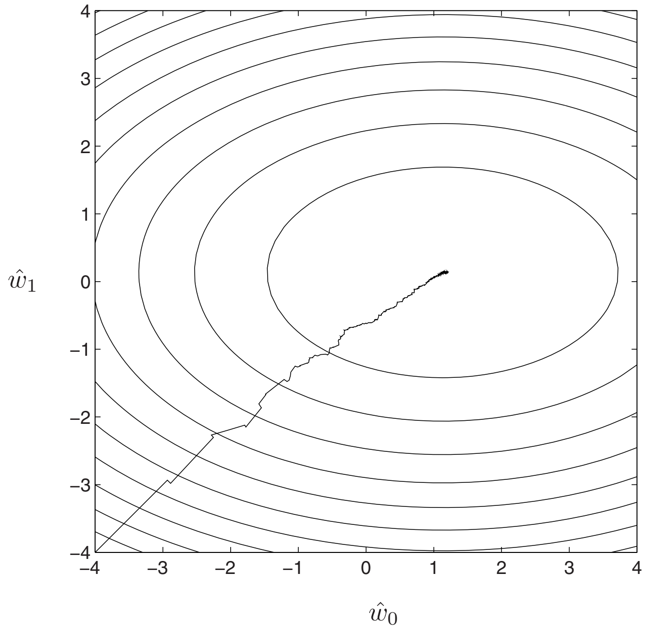


Figure 4.9 Convergence path of the transform-domain adaptive filter.

4.6 THE AFFINE PROJECTION ALGORITHM

There are situations where it is possible to recycle the old data signal in order to improve the convergence of the adaptive-filtering algorithms. Data-reusing algorithms [18], [19]-[24] are considered an alternative to increase the speed of convergence in adaptive-filtering algorithms in situations where the input signal is correlated. The penalty to be paid by data reusing is increased algorithm misadjustment, and as usual a trade-off between final misadjustment and convergence speed is achieved through the introduction of a convergence factor.

Let's assume we keep the last $L+1$ input signal vectors in a matrix as follows:

$$\mathbf{X}_{\text{ap}}(k) = \begin{bmatrix} x(k) & x(k-1) & \cdots & x(k-L+1) & x(k-L) \\ x(k-1) & x(k-2) & \cdots & x(k-L) & x(k-L-1) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x(k-N) & x(k-N-1) & \cdots & x(k-L-N+1) & x(k-L-N) \end{bmatrix}$$
$$= [\mathbf{x}(k)\,\mathbf{x}(k-1)\ldots\mathbf{x}(k-L)] \quad (4.74)$$

We can also define some vectors representing the partial reusing results at a given iteration k, such as the adaptive-filter output, the desired signal, and the error vectors.

These vectors are

$$\mathbf{y}_{\text{ap}}(k) = \mathbf{X}_{\text{ap}}^T(k)\mathbf{w}(k) = \begin{bmatrix} y_{\text{ap},0}(k) \\ y_{\text{ap},1}(k) \\ \vdots \\ y_{\text{ap},L}(k) \end{bmatrix} \tag{4.75}$$

$$\mathbf{d}_{\text{ap}}(k) = \begin{bmatrix} d(k) \\ d(k-1) \\ \vdots \\ d(k-L) \end{bmatrix} \tag{4.76}$$

$$\mathbf{e}_{\text{ap}}(k) = \begin{bmatrix} e_{\text{ap},0}(k) \\ e_{\text{ap},1}(k) \\ \vdots \\ e_{\text{ap},L}(k) \end{bmatrix} = \begin{bmatrix} d(k) - y_{\text{ap},0}(k) \\ d(k-1) - y_{\text{ap},1}(k) \\ \vdots \\ d(k-L) - y_{\text{ap},L}(k) \end{bmatrix} = \mathbf{d}_{\text{ap}}(k) - \mathbf{y}_{\text{ap}}(k) \tag{4.77}$$

The objective of the affine projection algorithm is to minimize

$$\begin{aligned} &\frac{1}{2}\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 \\ &\text{subject to :} \\ &\mathbf{d}_{\text{ap}}(k) - \mathbf{X}_{\text{ap}}^T(k)\mathbf{w}(k+1) = \mathbf{0} \end{aligned} \tag{4.78}$$

The affine projection algorithm maintains the next coefficient vector $\mathbf{w}(k+1)$ as close as possible to the current one[1] $\mathbf{w}(k)$, while forcing the *a posteriori*[2] error to be zero.

Using the method of Lagrange multipliers to turn the constrained minimization into an unconstrained one, the unconstrained function to be minimized is

$$F[\mathbf{w}(k+1)] = \frac{1}{2}\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 + \boldsymbol{\lambda}_{\text{ap}}^T(k)[\mathbf{d}(k) - \mathbf{X}_{\text{ap}}^T(k)\mathbf{w}(k+1)] \tag{4.79}$$

where $\boldsymbol{\lambda}_{\text{ap}}(k)$ is an $(L+1) \times 1$ vector of Lagrange multipliers. The above expression can be rewritten as

$$\begin{aligned} F[\mathbf{w}(k+1)] &= \frac{1}{2}\left[\mathbf{w}(k+1) - \mathbf{w}(k)\right]^T\left[\mathbf{w}(k+1) - \mathbf{w}(k)\right] \\ &\quad + \left[\mathbf{d}^T(k) - \mathbf{w}^T(k+1)\mathbf{X}_{\text{ap}}(k)\right]\boldsymbol{\lambda}_{\text{ap}}(k) \end{aligned} \tag{4.80}$$

The gradient of $F[\mathbf{w}(k+1)]$ with respect to $\mathbf{w}(k+1)$ is given by

$$\mathbf{g}_{\mathbf{w}}\left\{F[\mathbf{w}(k+1)]\right\} = \frac{1}{2}\left[2\mathbf{w}(k+1) - 2\mathbf{w}(k)\right] - \mathbf{X}_{\text{ap}}(k)\boldsymbol{\lambda}_{\text{ap}}(k) \tag{4.81}$$

[1]This procedure is known as minimal distance principle.

[2]The *a posteriori* error is the one computed with the current available data (up to instant k) using the already updated coefficient vector $\mathbf{w}(k+1)$.

Algorithm 4.5

The Affine Projection Algorithm

```
Initialization
  $\mathbf{x}(0) = \mathbf{w}(0) = [0\,0 \ldots 0]^T$
  choose $\mu$ in the range $0 < \mu \leq 2$
  $\gamma =$ small constant
Do for $k \geq 0$
  $\mathbf{e}_{\text{ap}}(k) = \mathbf{d}_{\text{ap}}(k) - \mathbf{X}_{\text{ap}}^T(k)\mathbf{w}(k)$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\text{ap}}(k)$
```

After setting the gradient of $F[\mathbf{w}(k+1)]$ with respect to $\mathbf{w}(k+1)$ equal to zero, we get

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{X}_{\text{ap}}(k)\boldsymbol{\lambda}_{\text{ap}}(k) \tag{4.82}$$

If we substitute equation (4.82) in the constraint relation of equation (4.78), we obtain

$$\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\boldsymbol{\lambda}_{\text{ap}}(k) = \mathbf{d}_{\text{ap}}(k) - \mathbf{X}_{\text{ap}}^T(k)\mathbf{w}(k) = \mathbf{e}_{\text{ap}}(k) \tag{4.83}$$

The update equation is now given by equation (4.82) with $\boldsymbol{\lambda}_{\text{ap}}(k)$ being the solution of equation (4.83), i.e.,

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\mathbf{e}_{\text{ap}}(k) \tag{4.84}$$

The above algorithm corresponds to the conventional affine projection algorithm [19] with unity convergence factor. A trade-off between final misadjustment and convergence speed is achieved through the introduction of a convergence factor as follows

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\mathbf{e}_{\text{ap}}(k) \tag{4.85}$$

Note that with the convergence factor the *a posteriori* error is no longer zero. In fact, when measurement noise is present in the environment, zeroing the *a posteriori* error is not a good idea since we are forcing the adaptive filter to compensate for the effect of a noise signal which is uncorrelated with the adaptive-filter input signal. The result is a high misadjustment when the convergence factor is one. The description of the affine projection algorithm is given in Algorithm 4.5, where an identity matrix multiplied by a small constant was added to the matrix $\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)$ in order to avoid numerical problems in the matrix inversion. The order of the matrix to be inverted depends on the number of data vectors being reused.

Let's define the hyperplane $\mathcal{S}(k)$ as follows

$$\mathcal{S}(k) = \{\mathbf{w}(k+1) \in \mathcal{R}^{N+1} : d(k) - \mathbf{w}^T(k+1)\mathbf{x}(k) = 0\} \tag{4.86}$$

It is noticed that the *a posteriori* error over this hyperplane is zero, that is, given the current input data stored in the vector $\mathbf{x}(k)$ the coefficients are updated to a point where the error computed with the coefficients updated is zero. This definition allows an insightful geometric interpretation for the affine projection algorithm.

In the affine projection algorithm the coefficients are computed such that they belong to an $L+1$-dimensional subspace $\in \mathbb{R}^{N+1}$, where $\mathbb{R}$ represents the set of real numbers, spanned by the $L+1$ columns of $\mathbf{X}_{\mathrm{ap}}(k)$. The objective of having $L+1$ *a posteriori* errors equal to zero has infinity number of solutions, such that any solution on $\mathcal{S}(k)$ can be added to a coefficient vector lying on $\mathcal{S}^{\perp}(k)$. By also minimizing $\frac{1}{2}\|\mathbf{w}(k+1)-\mathbf{w}(k)\|^2$ specifies a solution with minimum disturbance. The matrix $\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\mathbf{X}_{\mathrm{ap}}^T(k)$ represents an orthogonal projection operator on the $L+1$-dimensional subspace of $\mathbb{R}^{N+1}$ spanned by the $L+1$ columns of $\mathbf{X}_{\mathrm{ap}}(k)$. This projection matrix has $L+1$ eigenvalues equal to 1 and $N-L$ eigenvalues of value 0. On the other hand, the matrix $\mathbf{I}-\mu\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\mathbf{X}_{\mathrm{ap}}^T(k)$ has $L+1$ eigenvalues equal to 1 and $N-L$ eigenvalues of value $1-\mu$.

When $L=0$ and $L=1$ the affine projection algorithm has the normalized LMS and binormalized LMS algorithms [21] as special cases, respectively. In the binormalized case the matrix inversion has closed form solution. Fig. 4.10 illustrates the updating of the coefficient vector for a two-dimensional problem for the LMS algorithm, for the normalized LMS algorithm, for the normalized LMS algorithm with a single data reuse[3], and the binormalized LMS algorithm. Here we assume that the coefficients are originally at $\tilde{\mathbf{w}}$ when the new data vector $\mathbf{x}(k)$ becomes available and $\mathbf{x}(k-1)$ is still stored, and this scenario is used to illustrate the coefficient updating of related algorithms. In addition, it is assumed an environment with no additional noise and a system identification with sufficient order, where the LMS algorithm utilizes a small convergence factor whereas the remaining algorithms use unit convergence factor. The conventional LMS algorithm takes a step towards $\mathcal{S}(k)$ yielding a solution $\mathbf{w}(k+1)$, anywhere between points 1 and 3 in Fig. 4.10, that is closer to $\mathcal{S}(k)$ than $\tilde{\mathbf{w}}$. The NLMS algorithm with unit convergence factor performs a line search in the direction of $\mathbf{x}(k)$ to yield in a single step the solution $\mathbf{w}(k+1)$, represented by point 3 in Fig. 4.10, which belongs to $\mathcal{S}(k)$. A single reuse of the previous data using normalized LMS algorithm would lead to point 4. The binormalized LMS algorithm, which corresponds to an affine projection algorithm with two projections, yields the solution that belongs to $\mathcal{S}(k-1)$ and $\mathcal{S}(k)$, represented by point 5 in Fig. 4.10. As an illustration, it is possible to observe in Fig. 4.11 that by repeatedly re-utilizing the data vectors $\mathbf{x}(k)$ and $\mathbf{x}(k-1)$ to update the coefficients with the normalized LMS algorithm would reach point 5 in a zig-zag pattern after an infinite number of iterations. This approach is known as Kaczmarz method [22].

For a noise-free environment and sufficient-order identification problem, the optimal solution $\mathbf{w}_o$ is at the intersection of $L+1$ hyperplanes constructed with linearly independent input signal vectors. The affine projection algorithm with unit convergence factor updates the coefficient to the intersection. Fig. 4.12 illustrates the coefficient updating for a three-dimensional problem for the normalized

[3]In this algorithm the updating is performed in two steps: $\hat{\mathbf{w}}(k)=\mathbf{w}(k)+\frac{e(k)\mathbf{x}(k)}{\mathbf{x}^T(k)\mathbf{x}(k)}$ and $\mathbf{w}(k+1)=\hat{\mathbf{w}}(k)+\frac{\hat{e}(k-1)\mathbf{x}(k-1)}{\mathbf{x}^T(k-1)\mathbf{x}(k-1)}$, where in the latter case $\hat{e}(k-1)$ is computed with the previous data $d(k-1)$ and $\mathbf{x}(k-1)$ using the coefficients $\hat{\mathbf{w}}(k)$.

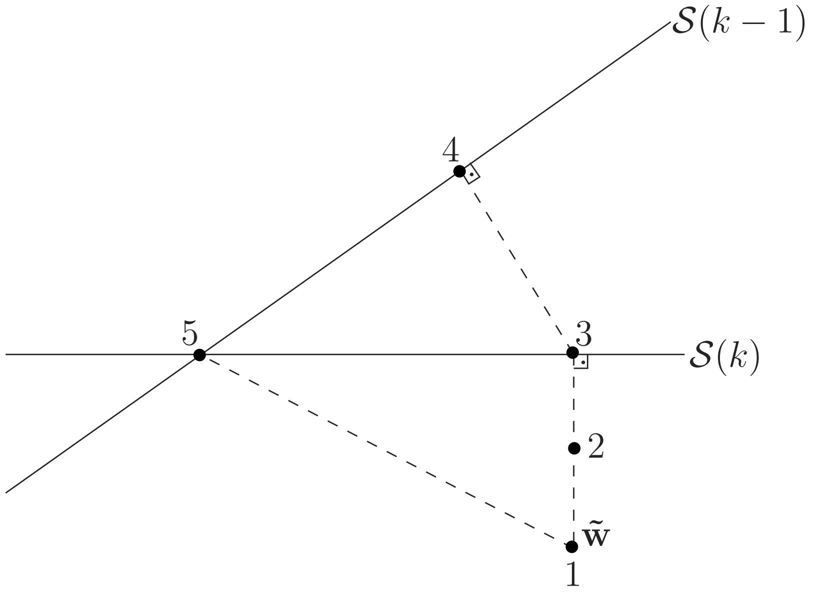


Figure 4.10 Coefficient vector updating for the normalized LMS algorithm and binormalized LMS algorithm.

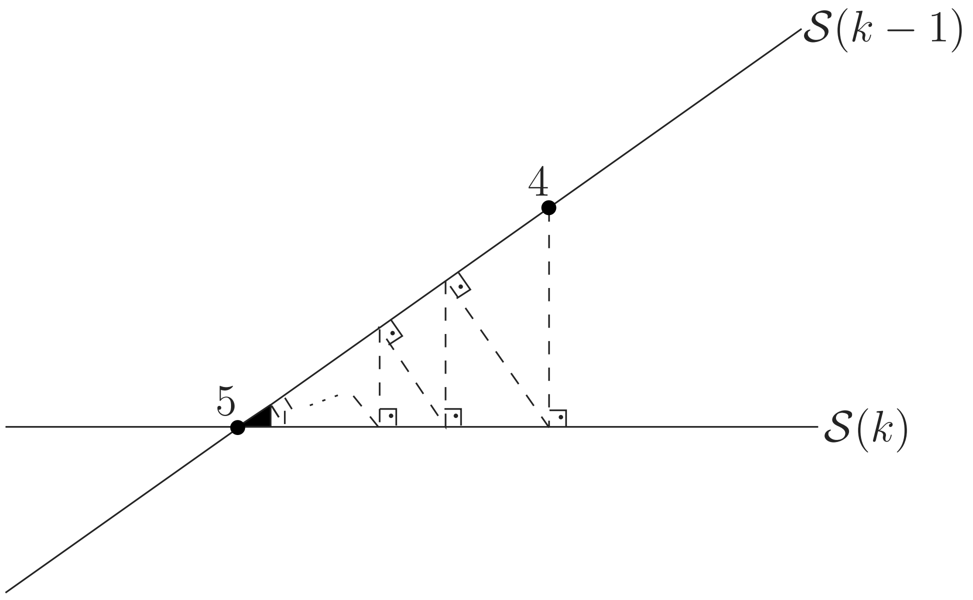


Figure 4.11 Multiple data reuse for the normalized LMS algorithm.

and binormalized LMS algorithms. It can be observed in Fig. 4.12 that $\mathbf{x}(k)$ and, consequently, $\mathbf{g_w}[e^2(k)]$ are orthogonal to the hyperplane $\mathcal{S}(k)$. Similarly, $\mathbf{x}(k-1)$ is orthogonal to the hyperplane $\mathcal{S}(k-1)$. The normalized LMS algorithm moves the coefficients from point 1 to point 2, whereas the binormalized LMS algorithm updates the coefficients to point 3 at the intersection of the two hyperplanes.

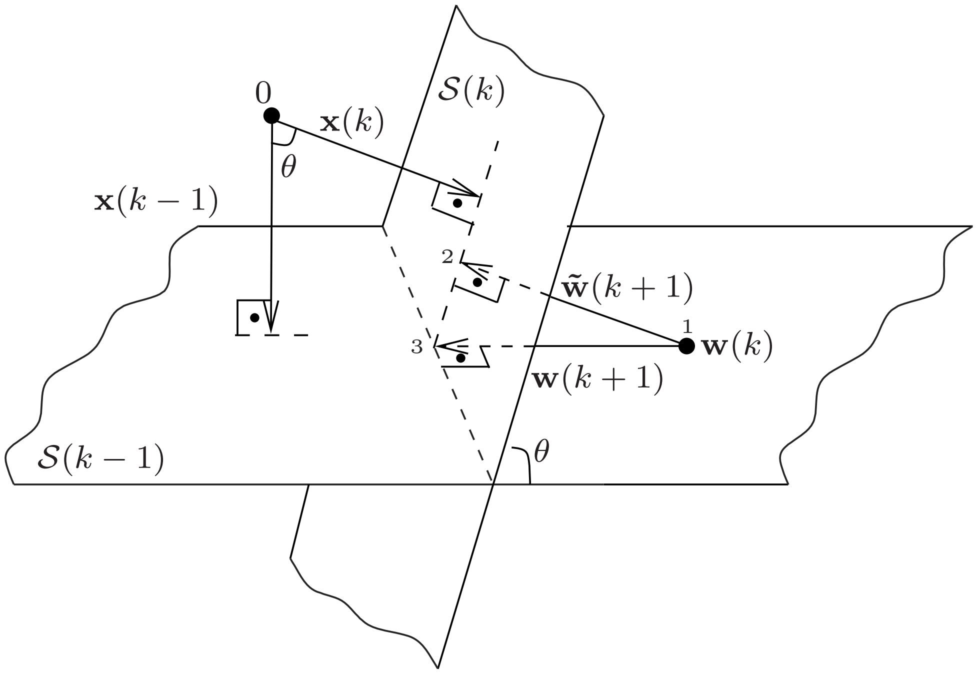


Figure 4.12 Three-dimensional coefficient vector updating for the normalized LMS algorithm and binormalized LMS algorithm.

The affine projection algorithm combines data reusing, orthogonal projections of L consecutive gradient directions, and normalization in order to achieve faster convergence than many other LMS-based algorithms. At each iteration, the affine projection algorithm yields the solution $\mathbf{w}(k+1)$ which is at the intersection of hyperplanes $\mathcal{S}(k), \mathcal{S}(k-1), \ldots, \mathcal{S}(k-L)$ and is as close as possible to $\mathbf{w}(k)$. The computational complexity of the affine projection algorithm is related to the number of data vectors being reused which ultimately determines the order of the matrix to be inverted. Some fast versions of the algorithm can be found in [20], [25]. It is also possible to reduce computations by employing data-selective strategies as will be discussed in Chapter 6.

4.6.1 Misadjustment in the Affine Projection Algorithm

The analysis of the affine projection algorithm is somewhat more involved than some of the LMS-based algorithms. The following framework provides an alternative analysis approach utilizing the concept of energy conservation [44]-[48]. This framework has been widely used in recent literature to analyze several adaptive-filtering algorithms [48]. In particular, the approach is very useful to analyze the behavior of the affine projection algorithm in a rather simple manner [47].

A general adaptive-filtering algorithm utilizes the following coefficient updating form

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu \mathbf{F}_{\mathbf{x}}(k)\mathbf{f}_{\mathbf{e}}(k) \tag{4.87}$$

where $\mathbf{F}_{\mathbf{x}}(k)$ is a matrix whose elements are functions of the input data and $\mathbf{f}_{\mathbf{e}}(k)$ is a vector whose elements are functions of the error. Assuming that the desired signal is given by

$$d(k) = \mathbf{w}_o^T \mathbf{x}(k) + n(k) \tag{4.88}$$

the underlying updating equation can be alternatively described by

$$\Delta\mathbf{w}(k+1) = \Delta\mathbf{w}(k) - \mu \mathbf{F}_{\mathbf{x}}(k)\mathbf{f}_{\mathbf{e}}(k) \tag{4.89}$$

where $\Delta\mathbf{w}(k) = \mathbf{w}(k) - \mathbf{w}_o$.

In the case of the affine projection algorithm

$$\mathbf{f}_{\mathbf{e}}(k) = -\mathbf{e}_{\text{ap}}(k) \tag{4.90}$$

according to equation (4.77). By premultiplying equation (4.89) by the input vector matrix of equation (4.74), the following expressions result

$$\begin{aligned} \mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k+1) &= \mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k) + \mu \mathbf{X}_{\text{ap}}^T(k)\mathbf{F}_{\mathbf{x}}(k)\mathbf{e}_{\text{ap}}(k) \\ -\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k) &= -\tilde{\mathbf{e}}_{\text{ap}}(k) + \mu \mathbf{X}_{\text{ap}}^T(k)\mathbf{F}_{\mathbf{x}}(k)\mathbf{e}_{\text{ap}}(k) \end{aligned} \tag{4.91}$$

where

$$\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k) = -\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k+1) \tag{4.92}$$

is the noiseless *a posteriori* error vector and

$$\tilde{\mathbf{e}}_{\text{ap}}(k) = -\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k) = \mathbf{e}_{\text{ap}}(k) - \mathbf{n}_{\text{ap}}(k) \tag{4.93}$$

is the noiseless *a priori* error vector with

$$\mathbf{n}_{\text{ap}}(k) = \begin{bmatrix} n(k) \\ n(k-1) \\ \vdots \\ n(k-L) \end{bmatrix}$$

being the standard noise vector.

For the regularized affine projection algorithm

$$\mathbf{F}_{\mathbf{x}}(k) = \mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k) + \gamma\mathbf{I}\right)^{-1}$$

where the matrix $\gamma\mathbf{I}$ is added to the matrix to be inverted in order to avoid numerical problems in the inversion operation in the cases $\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)$ is ill conditioned.

By solving equation (4.91), we get

$$\frac{1}{\mu}\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\left(\tilde{\mathbf{e}}_{\text{ap}}(k)-\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right)=\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)+\gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\text{ap}}(k)$$

If we replace the above equation in

$$\Delta\mathbf{w}(k+1)=\Delta\mathbf{w}(k)+\mu\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)+\gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\text{ap}}(k) \tag{4.94}$$

which corresponds to equation (4.89) for the affine projection case, it is possible to deduce that

$$\begin{aligned}&\Delta\mathbf{w}(k+1)-\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)=\\&\Delta\mathbf{w}(k)-\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\end{aligned} \tag{4.95}$$

From the above equation it is possible to prove that

$$\begin{aligned}&E\left[\|\Delta\mathbf{w}(k+1)\|^2\right]+E\left[\tilde{\mathbf{e}}_{\text{ap}}^T(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\right]=\\&E\left[\|\Delta\mathbf{w}(k)\|^2\right]+E\left[\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}^T(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right]\end{aligned} \tag{4.96}$$

Proof:

One can now calculate the Euclidean norm of both sides of equation (4.95)

$$\begin{aligned}&\left[\Delta\mathbf{w}(k+1)-\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\right]^T\\\times\;&\left[\Delta\mathbf{w}(k+1)-\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\right]=\\&\left[\Delta\mathbf{w}(k)-\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right]^T\\\times\;&\left[\Delta\mathbf{w}(k)-\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right]\end{aligned}$$

By performing the inner products one by one, the above equation becomes

$$\begin{aligned}&\Delta\mathbf{w}^T(k+1)\Delta\mathbf{w}(k+1)-\Delta\mathbf{w}^T(k+1)\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\\-\;&\left[\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\right]^T\Delta\mathbf{w}(k+1)\\+\;&\left[\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\right]^T\left[\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\text{ap}}(k)\right]=\\&\Delta\mathbf{w}^T(k)\Delta\mathbf{w}(k)-\Delta\mathbf{w}^T(k)\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\\-\;&\left[\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right]^T\Delta\mathbf{w}(k)\\+\;&\left[\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right]^T\left[\mathbf{X}_{\text{ap}}(k)\left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\right)^{-1}\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k)\right]\end{aligned}$$

Since $\tilde{\varepsilon}_{\mathrm{ap}}(k) = -\mathbf{X}_{\mathrm{ap}}^T(k)\Delta\mathbf{w}(k+1)$ and $\tilde{\mathbf{e}}_{\mathrm{ap}}(k) = -\mathbf{X}_{\mathrm{ap}}^T(k)\Delta\mathbf{w}(k)$

$$\begin{aligned}
&\|\Delta\mathbf{w}(k+1)\|^2 + \tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\\
+\ &\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k) + \tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k) =\\
&\|\Delta\mathbf{w}(k)\|^2 + \tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\\
+\ &\tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k) + \tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)
\end{aligned}$$

By removing the equal terms on both sides of the last equation the following equality holds

$$\begin{aligned}
&\|\Delta\mathbf{w}(k+1)\|^2 + \tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k) =\\
&\|\Delta\mathbf{w}(k)\|^2 + \tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)
\end{aligned} \tag{4.97}$$

As can be observed no approximations were utilized so far. Now by applying the expected value operation on both sides of the above equation, the expression of equation (4.96) holds.

□

If it is assumed that the algorithm has converged, that is, the coefficients remain in average unchanged, then $E\left[\|\Delta\mathbf{w}(k+1)\|^2\right] = E\left[\|\Delta\mathbf{w}(k)\|^2\right]$. As a result the following equality holds in the steady state.

$$E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right] = E\left[\tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\right] \tag{4.98}$$

In the above expression it is useful to remove the dependence on the *a posteriori* error, what can be achieved by applying equation (4.91) to the affine projection algorithm case.

$$\tilde{\varepsilon}_{\mathrm{ap}}(k) = \tilde{\mathbf{e}}_{\mathrm{ap}}(k) - \mu\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k) \tag{4.99}$$

By substituting equation (4.99) in equation (4.98) we get

$$\begin{aligned}
&E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right] = E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right.\\
-\ &\mu\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k) - \mu\mathbf{e}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\\
+\ &\left.\mu^2\mathbf{e}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k)\right]
\end{aligned} \tag{4.100}$$

The above expression can be simplified as

$$\mu^2 E\left[\mathbf{e}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\hat{\mathbf{R}}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{e}_{\mathrm{ap}}(k)\right] = \mu E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{e}_{\mathrm{ap}}(k) + \mathbf{e}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right] \tag{4.101}$$

where the following definitions are employed to simplify the discussion

$$\begin{aligned}\hat{\mathbf{R}}_{\text{ap}}(k) &= \mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)\\ \hat{\mathbf{S}}_{\text{ap}}(k) &= \left(\mathbf{X}_{\text{ap}}^T(k)\mathbf{X}_{\text{ap}}(k)+\gamma\mathbf{I}\right)^{-1}\end{aligned} \tag{4.102}$$

By rescuing the definition of the error squared of equation (3.39) and applying the expected value operator we obtain

$$\xi(k) = E[e^2(k)] = E[n^2(k)] - 2E[n(k)\Delta\mathbf{w}^T(k)\mathbf{x}(k)] + E[\Delta\mathbf{w}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\Delta\mathbf{w}(k)] \tag{4.103}$$

If the coefficients have weak dependency of the additional noise and applying the orthogonality principle, we can simplify the above expression as follows

$$\begin{aligned}\xi(k) &= \sigma_n^2 + E[\Delta\mathbf{w}^T(k)\mathbf{x}(k)\mathbf{x}^T(k)\Delta\mathbf{w}(k)]\\ &= \sigma_n^2 + E[\tilde{e}_{\text{ap},0}^2(k)]\end{aligned} \tag{4.104}$$

where $\tilde{e}_{\text{ap},0}(k)$ is the first element of vector $\tilde{\mathbf{e}}_{\text{ap}}(k)$.

In order to compute the excess of mean-square error we can remove the value of $E[\tilde{e}_{\text{ap},0}^2(k)]$ from equation (4.101). Since our aim is to compute $E[\tilde{e}_{\text{ap},0}^2(k)]$, we can substitute equation (4.93) in equation (4.101) in order to get rid of $\mathbf{e}_{\text{ap}}(k)$. The resulting expression is given by

$$\begin{aligned}&E\left[\mu(\tilde{\mathbf{e}}_{\text{ap}}(k)+\mathbf{n}_{\text{ap}}(k))^T\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)(\tilde{\mathbf{e}}_{\text{ap}}(k)+\mathbf{n}_{\text{ap}}(k))\right] =\\ &E\left[\tilde{\mathbf{e}}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)(\tilde{\mathbf{e}}_{\text{ap}}(k)+\mathbf{n}_{\text{ap}}(k)) + (\tilde{\mathbf{e}}_{\text{ap}}(k)+\mathbf{n}_{\text{ap}}(k))^T\hat{\mathbf{S}}_{\text{ap}}(k)\tilde{\mathbf{e}}_{\text{ap}}(k)\right]\end{aligned} \tag{4.105}$$

By considering the noise white and statistically independent of the input signal, the above relation can be further simplified as

$$\begin{aligned}&\mu E\left[\tilde{\mathbf{e}}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\tilde{\mathbf{e}}_{\text{ap}}(k) + \mathbf{n}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{n}_{\text{ap}}(k)\right] =\\ &2E\left[\tilde{\mathbf{e}}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)\tilde{\mathbf{e}}_{\text{ap}}(k)\right]\end{aligned} \tag{4.106}$$

The above expression, after some rearrangements, can be rewritten as

$$\begin{aligned}&2E\left\{\text{tr}[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)]\right\} - \mu E\left\{\text{tr}[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)]\right\} =\\ &\mu E\left\{\text{tr}[\mathbf{n}_{\text{ap}}(k)\mathbf{n}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)]\right\}\end{aligned} \tag{4.107}$$

where we used the property $\text{tr}[\mathbf{A}\cdot\mathbf{B}] = \text{tr}[\mathbf{B}\cdot\mathbf{A}]$.

In addition, if matrix $\hat{\mathbf{R}}_{\text{ap}}(k)$ is invertible it can be noticed that

$$\begin{aligned}\hat{\mathbf{S}}_{\text{ap}}(k) &= \left[\hat{\mathbf{R}}_{\text{ap}}(k)+\gamma\mathbf{I}\right]^{-1}\\ &= \hat{\mathbf{R}}_{\text{ap}}^{-1}(k)\left[\mathbf{I}-\gamma\hat{\mathbf{R}}_{\text{ap}}^{-1}(k)+\gamma^2\hat{\mathbf{R}}_{\text{ap}}^{-2}(k)-\gamma^3\hat{\mathbf{R}}_{\text{ap}}^{-3}(k)+\cdots\right]\\ &\approx \hat{\mathbf{R}}_{\text{ap}}^{-1}(k)\left[\mathbf{I}-\gamma\hat{\mathbf{R}}_{\text{ap}}^{-1}(k)\right] \approx \hat{\mathbf{R}}_{\text{ap}}^{-1}(k)\end{aligned} \tag{4.108}$$

where the last two relations are valid for $\gamma \ll 1$.

By assuming that the matrix $\hat{\mathbf{S}}_{\text{ap}}(k)$ is statistically independent of the noiseless *a priori* error after convergence, and of the noise, the equation (4.107) can be rewritten as

$$\begin{aligned}&2\text{tr}\left\{E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]E[\hat{\mathbf{S}}_{\text{ap}}(k)]\right\} - \mu\text{tr}\left\{E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]E[\hat{\mathbf{S}}_{\text{ap}}(k)]\right\}\\ &+\gamma\mu\text{tr}\left\{E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]\right\} = \mu\text{tr}\left\{E[\mathbf{n}_{\text{ap}}(k)\mathbf{n}^T_{\text{ap}}(k)]E[\hat{\mathbf{S}}_{\text{ap}}(k)]\right\} - \gamma\mu\text{tr}\left\{E[\mathbf{n}_{\text{ap}}(k)\mathbf{n}^T_{\text{ap}}(k)]\right\}\end{aligned} \tag{4.109}$$

This equation can be further simplified by assuming the noise is white[4] and γ is small leading to the following expression

$$(2-\mu)\text{tr}\{E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]E[\hat{\mathbf{S}}_{\text{ap}}(k)]\} = \mu\sigma_n^2\text{tr}\{E[\hat{\mathbf{S}}_{\text{ap}}(k)]\} \tag{4.110}$$

Our task now is to compute $E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]$ where we will assume in the process that this matrix is diagonal dominant whose final result has the following form

$$E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)] = \mathbf{A}E[\tilde{e}^2_{\text{ap},0}(k)] + \mu^2\mathbf{B}\sigma_n^2$$

Proof:

The i-th rows of equations (4.92) and (4.93) are given by

$$\tilde{\varepsilon}_{\text{ap},i}(k) = -\mathbf{x}^T(k-i)\Delta\mathbf{w}(k+1) \tag{4.111}$$

and

$$\tilde{e}_{\text{ap},i}(k) = -\mathbf{x}^T(k-i)\Delta\mathbf{w}(k) = e_{\text{ap},i}(k) - n(k-i) \tag{4.112}$$

for $i = 0,\ldots,L$. Using in equation (4.91) the fact that $\mathbf{X}^T_{\text{ap}}(k)\mathbf{F}_{\mathbf{X}}(k) \approx \mathbf{I}$ for small γ, then

$$-\tilde{\boldsymbol{\varepsilon}}_{\text{ap}}(k) = -\tilde{\mathbf{e}}_{\text{ap}}(k) + \mu\mathbf{e}_{\text{ap}}(k) \tag{4.113}$$

By properly utilizing in equations (4.111) and (4.112) the i-th row of equation (4.91), we obtain

$$\begin{aligned}\tilde{\varepsilon}_{\text{ap},i}(k) &= -\mathbf{x}^T(k-i)\Delta\mathbf{w}(k+1)\\ &= (1-\mu)\tilde{e}_{\text{ap},i}(k) - \mu n(k-i)\\ &= (1-\mu)\mathbf{x}^T(k-i)\Delta\mathbf{w}(k) - \mu n(k-i)\end{aligned} \tag{4.114}$$

Squaring the above equation, assuming the coefficients are weakly dependent on the noise which is in turn white noise, and following closely the procedure to derive equation (4.96) from equation (4.95), we get

$$E[(\mathbf{x}^T(k-i)\Delta\mathbf{w}(k+1))^2] = (1-\mu)^2E[(\mathbf{x}^T(k-i)\Delta\mathbf{w}(k))^2] + \mu^2\sigma_n^2 \tag{4.115}$$

[4]In this case, $E[\mathbf{n}_{\text{ap}}(k)\mathbf{n}^T_{\text{ap}}(k)] = \sigma_n^2\mathbf{I}$.

The above expression relates the squared values of the *a posteriori* and *a priori* errors. However, the same kind of relation holds for the previous time instant, that is

$$E[(\mathbf{x}^T(k-i-1)\Delta\mathbf{w}(k))^2] = (1-\mu)^2 E[(\mathbf{x}^T(k-i-1)\Delta\mathbf{w}(k-1))^2] + \mu^2\sigma_n^2$$

or

$$E[\tilde{e}^2_{\text{ap},i+1}(k)] = (1-\mu)^2 E[\tilde{e}^2_{\text{ap},i}(k-1)] + \mu^2\sigma_n^2 \tag{4.116}$$

Note that for $i=0$ this term corresponds to the second diagonal element of the matrix $E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]$. Specifically we can compute $E[\tilde{e}^2_{\text{ap},1}(k)]$ as

$$\begin{aligned} E[(\mathbf{x}^T(k-1)\Delta\mathbf{w}(k))^2] &= E[\tilde{e}^2_{\text{ap},1}(k)] \\ &= (1-\mu)^2 E[(\mathbf{x}^T(k-1)\Delta\mathbf{w}(k-1))^2] + \mu^2\sigma_n^2 \\ &= (1-\mu)^2 E[\tilde{e}^2_{\text{ap},0}(k-1)] + \mu^2\sigma_n^2 \end{aligned} \tag{4.117}$$

For $i=1$ equation (4.116) becomes

$$\begin{aligned} E[(\mathbf{x}^T(k-2)\Delta\mathbf{w}(k))^2] &= E[\tilde{e}^2_{\text{ap},2}(k)] \\ &= (1-\mu)^2 E[(\mathbf{x}^T(k-2)\Delta\mathbf{w}(k-1))^2] + \mu^2\sigma_n^2 \\ &= (1-\mu)^2 E[\tilde{e}^2_{\text{ap},1}(k-1)] + \mu^2\sigma_n^2 \end{aligned} \tag{4.118}$$

By substituting equation (4.117) in the above equation it follows that

$$E[\tilde{e}^2_{\text{ap},2}(k)] = (1-\mu)^4 E[\tilde{e}^2_{\text{ap},0}(k-2)] + [1+(1-\mu)^2]\mu^2\sigma_n^2 \tag{4.119}$$

By induction one can prove that

$$E[\tilde{e}^2_{\text{ap},i+1}(k)] = (1-\mu)^{2(i+1)} E[\tilde{e}^2_{\text{ap},0}(k-i-1)] + \left[1+\sum_{l=1}^{i}(1-\mu)^{2l}\right]\mu^2\sigma_n^2 \tag{4.120}$$

By assuming that $E[\tilde{e}^2_{\text{ap},0}(k)] \approx E[\tilde{e}^2_{\text{ap},0}(k-i)]$ for $i=0,\ldots,L$, then

$$E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)] = \mathbf{A}E[\tilde{e}^2_{\text{ap},0}(k)] + \mu^2\mathbf{B}\sigma_n^2 \tag{4.121}$$

with

$$\mathbf{A} = \begin{bmatrix} 1 & & & & \\ & (1-\mu)^2 & & \mathbf{0} & \\ & & (1-\mu)^4 & & \\ & \mathbf{0} & & \ddots & \\ & & & & (1-\mu)^{2L} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & & & & & & \\ & 1 & & & \mathbf{0} & & \\ & & 1+(1-\mu)^2 & & & & \\ & & & \ddots & & & \\ & \mathbf{0} & & & 1+\sum_{l=1}^{i}(1-\mu)^{2l} & & \\ & & & & & \ddots & \\ & & & & & & 1+\sum_{l=1}^{L-1}(1-\mu)^{2l} \end{bmatrix}$$

where it was also considered that the above matrix $E[\tilde{\mathbf{e}}_{\text{ap}}(k)\tilde{\mathbf{e}}^T_{\text{ap}}(k)]$ was diagonal dominant, as it is usually the case in practice. Note from the above relation that the convergence factor μ should be chosen in the range $0 < \mu < 2$, so that the elements of the noiseless *a priori* error remain bounded for any value of L.

□

We have available all the quantities required to calculate the excess of MSE in the affine projection algorithm. Specifically, we can substitute the result of equation (4.121) in equation (4.110) obtaining

$$(2-\mu)\left[E[\tilde{e}^2_{\text{ap},0}(k)]\text{tr}\{\mathbf{A}E[\hat{\mathbf{S}}_{\text{ap}}(k)]\} + \mu^2\sigma_n^2\text{tr}\{\mathbf{B}E[\hat{\mathbf{S}}_{\text{ap}}(k)]\}\right] = \mu\sigma_n^2\text{tr}\{E[\hat{\mathbf{S}}_{\text{ap}}(k)]\} \quad (4.122)$$

The second term on the left-hand side can be neglected in case the signal to noise ratio is high. For small μ this term also becomes substantially smaller than the term on the right-hand side. For μ close to one the referred terms become comparable only for large L, when the misadjustment becomes less sensitive to L. In the following discussions we will not consider the term multiplied by μ^2.

Assuming the diagonal elements of $E[\hat{\mathbf{S}}_{\text{ap}}(k)]$ are equal and the matrix $\mathbf{A}$ multiplying it on the left-hand side is a diagonal matrix, after a few manipulations it is possible to deduce that

$$\begin{aligned} E[\tilde{e}^2_{\text{ap},0}(k)] &= \frac{\mu}{2-\mu}\sigma_n^2\frac{\text{tr}\{E[\hat{\mathbf{S}}_{\text{ap}}(k)]\}}{\text{tr}\{\mathbf{A}E[\hat{\mathbf{S}}_{\text{ap}}(k)]\}} \\ &= \frac{(L+1)\mu}{2-\mu}\frac{1-(1-\mu)^2}{1-(1-\mu)^{2(L+1)}}\sigma_n^2 \end{aligned} \quad (4.123)$$

Therefore, the misadjustment for the affine projection algorithm is given by

$$M = \frac{(L+1)\mu}{2-\mu}\frac{1-(1-\mu)^2}{1-(1-\mu)^{2(L+1)}} \quad (4.124)$$

For large L and small $1-\mu$, this equation can be approximated by

$$M = \frac{(L+1)\mu}{(2-\mu)} \quad (4.125)$$

In [23], by considering a simplified model for the input signal vector consisting of vectors with discrete angular orientation and the independence assumption, an expression for the misadjustment of the affine projection algorithm was derived, that is

$$M = \frac{\mu}{2-\mu}E\left[\frac{1}{\|\mathbf{x}(k)\|^2}\right]\text{tr}[\mathbf{R}] \quad (4.126)$$

which is independent of L. It is observed in the experiments that higher number of reuses leads to higher misadjustment, as indicated in equation (4.125). The equivalent expression of (4.126) using the derivations presented here would lead to

$$M = \frac{(L+1)\mu}{2-\mu}E\left[\frac{1}{\|\mathbf{x}(k)\|^2}\right]\text{tr}[\mathbf{R}] \quad (4.127)$$

which can obtained from equation (4.123) by considering that

$$\mathrm{tr}\{E[\hat{\mathbf{S}}_{\mathrm{ap}}(k)]\} \approx (L+1)E\left[\frac{1}{\|\mathbf{x}(k)\|^2}\right]$$

and

$$\frac{1}{\mathrm{tr}\{\mathbf{A}E[\hat{\mathbf{S}}_{\mathrm{ap}}(k)]\}} \approx \mathrm{tr}[\mathbf{R}]$$

for μ close to one.

4.6.2 Behavior in Nonstationary Environments

In a nonstationary environment the error in the coefficients is described by the following vector

$$\Delta\mathbf{w}(k+1) = \mathbf{w}(k+1) - \mathbf{w}_o(k+1) \tag{4.128}$$

where $\mathbf{w}_o(k+1)$ is the optimal time-varying vector. For this case, equation (4.95) becomes

$$\Delta\mathbf{w}(k+1) = \Delta\hat{\mathbf{w}}(k) + \mu\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k) \tag{4.129}$$

where $\Delta\hat{\mathbf{w}}(k) = \mathbf{w}(k) - \mathbf{w}_o(k+1)$. By premultiplying the above expression by $\mathbf{X}_{\mathrm{ap}}^T(k)$ it follows that

$$\begin{aligned}\mathbf{X}_{\mathrm{ap}}^T(k)\Delta\mathbf{w}(k+1) &= \mathbf{X}_{\mathrm{ap}}^T(k)\Delta\hat{\mathbf{w}}(k) + \mu\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k)\\ -\tilde{\varepsilon}_{\mathrm{ap}}(k) &= -\tilde{\mathbf{e}}_{\mathrm{ap}}(k) + \mu\mathbf{X}_{\mathrm{ap}}^T(k)\mu\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k)\end{aligned} \tag{4.130}$$

By solving the equation (4.130), it is possible to show that

$$\frac{1}{\mu}\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}[\tilde{\mathbf{e}}_{\mathrm{ap}}(k) - \tilde{\varepsilon}_{\mathrm{ap}}(k)] = \left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}_{\mathrm{ap}}(k) \tag{4.131}$$

Following the same procedure to derive equation (4.95), we can now substitute equation (4.131) in equation (4.129) in order to deduce that

$$\begin{aligned}&\Delta\mathbf{w}(k+1) - \mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\\ &= \Delta\hat{\mathbf{w}}(k) - \mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\end{aligned} \tag{4.132}$$

By computing the energy on both sides of this equation as previously performed in equation (4.96), it is possible to show that

$$\begin{aligned}&E\left[\|\Delta\mathbf{w}(k+1)\|^2\right] + E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right]\\ &= E\left[\|\Delta\hat{\mathbf{w}}(k)\|^2\right] + E\left[\tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\right]\\ &= E\left[\|\Delta\mathbf{w}(k) + \Delta\mathbf{w}_o(k+1)\|^2\right] + E\left[\tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\right]\\ &\approx E\left[\|\Delta\mathbf{w}(k)\|^2\right] + E\left[\|\Delta\mathbf{w}_o(k+1)\|^2\right] + E\left[\tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\right]\end{aligned} \tag{4.133}$$

where $\Delta\mathbf{w}_o(k+1) = \mathbf{w}_o(k) - \mathbf{w}_o(k+1)$, and in the last equality we have assumed that $E\left[\Delta\mathbf{w}^T(k)\Delta\mathbf{w}_o(k+1)\right] \approx 0$. This assumption is valid for simple models for the time-varying behavior of the unknown system, such as random walk model [28][5]. We will adopt this assumption in order to simplify our analysis.

The time-varying characteristic of the unknown system leads to an excess mean-square error. As before, in order to calculate the excess MSE we assume that each element of the optimal coefficient vector is modeled as a first-order Markov process. As previously mentioned, this nonstationary environment can be considered somewhat simplified, but allows a manageable mathematical analysis. The first-order Markov process is described by

$$\mathbf{w}_o(k) = \lambda_{\mathbf{w}}\mathbf{w}_o(k-1) + \kappa_{\mathbf{w}}\mathbf{n}_{\mathbf{w}}(k) \tag{4.134}$$

where $\mathbf{n}_{\mathbf{w}}(k)$ is a vector whose elements are zero-mean white noise processes with variance $\sigma_{\mathbf{w}}^2$, and $\lambda_{\mathbf{w}} < 1$. If $\kappa_{\mathbf{w}} = 1$ this model may not represent a real system when $\lambda_{\mathbf{w}} \to 1$, since the $E[\mathbf{w}_o(k)\mathbf{w}_o^T(k)]$ will have unbounded elements if, for example, $\mathbf{n}_{\mathbf{w}}(k)$ is not exactly zero mean. A better model utilizes a factor $\kappa_{\mathbf{w}} = (1-\lambda_{\mathbf{w}})^{\frac{p}{2}}$, for $p \geq 1$, multiplying $\mathbf{n}_{\mathbf{w}}(k)$ in order to guarantee that $E[\mathbf{w}_o(k)\mathbf{w}_o^T(k)]$ is bounded.

In our derivations of the excess of MSE, the covariance of $\Delta\mathbf{w}_o(k+1) = \mathbf{w}_o(k) - \mathbf{w}_o(k+1)$ is required. That is

$$\begin{aligned}
\text{cov}[\Delta\mathbf{w}_o(k+1)] &= E\left[(\mathbf{w}_o(k+1) - \mathbf{w}_o(k))(\mathbf{w}_o(k+1) - \mathbf{w}_o(k))^T\right] \\
&= E\left[(\lambda_{\mathbf{w}}\mathbf{w}_o(k) + \kappa_{\mathbf{w}}\mathbf{n}_{\mathbf{w}}(k) - \mathbf{w}_o(k))(\lambda_{\mathbf{w}}\mathbf{w}_o(k) + \kappa_{\mathbf{w}}\mathbf{n}_{\mathbf{w}}(k) - \mathbf{w}_o(k))^T\right] \\
&= E\left\{[(\lambda_{\mathbf{w}}-1)\mathbf{w}_o(k) + \kappa_{\mathbf{w}}\mathbf{n}_{\mathbf{w}}(k)][(\lambda_{\mathbf{w}}-1)\mathbf{w}_o(k) + \kappa_{\mathbf{w}}\mathbf{n}_{\mathbf{w}}(k)]^T\right\}
\end{aligned} \tag{4.135}$$

Since each element of $\mathbf{n}_{\mathbf{w}}(k)$ is a zero-mean white noise process with variance $\sigma_{\mathbf{w}}^2$, and $\lambda_{\mathbf{w}} < 1$, by applying the result of equation (2.82), it follows that

$$\begin{aligned}
\text{cov}[\Delta\mathbf{w}_o(k+1)] &= \kappa_{\mathbf{w}}^2\sigma_{\mathbf{w}}^2\frac{(1-\lambda_{\mathbf{w}})^2}{1-\lambda_{\mathbf{w}}^2}\mathbf{I} + \kappa_{\mathbf{w}}^2\sigma_{\mathbf{w}}^2\mathbf{I} \\
&= \kappa_{\mathbf{w}}^2\left[\frac{1-\lambda_{\mathbf{w}}}{1+\lambda_{\mathbf{w}}} + 1\right]\sigma_{\mathbf{w}}^2\mathbf{I}
\end{aligned} \tag{4.136}$$

By employing this result, we can compute

$$E\left[\|\Delta\mathbf{w}_o(k+1)\|^2\right] = \text{tr}\{\text{cov}[\Delta\mathbf{w}_o(k+1)]\} = (N+1)\left[\frac{2\kappa_{\mathbf{w}}^2}{1+\lambda_{\mathbf{w}}}\right]\sigma_{\mathbf{w}}^2 \tag{4.137}$$

We are now in a position to solve equation (4.133) utilizing the result of equation (4.137). Again by assuming that the algorithm has converged, that is, the Euclidean norm of the coefficients increment

[5]In this model the coefficients change according to $\mathbf{w}_o(k) = \mathbf{w}_o(k-1) + \mathbf{n}_{\mathbf{w}}(k)$.

remains in average unchanged, then $E\left[\|\Delta\mathbf{w}(k+1)\|^2\right] = E\left[\|\Delta\mathbf{w}(k)\|^2\right]$. As a result, equation (4.133) can be rewritten as

$$\begin{aligned} E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right] &= E\left[\tilde{\varepsilon}_{\mathrm{ap}}^T(k)\left(\mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{X}_{\mathrm{ap}}(k)\right)^{-1}\tilde{\varepsilon}_{\mathrm{ap}}(k)\right] \\ &\quad +(N+1)\left[\frac{2\kappa_{\mathbf{w}}^2}{1+\lambda_{\mathbf{w}}}\right]\sigma_{\mathbf{w}}^2 \end{aligned} \tag{4.138}$$

Leading to the equivalent of equation (4.101) as follows

$$\begin{aligned} \mu^2 E\left[\mathbf{e}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\hat{\mathbf{R}}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{e}_{\mathrm{ap}}(k)\right] &= \mu E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{e}_{\mathrm{ap}}(k) + \mathbf{e}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right] \\ &\quad +(N+1)\left[\frac{2\kappa_{\mathbf{w}}^2}{1+\lambda_{\mathbf{w}}}\right]\sigma_{\mathbf{w}}^2 \end{aligned} \tag{4.139}$$

By solving this equation following precisely the same procedure as equation (4.101) was solved, we can derive the excess of MSE only due to the time-varying unknown system.

$$\xi_{\mathrm{lag}} = \frac{N+1}{\mu(2-\mu)}\left[\frac{2\kappa_{\mathbf{w}}^2}{1+\lambda_{\mathbf{w}}}\right]\sigma_{\mathbf{w}}^2 \tag{4.140}$$

By taking into consideration the additional noise and the time-varying parameters to be estimated, the overall excess of MSE is given by

$$\begin{aligned} \xi_{\mathrm{exc}} &= \frac{(L+1)\mu}{2-\mu}\frac{1-(1-\mu)^2}{1-(1-\mu)^{2(L+1)}}\sigma_n^2 + \frac{N+1}{\mu(2-\mu)}\left[\frac{2\kappa_{\mathbf{w}}^2}{1+\lambda_{\mathbf{w}}}\right]\sigma_{\mathbf{w}}^2 \\ &= \frac{1}{2-\mu}\left\{(L+1)\mu\frac{1-(1-\mu)^2}{1-(1-\mu)^{2(L+1)}}\sigma_n^2 + \frac{N+1}{\mu}\left[\frac{2\kappa_{\mathbf{w}}^2}{1+\lambda_{\mathbf{w}}}\right]\sigma_{\mathbf{w}}^2\right\} \end{aligned} \tag{4.141}$$

If $\kappa_{\mathbf{w}} = 1$, L is large, and $|1-\mu| < 1$, the above expression becomes simpler

$$\xi_{\mathrm{exc}} = \frac{1}{2-\mu}\left\{(L+1)\mu\sigma_n^2 + \frac{2(N+1)}{\mu(1+\lambda_{\mathbf{w}})}\sigma_{\mathbf{w}}^2\right\} \tag{4.142}$$

As can be observed, the contribution due to the lag is inversely proportional to the value of μ. This is an expected result since for small values of μ an adaptive-filtering algorithm will face difficulties in tracking the variations in the unknown system.

4.6.3 Transient Behavior

This subsection presents some considerations related to the behavior of the affine projection algorithm during the transient. In order to achieve this goal we start by removing the dependence of equation (4.96) on the noiseless *a posteriori* error through equation (4.99), very much like previously performed in the derivations of equations (4.100) and (4.101). The resulting expression is

$$\begin{aligned} E\left[\|\Delta\mathbf{w}(k+1)\|^2\right] &= E\left[\|\Delta\mathbf{w}(k)\|^2\right] + \mu^2 E\left[\mathbf{e}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\hat{\mathbf{R}}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{e}_{\mathrm{ap}}(k)\right] \\ &\quad -\mu E\left[\tilde{\mathbf{e}}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{e}_{\mathrm{ap}}(k) + \mathbf{e}_{\mathrm{ap}}^T(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\tilde{\mathbf{e}}_{\mathrm{ap}}(k)\right] \end{aligned} \tag{4.143}$$

Since from equation (4.93)

$$\begin{aligned}\mathbf{e}_{\text{ap}}(k) &= \tilde{\mathbf{e}}_{\text{ap}}(k) + \mathbf{n}_{\text{ap}}(k)\\ &= -\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k) + \mathbf{n}_{\text{ap}}(k)\end{aligned}$$

the above expression (4.143) can be rewritten as

$$\begin{aligned}&E\left[\|\Delta\mathbf{w}(k+1)\|^2\right] = E\left[\|\Delta\mathbf{w}(k)\|^2\right]\\ &+\mu^2 E\left[\left(-\Delta\mathbf{w}^T(k)\mathbf{X}_{\text{ap}}(k) + \mathbf{n}_{\text{ap}}^T(k)\right)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\left(-\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k) + \mathbf{n}_{\text{ap}}(k)\right)\right]\\ &-\mu E\left[\left(-\Delta\mathbf{w}^T(k)\mathbf{X}_{\text{ap}}(k)\right)\hat{\mathbf{S}}_{\text{ap}}(k)\left(-\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k) + \mathbf{n}_{\text{ap}}(k)\right)\right.\\ &\left.+\left(-\Delta\mathbf{w}^T(k)\mathbf{X}_{\text{ap}}(k) + \mathbf{n}_{\text{ap}}^T(k)\right)\hat{\mathbf{S}}_{\text{ap}}(k)\left(-\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k)\right)\right] \qquad (4.144)\end{aligned}$$

By considering the noise white and uncorrelated with the other quantities of this recursion, the above equation can be simplified to

$$\begin{aligned}E\left[\|\Delta\mathbf{w}(k+1)\|^2\right] &= E\left[\|\Delta\mathbf{w}(k)\|^2\right] - 2\mu E\left[\Delta\mathbf{w}^T(k)\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k)\right]\\ &+\mu^2 E\left[\Delta\mathbf{w}^T(k)\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k)\right]\\ &+\mu^2 E\left[\mathbf{n}_{\text{ap}}^T(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{n}_{\text{ap}}(k)\right] \qquad (4.145)\end{aligned}$$

By applying the property that $\text{tr}[\mathbf{AB}] = \text{tr}[\mathbf{BA}]$, this relation is equivalent to

$$\begin{aligned}\text{tr}\{\text{cov}[\Delta\mathbf{w}(k+1)]\} &= \text{tr}\left[\text{cov}[\Delta\mathbf{w}(k)]\right] - 2\mu\text{tr}\left\{E\left[\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k)\Delta\mathbf{w}^T(k)\right]\right\}\\ &+\mu^2\text{tr}\left\{E\left[\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\Delta\mathbf{w}(k)\Delta\mathbf{w}^T(k)\right]\right\}\\ &+\mu^2\text{tr}\left\{E\left[\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\right]E\left[\mathbf{n}_{\text{ap}}(k)\mathbf{n}_{\text{ap}}^T(k)\right]\right\} \qquad (4.146)\end{aligned}$$

By assuming that the $\Delta\mathbf{w}(k+1)$ is independent of the data and the noise is white, it follows that

$$\begin{aligned}\text{tr}\{\text{cov}[\Delta\mathbf{w}(k+1)]\} &= \text{tr}\left\{\left[\mathbf{I} - E\left(2\mu\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\right.\right.\right.\\ &\left.\left.\left.+\ \mu^2\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\right)\right]\text{cov}[\Delta\mathbf{w}(k)]\right\}\\ &+\mu^2\sigma_n^2\text{tr}\left\{E\left[\hat{\mathbf{S}}_{\text{ap}}(k)\hat{\mathbf{R}}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\right]\right\} \qquad (4.147)\end{aligned}$$

Now by recalling that

$$\hat{\mathbf{S}}_{\text{ap}}(k) \approx \hat{\mathbf{R}}_{\text{ap}}^{-1}(k)\left[\mathbf{I} - \gamma\hat{\mathbf{R}}_{\text{ap}}^{-1}(k)\right]$$

and by utilizing the unitary matrix $\mathbf{Q}$, that in the present discussion diagonalizes $E[\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)]$, the following relation is valid

$$\begin{aligned}\text{tr}\{\text{cov}[\Delta\mathbf{w}(k+1)]\mathbf{Q}\mathbf{Q}^T\} &= \text{tr}\left\{\mathbf{Q}\mathbf{Q}^T\left[\mathbf{I} - E\left(2\mu\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\right.\right.\right.\\ &\left.\left.\left.+\ (1-\gamma)\mu^2\mathbf{X}_{\text{ap}}(k)\hat{\mathbf{S}}_{\text{ap}}(k)\mathbf{X}_{\text{ap}}^T(k)\right)\right]\mathbf{Q}\mathbf{Q}^T\text{cov}[\Delta\mathbf{w}(k)]\mathbf{Q}\mathbf{Q}^T\right\}\\ &+(1-\gamma)\mu^2\sigma_n^2\text{tr}\left\{E\left[\hat{\mathbf{S}}_{\text{ap}}(k)\right]\right\} \qquad (4.148)\end{aligned}$$

Again by applying the property that $\mathrm{tr}[\mathbf{AB}] = \mathrm{tr}[\mathbf{BA}]$ and assuming γ small, it follows that

$$\begin{aligned}\mathrm{tr}\{\mathbf{Q}^T \mathrm{cov}[\Delta\mathbf{w}(k+1)]\mathbf{Q}\} &= \mathrm{tr}\left\{\mathbf{Q}\left[\mathbf{I} - \mathbf{Q}^T E\left(2\mu\mathbf{X}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}^T(k)\right.\right.\right. \\ &\quad + \left.\left. \mu^2\mathbf{X}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}^T(k)\right)\mathbf{Q}\right]\mathbf{Q}^T\mathrm{cov}[\Delta\mathbf{w}(k)]\mathbf{Q}\mathbf{Q}^T\Big\} \\ &\quad + \mu^2\sigma_n^2\mathrm{tr}\left\{E\left[\hat{\mathbf{S}}_{\mathrm{ap}}(k)\right]\right\}\end{aligned} \tag{4.149}$$

By defining

$$\Delta\mathbf{w}'(k+1) = \Delta\mathbf{w}(k+1)\mathbf{Q}$$

equation (4.149) can be rewritten as

$$\begin{aligned}\mathrm{tr}\{\mathrm{cov}[\Delta\mathbf{w}'(k+1)]\} &= \mathrm{tr}\left\{\mathbf{Q}^T\mathbf{Q}\left[\mathbf{I} - \mathbf{Q}^T E\left(-2\mu\mathbf{X}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}^T(k)\right.\right.\right. \\ &\quad + \left.\left. \mu^2\mathbf{X}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}^T(k)\right)\mathbf{Q}\right]\mathrm{cov}[\Delta\mathbf{w}'(k)]\Big\} \\ &\quad + \mu^2\sigma_n^2\mathrm{tr}\left\{E\left[\hat{\mathbf{S}}_{\mathrm{ap}}(k)\right]\right\} \\ &= \mathrm{tr}\left\{\left[\mathbf{I} - 2\mu\hat{\mathbf{\Lambda}} + \mu^2\hat{\mathbf{\Lambda}}\right]\mathrm{cov}[\Delta\mathbf{w}'(k)]\right\} + \mu^2\sigma_n^2\mathrm{tr}\left\{E\left[\hat{\mathbf{S}}_{\mathrm{ap}}(k)\right]\right\}\end{aligned} \tag{4.150}$$

where $\hat{\mathbf{\Lambda}}$ is a diagonal matrix whose elements are the eigenvalues of $E[\mathbf{X}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}^T(k)]$, denoted as $\hat{\lambda}_i$, for $i = 0, \ldots, N$.

By using the likely assumption that $\mathrm{cov}[\Delta\mathbf{w}'(k+1)]$ and $\hat{\mathbf{S}}_{\mathrm{ap}}(k)$ are diagonal dominant, we can disregard the trace operator in the above equation and observe that the geometric decaying curves have ratios $r_{\mathrm{cov}[\Delta\mathbf{w}(k)]} = (1 - 2\mu\hat{\lambda}_i + \mu^2\hat{\lambda}_i)$. As a result, according to the considerations in the derivation of equation (3.52), it is possible to infer that the convergence time constant is given by

$$\begin{aligned}\tau_{ei} &= \tau_{\mathrm{cov}[\Delta\mathbf{w}(k)]} \\ &= \frac{1}{\mu\hat{\lambda}_i}\frac{1}{2-\mu}\end{aligned} \tag{4.151}$$

since the error squared depends on the convergence of the diagonal elements of the covariance matrix of the coefficient-error vector, see discussions around equation (3.53). As can be observed, the time constants for error convergence are dependent on the inverse of the eigenvalues of $E[\mathbf{X}_{\mathrm{ap}}(k)\hat{\mathbf{S}}_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}^T(k)]$. However, since μ is not constrained by these eigenvalues, the speed of convergence is expected to be higher than for the LMS algorithm, particularly in situations where the eigenvalue spread of the input signal is high. Simulation results confirm the improved performance of the affine projection algorithm.

4.6.4 Complex Affine Projection Algorithm

Using the method of Lagrange multipliers to transform the constrained minimization into an unconstrained one, the unconstrained function to be minimized is

$$F[\mathbf{w}(k+1)] = \frac{1}{2}\|\mathbf{w}(k+1) - \mathbf{w}(k)\|^2 + \mathrm{re}\left\{\boldsymbol{\lambda}_{\mathrm{ap}}^T(k)[\mathbf{d}_{\mathrm{ap}}(k) - \mathbf{X}_{\mathrm{ap}}^T(k)\mathbf{w}^*(k+1)]\right\} \tag{4.152}$$

where $\boldsymbol{\lambda}_{\rm ap}(k)$ is a complex $(L+1)\times 1$ vector of Lagrange multipliers, and the real part operator is required in order to turn the overall objective function real valued. The above expression can be rewritten as

$$\begin{aligned} F[\mathbf{w}(k+1)] &= \frac{1}{2}[\mathbf{w}(k+1)-\mathbf{w}(k)]^H\,[\mathbf{w}(k+1)-\mathbf{w}(k)] \\ &\quad+\frac{1}{2}\boldsymbol{\lambda}^H_{\rm ap}(k)\left[\mathbf{d}^*_{\rm ap}(k)-\mathbf{X}^H_{\rm ap}(k)\mathbf{w}(k+1)\right] \\ &\quad+\frac{1}{2}\boldsymbol{\lambda}^T_{\rm ap}(k)\left[\mathbf{d}_{\rm ap}(k)-\mathbf{X}^T_{\rm ap}(k)\mathbf{w}^*(k+1)\right] \end{aligned} \tag{4.153}$$

The gradient of $F[\mathbf{w}(k+1)]$ with respect to $\mathbf{w}^*(k+1)$ is given by[6]

$$\frac{\partial F[\mathbf{w}(k+1)]}{\partial \mathbf{w}^*(k+1)} = \mathbf{g}_{\mathbf{w}^*}\{F[\mathbf{w}(k+1)]\} = \frac{1}{2}\,[\mathbf{w}(k+1)-\mathbf{w}(k)] - \frac{1}{2}\mathbf{X}_{\rm ap}(k)\boldsymbol{\lambda}_{\rm ap}(k) \tag{4.154}$$

After setting the gradient of $F[\mathbf{w}(k+1)]$ with respect to $\mathbf{w}^*(k+1)$ equal to zero, the expression below follows

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{X}_{\rm ap}(k)\boldsymbol{\lambda}_{\rm ap}(k) \tag{4.155}$$

By replacing equation (4.155) in the constraint relation $\mathbf{d}^*_{\rm ap}(k)-\mathbf{X}^H_{\rm ap}(k)\mathbf{w}(k+1)=\mathbf{0}$, we generate the expression

$$\mathbf{X}^H_{\rm ap}(k)\mathbf{X}_{\rm ap}(k)\boldsymbol{\lambda}_{\rm ap}(k) = \mathbf{d}^*_{\rm ap}(k) - \mathbf{X}^H_{\rm ap}(k)\mathbf{w}(k) = \mathbf{e}^*_{\rm ap}(k) \tag{4.156}$$

The update equation is now given by equation (4.155) with $\boldsymbol{\lambda}_{\rm ap}(k)$ being the solution of equation (4.156), i.e.,

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mathbf{X}_{\rm ap}(k)\left(\mathbf{X}^H_{\rm ap}(k)\mathbf{X}_{\rm ap}(k)\right)^{-1}\mathbf{e}^*_{\rm ap}(k) \tag{4.157}$$

This updating equation corresponds to the complex affine projection algorithm with unity convergence factor. As common practice, we introduce a convergence factor in order to trade-off final misadjustment and convergence speed as follows

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{X}_{\rm ap}(k)\left(\mathbf{X}^H_{\rm ap}(k)\mathbf{X}_{\rm ap}(k)\right)^{-1}\mathbf{e}^*_{\rm ap}(k) \tag{4.158}$$

The description of the complex affine projection algorithm is given in Algorithm 4.6, where as before a regularization is introduced through an identity matrix multiplied by a small constant added to the matrix $\mathbf{X}^H_{\rm ap}(k)\mathbf{X}_{\rm ap}(k)$ in order to avoid numerical problems in the matrix inversion.

4.7 SIMULATION EXAMPLES

In this section, some adaptive-filtering problems are described and solved by using some of the algorithms presented in this chapter.

[6]The reader should recall that when computing the gradient with respect to $\mathbf{w}^*(k+1)$, $\mathbf{w}(k+1)$ is treated as a constant.

Algorithm 4.6

Complex Affine Projection Algorithm

```
Initialization
  $\mathbf{x}(0) = \mathbf{w}(0) = [0\,0 \ldots 0]^T$
  choose $\mu$ in the range $0 < \mu \leq 2$
  $\gamma =$ small constant
Do for $k \geq 0$
  $\mathbf{e}^*_{\mathrm{ap}}(k) = \mathbf{d}^*_{\mathrm{ap}}(k) - \mathbf{X}^H_{\mathrm{ap}}(k)\mathbf{w}(k)$
  $\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\mathbf{X}_{\mathrm{ap}}(k)\left(\mathbf{X}^H_{\mathrm{ap}}(k)\mathbf{X}_{\mathrm{ap}}(k) + \gamma\mathbf{I}\right)^{-1}\mathbf{e}^*_{\mathrm{ap}}(k)$
```

Example 4.3: Transform-Domain LMS Algorithm

Use the transform-domain LMS algorithm to identify the system described in example of subsection 3.6.2. The transform is the DCT.

Solution:

All the results presented here for the transform-domain LMS algorithm are obtained by averaging the results of 200 independent runs.

We run the algorithm with a value of $\mu = 0.01$, with $\alpha = 0.05$ and $\gamma = 10^{-6}$. With this value of μ, the misadjustment of the transform-domain LMS algorithm is about the same as that of the LMS algorithm with $\mu = 0.02$. In Fig. 4.13, the learning curves for the eigenvalue spreads 20 and 80 are illustrated. First note that the convergence speed is about the same for different eigenvalue spreads, showing the effectiveness of the rotation performed by the transform in this case. If we compare these curves with those of Fig. 3.9 for the LMS algorithm, we conclude that the transform-domain LMS algorithm has better performance than the LMS algorithm for high eigenvalue spread. For an eigenvalue spread equal to 20, the transform-domain LMS algorithm requires around 200 iterations to achieve convergence, whereas the LMS requires at least 500 iterations. This improvement is achieved without increasing the misadjustment as can be verified by comparing the results of Tables 3.1 and 4.1.

The reader should bear in mind that the improvements in convergence of the transform-domain LMS algorithm can be achieved only if the transformation is effective. In this example, since the input signal is colored using a first-order all-pole filter, the cosine transform is known to be effective because it approximates the KLT.

The finite-precision implementation of the transform-domain LMS algorithm presents similar performance to that of the LMS algorithm, as can be verified by comparing the results of Tables 3.2 and 4.2. An eigenvalue spread of one is used in this example. The value of μ is 0.01, while the remaining parameter values are $\gamma = 2^{-b_d}$ and $\alpha = 0.05$. The value of μ in this case is chosen the same as for the LMS algorithm.

□

Table 4.1 Evaluation of the Transform-Domain LMS Algorithm

$\frac{\lambda_{\max}}{\lambda_{\min}}$	**Misadjustment**
1	0.2027
20	0.2037
80	0.2093

Table 4.2 Results of the Finite-Precision Implementation of the Transform-Domain LMS Algorithm

	$\xi(k)_Q$	$E[\|\|\Delta\mathbf{w}(k)_Q\|\|^2]$
No of bits	**Experiment**	**Experiment**
16	$1.627\ 10^{-3}$	$1.313\ 10^{-4}$
12	$1.640\ 10^{-3}$	$1.409\ 10^{-4}$
10	$1.648\ 10^{-3}$	$1.536\ 10^{-4}$

Example 4.4: Affine Projection Algorithm

An adaptive-filtering algorithm is used to identify the system described in example of subsection 3.6.2 using the affine projection algorithm using $L = 0$, $L = 1$ and $L = 4$. Do not consider the finite-precision case.

Solution:

Fig. 4.14 depicts the estimate of the MSE learning curve of the affine projection algorithm for the case of eigenvalue spread equal to 1, obtained by averaging the results of 200 independent runs. As can be noticed by increasing L the algorithm becomes faster. The chosen convergence factor is $\mu = 0.4$, and the measured misadjustments are $M = 0.32$ for $L = 0$, $M = 0.67$ for $L = 1$, and $M = 2.05$ for $L = 4$. In all cases $\gamma = 0$ is utilized, and for $L = 1$ in the first iteration we start with $L = 0$, whereas for $L = 4$ in the first four iterations we employ $L = 0, 1, 2,$ and 3, respectively. If we consider that the term $E\left[\frac{1}{\|\mathbf{X}(k)\|^2}\right] \approx \frac{1}{(N+1)\sigma_x^2}$, the expected misadjustment according to equation (4.126) is $M = 0.25$, which is somewhat close to the measured ones considering the above approximation as well as the approximations in the derivation of the theoretical formula.

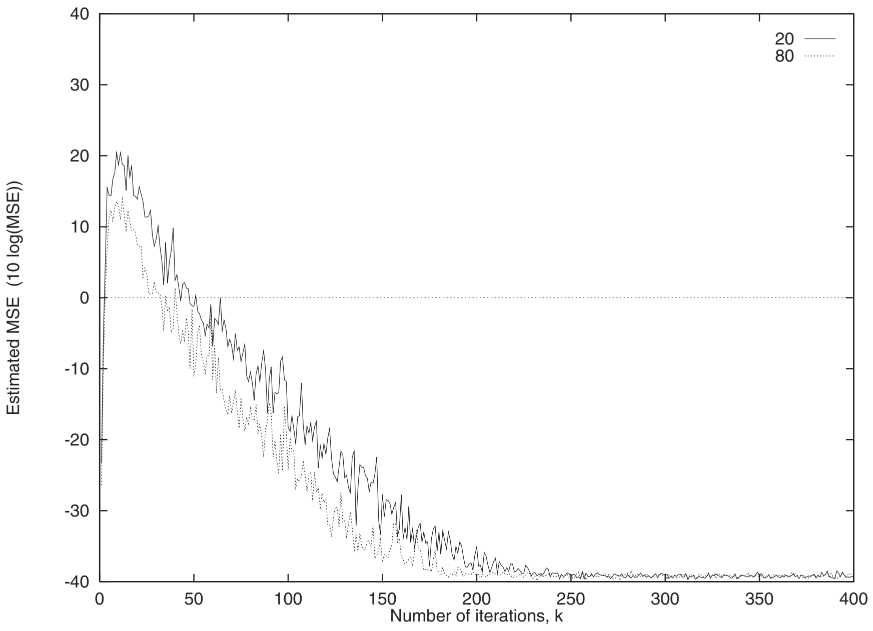


Figure 4.13 Learning curves for the transform-domain LMS algorithm for eigenvalue spreads: 20 and 80.

Fig. 4.15 depicts the average of the squared error obtained from 200 independent runs for the case of eigenvalue spread equal to 80. Again we verify that by increasing L the algorithm becomes faster. The chosen convergence factor is also $\mu = 0.4$, and the measured misadjustments for three values of the eigenvalue spread are listed in Table 4.3. It can be observed that higher eigenvalue spreads do not increase the misadjustment substantially.

Table 4.3 Evaluation of the Affine Projection Algorithm, $\mu = 0.4$

$\frac{\lambda_{\max}}{\lambda_{\min}}$	**Misadjustment,** $L = 0$		**Misadjustment,** $L = 1$		**Misadjustment,** $L = 4$	
	Experiment	**Theory**	**Experiment**	**Theory**	**Experiment**	**Theory**
1	0.32	0.25	0.67	0.37	2.05	0.81
20	0.35	0.25	0.69	0.37	2.29	0.81
80	0.37	0.25	0.72	0.37	2.43	0.81

In Fig. 4.16, it is shown the effect of using different values for the convergence factor, when $L = 1$ and the eigenvalue spread is equal to 1. For $\mu = 0.2$ the misadjustment is $M = 0.30$, for $\mu = 0.4$ the misadjustment is $M = 0.67$, and for $\mu = 1$ the misadjustment is $M = 1.56$.

$\square$

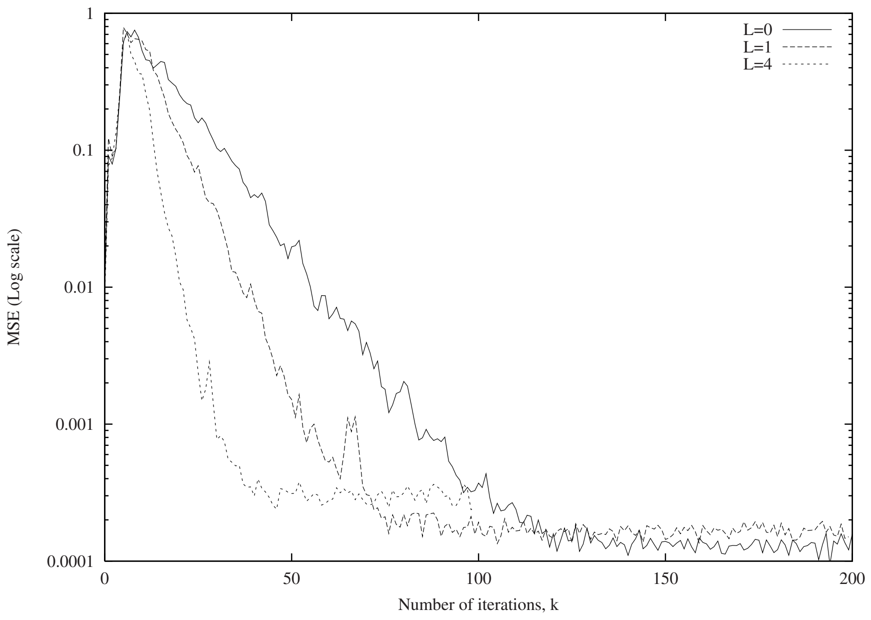


Figure 4.14 Learning curves for the affine projection algorithms for $L = 0$, $L = 1$, and $L = 4$, eigenvalue spread equal 1.

4.7.1 Signal Enhancement Simulation

In this subsection, a signal enhancement simulation environment is described. This example will also be employed in some of the following chapters.

In a signal enhancement problem, the reference signal is

$$r(k) = \sin(0.2\pi k) + n_r(k)$$

where $n_r(k)$ is zero-mean Gaussian white noise with variance $\sigma_{n_r}^2 = 10$. The input signal is given by $n_r(k)$ passed through a filter with the following transfer function

$$H(z) = \frac{0.4}{z^2 - 1.36z + 0.79}$$

The adaptive filter is a 20th-order FIR filter. In all examples, a delay $L = 10$ is applied to the reference signal.

Example 4.5: Quantized-Error and Normalized LMS Algorithms

Using the sign-error, power-of-two error with $b_d = 12$, and normalized LMS algorithms:
(a) Choose an appropriate μ in each case and run an ensemble of 50 experiments. Plot the average

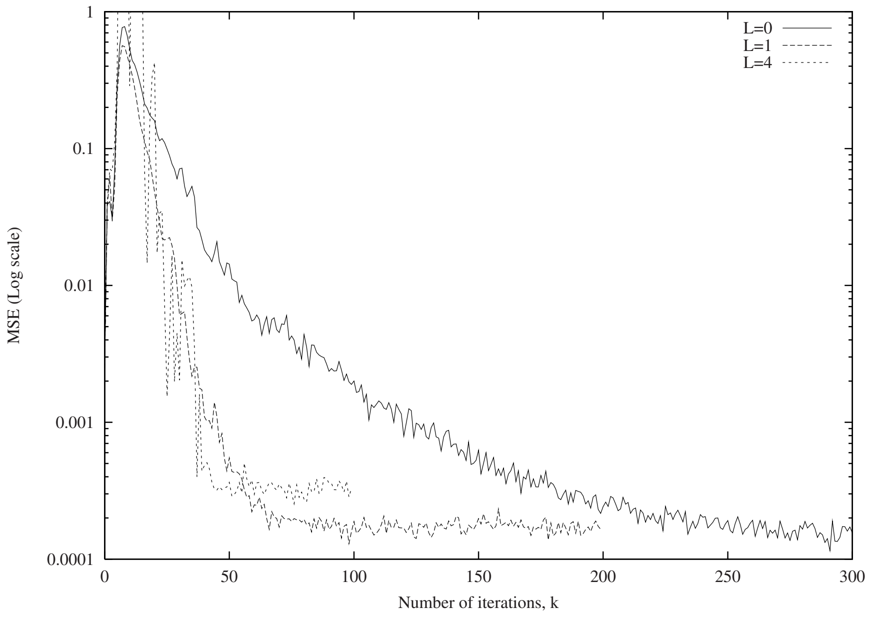


Figure 4.15 Learning curves for the affine projection algorithms for $L = 0$, $L = 1$, and $L = 4$, eigenvalue spread equal 80.

learning curve.
(b) Plot the output errors and comment on the results.

Solution:

The maximum value of μ for the LMS algorithm in this example is 0.005. The value of μ for both the sign-error and power-of-two LMS algorithms is chosen 0.001. The coefficients of the adaptive filter are initialized with zero. For the normalized LMS algorithm $\mu_n = 0.4$ and $\gamma = 10^{-6}$ are used. Fig. 4.17 depicts the learning curves for the three algorithms. The results show that the sign-error and power-of-two error algorithms present similar convergence speed, whereas the normalized LMS algorithm is faster to converge. The reader should notice that the MSE after convergence is not small since we are dealing with an example where the signal to noise ratio is low.

The DFT with 128 points of the input signal is shown in Fig. 4.18 where the presence of the sinusoid cannot be noted. In the same figure are shown the DFT of the error and the error signal itself, for the experiment using the normalized LMS algorithm. In the cases of DFT, the result presented is the magnitude of the DFT outputs. As can be verified, the output error tends to produce a signal with the same period of the sinusoid after convergence and the DFT shows clearly the presence of the sinusoid. The other two algorithms lead to similar results.

□

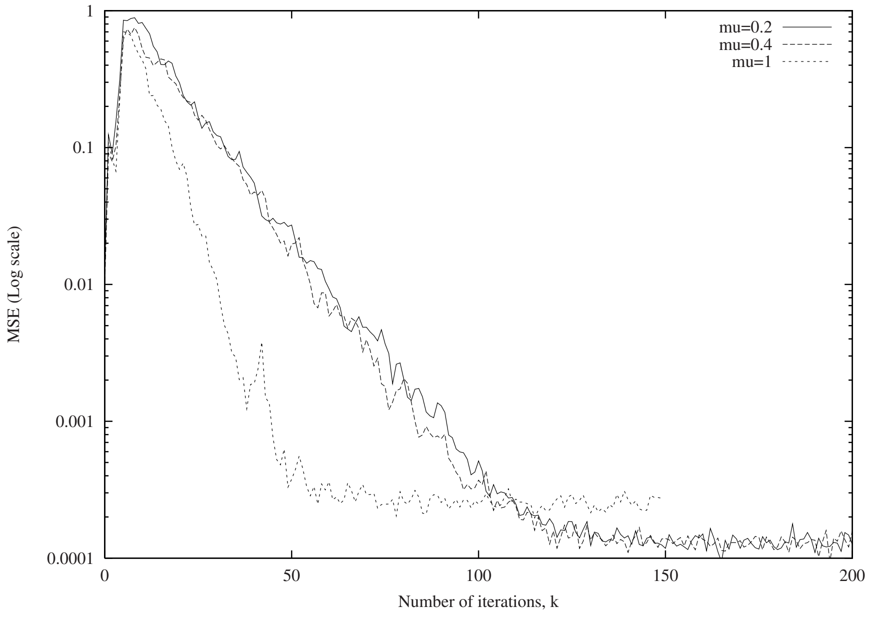


Figure 4.16 Learning curves for the affine projection algorithms for $\mu = 0.2$, $\mu = 0.4$, and $\mu = 1$.

4.7.2 Signal Prediction Simulation

In this subsection a signal prediction simulation environment is described. This example will also be used in some of the following chapters.

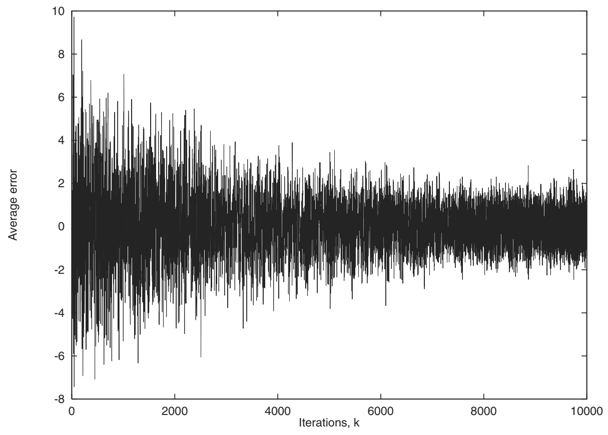
In a prediction problem the input signal is

$$x(k) = -\sqrt{2}\ \sin(0.2\pi k) + \sqrt{2}\ \sin(0.05\pi k) + n_x(k)$$

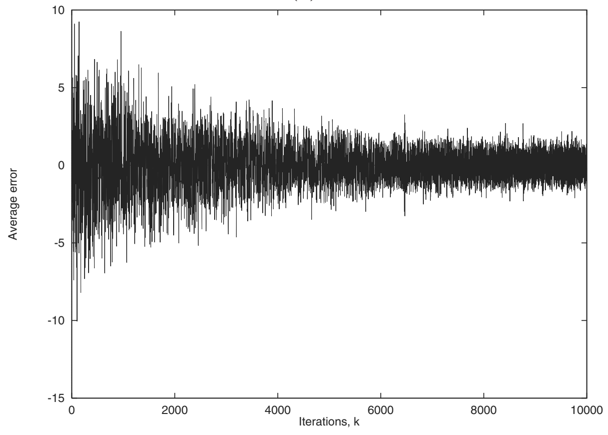
where $n_x(k)$ is zero-mean Gaussian white noise with variance $\sigma^2_{n_x} = 1$. The adaptive filter is a fourth-order FIR filter.
(a) Run an ensemble of 50 experiments and plot the average learning curve.
(b) Determine the zeros of the resulting FIR filter and comment on the results.

Example 4.6: Quantized-Error and Normalized LMS Algorithms

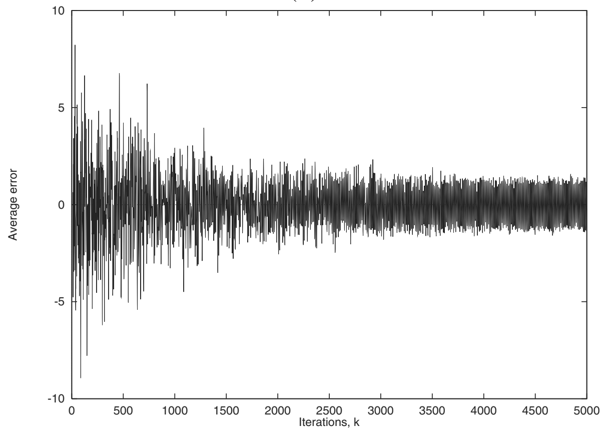
We solve the above problem using the sign-error, power-of-two error with $b_d = 12$, and normalized LMS algorithms.



(a)



(b)



(c)

Figure 4.17 Learning curves for the (a) Sign-error, (b) Power-of-two, and (c) Normalized LMS algorithms.

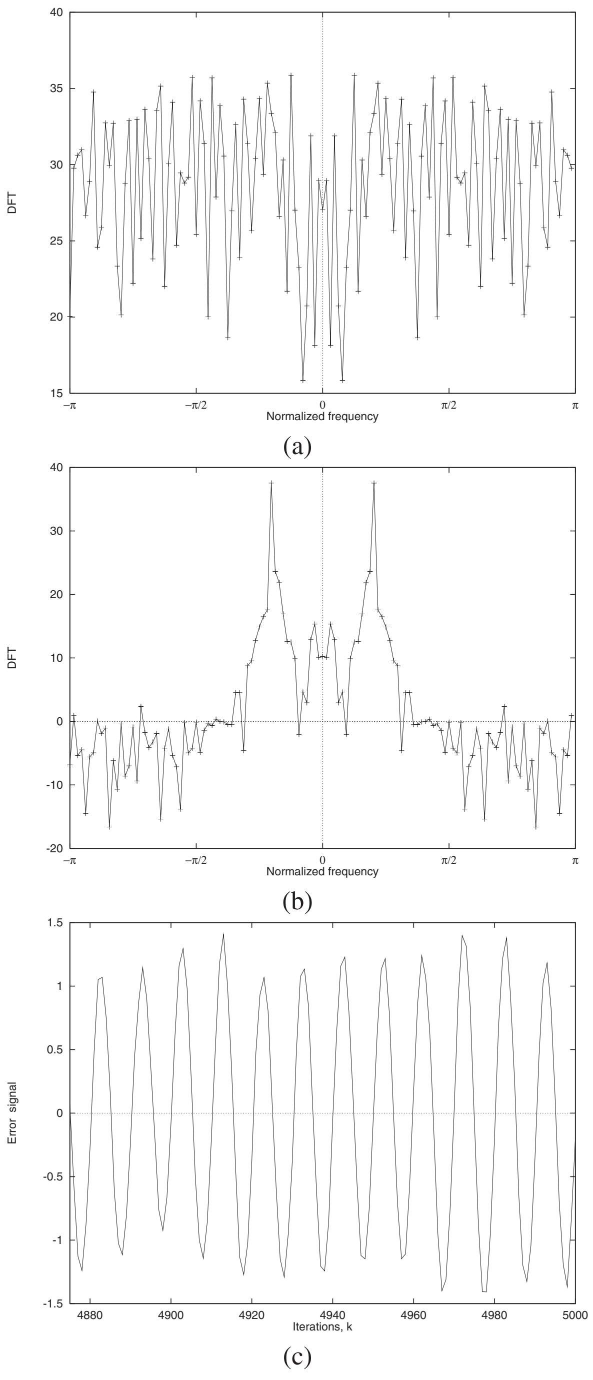


Figure 4.18 (a) DFT of the input signal, (b) DFT of the error signal, (c) The output error for the normalized LMS algorithm.

Solution:

In the first step, each algorithm is tested in order to determine experimentally the maximum value of μ in which the convergence is achieved. The choice of the convergence factor is $\mu_{\max}/5$ for each algorithm. The chosen values of μ for the sign-error and power-of-two LMS algorithms are 0.0028 and 0.0044, respectively. For the normalized LMS algorithm, $\mu_n = 0.4$ and $\gamma = 10^{-6}$ are used. The coefficients of the adaptive filter are initialized with zero. In all cases, we notice a strong attenuation of the predictor response around the frequencies of the two sinusoids. See, for example, the response depicted in Fig. 4.19 obtained by running the power-of-two LMS algorithm. The learning curves for the three algorithms are depicted in Fig. 4.20. The zeros of the transfer function from the input to the output error are calculated for the power-of-two algorithm:

$$-0.3939;\ -0.2351 \pm \jmath 0.3876;\ -0.6766 \pm \jmath 0.3422$$

Notice that the predictor tends to place its zeros at low frequencies, in order to attenuate the two low-frequency sinusoids.

In the experiments, we notice that for a given additional noise, smaller convergence factor leads to higher attenuation at the sinusoid frequencies. This is an expected result since the excess MSE is smaller. Another observation is that the attenuation also grows as the signal to noise ratio is reduced, again due to the smaller MSE.

□

4.8 CONCLUDING REMARKS

In this chapter, a number of adaptive-filtering algorithms were presented derived from the LMS algorithm. There were two basic directions followed in the derivation of the algorithms: one direction was to search for simpler algorithms from the computational point of view, and the other was to sophisticate the LMS algorithm looking for improvements in performance. The simplified algorithms lead to low-power, low-complexity and/or high-speed integrated circuit implementations [29], at a cost of increasing the misadjustment and/or of losing convergence speed among other things [30]. The simplified algorithms discussed here were the quantized-error algorithms.

We also introduced the LMS-Newton algorithm, whose performance is independent of the eigenvalue spread of the input signal correlation matrix. This algorithm is related to the RLS algorithm which will be discussed in the following chapter, although some distinctive features exist between them [39]. Newton-type algorithms with reduced computational complexity are also known [40]-[41], and the main characteristic of this class of algorithms is to reduce the computation involving the inverse of the estimate of **R**.

In the normalized LMS algorithm, the straightforward objective was to find the step size that minimizes the instantaneous output error. There are many papers dealing with the analysis [31]-[33] and applications [34] of the normalized LMS algorithm. The idea of using variable step size in the LMS

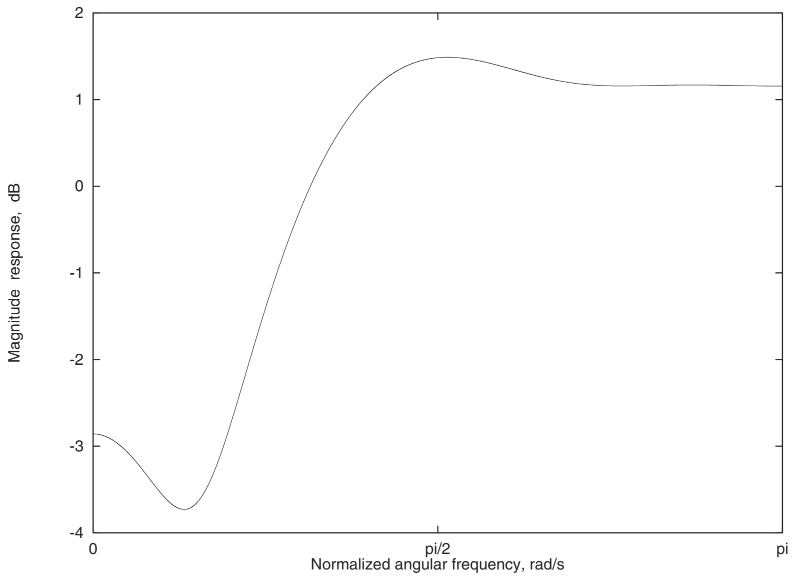
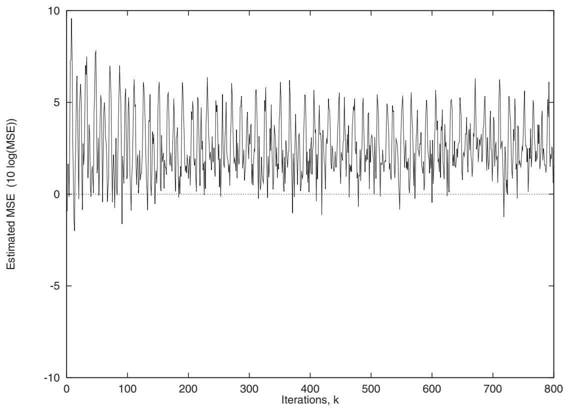


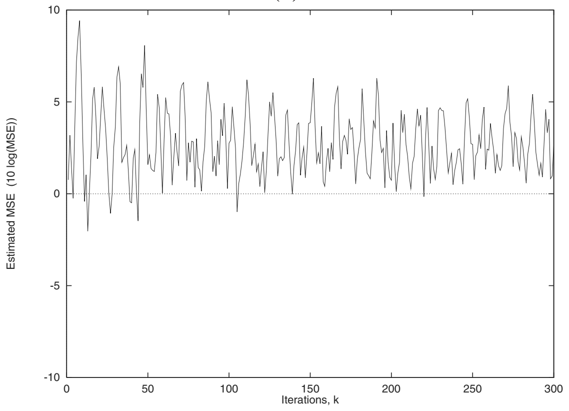
Figure 4.19 Magnitude response of the FIR adaptive filter at a given iteration after convergence using the power-of-two LMS algorithm.

and normalized LMS algorithms can lead to a number of interesting algorithms [35]-[37], that in some cases are very efficient in tracking nonstationary environments [38].

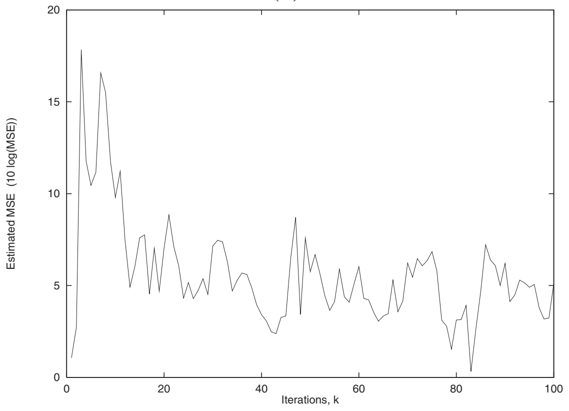
The transform-domain LMS algorithm aimed at reducing the eigenvalue spread of the input signal correlation matrix. Several frequency-domain adaptive algorithms, which are related in some sense to the transform-domain LMS algorithm, have been investigated in the recent years [42]. Such algorithms exploit the whitening property associated with the normalized transform-domain LMS algorithm, and most of them update the coefficients at a rate lower than the input sampling rate. One of the resulting structures, presented in [43], can be interpreted as a direct generalization of the transform-domain LMS algorithm and is called generalized adaptive subband decomposition structure. Such structure consists of a small-size fixed transform, which is applied to the input sequence, followed by sparse adaptive subfilters which are updated at the input rate. In high-order adaptive-filtering problems, the use of this structure with appropriately chosen transform-size and sparsity factor can lead to significant convergence rate improvement for colored input signals when compared to the standard LMS algorithm. The convergence rate improvement is achieved without the need for large transform sizes. Other algorithms to deal with high-order adaptive filters are discussed in Chapter 12.
The affine projection algorithm is very appealing in applications requiring a trade-off between convergence speed and computational complexity. Although the algorithms in the affine projection family might have high misadjustment, their combination with deterministic objective functions leading to data selective updating results in computationally efficient algorithms with low misadjustment and high convergence speed [24], as will be discussed in Chapter 6. Several simulation examples involving the LMS-based algorithms were presented in this chapter. These examples aid the reader to understand what are the main practical characteristics of the LMS-based algorithms.



(a)



(b)



(c)

Figure 4.20 Learning curves for the (a) Sign-error, (b) Power-of-two, and (c) Normalized LMS algorithms.

4.9 REFERENCES

1. T. A. C. M. Claasen and W. F. G. Mecklenbräuker, “Comparison of the convergence of two algorithms for adaptive FIR filters,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-29, pp. 670-678, June 1981.

2. N. A. M. Verhoeckx and T. A. C. M. Claasen, “Some considerations on the design of adaptive digital filters equipped with the sign algorithm,” *IEEE Trans. on Communications*, vol. COM-32, pp. 258-266, March 1984.

3. N. J. Bershad, “Comments on ‘Comparison of the convergence of two algorithms for adaptive FIR digital filters’,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-33, pp. 1604-1606, Dec. 1985.

4. P. Xue and B. Liu, “Adaptive equalizer using finite-bit power-of-two quantizer,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-34, pp. 1603-1611, Dec. 1986.

5. V. J. Mathews and S. H. Cho, “Improved convergence analysis of stochastic gradient adaptive filters using the sign algorithm,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-35, pp. 450-454, April 1987.

6. W. A. Sethares and C. R. Johnson, Jr., “A comparison of two quantized state adaptive algorithms,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-37, pp. 138-143, Jan. 1989.

7. V. J. Mathews, “Performance analysis of adaptive filters equipped with dual sign algorithm,” *IEEE Trans. on Signal Processing*, vol. 39, pp. 85-91, Jan. 1991.

8. E. Eweda, “Convergence analysis and design of an adaptive filter with finite-bit power-of-two quantizer error,” *IEEE Trans. on Circuits and Systems II : Analog and Digital Signal Processing*, vol. 39, pp. 113-115, Feb. 1992.

9. W. A. Sethares, I. M. X. Mareels, B. D. O. Anderson, C. R. Johnson, Jr., and R. R. Bitmead, “Excitation conditions for signed regressor least mean square adaptation,” *IEEE Trans. on Circuits and Systems*, vol. 35, pp. 613-624, June 1988.

10. S. H. Cho and V. J. Mathews, “Tracking analysis of the sign algorithm in nonstationary environments,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. 38, pp. 2046-2057, Dec. 1990.

11. J. C. M. Bermudez and N. J. Bershad, “A nonlinear analytical model for the quantized LMS algorithm: The arbitrary step size case,” *IEEE Trans. on Signal Processing*, vol. 44, pp. 1175-1183, May 1996.

12. S. S. Narayan, A. M. Peterson, and M. J. Narasimha, “Transform domain LMS algorithm,” *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-31, pp. 609-615, June 1983.

13. D. F. Marshall, W. K. Jenkins, and J. J. Murphy, “The use of orthogonal transform for improving performance of adaptive filters, ” *IEEE Trans. on Circuits and Systems*, vol. 36, pp. 474-484, April 1989.

14. J. C. Lee and C. K. Un, "Performance of transform-domain LMS adaptive digital filters," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-34, pp. 499-510, June 1986.

15. F. F. Yassa, "Optimality in the choice of convergence factor for gradient based adaptive algorithms," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-35, pp. 48-59, Jan. 1987.

16. B. Widrow and S. D. Stearns, *Adaptive Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1985.

17. P. S. R. Diniz and L. W. Biscainho, "Optimal variable step size for the LMS/Newton algorithm with application to subband adaptive filtering," *IEEE Trans. on Signal Processing*, vol. SP-40, pp. 2825-2829, Nov. 1992.

18. S. Roy and J. J. Shynk, "Analysis of the data-reusing LMS algorithm," *Proc. Midwest Symposium on Circuits and Systems*, Urbana, IL, pp. 1127-1130, Aug. 1989.

19. K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communications in Japan*, vol. 67-A, pp. 19-27, 1984.

20. S. L. Gay and S. Tavathia, "The fast affine projection algorithm," *Proc. IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, Detroit, MI, pp. 3023-3026, May 1995.

21. J. A. Apolinário, M. L. R. de Campos, and P. S. R. Diniz, "The binormalized data-reusing LMS algorithm," *IEEE Trans. on Signal Processing*, vol. 48, pp. 3235-3242, Nov. 2000.

22. R. A. Soni, K. A. Gallivan, and W. K. Jenkins, "Low-complexity data-reusing methods in adaptive filtering," *IEEE Trans. on Signal Processing*, vol. 52, pp. 394-405, Feb. 2004.

23. S. G. Sankaran and A. A. (Louis) Beex, "Convergence behavior of affine projection algorithms," *IEEE Trans. on Signal Processing*, vol. 48, pp. 1086-1096, April 2000.

24. S. Werner and P. S. R. Diniz, "Set-membership affine projection algorithm," *IEEE Signal Processing Letters*, vol. 8, pp. 231-235, Aug 2001.

25. G.-O. Glentis, K. Berberidis, and S. Theodoridis, "Efficient least squares adaptive algorithms for FIR transversal filtering," *IEEE Signal Processing Magazine*, vol. 16, pp. 13-41, July 1999.

26. N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video*, Prentice Hall, Englewood Cliffs, NJ, 1984.

27. R. Price, "A useful theorem for nonlinear devices having Gaussian inputs," *IRE Trans. on Information Theory*, vol. IT-4, pp. 69-72, June 1958.

28. A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 3rd edition, McGraw-Hill, New York, NY, 1991.

29. H. Samueli, B. Daneshrad, R. B. Joshi, B. C. Wong, and H. T. Nicholas, III, "A 64-tap CMOS echo canceller/decision feedback equalizer for 2B1Q HDSL ," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 839-847, Aug. 1991.

30. C. R. Johnson, Jr., *Lectures on Adaptive Parameter Estimation*, Prentice Hall, Englewood Cliffs, NJ, 1988.

31. D. T. Slock, "On the convergence behavior of the LMS and normalized LMS algorithms," *IEEE Trans. on Signal Processing*, vol. 40, pp. 2811-2825, Sept. 1993.

32. N. J. Bershad, "Analysis of the normalized LMS algorithm with Gaussian inputs," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-34, pp. 793-806, Aug. 1986.

33. M. Tarrab and A. Feuer, "Convergence and performance analysis of the normalized LMS algorithm with uncorrelated Gaussian data," *IEEE Trans. on Information Theory*, vol. IT-34, pp. 680-691, July 1988.

34. J. F. Doherty, "An adaptive algorithm for stable decision-feedback filtering," *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, vol. 40, pp. 1-8, Jan. 1993.

35. W. B. Mikhael, F. H. Fu, L. G. Kazovsky, G. S. Kang, and L. J. Fransen, "Adaptive filter with individual adaptation of parameters," *IEEE Trans. on Circuits and Systems*, vol. 33, pp. 677-686, July 1986.

36. R. W. Harris, D. M. Chabries, and F. A. Bishop, "A variable step (VS) adaptive filter algorithm," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-34, pp. 309-316, April 1986.

37. C. S. Modlin and J. M. Cioffi, "A fast decision feedback LMS algorithm using multiple step sizes," *Proc. IEEE Inter. Conf. on Communications*, New Orleans, pp. 1201-1205, May 1994.

38. S. D. Peters and A. Antoniou, "Environment estimation for enhanced NLMS adaptation," *Proc. IEEE Pac. Rim Conf. on Comm., Comp. and Sig. Proc.*, Victoria, Canada, pp. 342-345, May 1993.

39. P. S. R. Diniz, M. L. R. de Campos, and A. Antoniou, "Analysis of LMS-Newton adaptive filtering algorithms with variable convergence factor," *IEEE Trans. on Signal Processing*, vol. 43, pp. 617-627, March 1995.

40. D. F. Marshall and W. K. Jenkins, "A fast quasi-Newton adaptive filtering algorithm," *IEEE Trans. on Signal Processing*, vol. 40, pp. 1652-1662, July 1993.

41. G. V. Moustakides and S. Theodoridis, "Fast Newton transversal filters - A new class of adaptive estimation algorithm," *IEEE Trans. on Signal Processing*, vol. 39, pp. 2184-2193, Oct. 1991.

42. J. J. Shynk, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Processing Magazine*, vol. 9, pp. 15-37, Jan. 1992.

43. M. R. Petraglia and S. K. Mitra,"Adaptive FIR filter structure based on the generalized subband decomposition of FIR filters," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 40, pp. 354-362, June 1993.

44. A. H. Sayed and M. Rupp, "Error-energy bounds for adaptive gradient algorithms," *IEEE Trans. on Signal Processing*, vol. 44, pp. 1982-1989, August 1996.

45. N. R. Yousef and A. H. Sayed, "A unified approach to the steady-state and tracking analyses of adaptive filters," *IEEE Trans. on Signal Processing*, vol. 49, pp. 314-324, Feb. 2001.

46. T. Y. Al-Naffouri and A. H. Sayed, "Transient analysis of adaptive filters with error nonlinearities," *IEEE Trans. on Signal Processing*, vol. 51, pp. 653-663, March 2003.

47. H.-C. Shin and A. H. Sayed, "Mean-square performance of a family of affine projection algorithms," *IEEE Trans. on Signal Processing*, vol. 52, pp. 90-102, Jan. 2004.

48. A. H. Sayed, *Fundamentals of Adaptive Filtering*, John Wiley & Sons, Hoboken, NJ, 2003.

49. M. L. R. de Campos and A. Antoniou, "A new quasi-Newton adaptive filtering algorithm," *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, vol. 44, pp. 924-934, Nov. 1997.

4.10 PROBLEMS

1. From equation (4.16) derive the difference equation for $\mathbf{v}'(k)$ given by equation (4.19).

2. Prove the validity of equation (4.27).

3. The sign-error algorithm is used to predict the signal $x(k) = \sin(\pi k/3)$ using a second-order FIR filter with the first tap fixed at 1, by minimizing the mean square value of $y(k)$. This is an alternative way to interpret how the predictor works. Calculate an appropriate μ, the output signal $y(k)$, and the filter coefficients for the first 10 iterations. Start with $\mathbf{w}^T(0) = [1\ 0\ 0]$.

4. Derive an LMS-Newton algorithm leading to zero *a posteriori* error.

5. Derive the updating equations of the affine projection algorithm, for $L = 1$.

6. Use the sign-error algorithm to identify a system with the transfer function given below. The input signal is a uniformly distributed white noise with variance $\sigma_x^2 = 1$, and the measurement noise is Gaussian white noise uncorrelated with the input with variance $\sigma_n^2 = 10^{-3}$. The adaptive filter has 12 coefficients.

$$H(z) = \frac{1 - z^{-12}}{1 + z^{-1}}$$

(a) Calculate the upper bound for μ ($\mu_{\max}$) to guarantee the algorithm stability.
(b) Run the algorithm for $\mu_{\max}/2$, $\mu_{\max}/5$, and $\mu_{\max}/10$. Comment on the convergence behavior in each case.
(c) Measure the misadjustment in each example and compare with the results obtained by equation (4.28).
(d) Plot the obtained FIR filter frequency response at any iteration after convergence is achieved and compare with the unknown system.

7. Repeat the previous problem using an adaptive filter with 8 coefficients and interpret the results.

8. Repeat problem 6 when the input signal is a uniformly distributed white noise with variance $\sigma_{n_x}^2 = 0.5$, filtered by an all-pole filter given by

$$H(z) = \frac{z}{z - 0.9}$$

9. In problem 6, consider that the additional noise has the following variances (a) $\sigma_n^2 = 0$, (b) $\sigma_n^2 = 1$. Comment on the results obtained in each case.

10. Perform the equalization of a channel with the following impulse response

$$h(k) = ku(k) - (2k - 9)u(k - 5) + (k - 9)u(k - 10)$$

using a known training signal consisting of a binary (-1,1) random signal. An additional Gaussian white noise with variance 10^{-2} is present at the channel output.
(a) Apply the sign-error with an appropriate μ and find the impulse response of an equalizer with 15 coefficients.
(b) Convolve the equalizer impulse response at an iteration after convergence, with the channel impulse response and comment on the result.

11. In a system identification problem, the input signal is generated by an autoregressive process given by

$$x(k) = -1.2x(k - 1) - 0.81x(k - 2) + n_x(k)$$

where $n_x(k)$ is zero-mean Gaussian white noise with variance such that $\sigma_x^2 = 1$. The unknown system is described by

$$H(z) = 1 + 0.9z^{-1} + 0.1z^{-2} + 0.2z^{-3}$$

The adaptive filter is also a third-order FIR filter. Using the sign-error algorithm:
(a) Choose an appropriate μ, run an ensemble of 20 experiments, and plot the average learning curve.
(b) Measure the excess MSE and compare the results with the theoretical value.

12. In the previous problem, calculate the time constant τ_{wi} and the expected number of iterations to achieve convergence.

13. The sign-error 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 $\sigma_{\mathbf{w}}^2 = 0.001$. The initial time-varying system multiplier coefficients are

$$\mathbf{w}_o^T = [0.03490 \quad -0.011 \quad -0.06864 \;\; 0.22391 \;\; 0.55686 \;\; 0.35798 \quad -0.0239 \quad -0.07594]$$

The input signal is Gaussian white noise with variance $\sigma_x^2 = 0.7$, and the measurement noise is also Gaussian white noise independent of the input signal and of the elements of $\mathbf{n_w}(k)$, with variance $\sigma_n^2 = 0.01$.
For $\mu = 0.01$, simulate the experiment described and measure the excess MSE.

14. Reduce the value of $\lambda_{\mathbf{w}}$ to 0.95 in problem 13, simulate, and comment on the results.

15. Suppose a 15th-order FIR digital filter with multiplier coefficients given below, is identified through an adaptive FIR filter of the same order using the sign-error algorithm. Use fixed-point arithmetic and run simulations for the following case.

Additional noise: white noise with variance	$\sigma_n^2 = 0.0015$
Coefficient wordlength:	$b_c = 16$ bits
Signal wordlength:	$b_d = 16$ bits
Input signal: Gaussian white noise with variance	$\sigma_x^2 = 0.7$
	$\mu = 0.01$

$\mathbf{w}_o^T = [0.0219360 \quad 0.0015786 \quad -0.0602449 \quad -0.0118907 \quad 0.1375379 \quad 0.0574545 \quad -0.3216703 \quad -0.5287203 \quad -0.2957797 \quad 0.0002043 \quad 0.290670 \quad -0.0353349 \quad -0.068210 \quad 0.0026067 \quad 0.0010333 \quad -0.0143593]$

Plot the learning curves of the estimates of $E[||\Delta\mathbf{w}(k)_Q||^2]$ and $\xi(k)_Q$ obtained through 25 independent runs, for the finite- and infinite-precision implementations.

16. Repeat the above problem for the following cases
(a) $\sigma_n^2 = 0.01$, $b_c = 12$ bits, $b_d = 12$ bits, $\sigma_x^2 = 0.7$, $\mu = 10^{-4}$.
(b) $\sigma_n^2 = 0.1$, $b_c = 10$ bits, $b_d = 10$ bits, $\sigma_x^2 = 0.8$, $\mu = 2.0\ 10^{-5}$.
(c) $\sigma_n^2 = 0.05$, $b_c = 14$ bits, $b_d = 16$ bits, $\sigma_x^2 = 0.8$, $\mu = 3.5\ 10^{-4}$.

17. Repeat problem 15 for the case where the input signal is a first-order Markov process with $\lambda_{\mathbf{x}} = 0.95$.

18. Repeat problem 6 for the dual-sign algorithm given $\gamma = 16$ and $\rho = 1$, and comment on the results.

19. Repeat problem 6 for the power-of-two error algorithm given $b_d = 6$ and $\tau = 2^{-b_d}$, and comment on the results.

20. Repeat problem 6 for the sign-data and sign-sign algorithms and compare the results.

21. Show the validity of the matrix inversion lemma defined in equation (4.51).

22. For the setup described in problem 8, choose an appropriate μ and run the LMS-Newton algorithm.
(a) Measure the misadjustment.
(b) Plot the frequency response of the FIR filter obtained after convergence is achieved and compare with the unknown system.

23. Repeat problem 8 using the normalized LMS algorithm.

24. Repeat problem 8 using the transform-domain LMS algorithm with DCT. Compare the results with those obtained with the standard LMS algorithm.

25. Repeat problem 8 using the affine projection algorithm.

26. Repeat problem 8 using the transform-domain LMS algorithm with DCT.

27. For the input signal described in problem 8, derive the autocorrelation matrix of order one (2×2). Apply the DCT and the normalization to $\mathbf{R}$ in order to generate $\hat{\mathbf{R}} = \mathbf{\Sigma}^{-2}\mathbf{T}\mathbf{R}\mathbf{T}^T$. Compare the eigenvalue spreads of $\mathbf{R}$ and $\hat{\mathbf{R}}$.

28. Repeat the previous problem for **R** with dimension 3 by 3.

29. Use the complex affine projection algorithm with $L = 3$ to equalize a channel with the transfer function given below. The input signal is a four QAM signal representing a randomly generated bit stream with the signal to noise ratio $\frac{\sigma_{\tilde{x}}^2}{\sigma_n^2} = 20$ at the receiver end, that is, $\tilde{x}(k)$ is the received signal without taking into consideration the additional channel noise. The adaptive filter has 10 coefficients.

$$H(z) = (0.34 - 0.27\jmath) + (0.87 + 0.43\jmath)z^{-1} + (0.34 - 0.21\jmath)z^{-2}$$

(a) Run the algorithm for $\mu = 0.1$, $\mu = 0.4$, and $\mu = 0.8$. Comment on the convergence behavior in each case.
(b) Plot the real versus imaginary parts of the received signal before and after equalization.
(c) Increase the number of coefficients to 20 and repeat the experiment in (b).

30. Repeat the problem 29 for the case of the normalized LMS algorithm.

31. In a system identification problem the input signal is generated from a four QAM of the form

$$x(k) = x_{\text{re}}(k) + \jmath x_{\text{im}}(k)$$

where $x_{\text{re}}(k)$ and $x_{\text{im}}(k)$ assume values ± 1 randomly generated. The unknown system is described by

$$H(z) = 0.32 + 0.21\jmath + (-0.3 + 0.7\jmath)z^{-1} + (0.5 - 0.8\jmath)z^{-2} + (0.2 + 0.5\jmath)z^{-3}$$

The adaptive filter is also a third-order complex FIR filter, and the additional noise is composed of zero-mean Gaussian white noises in the real and imaginary parts with variance $\sigma_n^2 = 0.4$. Using the complex affine projection algorithm with $L = 1$, choose an appropriate μ, run an ensemble of 20 experiments, and plot the average learning curve.

32. Repeat the problem 31 utilizing the affine projection algorithm with $L = 4$.

33. Derive a complex transform-domain LMS algorithm for the case the transformation matrix is the DFT.

34. The Quasi-Newton algorithm first proposed in [49] is described by the following set of equations

$$\begin{aligned}
e(k) &= d(k) - \mathbf{w}^T(k)\mathbf{x}(k) \\
\mu(k) &= \frac{1}{2\mathbf{x}^T(k)\hat{\mathbf{R}}^{-1}(k)\mathbf{x}(k)} \\
\mathbf{w}(k+1) &= \mathbf{w}(k) + 2\,\mu(k)\,e(k)\,\hat{\mathbf{R}}^{-1}(k)\mathbf{x}(k) \\
\hat{\mathbf{R}}^{-1}(k+1) &= \hat{\mathbf{R}}^{-1}(k) - 2\mu(k)\,(1-\mu(k))\,\hat{\mathbf{R}}^{-1}(k)\mathbf{x}(k)\mathbf{x}^T(k)\hat{\mathbf{R}}^{-1}(k)
\end{aligned} \tag{4.159}$$

(a) Apply this algorithm as well as the binormalized LMS algorithm to identify the system

$$H(z) = 1 + z^{-1} + z^{-2}$$

when the additional noise is a uniformly distributed white noise with variance $\sigma_n^2 = 0.01$, and the input signal is a Gaussian noise with unit variance filtered by an all-pole filter given by

$$G(z) = \frac{0.19z}{z - 0.9}$$

Through simulations, compare the convergence speed of the two algorithms when their misadjustments are approximately the same. The later condition can be met by choosing the μ in the binormalized LMS algorithm appropriately.