6

Complex Valued Adaptive Filters

This Chapter presents algorithms for the training of linear and nonlinear feedforward complex adaptive filters. Stochastic gradient learning algorithms are introduced for:

- linear transversal adaptive filters
- fully complex feedforward adaptive filters
- split-complex feedforward adaptive filters
- dual univariate adaptive filters

and are supported by convergence studies and simulations.

As adaptive filtering in the complex domain gives us more degrees of freedom than processing in the real domain, there are several equivalent formulations for the operation of such filters. For instance, the operation of the Complex Least Mean Square (CLMS) algorithm for transversal adaptive filters (adaptive linear combiner) in its original form [307] is given by

$$y = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k) = \mathbf{w}^{\mathrm{T}}(k)\mathbf{x}(k) \quad \rightarrow \quad \Delta\mathbf{w}(k) = \mu e(k)\mathbf{x}^{*}(k) \tag{6.1}$$

and two other frequently used forms are

$$y = \mathbf{w}^{\mathrm{H}}(k)\mathbf{x}(k) = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}^{*}(k) \quad \rightarrow \quad \Delta\mathbf{w}(k) = \mu e^{*}(k)\mathbf{x}(k) \tag{6.2}$$

and

$$y = \mathbf{x}^{\mathrm{H}}(k)\mathbf{w}(k) = \mathbf{w}^{\mathrm{T}}(k)\mathbf{x}^{*}(k) \quad \rightarrow \quad \Delta\mathbf{w}(k) = \mu e(k)\mathbf{x}(k) \tag{6.3}$$

The formulations (Equations 6.1–6.3) produce identical results and can be transformed from one into another by deterministic mappings. In the convergence analysis, the most convenient form will be used.

Complex Valued Nonlinear Adaptive Filters: Noncircularity, Widely Linear and Neural Models
Danilo Mandic and Vanessa Su Lee Goh

6.1 Adaptive Filtering Configurations

Figure 6.1 shows the block diagram of an adaptive filter as a closed loop system consisting of the filter architecture, which can be linear, nonlinear, feedforward or feedback, and the control algorithm. At every time instant k, the coefficients of the filter $\mathbf{w}(k)$ are adjusted based on the output from the control algorithm, thus providing a closed loop adaptation. The optimisation criterion within the control algorithm is a function of the instantaneous output error $e(k) = d(k) - y(k)$, where $d(k)$ is the desired response, $y(k)$ is the filter output, and $x(k)$ is the input signal.

The simplest adaptive filter is a feedforward linear combiner shown in Figure 6.2. The output of this adaptive finite impulse response (FIR) filter of length N is given by

$$y(k) = \sum_{n=1}^{N} w_n(k)x(k-n+1) = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k) \tag{6.4}$$

where $\mathbf{w}(k) = [w_1(k), \ldots, w_N(k)]^{\mathrm{T}}$, $\mathbf{x}(k) = [x(k), \ldots, x(k-N+1)]^{\mathrm{T}}$, and symbol $(\cdot)^{\mathrm{T}}$ denotes the vector transpose operator.

We distinguish between four basic adaptive filtering configurations:

- **System identification configuration, Figure 6.3(a).** In order to model the time varying parameters of an unknown system, the adaptive filter is connected in parallel to the system whose parameters are to be estimated. The unknown system and the adaptive filter share the same input $x(k)$, whereas the output of the unknown system serves as a desired response for the adaptive filter. This configuration is used typically in echo cancellation in acoustics and communications.
- **Noise cancellation configuration, Figure 6.3(b).** In this configuration, the noisy input $s(k) + N_0(k)$ serves as a 'desired' response, whereas the 'reference input' is an external noise source $N_1(k)$ which is correlated with the noise $N_0(k)$. The output of the filter $y(k)$

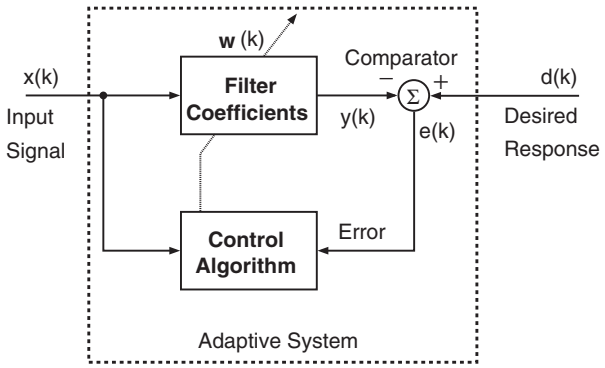


Figure 6.1 Block diagram of an adaptive filter as a closed loop system

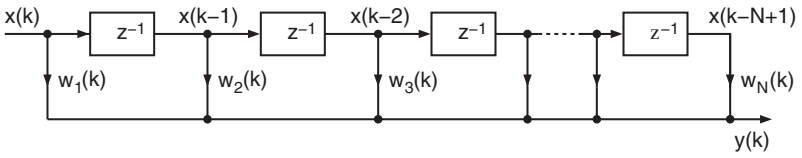


Figure 6.2 Linear adaptive filter

provides an estimate of the noise $\hat{N}_0(k)$, which is then subtracted from the primary input $s(k) + N_0(k)$ to produce a denoised signal $\hat{s}(k)$. This scheme is used in numerous noise cancellation applications in acoustics and biomedicine.

- **Inverse system modelling configuration, Figure 6.4(a).** The goal of inverse system modelling is to produce an estimate of the inverse of the transfer function of an unknown system. To achieve this, the adaptive filter is connected in series with the unknown system. The desired signal is the input signal delayed, due to signal propagation and the operation in discrete time. Typical applications of this scheme are in channel equalisation in digital communications and in control of industrial plants.
- **Adaptive prediction configuration, Figure 6.4(b).** In this configuration, the goal is to predict the value of the input signal M steps ahead, that is, to produce an estimate $\hat{x}(k + M)$, where M is the prediction horizon. The desired response (teaching signal) is the M steps ahead advanced version of the input signal. The range of applications of adaptive prediction are numerous, from quantitative finance to vehicle navigation systems.

Prediction is at the core of adaptive filtering and most of the simulations in this book will be conducted in the one step ahead adaptive prediction setting. In this case, the output of the linear adaptive filter from Figure 6.2 can be written as

$$y(k) = \sum_{n=1}^{N} w_n(k)x(k-n) = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k) \tag{6.5}$$

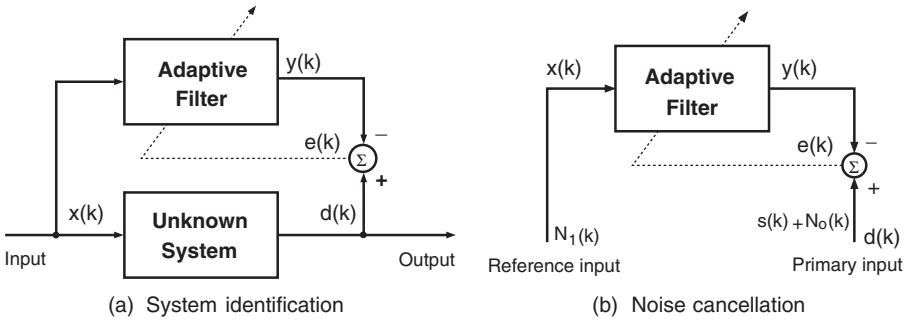


Figure 6.3 Adaptive filtering configurations

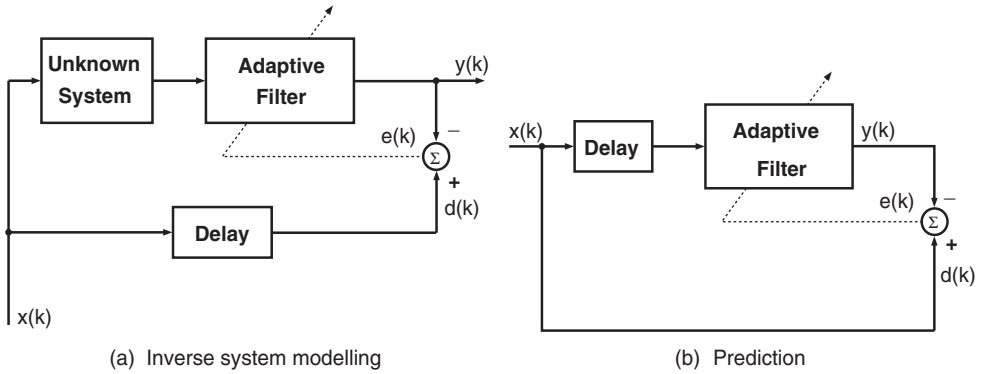


Figure 6.4 Adaptive filtering configurations

where the input in the filter memory (tap input vector) is $\mathbf{x}(k) = [x(k-1), \ldots, x(k-N)]^{\mathrm{T}}$ and the weight vector (vector of adaptive filter coefficients) is $\mathbf{w}(k) = [w_1(k), \ldots, w_N(k)]^{\mathrm{T}}$.

Wiener filter. Consider a linear feedforward filter with fixed coefficients $\mathbf{w}$, for which the optimisation task is to minimise the error criterion (cost function)

$$J(\mathbf{w}) = E\big[|e(k)|^2\big] = E\big[e(k)e^*(k)\big] = E\big[e_r^2(k) + e_i^2(k)\big] \tag{6.6}$$

This is a 'deterministic' function of the weight vector $\mathbf{w}$, and represents the estimated power of the output error. The aim is to find an 'optimal' vector with fixed coefficients $\mathbf{w}_{\mathrm{o}}$ which minimises $J(\mathbf{w})$. The Wiener filter provides a block solution which produces the best estimate of the desired signal in the mean square sense – for stationary input signals this solution is optimal in terms of second order statistics.

It is convenient to analyse the mean square error (Equation 6.6) using the description (Equation 6.2), where $e(k) = d(k) - \mathbf{w}^{\mathrm{H}}\mathbf{x}(k)$. Upon evaluating Equation (6.6), we have[1]

$$\begin{aligned} E\big[e(k)e^*(k)\big] &= E\left[\big(d(k) - \mathbf{w}^{\mathrm{H}}\mathbf{x}(k)\big)\big(d(k) - \mathbf{w}^{\mathrm{H}}\mathbf{x}(k)\big)^*\right] \\ &= E\big[d(k)d^*(k) - \mathbf{w}^{\mathrm{H}}\mathbf{x}(k)d^*(k) - \mathbf{w}^{\mathrm{T}}\mathbf{x}^*(k)d(k) + \mathbf{w}^{H}\mathbf{x}(k)\mathbf{w}^{T}\mathbf{x}^*(k)\big] \end{aligned} \tag{6.7}$$

Due to the linearity of the statistical expectation operator $E[\cdot]$, we have

$$\begin{aligned} E\big[e(k)e^*(k)\big] &= E\big[\,|d(k)|^2\,\big] - \mathbf{w}^{\mathrm{H}}E\big[\mathbf{x}(k)d^*(k)\big] - \mathbf{w}^{\mathrm{T}}E\big[\mathbf{x}^*(k)d(k)\big] + \mathbf{w}^{\mathrm{H}}E\big[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\big]\mathbf{w} \\ &= \sigma_d^2 - \mathbf{w}^{\mathrm{H}}\mathbf{p} - \mathbf{p}^{\mathrm{H}}\mathbf{w} + \mathbf{w}^{\mathrm{H}}\mathbf{R}\mathbf{w} \end{aligned} \tag{6.8}$$

where $\sigma_d^2 = E[|d(k)|^2]$ is the power of the desired response, $\mathbf{R} = E\big[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\big]$ denotes the complex valued input correlation matrix, $\mathbf{p} = E\big[d^*(k)\mathbf{x}(k)\big]$ is the crosscorrelation vector between the desired response and the input signal, subscripts $(\cdot)_r$ and $(\cdot)_i$ denote respectively the real and imaginary part of the complex number, $(\cdot)^*$ is the complex conjugation operator, and $(\cdot)^{\mathrm{H}}$ is the Hermitian transpose operator.

[1] Based on the identity $\mathbf{a}^{\mathrm{T}}\mathbf{b} = \mathbf{b}^{\mathrm{T}}\mathbf{a}$, we have $\mathbf{w}^{\mathrm{T}}\mathbf{x}^*(k) = \mathbf{x}^{\mathrm{H}}(k)\mathbf{w}$ and $\mathbf{w}^{\mathrm{T}}\mathbf{p}^* = \mathbf{p}^{\mathrm{H}}\mathbf{w}$.

We can solve for the optimal weight vector $\mathbf{w}_o$ by differentiating Equation (6.8) with respect to $\mathbf{w}$ and setting the result to zero, that is[2]

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = 2\mathbf{R}\mathbf{w} - 2\mathbf{p} = \mathbf{0} \tag{6.9}$$

which gives the Wiener–Hopf solution[3]

$$\mathbf{w}_o = \arg\min_{\mathbf{w}} J(\mathbf{w}) = \mathbf{R}^{-1}\mathbf{p} \tag{6.10}$$

The minimum achievable mean square error is calculated by replacing the optimal weight vector (Equation 6.10) back into Equation (6.8), to give [308]

$$J_{\min} = J(\mathbf{w}_o) = \sigma_d^2 - \mathbf{p}^{\mathrm{H}}\mathbf{R}^{-1}\mathbf{p} \tag{6.11}$$

To emphasise that the cost function (Equation 6.6) is quadratic in $\mathbf{w}$ and hence has a global minimum $J_{\min}$ for $\mathbf{w} = \mathbf{w}_o$, we can rewrite Equation (6.8) as

$$J(\mathbf{w}) = J_{\min} + (\mathbf{w} - \mathbf{w}_o)^{\mathrm{H}}\,\mathbf{R}\,(\mathbf{w} - \mathbf{w}_o) \tag{6.12}$$

Notice that the mean square error $J(\mathbf{w})$ comprises two terms – the minimum achievable mean square error $J_{\min}$ and a term that is quadratic in the weight error vector $\mathbf{v} = \mathbf{w} - \mathbf{w}_o$, which for the Wiener solution $\mathbf{w} = \mathbf{w}_o$ vanishes. The weight error vector $\mathbf{v}$ is also called the *misalignment vector*, and plays an important role in the analysis of adaptive filtering algorithms.

For nonstationary data the optimal weight vector is time varying, and hence the Wiener solution is suboptimal.

6.2 The Complex Least Mean Square Algorithm

Due to the block nature of the Wiener solution and the requirement of stationarity of the input, together with a possibly prohibitively large correlation matrix, this algorithm is not suitable for real world real time applications. One way to mitigate this problem is to make use of the parabolic shape of the 'error surface' defined by $J(\mathbf{w}) = E[|e(k)|^2]$. As the convexity of the error surface guarantees the existence of the solution, we can reach the optimal solution $\mathbf{w} = \mathbf{w}_o$ recursively, by performing 'steepest descent' towards the minimum $J_{\min} = J(\mathbf{w}_o)$, that is

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\nabla_{\mathbf{w}} J(\mathbf{w}) \tag{6.13}$$

where μ is the stepsize, a small positive constant.

[2]The aim is to minimise the output error power, hence the cost function is a real function of complex variable, and does not have a derivative directly in $\mathbb{C}$. We can, however use the $\mathbb{CR}$ calculus to find $\nabla_{\mathbf{w}} J$, as explained in Chapter 5.
[3]For stationary inputs the correlation matrix $\mathbf{R}$ is almost always positive semidefinite, that is, $\mathbf{u}^{\mathrm{H}}\mathbf{R}\mathbf{u} \geq 0$ for every nonzero $\mathbf{u}$, and therefore nonsingular and invertible.

The complex least mean square (CLMS) algorithm, introduced in 1975 [307], performs 'stochastic gradient descent', whereby all the statistical quantities in the Wiener filtering problem are replaced by their instantaneous estimates, to give[4]

$$\begin{aligned} E\left[|e^2(k)|\right] &\rightarrow \frac{1}{2}|e^2(k)| \\ E\left[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\right] &\rightarrow \mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k) \\ E\left[\mathbf{x}(k)d(k)\right] &\rightarrow \mathbf{x}(k)d(k) \end{aligned} \tag{6.14}$$

The 'stochastic' cost function

$$J(k) = \frac{1}{2}|e(k)|^2 \tag{6.15}$$

is now time varying, and based on Equation (6.13) the weight vector update can be expressed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\nabla_{\mathbf{w}} J(k)_{|\mathbf{w}=\mathbf{w}(k)} \tag{6.16}$$

The gradient of the cost function with respect to the complex valued weight vector $\mathbf{w}(k) = \mathbf{w}_r(k) + \jmath\mathbf{w}_i(k)$ can be expressed as[5]

$$\nabla_{\mathbf{w}} J(k) = \nabla_{\mathbf{w}_r} J(k) + \jmath\nabla_{\mathbf{w}_i} J(k) = \frac{\partial J(k)}{\partial \mathbf{w}_r(k)} + \jmath\frac{\partial J(k)}{\partial \mathbf{w}_i(k)} \tag{6.17}$$

As the output error of the linear adaptive filter in Figure 6.2 is given by

$$e(k) = d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)$$

the gradient with respect to the real part of the complex weight vector can be evaluated as

$$\nabla_{\mathbf{w}_r} J(k) = \frac{1}{2}\frac{\partial\left[e(k)e^*(k)\right]}{\partial \mathbf{w}_r(k)} = \frac{1}{2}e(k)\nabla_{\mathbf{w}_r} e^*(k) + \frac{1}{2}e^*(k)\nabla_{\mathbf{w}_r} e(k) \tag{6.18}$$

where

$$\nabla_{\mathbf{w}_r} e(k) = -\mathbf{x}(k) \qquad \nabla_{\mathbf{w}_r} e^*(k) = -\mathbf{x}^*(k)$$

The real part of the gradient of the cost function is then obtained as

$$\nabla_{\mathbf{w}_r} J(k) = -\frac{1}{2}e(k)\mathbf{x}^*(k) - \frac{1}{2}e^*(k)\mathbf{x}(k) \tag{6.19}$$

[4]For convenience, the cost function is scaled by $\frac{1}{2}$. This makes the weight updates easier to manipulate, and does not influence the result, as this factor will be cancelled after performing the derivative of the squared error term.

[5]We have already calculated this gradient in Section 5.4.4, using the $\mathbb{R}^*$-derivative – see Equation (5.42). In this Chapter, we provide a step by step derivation of CLMS – $\mathbb{CR}$ calculus will be used to simplify the derivation of learning algorithms for feedback filters in Chapters 7 and 15.

Similarly, the gradient of the cost function with respect to the imaginary part of the weight vector can be calculated as

$$\nabla_{\mathbf{w}_i} J(k) = \frac{J}{2} e(k)\mathbf{x}^*(k) - \frac{J}{2} e^*(k)\mathbf{x}(k) \tag{6.20}$$

By combining Equations (6.19) and (6.20) we obtain the gradient term in Equation (6.16) in the form

$$\nabla_{\mathbf{w}} J(k) = \nabla_{\mathbf{w}_r} J(k) + J\nabla_{\mathbf{w}_i} J(k) = -e(k)\mathbf{x}^*(k)$$

Finally, the stochastic gradient adaptation for the weight vector can be expressed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}^*(k), \qquad \mathbf{w}(0) = \mathbf{0} \tag{6.21}$$

This completes the derivation of the complex least mean square (CLMS) algorithm [307].

Two other frequently used formulations for the CLMS are

$$y = \mathbf{w}^{\mathrm{H}}(k)\mathbf{x}(k) \quad \rightarrow \quad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e^*(k)\mathbf{x}(k) \tag{6.22}$$

and

$$y = \mathbf{x}^{\mathrm{H}}(k)\mathbf{w}(k) \quad \rightarrow \quad \mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k) \tag{6.23}$$

The above three formulations for CLMS are equivalent; forms (6.22) and (6.23) are sometimes more convenient for matrix manipulation.

6.2.1 Convergence of the CLMS Algorithm

To illustrate that for stationary signals the CLMS converges to the optimal Wiener solution, consider the expected value for the CLMS update

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu E\left[e(k)\mathbf{x}^*(k)\right] \tag{6.24}$$

which has converged when $\mathbf{w}(k+1) = \mathbf{w}(k) = \mathbf{w}(\infty)$, that is, when the iteration (6.24) reaches its fixed point[6] [199, 256], and the weight update $\Delta\mathbf{w}(k) = \mu E[e(k)\mathbf{x}^*(k)] = \mathbf{0}$. This is achieved for[7]

$$\mathbf{0} = E\left[d^*(k)\mathbf{x}(k)\right] - E\left[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\right]\mathbf{w}(k) \quad \Longleftrightarrow \quad \mathbf{R}\mathbf{w} = \mathbf{p} \tag{6.25}$$

that is, for the Wiener solution. The condition $E[e(k)\mathbf{x}^*(k)] = \mathbf{0}$ is called the 'orthogonality condition' and states that the output error of the filter and the tap input vector are orthogonal ($e \perp \mathbf{x}$) when the filter has converged to the optimal solution.

[6] For more detail see Appendix P, Appendix O, and Appendix N.

[7] Upon applying the complex conjugation operator and setting $\mathbf{w}^* = \mathbf{w}$.

It is of greater interest, however, to analyse the evolution of the weights in time. As with any other estimation problem, we need to analyse the 'bias' and 'variance' of the estimator, that is:

- Convergence in the mean, to ascertain whether $\mathbf{w}(k) \to \mathbf{w}_o$ when $k \to \infty$;
- Convergence in the mean square, in order to establish whether the variance of the weight error vector $\mathbf{v}(k) = \mathbf{w}(k) - \mathbf{w}_o(k)$ approaches $J_{\min}$ as $k \to \infty$.

The analysis of convergence of linear adaptive filters is made mathematically tractable if we use so called *independence assumptions*, such that the filter coefficients are statistically independent of the data currently in filter memory, and $\{d(l), x(l)\}$ is independent of $\{d(k), x(k)\}$ for $k \neq l$.

Convergence in the mean. For convenience, the analysis will be based on the form (6.23). We can assume without loss in generality that the desired response

$$d(k) = \mathbf{x}^{\mathrm{H}}(k)\mathbf{w}_o + q(k) \tag{6.26}$$

where $q(k)$ is complex white Gaussian noise, with zero mean and variance σ_q^2, which is uncorrelated with $\mathbf{x}(k)$. Then, we have

$$\begin{aligned} e(k) &= \mathbf{x}^{\mathrm{H}}(k)\mathbf{w}_o + q(k) - \mathbf{x}^{\mathrm{H}}(k)\mathbf{w}(k) \\ \mathbf{w}(k+1) &= \mathbf{w}(k) + \mu\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}\mathbf{w}_o - \mu\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\mathbf{w}(k) + \mu q(k)\mathbf{x}(k) \end{aligned}$$

Subtracting the optimal weight vector $\mathbf{w}_o$ from both sides of the last equation, the weight error vector $\mathbf{v}(k) = \mathbf{w}(k) - \mathbf{w}_o$ can be expressed as[8]

$$\mathbf{v}(k+1) = \mathbf{v}(k) - \mu\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\mathbf{v}(k) + \mu q(k)\mathbf{x}(k) \tag{6.27}$$

Applying the statistical expectation operator to both sides of Equation (6.27) and employing the independence assumptions, we have

$$E[\mathbf{v}(k+1)] = \big(\mathbf{I} - \mu E[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)]\big)E[\mathbf{v}(k)] + \mu E[q(k)\mathbf{x}(k)] = \big(\mathbf{I} - \mu\mathbf{R}\big)E[\mathbf{v}(k)] \tag{6.28}$$

Unless the correlation matrix $\mathbf{R}$ is diagonal, there will be cross–coupling between the coefficients of the weight error vector. Since $\mathbf{R}$ is Hermitian and positive semidefinite, it can be rotated into a diagonal matrix by a unitary transformation[9]

$$\mathbf{R} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{\mathrm{H}} \tag{6.29}$$

where $\mathbf{\Lambda} = diag\,(\lambda_1, \lambda_2, \ldots, \lambda_N)$ is a diagonal matrix comprising the real and positive eigenvalues of the correlation matrix, $\mathbf{Q}$ is the matrix of the corresponding eigenvectors, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$.

[8]The same result can be obtained from the original formulation (6.21) where $d(k) = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}_o + q(k)$, based on $\Delta\mathbf{w}(k) = \mu\mathbf{x}^*(k)e(k)$, and using the identity $\mathbf{x}^*(k)\mathbf{x}^{\mathrm{T}}(k) = \mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)$.

[9]The eigenvectors may be chosen to be orthonormal in which case $\mathbf{Q}$ is unitary.

Rotating the weight error vector $\mathbf{v}(k)$ by the eigenmatrix $\mathbf{Q}$, that is, $\mathbf{v}'(k) = \mathbf{Q}\mathbf{v}(k)$, decouples the evolution of its coefficients. This rotation allows us to express the so called 'modes of convergence' solely in terms of the corresponding eigenvalues of the correlation matrix[10]

$$\mathbf{v}'(k+1) = (\mathbf{I} - \mu\mathbf{\Lambda})\,\mathbf{v}'(k) \tag{6.30}$$

Since $(\mathbf{I} - \mu\mathbf{\Lambda})$ is diagonal and the nth component of $\mathbf{v}'$ represents the projection of the vector $\mathbf{v}(k)$ onto the nth eigenvector of $\mathbf{R}$, every element of $\mathbf{v}'(k)$ evolves independently, and Equation (6.30) converges to zero if $|\mathbf{I} - \mu\lambda_n| < 1$. As the fastest mode of convergence corresponds to the maximum eigenvalue $\lambda_{\max}$, the condition for the convergence in the mean of the CLMS algorithm becomes[11]

$$0 < \mu < \frac{2}{\lambda_{\max}} \approx \frac{2}{\text{tr}[\mathbf{R}]} \tag{6.31}$$

The trace of the input correlation matrix is equal to the product of the filter length and input signal power, and so an easier to estimate bound on the learning rate is given by

$$0 < \mu < \frac{2}{NE\left[|x(k)|^2|\right]} \tag{6.32}$$

Convergence in the mean square. To evaluate the mean square error, the CLMS must converge in the mean, and hence its learning rate must obey $0 < \mu < 2/\lambda_{\max}$. As the filter coefficients begin to converge in the mean, they start fluctuating around their optimum values, defined by $\mathbf{w}_{\text{o}}$. This is due to the 'stochastic' gradient approximation used for the update of $\mathbf{w}(k)$, that is, the use of instantaneous estimates for the statistical moments within CLMS, as shown in Equation (6.14). As a result, the mean square error $\xi(k) = E[|e(k)|^2]$ exceeds the minimum mean square error $J_{\min}$ (6.11) by an amount referred to as the *excess mean square error*, denoted by $\xi_{\text{EMSE}}(k)$, that is [110]

$$\xi(k) = J_{\min} + \xi_{\text{EMSE}}(k) \tag{6.33}$$

The excess mean square error depends on second-order statistical properties of the desired response, input, and weight error vector. The plot showing time evolution of the mean square error is called the *learning curve*.

For convergence, it is of principal importance to preserve the asymptotic boundedness[12] of the mean square error $\xi(k)$. Based on Equation (6.12) and the signal model (6.26), the minimum mean square error is $J_{\min} = \sigma_q^2$, which gives[13]

$$\xi(k) = \sigma_q^2 + E\left[\mathbf{v}^{\text{H}}(k)\mathbf{R}\mathbf{v}(k)\right] = \sigma_q^2 + \text{tr}\left[\mathbf{R}\mathbf{K}(k)\right] \tag{6.34}$$

[10]Modes of convergence are defined as $v^n(k+1) = (1 - \mu\lambda_n)v^n(k)$, $n = 1, \ldots, N$. As $v^n(k+1) = (1 - \lambda_n)^k v^n(0)$, we require $|1 - \mu\lambda_n| < 1$.

[11]Using the identity $\lambda_{\max} \leq \sum$(diagonal elements of $\mathbf{R}$) = tr[$\mathbf{R}$].

[12]For more detail on asymptotic stability, see Appendix O.

[13]Using the identity $E[\mathbf{v}^{\text{H}}(k)\mathbf{R}\mathbf{v}(k)] = \text{tr}[\mathbf{R}\mathbf{K}(k)] = \text{tr}[\mathbf{K}(k)\mathbf{R}]$.

where the weight error correlation matrix $\mathbf{K}(k) = E[\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k)]$ and $\xi_{\mathrm{EMSE}}(k) = \mathrm{tr}[\mathbf{R}\mathbf{K}(k)]$. Since the correlation matrix $\mathbf{R}$ is bounded, the CLMS will converge in the mean square if the elements of $\mathbf{K}(k)$ remain bounded as $k \to \infty$. To analyse $\mathbf{K}(k)$ we can multiply Equation (6.27) by $\mathbf{v}^{H}(k)$, to give

$$\begin{aligned}\mathbf{v}(k+1)\mathbf{v}^{\mathrm{H}}(k+1) = \mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k) - \mu\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k) - \mu\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k)\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k) \\ + \mu^2 q(k)q^*(k)\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k) + \mu^2\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k)\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k) + xt(k)\end{aligned} \tag{6.35}$$

where $xt(k)$ comprises the crossterms. Owing to the independence assumptions, the cross-terms vanish upon the application of the statistical expectation operator, and

$$\begin{aligned}E\left[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k)\right] &= \mathbf{R}\mathbf{K}(k) \\ E\left[\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k)\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\right] &= \mathbf{K}(k)\mathbf{R} \\ E\left[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\mathbf{v}(k)\mathbf{v}^{\mathrm{H}}(k)\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\right] &= \mathbf{R}\mathbf{K}(k)\mathbf{R} + \mathbf{R}\mathrm{tr}[\mathbf{R}\mathbf{K}(k)] \\ E\left[|q(k)|^2\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)\right] &= \sigma_q^2\mathbf{R}\end{aligned} \tag{6.36}$$

to yield[14]

$$\mathbf{K}(k+1) = \mathbf{K}(k) - \mu\Big(\mathbf{R}\mathbf{K}(k) + \mathbf{K}(k)\mathbf{R}\Big) + \mu^2\Big(\mathbf{R}\mathbf{K}(k) + \mathrm{tr}[\mathbf{R}\mathbf{K}(k)]\Big)\mathbf{R} + \mu^2\sigma_q^2\mathbf{R} \tag{6.37}$$

Evaluation of the excess mean square error based on Equation (6.37) is rather mathematically demanding, however, similarly to the analysis of convergence in the mean, since $\mathbf{R}$ is Hermitian and positive semidefinite, it can be rotated into a diagonal matrix by a unitary transformation[15] $\mathbf{R} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-1}$. Then, since $\mathbf{v}'(k) = \mathbf{Q}\mathbf{v}(k)$, the rotated weight error correlation matrix becomes

$$\widetilde{\mathbf{K}}(k) = \mathbf{Q}\mathbf{K}(k)\mathbf{Q}^{\mathrm{H}} \tag{6.38}$$

To simplify the analysis of Equation (6.37), consider a white iid[16] input for which $\mathbf{R} = \sigma_x^2\mathbf{I}$, to yield

$$\widetilde{\mathbf{K}}(k+1) = \left(1 - \mu\sigma_x^2\right)^2\widetilde{\mathbf{K}}(k) + \mu^2\sigma_x^4\mathrm{tr}[\widetilde{\mathbf{K}}(k)]\mathbf{I} + \mu^2\sigma_q^2\sigma_x^2\mathbf{I} \tag{6.39}$$

Since $\widetilde{\mathbf{K}}(k)$ is a correlation matrix, for every element $\kappa_{mn}(k) \in \widetilde{\mathbf{K}}(k)$, we have

$$|\kappa_{mn}(k)|^2 \leq \kappa_{mm}(k)\kappa_{nn}(k) \tag{6.40}$$

[14]We use the property that for zero mean, complex, jointly Gaussian x_1, x_2, x_3, x_4, the Gaussian Moment Factoring Theorem states that $E\left[x_1x_2^{\mathrm{H}}x_3x_4^{\mathrm{H}}\right] = x_1x_2^{\mathrm{H}} \cdot x_3x_4^{\mathrm{H}} + x_1x_4^{\mathrm{H}} \cdot x_2^{\mathrm{H}}x_3$. For a detailed analysis, we refer to [62, 76, 127].
[15]Here, $\mathbf{\Lambda} = \mathrm{diag}\,(\lambda_1, \lambda_2, \ldots, \lambda_N)$ is a diagonal matrix comprising the real and positive eigenvalues of the correlation matrix, $\mathbf{Q}$ is the matrix of the corresponding eigenvectors, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$.
[16]Independent identically distributed.

Thus, for convergence of Equation (6.39) it is sufficient to look only at the evolution of the diagonal elements of $\widetilde{\mathbf{K}}(k)$; the recursion for the calculation of these coefficients is provided in the classic result by Horowitz and Senne [127]. Consider a vector comprising the diagonal elements of $\widetilde{\mathbf{K}}(k)$, given by

$$\mathbf{s}(k) = [\kappa_{11}(k), \ldots, \kappa_{NN}(k)]^{\mathrm{T}}$$

Then, from Equation (6.39) we have

$$\mathbf{s}(k+1) = \left[\left(1-\mu\sigma_x^2\right)^2\mathbf{I} + \mu^2\sigma_x^4\mathbf{1}\mathbf{1}^{\mathrm{T}}\right]\mathbf{s}(k) + \mu^2\sigma_q^2\sigma_x^2\mathbf{1} \tag{6.41}$$

where $\mathbf{1}$ is an $N \times 1$ vector of ones. In the steady state, $\mathbf{s}(k+1) = \mathbf{s}(k) = \mathbf{s}(\infty)$, and Equation (6.41) can be expressed as

$$\mathbf{s}(\infty) = \frac{\mu\sigma_q^2\mathbf{1}}{2\mathbf{I} - \mu\sigma_x^2\left(\mathbf{I} + \mathbf{1}\mathbf{1}^{\mathrm{T}}\right)} \tag{6.42}$$

We can use the matrix inversion lemma[17] [110] to find the inverse of the denominator of Equation (6.42), to give

$$\kappa_{pp}(\infty) = \mu\sigma_q^2 \frac{\frac{1}{2-\mu\sigma_x^2}}{1-\mu\sum_{n=1}^{N}\frac{\sigma_x^2}{2-\mu\sigma_x^2}} \qquad p = 1, \ldots, N \tag{6.43}$$

The steady state excess mean square error $\xi_{\mathrm{EMSE}}(\infty) = \mathrm{tr}[\mathbf{R}\mathbf{K}(\infty)] = \mathrm{tr}[\mathbf{K}(\infty)\mathbf{R}]$ now becomes

$$\xi_{\mathrm{EMSE}}(\infty) = \sigma_x^2\mathbf{s}^{\mathrm{T}}(\infty)\mathbf{1} = \sigma_q^2\frac{\sum_{n=1}^{N}\frac{\mu\sigma_x^2}{2-\mu\sigma_x^2}}{1-\sum_{n=1}^{N}\frac{\mu\sigma_x^2}{2-\mu\sigma_x^2}} \tag{6.44}$$

It is convenient to assess the performance of adaptive filters in terms of the misadjustment

$$\mathcal{M} = \frac{\xi_{\mathrm{EMSE}}(\infty)}{\xi_{\min}} = \frac{\xi_{\mathrm{EMSE}}(\infty)}{\sigma_q^2} \tag{6.45}$$

From Equations (6.34) and (6.43), for small μ the expression for misadjustment simplifies into

$$\mathcal{M} = \mu\frac{\frac{1}{2}\mathrm{tr}[\mathbf{R}]}{1-\frac{1}{2}\mu\mathrm{tr}[\mathbf{R}]} = \mu\frac{\sigma_x^2 N}{2-\mu\sigma_x^2 N} \approx \frac{1}{2}\mu\sigma_x^2 N \tag{6.46}$$

[17] The form of the matrix inversion lemma used states that for a positive definite $N \times N$ matrix $\mathbf{A}$, scalar a, and $N \times 1$ vector $\mathbf{a}$, we have

$$\left(\mathbf{A} + a\mathbf{a}\mathbf{a}^{\mathrm{H}}\right)^{-1} = \mathbf{A}^{-1} - \frac{a\mathbf{A}^{-1}\mathbf{a}\mathbf{a}^{\mathrm{H}}\mathbf{A}^{-1}}{1 + a\mathbf{a}^{\mathrm{H}}\mathbf{A}^{-1}\mathbf{a}}$$

For convergence in the mean square, the misadjustment must remain bounded and positive, that is $1 - \frac{1}{2}\mu \text{tr}[\mathbf{R}] > 0$. Thus, the mean square error $\xi(k)$ converges asymptotically to $\xi(\infty) = J_{\min} = \sigma_q^2$ for

$$0 < \mu < \frac{2}{\text{tr}[\mathbf{R}]} = \frac{2}{\sigma_x^2 N} \tag{6.47}$$

For mathematical tractability, the above bound on the learning rate has been derived for a white iid input. In practical applications this bound is considerably lower, and depends on the condition number of the input correlation matrix $\nu = \lambda_{\max}/\lambda_{\min}$, or equivalently on the flatness of the power spectrum of the input.

From Equation (6.46), as the misadjustment is directly proportional to the stepsize μ, the requirements of fast convergence and good steady state properties of an adaptive filtering algorithm are contradictory. An algorithm will have fast initial convergence for a large μ, whereas it will converge to the optimal solution for a small μ. For the same learning rate μ, shorter filters will exhibit lower misadjustment. Chapter 8 introduces adaptive filters with variable learning rates, which can cope better with the requirements of fast initial convergence and low misadjustment.

6.3 Nonlinear Feedforward Complex Adaptive Filters

Following on the derivation of the CLMS algorithm, we shall now introduce stochastic gradient learning algorithms for training feedforward nonlinear filters, for which a generic block diagram is shown in Figure 6.5. The nonlinearity $\Phi(\cdot)$ can be from any class addressed in Chapter 4.

To simplify the notation, the output of the filter in Figure 6.5, can be expressed as

$$\begin{aligned} y(k) = \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)\right) \rightarrow \Phi(k) = u\left(\sigma(k), \tau(k)\right) + \jmath v\left(\sigma(k), \tau(k)\right) = u(k) + \jmath v(k) \\ \mathbf{x}^{T}(k)\mathbf{w}(k) \rightarrow \sigma(k) + \jmath\tau(k) \end{aligned} \tag{6.48}$$

6.3.1 Fully Complex Nonlinear Adaptive Filters

As fully complex nonlinearities are complex functions of complex variables, the Complex Nonlinear Gradient Descent (CNGD) algorithm for training this class of filters can be derived similarly to the derivation of CLMS. If $\Phi(k)$ is analytic on a region in $\mathbb{C}$, then based on the

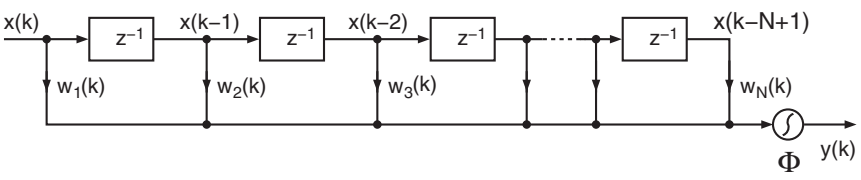


Figure 6.5 Nonlinear adaptive filter

cost function

$$J(k) = \frac{1}{2}|e(k)|^2 \tag{6.49}$$

we have

$$\nabla_{\mathbf{w}} J(k) = \frac{\partial J(k)}{\partial \mathbf{w}_r(k)} + \jmath \frac{\partial J(k)}{\partial \mathbf{w}_i(k)} = \nabla_{\mathbf{w}_r} J(k) + \jmath \nabla_{\mathbf{w}_i} J(k)$$

where

$$\begin{aligned} \nabla_{\mathbf{w}_r} J(k) &= \frac{1}{2} e(k) \nabla_{\mathbf{w}_r} e^*(k) + \frac{1}{2} e^*(k) \nabla_{\mathbf{w}_r} e(k) \\ \nabla_{\mathbf{w}_i} J(k) &= \frac{1}{2} e(k) \nabla_{\mathbf{w}_i} e^*(k) + \frac{1}{2} e^*(k) \nabla_{\mathbf{w}_i} e(k) \end{aligned} \tag{6.50}$$

Since the output error of the linear adaptive filter in Figure 6.2 can be expressed as

$$e(k) = d(k) - \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)\right)$$

and the partial derivatives for the 'net input' $net(k) = \mathbf{x}^T(k)\mathbf{w}(k)$ from Equation (6.48) are

$$\begin{aligned} \frac{\partial \sigma(k)}{\partial \mathbf{w}_r(k)} &= \mathbf{x}_r(k) \quad \frac{\partial \sigma(k)}{\partial \mathbf{w}_i(k)} = -\mathbf{x}_i(k) \\ \frac{\partial \tau(k)}{\partial \mathbf{w}_r(k)} &= \mathbf{x}_i(k) \quad \frac{\partial \tau(k)}{\partial \mathbf{w}_i(k)} = \mathbf{x}_r(k) \end{aligned} \tag{6.51}$$

Thus, the gradients with respect to the real and parts of the weight vector in Equation (6.50) become

$$\frac{\partial J(k)}{\partial \mathbf{w}_r(k)} = -e_r(k)\left[u_\sigma(k)\mathbf{x}_r(k) + u_\tau(k)\mathbf{x}_i(k)\right] - e_i(k)\left[v_\sigma(k)\mathbf{x}_r(k) + v_\tau(k)\mathbf{x}_i(k)\right] \tag{6.52}$$

$$\frac{\partial J(k)}{\partial \mathbf{w}_i(k)} = -e_r(k)\left[-u_\sigma(k)\mathbf{x}_i(k) + u_\tau(k)\mathbf{x}_r(k)\right] - e_i(k)\left[-v_\sigma(k)\mathbf{x}_i(k) + v_\tau(k)\mathbf{x}_r(k)\right] \tag{6.53}$$

where $u_\sigma = \partial u/\partial\sigma$, $u_\tau = \partial u/\partial\tau$, $v_\sigma = \partial v/\partial\sigma$, $v_\tau = \partial v/\partial\tau$ exist and are bounded.

By employing the Cauchy–Riemann equations (see Chapter 5), we have

$$u_\sigma(k) = v_\tau(k) \qquad u_\tau(k) = -v_\sigma(k) \tag{6.54}$$

and the gradient $\nabla_{\mathbf{w}} J(k) = \nabla_{\mathbf{w}_r} J(k) + \jmath \nabla_{\mathbf{w}_i} J(k)$ simplifies into

$$\begin{aligned} \nabla_{\mathbf{w}} J(k) &= -\mathbf{x}^*(k)\Big[e_r(k)\big(u_\sigma(k) - \jmath v_\sigma(k)\big) + e_i(k)\big(v_\sigma(k) + \jmath u_\sigma(k)\big)\Big] \\ &= -\mathbf{x}^*(k)\big[\Phi'^*(k)e_r(k) + \jmath\Phi'^*(k)e_i(k)\big] \\ &= -\mathbf{x}^*(k)\Phi'^*(k)e(k) \end{aligned} \tag{6.55}$$

where, for convenience, $\Phi'(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)) = \Phi'(k)$.

Finally, the weight update for the complex nonlinear gradient descent (CNGD) algorithm for training fully complex nonlinear adaptive filters is given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\Phi'^{*}(k)\mathbf{x}^{*}(k), \qquad \mathbf{w}(0) = \mathbf{0} \tag{6.56}$$

The nonlinear nature of the filter is reflected in the first derivative of a fully complex nonlinearity within the update. This weight update has the same generic form as CLMS (Equation 6.21) and hence it degenerates into CLMS for a linear Φ. Properties of the class of fully complex nonlinear functions are given in Chapter 4 and Appendix E, whereas their representation as Möbius transformations is addressed in Chapter 11.

6.3.2 Derivation of CNGD using $\mathbb{CR}$ calculus

The complex nonlinear gradient descent algorithm can be alternatively derived using the $\mathbb{CR}$ calculus. From Equation (5.40), the gradient of the cost function is calculated along the conjugate direction of the weights, that is $\nabla_{\mathbf{w}} J = 2\partial J(k)/\partial \mathbf{w}^{*}(k)$.

For most nonlinear activation functions used in this work[18]

$$\partial\Phi^{*}/\partial net = \partial\Phi/\partial net^{*} = \Phi^{'*} \qquad \partial net^{*}(k)/\partial \mathbf{w}^{*}(k) = \mathbf{x}^{*}(k) \tag{6.57}$$

Thus $\nabla_{\mathbf{w}} J(k) = -e(k)\Phi'^{*}(k)\mathbf{x}^{*}(k)$ and we obtain the algorithm (6.56).

Convergence of feedforward nonlinear adaptive filters. Because of the effects of nonlinearity, it is difficult to derive directly the conditions for convergence of fully complex adaptive feedforward filters. However, for a contractive Φ, an approximate analysis can be conducted based on the contraction mapping theorem (see Appendix P) [190], and the convergence analysis in Section 6.2.1. We shall next analyse the mean weight vector convergence for fully complex feedforward adaptive filters.

We can express, without loss in generality, the desired response as

$$d(k) = \Phi(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}_{\mathrm{o}}) + q(k) \tag{6.58}$$

where $q(k)$ is complex Gaussian noise with variance σ_q^2, uncorrelated with $\mathbf{x}(k)$. The instantaneous output error and the weight update can now be expressed as

$$\begin{aligned} e(k) &= \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}_{\mathrm{o}}\right) + q(k) - \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)\right) \\ \mathbf{w}(k+1) &= \mathbf{w}(k) + \mu q(k)\Phi'^{*}(k)\mathbf{x}^{*}(k) - \mu\Phi'^{*}(k)\mathbf{x}^{*}(k)\left[\Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)\right) - \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}_{\mathrm{o}}\right)\right] \end{aligned} \tag{6.59}$$

If Φ is a contraction,[19] then the term in the square brackets can be approximated as (recall that $\mathbb{C}$ is not an ordered field – see Appendix A)

$$|\Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)\right) - \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}_{\mathrm{o}}\right)| \leq \gamma|\mathbf{x}^{\mathrm{T}}(k)(\mathbf{w}(k) - \mathbf{w}_{\mathrm{o}})| = \beta|\mathbf{x}^{\mathrm{T}}(k)\mathbf{v}(k)| \tag{6.60}$$

[18] In addition, for the class of elementary transcendental functions, which are typical fully complex nonlinear activation functions used in this work, we have $(\Phi')^{*} = (\Phi^{*})' = \Phi'^{*}$.

[19] By the contraction mapping theorem $|\Phi(b) - \Phi(a)| \leq \gamma|b - a|,\ \gamma < 1,\ a, b \in S \subset \mathbb{C}$ (see Appendix P).

Subtract the optimal weight vector $\mathbf{w}_o$ from both sides of Equation (6.59) and for simplicity ignore the modulus in Equation (6.60) to obtain the recursion for the weight error vector

$$\mathbf{v}(k+1) = \mathbf{v}(k) - \mu\beta\Phi'^{*}(k)\mathbf{x}^{*}(k)\mathbf{x}^{\mathrm{T}}(k)\mathbf{v}(k) + \mu q(k)\Phi'^{*}(k)\mathbf{x}^{*}(k) \tag{6.61}$$

For Φ a contraction, its first derivative $|\Phi'(k)| < 1$ and can be replaced by a constant $\gamma < 1$. Upon applying the statistical expectation operator and using the independence assumptions $(q \perp \mathbf{x})$

$$E\big[|\mathbf{v}(k+1)|\big] \leq \big|\mathbf{I} - \mu\beta\gamma E[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)]\big| E\big[|\mathbf{v}(k)|\big] \tag{6.62}$$

Similarly to Equation (6.32), the modes of convergence for Equation (6.62) are dominated by the largest eigenvalue of $\mathbf{R} = E[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)]$, and for a white iid input the bound on the learning rate which preserves convergence in the mean is

$$0 < \mu < \frac{2}{\alpha \parallel \mathbf{x}(k) \parallel_2^2} \tag{6.63}$$

where α is a positive parameter derived from the first derivative of Φ.

6.3.3 Split-complex Approach

This class of filters has the same general architecture as fully complex feedforward adaptive filters shown in Figure 6.5. As has been shown in Chapter 4, we differentiate between the real–imaginary and amplitude–phase split-complex approaches. The former is suitable for signals which exhibit symmetry around the real and imaginary axes, and the latter is best suited for rotational processes.

Real–imaginary split-complex approach (RISC). Split-complex nonlinear activation functions were originally introduced for complex neural networks employed for binary classification [166] and nonlinear equalisation in communications [27]. The output of a feedforward nonlinear adaptive filter with a RISC nonlinearity is given by

$$y(k) = \Phi(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)) = \Phi\big(net(k)\big) = \sigma\big(net_r(k)\big) + \jmath\sigma\big(net_i(k)\big) \tag{6.64}$$

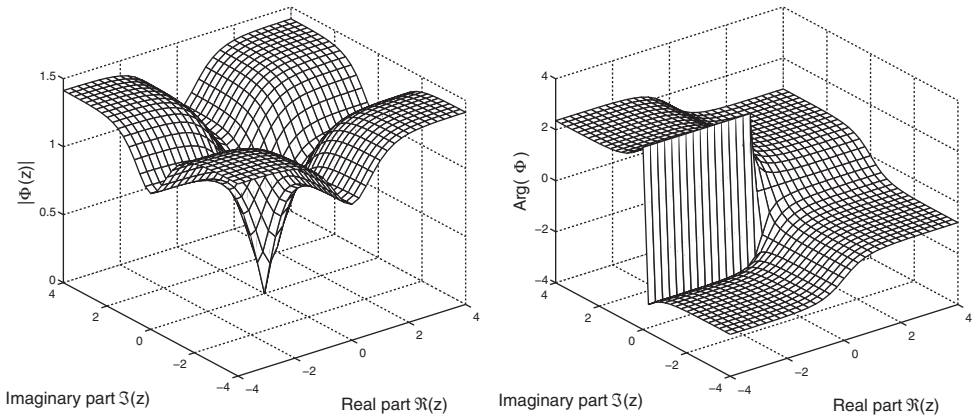
where the real and imaginary parts of the complex net input $net(k)$ are processed separately by real valued sigmoid functions σ. Figure 6.6 shows one such nonlinearity – a hyperbolic tangent split complex activation function.

As the gradients with respect to the real and imaginary parts of the output are calculated independently – in the same way as in real valued nonlinear adaptive filters, the weight update in the RISC approach becomes

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu\Big(e_r(k)\sigma'\big(net_r(k)\big) + \jmath e_i(k)\sigma'\big(net_i(k)\big)\Big)\mathbf{x}^{*}(k) \tag{6.65}$$

Amplitude–phase split-complex approach (APSC). Examples include filters with the output nonlinearity proposed by Georgiou and Koutsougeras [88], given by

$$\Phi(z) = \frac{z}{c + |z|/r} \tag{6.66}$$



(a) Magnitude (b) Phase

Figure 6.6 Real–imaginary split-complex function $\Phi(z) = \tanh(u) + \jmath \tanh(v)$

and filters based on the output nonlinearity introduced by Hirose [118], given by

$$\Phi(z) = \tanh(|z|/m)e^{\jmath \arg(z)} \tag{6.67}$$

where c, r, and m are real positive constants. The filter output for the class of APSC nonlinearities has a general form

$$y(k) = \Phi\left(\mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)\right) = \Phi\left(net(k)\right) = \sigma\left(|net(k)|\right)e^{\jmath \arg(net(k))} \tag{6.68}$$

for which the update becomes (for $e^{\jmath\varphi} = e^{\jmath \arg(net(k))}$)

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu \mathbf{x}^*(k)e^{\jmath\varphi}\left[\Re\left\{e(k)\sigma'(|net(k)|)e^{\jmath\varphi}\right\} + \jmath\Im\left\{\tfrac{1}{|net(k)|}e(k)\sigma(|net(k)|)e^{\jmath\varphi}\right\}\right] \tag{6.69}$$

where the symbols $\Re\{\cdot\}$ and $\Im\{\cdot\}$ denote respectively the real and imaginary part of a complex number.

Boundendess vs differentiability. The output of split-complex nonlinear adaptive filters is bounded by virtue of the saturation type real valued nonlinearities within the activation functions. However, split-complex functions are not differentiable in the complex sense, and hence the stochastic gradient learning algorithms for RISC and APSC adaptive filters do not have the same generic form as CLMS.

6.3.4 Dual Univariate Adaptive Filtering Approach (DUAF)

The dual univariate approach deals with complex data by splitting the input signal into its real and imaginary parts and treating them as independent real valued quantities [156, 213], as shown in Figure 6.7. Dual univariate filters therefore perform suboptimally when dealing with complex valued signals with rich nonlinear behaviour and coupling between the real and

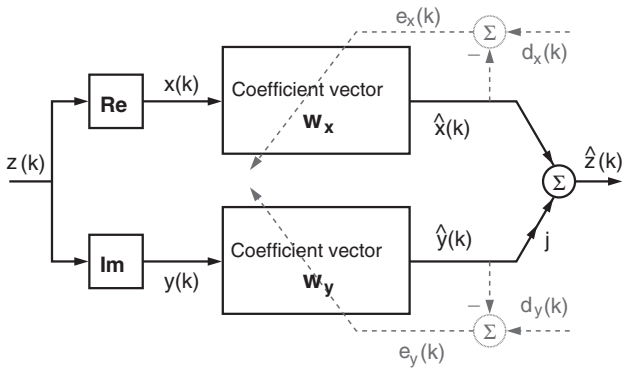


Figure 6.7 Dual univariate real adaptive filter

imaginary parts. However, they are simple and fast, and can be used for processes for which the real and imaginary part are not heavily correlated.

Any real valued adaptive filter can be used for the x and y channel, these can be linear, nonlinear, feedforward or recurrent [190], and the two channels do not have to share the same filter. In the simplest case, the weight vectors $\mathbf{w}_x(k)$ and $\mathbf{w}_y(k)$ are updated using standard LMS, to give

$$\begin{aligned} \mathbf{w}_x(k+1) &= \mathbf{w}_x(k) + \mu e_x(k)\mathbf{x}(k) \\ \mathbf{w}_y(k+1) &= \mathbf{w}_y(k) + \mu e_y(k)\mathbf{y}(k) \end{aligned} \tag{6.70}$$

where the corresponding instantaneous output errors are

$$e_x(k) = d_x(k) - \hat{x}(k) \qquad e_y(k) = d_y(k) - \hat{y}(k) \tag{6.71}$$

6.4 Normalisation of Learning Algorithms

The weight update $\Delta\mathbf{w}(k)$ within stochastic gradient learning algorithms is proportional to the input vector $\mathbf{x}(k)$, which causes gradient noise amplification for large magnitudes of the input. For instance, within the CLMS algorithm, we have $\Delta\mathbf{w}(k) = \mu e(k)\mathbf{x}^*(k)$. Learning algorithms can be made independent of the input signal power by normalising their updates, as shown below.

Normalised CLMS (NCLMS). Observe from Equation (6.32) that for the convergence of CLMS in the mean square, the bound on the stepsize is given by

$$0 < \mu < \frac{2}{\sigma_x^2 N}$$

and that the input signal power σ_x^2 can be estimated as[20]

$$\sigma_x^2 = E\left[|x(k)|^2\right] \approx \frac{1}{N}\sum_{n=1}^{N}|x(k-n)|^2 = \frac{1}{N}\mathbf{x}^{\mathrm{H}}(k)\mathbf{x}(k)$$

which leads to the following bound on the stepsize

$$0 < \mu < \frac{2}{\mathbf{x}^{\mathrm{H}}(k)\mathbf{x}(k)} \tag{6.72}$$

This stepsize can be incorporated into the CLMS, to give the normalised CLMS (NCLMS) algorithm

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\eta}{\parallel \mathbf{x}(k) \parallel_2^2} e(k)\mathbf{x}^*(k) \tag{6.73}$$

where $0 < \eta < 2$. The effective stepsize

$$\mu(k) = \frac{\eta}{\parallel \mathbf{x}(k) \parallel_2^2} \tag{6.74}$$

is time varying, it alters the magnitude, but not the direction of the estimated gradient vector and greatly reduces the gradient noise, resulting in faster convergence as compared with CLMS [110].

Another, albeit approximate, way to perform normalisation of stochastic gradient learning algorithms for feedforward adaptive filters is to find the stepsize which minimises the error $e(k+1)$ based on its Taylor series expansion [187, 190, 279], given by

$$e(k+1) = e(k) + \sum_{n=1}^{N}\frac{\partial e(k)}{\partial w_n(k)}\Delta w_n(k) + \sum_{n=1}^{N}\sum_{m=1}^{N}\frac{\partial e^2(k)}{\partial w_n(k)\partial w_m(k)}\Delta w_n(k)\Delta w_m(k) + \cdots \tag{6.75}$$

For the CLMS (in the prediction setting $\mathbf{x}(k) = [x(k-1), \ldots, x(k-N]^{\mathrm{T}}$) the second- and higher-order terms in Equation (6.75) vanish, and

$$\frac{\partial e(k)}{\partial w_n(k)} = -x(k-n) \qquad \Delta w_n(k) = \mu e(k)x^*(k-n)$$

which gives

$$e(k+1) = e(k)\left[1 - \mu \parallel \mathbf{x}(k) \parallel_2^2\right]$$

The error $e(k+1)$ is zero for $e(k) = 0$ (trivial solution), and for

$$\mu(k) = \frac{1}{\parallel \mathbf{x}(k) \parallel_2^2} \tag{6.76}$$

that is, the stepsize of NCLMS.

[20] For convenience, we consider the prediction setting, where $\mathbf{x}(k) = [x(k-1), \ldots, x(k-N)]^{\mathrm{T}}$.

Normalised CNGD. Similarly to NCLMS, the CNGD algorithm for fully complex nonlinear adaptive filters can be normalised based on the Taylor series expansion (6.75), where

$$\frac{\partial e(k)}{\partial w_n(k)} = -\Phi'(k)x(k-n) \qquad \Delta w_n(k) = \mu e(k)\Phi'^*(k)x^*(k-n)$$

This gives the following 'optimal' learning rate for the complex nonlinear gradient descent algorithm

$$\mu(k) = \frac{1}{|\Phi'(k)|^2 \parallel \mathbf{x}(k) \parallel_2^2} \tag{6.77}$$

Finally, the normalised complex nonlinear gradient descent (NCNGD) algorithm can be expressed as

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{\eta}{|\Phi'(k)|^2 \parallel \mathbf{x}(k) \parallel_2^2} e(k)\Phi'^*(k)\mathbf{x}^*(k) \tag{6.78}$$

This alternative derivation of optimal stepsizes is only approximate – strictly speaking the normalisation should be based on the minimisation of the *a posteriori* error[21] $d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k+1)$ [61, 248].

6.5 Performance of Feedforward Nonlinear Adaptive Filters

Simulations have been conducted to illustrate:

- performance comparison between the linear CLMS and the fully complex, split complex, and dual univariate approaches;
- learning curves, showing convergence of learning algorithms when processing linear and nonlinear signals;
- performance comparison within the class of fully complex filters, for all the elementary transcendental functions used as nonlinearities.

Simulations were performed in a one step ahead prediction setting, on a linear AR(4) signal given by

$$y(k) = 1.79y(k-1) - 1.85y(k-2) + 1.27y(k-3) - 0.41y(k-4) + n(k) \tag{6.79}$$

where $n(k)$ is zero mean complex valued white Gaussian noise with variance $\sigma_n^2 = 1$, together with benchmark nonlinear Lorenz and Ikeda series, and a segment of real world wind data (speed and direction) recorded by an ultrasonic anemometer.[22] The Lorenz, Ikeda, and wind signals were made complex valued by convenience of representation, as explained in Section 2.2; for instance, for the complex wind vector we have $\mathbf{v}(k) = v(k)e^{j\phi(k)}$, where v denotes the wind speed and $\phi(k)$ direction.

[21] For more detail see Chapter 10 and Appendix M.

[22] These datasets are described in detail in Chapters 8 and 13.

Table 6.1 Prediction performance (in R_p [dB]) for the fully complex (FCAF), real–imaginary split-complex (RISC), linear CLMS, and dual univariate (DUAF) adaptive filters

	AR(4)	Lorenz	Ikeda	Wind
FACF	2.5222	15.2475	1.2364	10.2217
RISC	2.4969	14.4656	1.1535	9.8896
CLMS	2.5224	14.5456	1.1967	9.8939
DUAF	1.6528	11.7529	–0.3072	8.7353

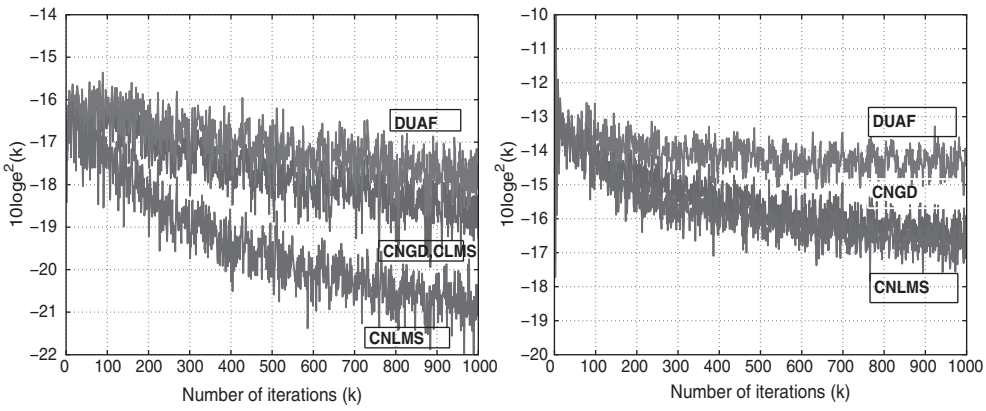
The quantitative performance criterion was the prediction gain

$$R_p = 10 \log \frac{\sigma_y^2}{\sigma_e^2}[dB] \tag{6.80}$$

For the AR(4) and Ikeda series, the results were averaged over 200 independent simulations, whereas for the complex Lorenz and wind signal single trial simulations were performed. In all cases, the filter tap length was $N = 4$, the signals were standardised to zero mean and maximum magnitude $|x|_{\max} = 0.8$, and the learning rate was $\mu = 0.01$.

Table 6.1 shows the prediction performance for all the feedforward adaptive filtering architectures considered. The data were 1000 samples long, and R_p was calculated over the last 100 samples. The fully complex filter used the complex tanh as the output nonlinearity and exhibited the best performance for nonlinear signals, whereas for the linear AR(4) process its performance was similar to that of CLMS. The performances of the split complex filter, which used the real tanh, and the CLMS were similar. The dual univariate approach had the worst performance, as by design, it did not take into account the correlation between the real and imaginary channel of complex quantities.

Figure 6.8 shows learning curves for the linear CLMS, nonlinear CNGD, DUAF, and normalised CLMS (NCLMS). This was achieved by averaging 200 independent simulations in a



(a) Prediction of the AR(4) process (b) Prediction of the Ikeda signal

Figure 6.8 Learning curves for one step ahead prediction of a linear and nonlinear signal

Table 6.2 Prediction performance (in R_p [dB]) for the fully complex filter and all elementary transcendental functions

	tan	sin	arctan	arcsin	tanh	sinh	arctanh	arcsinh
AR(4)	4.8968	4.9318	4.9057	4.9149	4.8955	4.9052	4.8820	4.8855
Lorenz	15.4061	16.9497	16.7381	16.0620	16.9510	16.0966	15.3699	16.9214
Ikeda	2.1739	2.3037	2.3361	2.2226	2.3383	2.2332	2.1846	2.3187
Wind	11.2484	11.7029	10.7832	12.0004	11.2454	11.9136	11.4867	11.5850

one step ahead prediction setting, for the linear AR(4) model (6.79) and nonlinear Ikeda map. Figure 6.8(a) illustrates that, for the linear signal, the fully complex nonlinear filter trained by CNGD performed similarly to CLMS, DUAF exhibited slightly inferior performance, whereas the NCLMS outperformed all the other algorithms. For the nonlinear Ikeda map, however, the CNGD had considerable performance advantage over DUAF, and approached the performance of NCLMS, as shown in Figure 6.8(b).

In the second set of simulations, only the fully complex nonlinearities were considered and the prediction performances were evaluated over the last 1000 samples of 6000 sample long datasets, and for all the elementary transcendental functions, as shown in Table 6.2. As the filters exhibited similar performances for all the fully complex nonlinearities, for convenience, we will most frequently use the tanh function.

6.6 Summary: Choice of a Nonlinear Adaptive Filter

The linear adaptive filter shown in Figure 6.2 is a simple and powerful adaptive filtering architecture. Its output is a linear combination of the inputs in filter memory and the set of adaptive filter coefficients (filter weights). However, due to its linear nature, this filter may perform suboptimally when processing nonlinear signals. To this end, we have introduced a class of nonlinear adaptive filters which comprise the standard linear adaptive filter and the output nonlinearity, as shown in Figure 6.5. This structure, called a nonlinear adaptive filter or a dynamical perceptron, is also a basic building block for neural networks. Properties and convergence of linear and nonlinear complex adaptive filters have been analysed, and their performance has been illustrated for both signals complex by design and by convenience of representation.

To summarise:

- The complex least mean square (CLMS) algorithm has been introduced for training complex linear adaptive filters, and its convergence in the mean and in the mean square has been addressed.
- Two classes of complex nonlinear adaptive filters have been introduced – fully complex and split-complex, together with the dual univariate approach.
- When selecting a nonlinear function at the output of a complex nonlinear adaptive filter, we need to choose between differentiability (fully complex) and boundedness (split-complex); this choice depends on the application – split-complex filters are more commonly used in neural networks for classification, whereas fully complex filters are a more natural choice in nonlinear adaptive filtering.

- Nonlinear adaptive filters based on fully complex nonlinearities retain the same generic form in their updates as linear adaptive filters, and provide the most consistent performance.
- When using fully complex filters we often need to standardise inputs; to allow for more freedom in the processing of general complex signals, Chapter 9 introduces nonlinear adaptive filters with an adaptive amplitude of nonlinearity.
- For best performance, the learning rate should be large at the beginning of the adaptation – for fast convergence, and small in the steady state – for good misadjustment. These are contradictory requirements, and to deal with these issues filters with a variable stepsize are introduced in Chapter 8.
- Learning algorithms in this chapter have been introduced based on standard complex statistics and are optimal when processing circular signals, that is, signals with rotation invariant distributions. However, when processing noncircular data (see Chapter 12), it is more appropriate to use so called widely linear models, which take into account both the covariance $C = E[\mathbf{x}(k)\mathbf{x}^{\mathrm{H}}(k)]$ and pseudocovariance $\mathbb{P} = E[\mathbf{x}(k)\mathbf{x}^{\mathrm{T}}(k)]$ functions. One such adaptive filtering algorithm is the augmented CLMS (ACLMS), introduced in Chapter 13.

Chapter 7 extends the class of feedforward nonlinear adaptive filters to allow feedback. Feedback architectures are very useful when modelling systems with long impulse responses which would require long feedforward filters.