Equations (1) and (2) provide two useful rules for the differentiation of a real-valued function with respect to a vector.

3. **Positive definite matrix**
An m-by-m matrix $\mathbf{R}$ is said to be nonnegative definite if it satisfies the condition

$$\mathbf{a}^T\mathbf{R}\mathbf{a} \geq 0 \qquad \text{for any vector } \mathbf{a} \in \mathbb{R}^m$$

If this condition is satisfied with the inequality sign, the matrix $\mathbf{R}$ is said to be positive definite.

An important property of a positive definite matrix $\mathbf{R}$ is that it is *nonsingular,* that is, the inverse matrix $\mathbf{R}^{-1}$ exists.

Another important property of a positive definite matrix $\mathbf{R}$ is that its eigenvalues, or roots or the characteristic equation

$$\det(\mathbf{R}) = 0$$

are all positive.

4. **Robustness**
The H^∞ criterion is due to Zames (1981), and it is developed in Zames and Francis (1983). The criterion is discussed in Doyle et al. (1989), Green and Limebeer (1995), and Hassibi et al. (1998).

5. To overcome the limitations of the LMS algorithm, namely, slow rate of convergence and sensitivity to variations in the condition number of the correlation matrix $\mathbf{R}_x$, we may use the *recursive least-squares (RLS) algorithm,* which follows from a recursive implementation of the linear least-squares filter described in Section 3.4. The RLS algorithm is a special case of the Kalman filter, which is known to be the optimum linear filter for a nonstationary environment. Most importantly, the Kalman filter exploits all past data extending up to and including the time instant at which the computations are made. For more details about the RLS algorithm and its relationship to the Kalman filter, see Haykin (1996). The Kalman filter is discussed in Chapter 15.

PROBLEMS

Unconstrained optimization

3.1 Explore the method of steepest descent involving a single weight w by considering the following cost function:

$$\mathscr{E}(w) = \frac{1}{2}\sigma^2 - r_{xd}w + \frac{1}{2}r_x w^2$$

where σ^2, r_{xd}, and r_x are constants.

3.2 Consider the cost function

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}\sigma^2 - \mathbf{r}_{xd}^T\mathbf{w} + \frac{1}{2}\mathbf{w}^T\mathbf{R}_x\mathbf{w}$$

where σ^2 is some constant, and

$$\mathbf{r}_{xd} = \begin{bmatrix} 0.8182 \\ 0.354 \end{bmatrix}$$

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0.8182 \\ 0.8182 & 1 \end{bmatrix}$$

(a) Find the optimum value $\mathbf{w}^*$ for which $\mathscr{E}(\mathbf{w})$ reaches its minimum value.

(b) Use the method of steepest descent to compute $\mathbf{w}^*$ for the following two values of learning-rate parameter:

(i) $\eta = 0.3$

(ii) $\eta = 1.0$

For each case, plot the trajectory traced by the evolution of the weight vector $\mathbf{w}(n)$ in the W-plane.

Note: The trajectories obtained for cases (i) and (ii) of part (b) should correspond to the pictures displayed in Fig. 3.2.

3.3 Consider the cost function of Eq. (3.24) that represents a modified form of the sum of error squares defined in Eq. (3.17). Show that the application of the Gauss–Newton method to Eq. (3.24) yields the weight-update described in Eq. (3.23).

LMS Algorithm

3.4 The correlation matrix $\mathbf{R}_x$ of the input vector $\mathbf{x}(n)$ in the LMS algorithm is defined by

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

Define the range of values for the learning-rate parameter η of the LMS algorithm for it to be convergent in the mean square.

3.5 The *normalized LMS algorithm* is described by the following recursion for the weight vector:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\eta}{\|\mathbf{x}(n)\|^2} e(n)\mathbf{x}(n)$$

where η is a positive constant and $\|\mathbf{x}(n)\|$ is the Euclidean norm of the input vector $\mathbf{x}(n)$. The error signal $e(n)$ is defined by

$$e(n) = d(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n)$$

where $d(n)$ is the desired response. For the normalized LMS algorithm to be convergent in the mean square, show that

$$0 < \eta < 2$$

3.6 The LMS algorithm is used to implement the generalized sidelobe canceler shown in Fig. 2.16. Set up the equations that define the operation of this system, assuming the use of a single neuron for the neural network.

3.7 Consider a linear predictor with its input vector made up of the samples $x(n-1)$, $x(n-2), \dots, x(n-m)$, where m is the prediction order. The requirement is to use the LMS algorithm to make a prediction $\hat{x}(n)$ of the input sample $x(n)$. Set up the recursions that may be used to compute the tap weight $w_1, w_2, \dots, w_m$ of the predictor.

3.8 The ensemble-averaged counterpart to the sum of error squares viewed as a cost function is the mean-square value of the error signal:

$$J(\mathbf{w}) = \frac{1}{2} E[e^2(n)]$$

$$= \frac{1}{2} E[(d(n) - \mathbf{x}^T(n)\mathbf{w})^2]$$

(a) Assuming that the input vector $\mathbf{x}(n)$ and desired response $d(n)$ are drawn from a stationary environment, show that

$$J(\mathbf{w}) = \frac{1}{2}\sigma_d^2 - \mathbf{r}_{\mathbf{x}d}^T \mathbf{w} + \frac{1}{2}\mathbf{w}^T\mathbf{R}_\mathbf{x}\mathbf{w}$$

where

$$\sigma_d^2 = E[d^2(n)]$$
$$\mathbf{r}_{\mathbf{x}d} = E[\mathbf{x}(n)\, d(n)]$$
$$\mathbf{R}_\mathbf{x} = E[\mathbf{x}(n)\mathbf{x}^T(n)]$$

(b) For this cost function, show that the gradient vector and Hessian matrix of $J(\mathbf{w})$ are as follows, respectively:

$$\mathbf{g} = -\mathbf{r}_{\mathbf{x}d} + \mathbf{R}_\mathbf{x}\mathbf{w}$$
$$\mathbf{H} = \mathbf{R}_\mathbf{x}$$

(c) In the *LMS/Newton algorithm*, the gradient vector $\mathbf{g}$ is replaced by its instantaneous value (Widrow and Stearns, 1985). Show that this algorithm, incorporating a learning-rate parameter η, is described by

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta\, \mathbf{R}_\mathbf{x}^{-1}\mathbf{x}(n)(d(n) - \mathbf{x}^T(n)\mathbf{w}(n))$$

The inverse of the correlation matrix $\mathbf{R}_\mathbf{x}$, assumed to be positive definite, is calculated ahead of time.

3.9 In this problem we revisit the correlation matrix memory discussed in Section 2.11. A shortcoming of this memory is that when a key pattern $\mathbf{x}_j$ is presented to it, the actual response $\mathbf{y}$ produced by the memory may not be close enough (in a Euclidean sense) to the desired response (memorized pattern) $\mathbf{y}_j$ for the memory to associate perfectly. This shortcoming is inherited from the use of Hebbian learning that has no provision for feedback from the output to the input. As a remedy for this shortcoming, we may incorporate an error-correction mechanism into the design of the memory, forcing it to associate properly (Anderson, 1983).

Let $\hat{\mathbf{M}}(n)$ denote the memory matrix learned at iteration n of the error-correction learning process. The memory matrix $\hat{\mathbf{M}}(n)$ learns the information represented by the associations:

$$\mathbf{x}_k \rightarrow \mathbf{y}_k, \qquad k = 1, 2, \ldots, q$$

(a) Adapting the LMS algorithm for the problem at hand, show that the updated value of the memory matrix is defined by

$$\hat{\mathbf{M}}(n+1) = \hat{\mathbf{M}}(n) + \eta[\mathbf{y}_k - \hat{\mathbf{M}}(n)\mathbf{x}_k]\mathbf{x}_k^T$$

where η is the learning-rate parameter.

(b) For autoassociation, $\mathbf{y}_k = \mathbf{x}_k$. For this special case, show that as the number of iterations, n, approaches infinity, the memory autoassociates perfectly, as shown by

$$\mathbf{M}(\infty)\mathbf{x}_k = \mathbf{x}_k, \qquad k = 1, 2, \ldots, q$$

(c) The result described in part (b) may be viewed as an *eigenvalue problem*. In that context, $\mathbf{x}_k$ represents an eigenvector of $\mathbf{M}(\infty)$. What are the eigenvalues of $\mathbf{M}(\infty)$?

3.10 In this problem we investigate the effect of bias on the condition number of a correlation matrix, and therefore the performance of the LMS algorithm.

Consider a random vector $\mathbf{X}$ with covariance matrix

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

and mean vector

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$$

(a) Calculate the condition number of the covariance matrix $\mathbf{C}$.

(b) Calculate the condition number of the correlation matrix $\mathbf{R}$.

Comment on the effect of the bias $\boldsymbol{\mu}$ on the performance of the LMS algorithm.

Rosenblatt's Perceptron

3.11 In this problem, we consider another method for deriving the update equation for Rosenblatt's perceptron. Define the *perceptron criterion function* (Duda and Hart, 1973):

$$J_p(\mathbf{w}) = \sum_{\mathbf{x} \in \mathscr{X}(\mathbf{w})} (-\mathbf{w}^T\mathbf{x})$$

where $\mathscr{X}(\mathbf{w})$ is the set of samples misclassified by the choice of weight vector $\mathbf{w}$. Note that $J_p(\mathbf{w})$ is defined as zero if there are no misclassified samples, and the output is misclassified if $\mathbf{w}_{\mathbf{x}}^T \leq 0$.

(a) Demonstrate geometrically that $J_p(\mathbf{w})$ is proportional to the sum of Euclidean distances from the misclassified samples to the decision boundary.

(b) Determine the gradient of $J_p(\mathbf{w})$ with respect to the weight vector $\mathbf{w}$.

(c) Using the result obtained in part (b), show that the weight-update for the perceptron is:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n) \sum_{\mathbf{x} \in \mathscr{X}(\mathbf{w}(n))} \mathbf{x}$$

where $\mathscr{X}(\mathbf{w}(n))$ is the set of samples misclassified by the use of weight vector $\mathbf{w}(n)$, and $\eta(n)$ is the learning-rate parameter. Show that this result, for the case of a single-sample correction, is basically the same as that described by Eqs. (3.54) and (3.55).

3.12 Verify that Eqs. (3.68)–(3.71), summarizing the perceptron convergence algorithm, are consistent with Eqs. (3.54) and (3.55).

3.13 Consider two one-dimensional, Gaussian-distributed classes $\mathscr{C}_1$ and $\mathscr{C}_2$ that have a common variance equal to 1. Their mean values are

$$\mu_1 = -10$$
$$\mu_2 = +10$$

These two classes are essentially linearly separable. Design a classifier that separates these two classes.

3.14 Suppose that in the signal-flow graph of the perceptron shown in Fig. 3.6 the hard limiter is replaced by the sigmoidal nonlinearity:

$$\varphi(v) = \tanh\left(\frac{v}{2}\right)$$

where v is the induced local field. The classification decisions made by the perceptron are defined as follows:

> *Observation vector* $\mathbf{x}$ *belongs to class* $\mathscr{C}_1$ *if the output* $y > \theta$ *where* θ *is a threshold; otherwise,* $\mathbf{x}$ *belongs to class* $\mathscr{C}_2$.

Show that the decision boundary so constructed is a hyperplane.

3.15 **(a)** The perceptron may be used to perform numerous logic functions. Demonstrate the implementation of the binary logic functions AND, OR, and COMPLEMENT.

(b) A basic limitation of the perceptron is that it cannot implement the EXCLUSIVE OR function. Explain the reason for this limitation.

3.16 Equations (3.86) and (3.87) define the weight vector and bias of the Bayes classifier for a Gaussian environment. Determine the composition of this classifier for the case when the covariance matrix $\mathbf{C}$ is defined by

$$\mathbf{C} = \sigma^2\mathbf{I}$$

where σ^2 is a constant.