

Radial-Basis Function Networks

INTRODUCTION

The design of a supervised neural network may be pursued in a variety of ways. The back-propagation algorithm for the design of a multilayer perceptron (under supervision) as described in the previous chapter may be viewed as the application of a recursive technique known in statistics as *stochastic approximation*. In this chapter we take a completely different approach by viewing the design of a neural network as a *curve-fitting (approximation) problem* in a high-dimensional space. According to this viewpoint, learning is equivalent to finding a surface in a multidimensional space that provides a best fit to the training data, with the criterion for "best fit" being measured in some statistical sense. Correspondingly, generalization is equivalent to the use of this multidimensional surface to interpolate the test data. Such a viewpoint is the motivation behind the method of radial-basis functions in the sense that it draws upon research work on traditional strict interpolation in a multidimensional space. In the context of a neural network, the hidden units provide a set of "functions" that constitute an arbitrary "basis" for the input patterns (vectors) when they are expanded into the hidden space; these functions are called *radial-basis functions*.¹ Radial-basis functions were first introduced in the solution of the real multivariate interpolation problem. The early work on this subject is surveyed in Powell (1985), and more recent work is surveyed in Light (1992b). It is now one of the main fields of research in numerical analysis.

The construction of a *radial-basis function (RBF) network*, in its most basic form, involves three layers with entirely different roles. The input layer is made up of source nodes (sensory units) that connect the network to its environment. The second layer, the *only* hidden layer in the network, applies a nonlinear transformation from the input space to the hidden space; in most applications the hidden space is of high dimensionality. The output layer is linear, supplying the response of the network to the activation pattern (signal) applied to the input layer. A mathematical justification for the rationale of a nonlinear transformation followed by a linear transformation may be traced back to an early

paper by Cover (1965). According to this paper, a pattern-classification problem cast in a high-dimensional space is more likely to be linearly separable than in a low-dimensional space—hence the reason for frequently making the dimension of the hidden space in an RBF network high. Another important point is the fact that the dimension of the hidden space is directly related to the capacity of the network to approximate a smooth input-output mapping (Mhaskar, 1996; Niyogi and Girosi, 1996); the higher the dimension of the hidden space, the more accurate the approximation will be.

Organization of the Chapter

The main body of the chapter is organized as follows. We lay the foundations for the construction of an RBF network in Sections 5.2 and 5.4. We do this in two stages. First, we describe Cover's theorem on the separability of patterns; the XOR problem is used here to illustrate the application of this theorem. In Section 5.3 we consider the interpolation problem and its relationship to RBF networks.

After developing an understanding of how the RBF network functions, we move on to the second part of the chapter that consists of Sections 5.4 through 5.9. In Section 5.4 we discuss the viewpoint that supervised learning is an ill-posed hypersurface reconstruction problem. In Section 5.5 we present a detailed treatment of Tikhonov's regularization theory and its application to RBF networks. This theory naturally leads to the formulation of regularization networks in Section 5.6. This class of RBF networks is computationally demanding. To reduce computational complexity, in Section 5.7 we discuss a modified form of regularization networks referred to as generalized RBF networks. In Section 5.8 we revisit the XOR problem and show how it can be solved using an RBF network. In Section 5.9 we complete the study of regularization theory by describing the method of generalized cross-validation for selecting a suitable value for the regularization parameter.

Section 5.10 discusses the approximation properties of RBF networks. Section 5.11 presents a comparison between RBF networks and multilayer perceptrons, both of which are important examples of layered feedforward networks.

In Section 5.12 we discuss kernel regression estimation as the basis of another viewpoint of RBF networks. We relate RBF networks to a large body of the statistics literature dealing with density estimation and kernel regression theory.

The last part of the chapter consists of Sections 5.13 and 5.14. In Section 5.13 we describe four different learning strategies for the design of RBF networks. In Section 5.14 we describe a computer experiment on pattern classification using RBF networks.

The chapter concludes with some final thoughts on RBF networks in Section 5.15.

5.2 COVER'S THEOREM ON THE SEPARABILITY OF PATTERNS

When a radial-basis function (RBF) network is used to perform a *complex* pattern-classification task, the problem is basically solved by transforming it into a high-dimensional space in a nonlinear manner. The underlying justification is found in *Cover's theorem on the separability of patterns*, which, in qualitative terms, may be stated as follows (Cover, 1965):

A complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space.

From the work we did on single-layer perceptrons in Chapter 3, we know that once we have linearly separable patterns, the classification problem is relatively easy to solve. Accordingly, we may develop a great deal of insight into the operation of an RBF network as a pattern classifier by studying the separability of patterns.

Consider a family of surfaces where each naturally divides an input space into two regions. Let $\mathcal{X}$ denote a set of N patterns (vectors) $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$, each of which is assigned to one of two classes $\mathcal{X}_1$ and $\mathcal{X}_2$. This *dichotomy* (binary partition) of the points is said to be separable with respect to the family of surfaces if a surface exists in the family that separates the points in the class $\mathcal{X}_1$ from those in the class $\mathcal{X}_2$. For each pattern $\mathbf{x} \in \mathcal{X}$, define a vector made up of a set of real-valued functions $\{\varphi_i(\mathbf{x})\}_{i=1, 2, \dots, m_1}$, as shown by

$$\boldsymbol{\varphi}(\mathbf{x}) = [\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), \dots, \varphi_{m_1}(\mathbf{x})]^T \quad (5.1)$$

Suppose that the pattern $\mathbf{x}$ is a vector in an m_0 -dimensional input space. The vector $\boldsymbol{\varphi}(\mathbf{x})$ then maps points in m_0 -dimensional input space into corresponding points in a new space of dimension m_1 . We refer to $\boldsymbol{\varphi}(\mathbf{x})$ as a *hidden function*, because it plays a role similar to that of a hidden unit in a feedforward neural network. Correspondingly, the space spanned by the set of hidden functions $\{\boldsymbol{\varphi}(\mathbf{x})\}_{i=1}^m$ is referred to as the *hidden space* or *feature space*.

A dichotomy $\{\mathcal{X}_1, \mathcal{X}_2\}$ of $\mathcal{X}$ is said to be $\boldsymbol{\varphi}$ -separable if there exists an m_1 -dimensional vector $\mathbf{w}$ such that we may write (Cover, 1965)

$$\begin{aligned} \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) &> 0, & \mathbf{x} \in \mathcal{X}_1 \\ \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) &< 0, & \mathbf{x} \in \mathcal{X}_2 \end{aligned} \quad (5.2)$$

The hyperplane defined by the equation

$$\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) = 0$$

describes the separating surface in the $\boldsymbol{\varphi}$ -space (i.e., hidden space). The inverse image of this hyperplane, that is,

$$\mathbf{x}: \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) = 0 \quad (5.3)$$

defines the *separating surface* in the input space.

Consider a natural class of mappings obtained by using a linear combination of r -wise products of the pattern vector coordinates. The separating surfaces corresponding to such mappings are referred to as *r th-order rational varieties*. A rational variety of order r in a space of dimension m_0 is described by an r th degree homogeneous equation in the coordinates of the input vector $\mathbf{x}$, as shown by

$$\sum_{0 \leq i_1 \leq i_2 \leq \dots \leq i_r \leq m_0} a_{i_1 i_2 \dots i_r} x_{i_1} x_{i_2} \dots x_{i_r} = 0 \quad (5.4)$$

where x_i is the i th component of input vector $\mathbf{x}$, and x_0 is set equal to unity in order to express the equation in a homogeneous form. An r th order product of entries x_i of

$\mathbf{x}$, that is, $x_{i_1} x_{i_2} \dots x_{i_r}$, is called a *monomial*. For an input space of dimensionality m_0 , there are

$$\frac{(m_0 - r)!}{m_0! r!}$$

monomials in Eq. (5.4). Examples of the type of separating surfaces described by Eq. (5.4) are *hyperplanes* (first-order rational varieties), *quadrics* (second-order rational varieties), and *hyperspheres* (quadrics with certain linear constraints on the coefficients). These examples are illustrated in Fig. 5.1 for a configuration of five points in a two-dimensional input space. In general, linear separability implies spherical separability which implies quadric separability; however, the converses are not necessarily true.

In a probabilistic experiment, the separability of a set of patterns becomes a random event that depends on the dichotomy chosen and the distribution of the patterns in the input space. Suppose that the activation patterns $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$ are chosen independently, according to a probability measure imposed on the input space. Suppose also that all the possible dichotomies of $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ are equiprobable. Let $P(N, m_1)$ denote the probability that a particular dichotomy picked at random is $\boldsymbol{\varphi}$ -separable, where the class of separating surfaces chosen has m_1 degrees of freedom. Following Cover (1965), we may then state that

$$P(N, m_1) = \left(\frac{1}{2}\right)^{N-1} \sum_{m=0}^{m_1-1} \binom{N-1}{m} \quad (5.5)$$

where the binomial coefficients comprising $N-1$ and m are themselves defined for all integer l and m by

$$\binom{l}{m} = \frac{l(l-1)\dots(l-m+1)}{m!}$$

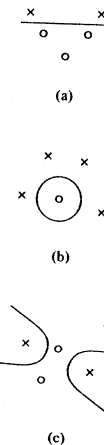


FIGURE 5.1 Three examples of $\boldsymbol{\varphi}$ -separable dichotomies of different sets of five points in two dimensions: (a) linearly separable dichotomy; (b) spherically separable dichotomy; (c) quadrically separable dichotomy.

Equation (5.5) embodies the essence of *Cover's separability theorem* for random patterns.² It is a statement of the fact that the cumulative binomial distribution corresponding to the probability that $(N - 1)$ flips of a fair coin will result in $(m_1 - 1)$ or fewer heads.

Although the hidden-unit surfaces envisioned in the derivation of Eq. (5.5) are of a polynomial form and therefore different from those commonly used in radial-basis function networks, the essential content of the equation has general applicability. Specifically, the higher we make the dimension m_1 of the hidden space, the closer the probability $P(N, m_1)$ will be to unity. To sum up, Cover's theorem on the separability of patterns encompasses two basic ingredients:

1. Nonlinear formulation of the hidden function defined by $\varphi_i(\mathbf{x})$, where $\mathbf{x}$ is the input vector and $i = 1, 2, \dots, m_1$.
2. High dimensionality of the hidden space compared to the input space; that dimensionality is determined by the value assigned to m_1 (i.e., the number of hidden units).

In general, as stated previously, a complex pattern-classification problem cast in high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space. We emphasize, however, that in some cases the use of nonlinear mapping (i.e., point 1) may be sufficient to produce linear separability without having to increase the dimensionality of the hidden-unit space, as illustrated in the following example.

Example 5.1. The XOR Problem

To illustrate the significance of the idea of φ -separability of patterns, consider the simple yet important XOR problem. In the XOR problem there are four points (patterns): $(1, 1)$, $(0, 1)$, $(0, 0)$, and $(1, 0)$, in a two-dimensional input space, as depicted in Fig. 5.2a. The requirement is to construct a pattern classifier that produces the binary output 0 in response to the input pattern $(1, 1)$, or $(0, 0)$, and the binary output 1 in response to the input pattern $(0, 1)$ or $(1, 0)$. Thus points that are closest in the input space, in terms of the Hamming distance, map to regions that are maximally apart in the output space.

Define a pair of Gaussian hidden functions as follows:

$$\begin{aligned}\varphi_1(\mathbf{x}) &= e^{-\|\mathbf{x} - \mathbf{t}_1\|^2}, & \mathbf{t}_1 &= [1, 1]^T \\ \varphi_2(\mathbf{x}) &= e^{-\|\mathbf{x} - \mathbf{t}_2\|^2}, & \mathbf{t}_2 &= [0, 0]^T\end{aligned}$$

We may then construct the results summarized in Table 5.1 for the four different input patterns of interest. The input patterns are mapped onto the $\varphi_1 - \varphi_2$ plane as shown in Fig. 5.2b. Here we now see that the input patterns $(0, 1)$ and $(1, 0)$ are linearly separable from the remaining input patterns $(1, 1)$ and $(0, 0)$. Thereafter, the XOR problem may be readily solved by using the functions $\varphi_1(\mathbf{x})$ and $\varphi_2(\mathbf{x})$ as the inputs to a linear classifier such as the perceptron.

In this example there is no increase in the dimensionality of the hidden space compared to the input space. In other words, nonlinearity exemplified by the use of Gaussian hidden functions is sufficient to transform the XOR problem into a linearly separable one.

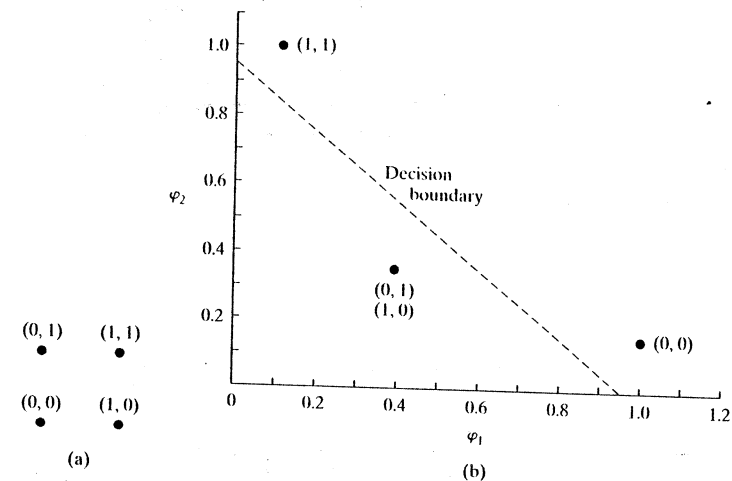


FIGURE 5.2 (a) The four patterns of the XOR problem; (b) Decision-making diagram.

TABLE 5.1 Specification of the Hidden Functions for the XOR Problem of Example 5.1

Input Pattern, $\mathbf{x}$	First Hidden Function, $\varphi_1(\mathbf{x})$	Second Hidden Function, $\varphi_2(\mathbf{x})$
$(1,1)$	1	0.1353
$(0,1)$	0.3678	0.3678
$(0,0)$	0.1353	1
$(1,0)$	0.3678	0.3678

Separating Capacity of a Surface

Equation (5.5) has an important bearing on the expected maximum number of randomly assigned patterns that are linearly separable in a multidimensional space. To explore this issue, let $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$ be a sequence of random patterns (vectors) as previously described. Let N be a random variable defined as the largest integer such that this sequence is φ -separable, where φ has m_1 degrees of freedom. Then, from Eq. (5.5) we deduce that the probability that $N = n$ is given by

$$\begin{aligned}\text{Prob}(N = n) &= P(n, m_1) - P(n + 1, m_1) \\ &= \binom{1}{2}^n \binom{n-1}{m_1-1}, \quad n = 0, 1, 2, \dots, \end{aligned} \quad (5.6)$$

For an interpretation of this result, we recall the definition of a *negative binomial distribution*. This distribution equals the probability that k failures precede the r th success

in a long, repeated sequence of *Bernoulli trials*. In such a probabilistic experiment there are only two possible outcomes for each trial, success or failure, and their probabilities remain the same throughout the experiment. Let p and q denote the probabilities of success and failure, respectively, with $p + q = 1$. The negative binomial distribution is defined by (Feller, 1968)

$$f(k; r, p) = p^r q^k \binom{r+k-1}{k}$$

For the special case of $p = q = 1/2$ (i.e., success and failure are equiprobable) and $k + r = n$, the negative binomial distribution reduces to

$$f\left(k; n - k, \frac{1}{2}\right) = \left(\frac{1}{2}\right)^n \binom{n-1}{k}, \quad n = 0, 1, 2, \dots$$

With this definition, we now see that the result described in Eq. (5.6) is just the negative binomial distribution, shifted by m_1 units to the right, and with parameters m_1 and $1/2$. Thus N corresponds to the "waiting time" for the m_1 -th failure in a sequence of tosses of a fair coin. The expectation of the random variable N and its median are respectively:

$$E[N] = 2m_1 \quad (5.7)$$

and

$$\text{Median}[N] = 2m_1 \quad (5.8)$$

We therefore have a corollary to Cover's theorem in the form of a celebrated asymptotic result that may be stated as (Cover, 1965):

The expected maximum number of randomly assigned patterns (vectors) that are linearly separable in a space of dimensionality m_1 is equal to $2m_1$.

This result suggests that $2m_1$ is a natural definition of the *separating capacity* of a family of decision surfaces having m_1 degrees of freedom. In a way, the separating capacity of a surface is closely related to the notion of VC dimension that is discussed in Chapter 2.

3 INTERPOLATION PROBLEM

The important point that emerges from Cover's theorem on the separability of patterns is that in solving a nonlinearly separable pattern-classification problem, there is usually practical benefit to be gained by mapping the input space into a new space of high enough dimension. Basically, a nonlinear mapping is used to transform a nonlinearly separable classification problem into a linearly separable one. In a similar way, we may use a nonlinear mapping to transform a difficult nonlinear filtering problem into an easier one that involves linear filtering.

Consider then a feedforward network with an input layer, a single hidden layer, and an output layer consisting of a single unit. We have purposely chosen a single output unit to simplify the exposition without loss of generality. The network is designed to perform a *nonlinear mapping* from the input space to the hidden space, followed by

a *linear mapping* from the hidden space to the output space. Let m_o denote the dimension of the input space. Then, in an overall fashion, the network represents a map from the m_o -dimensional input space to the single-dimensional output space, written as

$$s: \mathbb{R}^{m_o} \rightarrow \mathbb{R}^1 \quad (5.9)$$

We may think of the map s as a *hypersurface* (graph) $\Gamma \subset \mathbb{R}^{m_o+1}$, just as we think of the elementary map $s: \mathbb{R}^1 \rightarrow \mathbb{R}^1$, where $s(x) = x^2$, as a parabola drawn in $\mathbb{R}^2$ space. The surface Γ is a multidimensional plot of the output as a function of the input. In a practical situation, the surface Γ is unknown and the training data are usually contaminated with noise. The training phase and generalization phase of the learning process may be respectively viewed as follows (Broomhead and Lowe, 1988):

- The training phase constitutes the optimization of a fitting procedure for the surface Γ , based on known data points presented to the network in the form of input-output examples (patterns).
- The generalization phase is synonymous with interpolation between the data points, with the interpolation being performed along the constrained surface generated by the fitting procedure as the optimum approximation to the true surface Γ .

Thus we are led to the theory of *multivariable interpolation* in high-dimensional space, which has a long history (Davis, 1963). The interpolation problem, in its *strict* sense, may be stated:

Given a set of N different points $\{\mathbf{x}_i \in \mathbb{R}^{m_o} | i = 1, 2, \dots, N\}$ and a corresponding set of N real numbers $\{d_i \in \mathbb{R}^1 | i = 1, 2, \dots, N\}$, find a function $F: \mathbb{R}^{m_o} \rightarrow \mathbb{R}^1$ that satisfies the interpolation condition:

$$F(\mathbf{x}_i) = d_i, \quad i = 1, 2, \dots, N \quad (5.10)$$

For strict interpolation as specified here, the interpolating surface (i.e., function F) is constrained to pass through *all* the training data points.

The *radial-basis functions* (RBF) technique consists of choosing a function F that has the following form (Powell, 1988):

$$F(\mathbf{x}) = \sum_{i=1}^N w_i \varphi(\|\mathbf{x} - \mathbf{x}_i\|) \quad (5.11)$$

where $\{\varphi(\|\mathbf{x} - \mathbf{x}_i\|) | i = 1, 2, \dots, N\}$ is a set of N arbitrary (generally nonlinear) functions, known as *radial-basis functions*, and $\|\cdot\|$ denotes a *norm* that is usually Euclidean. The known data points $\mathbf{x}_i \in \mathbb{R}^{m_o}$, $i = 1, 2, \dots, N$ are taken to be the *centers* of the radial-basis functions.

Inserting the interpolation conditions of Eq. (5.10) in (5.11), we obtain the following set of simultaneous linear equations for the unknown coefficients (weights) of the expansion $\{w_i\}$:

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix} \quad (5.12)$$

where

$$\varphi_{ji} = \varphi(\|x_j - x_i\|), \quad (j, i) = 1, 2, \dots, N \quad (5.13)$$

Let

$$\mathbf{d} = [d_1, d_2, \dots, d_N]^T$$

$$\mathbf{w} = [w_1, w_2, \dots, w_N]^T$$

The N -by-1 vectors $\mathbf{d}$ and $\mathbf{w}$ represent the *desired response vector* and *linear weight vector*, respectively, where N is the *size of the training sample*. Let Φ denote an N -by- N matrix with elements φ_{ji} :

$$\Phi = \{\varphi_{ji} | (j, i) = 1, 2, \dots, N\} \quad (5.14)$$

We call this matrix the *interpolation matrix*. We may then rewrite Eq. (5.12) in the compact form

$$\Phi \mathbf{w} = \mathbf{x} \quad (5.15)$$

Assuming that Φ is nonsingular and therefore that the inverse matrix Φ^{-1} exists, we may go on to solve Eq. (5.15) for the weight vector $\mathbf{w}$ as shown by

$$\mathbf{w} = \Phi^{-1} \mathbf{x} \quad (5.16)$$

The vital question is: How can we be sure that the interpolation matrix Φ is nonsingular? It turns out that for a large class of radial-basis functions and under certain conditions, the answer to this question is given in the following important theorem.

Micchelli's Theorem

In Micchelli (1986), the following theorem is proved:

Let $\{x_i\}_{i=1}^N$ be a set of distinct points in $\mathbb{R}^m$. Then the N -by- N interpolation matrix Φ , whose ji -th element is $\varphi_{ji} = \varphi(\|x_j - x_i\|)$, is nonsingular.

There is a large class of radial-basis functions that is covered by Micchelli's theorem; it includes the following functions that are of particular interest in the study of RBF networks:

1. Multiquadrics:

$$\varphi(r) = (r^2 + c^2)^{1/2} \quad \text{for some } c > 0 \text{ and } r \in \mathbb{R} \quad (5.17)$$

2. Inverse multiquadrics:

$$\varphi(r) = \frac{1}{(r^2 + c^2)^{1/2}} \quad \text{for some } c > 0 \text{ and } r \in \mathbb{R} \quad (5.18)$$

3. Gaussian functions:

$$\varphi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \text{for some } \sigma > 0 \text{ and } r \in \mathbb{R} \quad (5.19)$$

The multiquadrics and inverse multiquadrics are both due to Hardy (1971).

For the radial-basis functions listed in Eqs. (5.17) to (5.19) to be nonsingular, the points $\{x_i\}_{i=1}^N$ must all be different (i.e., distinct). This is all that is required for nonsingularity of the interpolation matrix Φ , whatever the values of size N of the data points or dimensionality m_0 of the vectors (points) x_i .

The inverse multiquadrics of Eq. (5.18) and the Gaussian functions of (5.19) share a common property: They are both *localized* functions, in the sense that $\varphi(r) \rightarrow 0$ as $r \rightarrow \infty$. In both of these cases the interpolation matrix Φ is positive definite. By contrast, the multiquadrics of Eq. (5.17) are *nonlocal* in that $\varphi(r)$ becomes unbounded as $r \rightarrow \infty$; and the corresponding interpolation matrix Φ has $N-1$ negative eigenvalues and only one positive eigenvalue, with the result that it is *not* positive definite (Micchelli, 1986). What is remarkable, however, is that an interpolation matrix Φ based on Hardy's multiquadrics is nonsingular, and therefore suitable for use in the design of RBF networks.

What is even more remarkable is that radial-basis functions that *grow* at infinity, such as multiquadrics, can be used to approximate a smooth input-output mapping with greater accuracy than those that yield a positive-definite interpolation matrix. This surprising result is discussed in Powell (1988).

5.4 SUPERVISED LEARNING AS AN ILL-POSED HYPERSURFACE RECONSTRUCTION PROBLEM

The strict interpolation procedure described may not be a good strategy for the training of RBF networks for certain classes of tasks because of poor generalization to new data for the following reason: When the number of data points in the training sample is much larger than the number of degrees of freedom of the underlying physical process, and we are constrained to have as many radial-basis functions as data points, the problem is overdetermined. Consequently, the network may end up fitting misleading variations due to idiosyncrasies or noise in the input data, thereby resulting in degraded generalization performance (Broomhead and Lowe, 1988).

To develop a deep understanding of the overfitting problem and how to cure it, we first go back to the viewpoint that the design of a neural network trained to retrieve an output pattern when presented with an input pattern is equivalent to learning a hypersurface (i.e., multidimensional mapping) that defines the output in terms of the input. In other words, *learning is viewed as a problem of hypersurface reconstruction, given a set of data points that may be sparse*.

According to Keller (1976) and Kirsch (1996), two related problems are said to be the *inverse* of each other if the formulation of each of them requires partial or full knowledge of the other. Ordinarily we find that one of the problems has been studied earlier, and perhaps in greater detail than the other. This particular problem is called the *direct problem*, in which case the other one is called the *inverse problem*. However, from a mathematical perspective, there is another more important difference between a direct problem and an inverse problem. Specifically, a problem of interest may be well-posed or ill-posed. The term "well-posed" has been used in applied mathematics since the time of Hadamard in the early 1900s. To explain this terminology, assume that

we have a domain X and a range Y taken to be metric spaces, and that are related by a fixed but unknown mapping f . The problem of reconstructing the mapping f is said to be *well-posed* if three conditions are satisfied (Tikhonov and Arsenin, 1977; Morozov, 1993; Kirsch, 1996):

1. *Existence*. For every input vector $\mathbf{x} \in X$, there does exist an output $y = f(\mathbf{x})$, where $y \in Y$.
2. *Uniqueness*. For any pair of input vectors $\mathbf{x}, \mathbf{t} \in X$, we have $f(\mathbf{x}) = f(\mathbf{t})$ if, and only if, $\mathbf{x} = \mathbf{t}$.
3. *Continuity*. The mapping is continuous, that is, for any $\varepsilon > 0$ there exists $\delta = \delta(\varepsilon)$ such that the condition $\rho_x(\mathbf{x}, \mathbf{t}) < \delta$ implies that $\rho_y(f(\mathbf{x}), f(\mathbf{t})) < \varepsilon$, where $\rho(\cdot, \cdot)$ is the symbol for distance between the two arguments in their respective spaces. This criterion is illustrated in Fig 5.3. The property of continuity is also referred to as *stability*.

If any of these conditions is not satisfied, the problem is said to be *ill-posed*. Basically, an ill-posed problem means that large data sets may contain a surprisingly small amount of information about the desired solution.

In the context of our present situation, the physical phenomenon responsible for generating the training data (e.g., speech, pictures, radar signals, sonar signals, seismic data) is a well-posed direct problem. However, learning from such physical forms of data, viewed as a hypersurface reconstruction problem, is an ill-posed inverse problem for the following reasons. First, the existence criterion may be violated in that a distinct output may not exist for every input. Second, there may not be as much information in the training sample as we really need to reconstruct the input-output mapping uniquely, hence the uniqueness criterion is likely to be violated. Third, the unavoidable presence of noise or imprecision in real-life training data adds uncertainty to the reconstructed input-output mapping. In particular, if the noise level in the input is too high, it is possible for the neural network to produce an output outside of the range Y for a specified input $\mathbf{x}$ in the domain X ; in other words, there is likelihood for the continuity criterion to be violated. If a learning problem lacks the property of continuity, then the computed input-output mapping has nothing to do with the true solution to the learning problem. There is no way to overcome this difficulty unless some prior information about the input-output mapping is available. In this context, it is rather appropriate that we remind ourselves of a statement made by Lanczos on linear differential operators (Lanczos, 1964): "A lack of information cannot be remedied by any mathematical trickery."

The important issue of how to make an ill-posed problem into a well-posed one via regularization is discussed in the next section.³

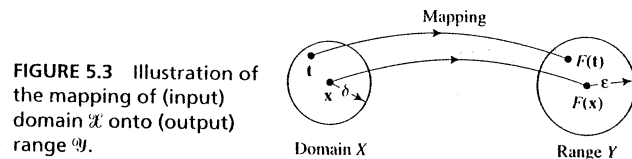


FIGURE 5.3 Illustration of the mapping of (input) domain X onto (output) range Y .

5.5 REGULARIZATION THEORY

In 1963, Tikhonov proposed a new method called *regularization* for solving ill-posed problems.⁴ In the context of a hypersurface reconstruction problem, the basic idea of regularization is to *stabilize* the solution by means of some auxiliary nonnegative functional that embeds prior information about the solution. The most common form of prior information involves the assumption that the input-output mapping function (i.e., solution to the reconstruction problem) is *smooth*, in the sense that similar inputs correspond to similar outputs.

To be specific, let the set of input-output data (i.e., training sample) available for approximation be described by

$$\begin{aligned} \text{Input signal:} \quad & \mathbf{x}_i \in \mathbb{R}^{m_0}, \quad i = 1, 2, \dots, N \\ \text{Desired response:} \quad & d_i \in \mathbb{R}^1, \quad i = 1, 2, \dots, N \end{aligned} \quad (5.20)$$

Note that the output is assumed to be one-dimensional. This assumption does not in any way limit the general applicability of the regularization theory being developed here. Let the approximating function be denoted by $F(\mathbf{x})$, where (for convenience of presentation) we have omitted the weight vector $\mathbf{w}$ of the network from the argument of the function F . Basically, Tikhonov's regularization theory involves two terms:

1. *Standard Error Term*. This first term, denoted by $\mathcal{E}_s(F)$, measures the standard error (distance) between the desired (target) response d_i and the actual response y_i for training example $i = 1, 2, \dots, N$. Specifically, we define

$$\begin{aligned} \mathcal{E}_s(F) &= \frac{1}{2} \sum_{i=1}^N (d_i - y_i)^2 \\ &= \frac{1}{2} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)]^2 \end{aligned} \quad (5.21)$$

where we have introduced the scaling factor $\frac{1}{2}$ for the sake of consistency with material presented in previous chapters.

2. *Regularizing Term*. This second term, denoted for $\mathcal{E}_c(F)$, depends on the "geometric" properties of the approximating function $F(\mathbf{x})$. Specifically, we write

$$\mathcal{E}_c(F) = \frac{1}{2} \|\mathbf{D}F\|^2 \quad (5.22)$$

where $\mathbf{D}$ is a *linear differential operator*. Prior information about the form of the solution [i.e., the input-output mapping function $F(\mathbf{x})$] is embedded in the operator $\mathbf{D}$, which naturally makes the selection of $\mathbf{D}$ problem-dependent. We also refer to $\mathbf{D}$ as a *stabilizer* because it stabilizes the solution to the regularization problem, making it smooth and thereby satisfying the property of continuity. However, smoothness implies continuity, but the reverse is not necessarily true.

The analytic approach used to handle the situation described in Eq. (5.22) builds on the concept of a *function space*,⁵ which refers to a *normed space*⁶ of functions. In such a space of many (strictly speaking, infinitely many) dimensions, a continuous function is

represented by a *vector*. By using this geometrical image, an insightful link is established between matrices and linear differential operators. The analysis of linear systems thereby becomes translatable to the analysis of linear differential equations (Lanczos, 1964).

Thus, the symbol $\|\cdot\|$ in Eq. (5.22) denotes a norm imposed on the function space to which $\mathbf{D}F(\mathbf{x})$ belongs. Ordinarily the function space used here is the L_2 space that consists of all real-valued functions $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^m$, for which $\|f(\mathbf{x})\|^2$ is Lebesgue integrable. The function $f(\mathbf{x})$ used here denotes the actual function that defines the underlying physical process responsible for generating the set of input-output data pairs $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$; see note 7 for more details.

The quantity to be minimized in regularization theory is

$$\begin{aligned}\mathcal{E}(F) &= \mathcal{E}_s(F) + \lambda \mathcal{E}_c(F) \\ &= \frac{1}{2} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)]^2 + \frac{1}{2} \lambda \|\mathbf{D}F\|^2\end{aligned}\quad (5.23)$$

where λ is a positive real number called the *regularization parameter* and $\mathcal{E}(F)$ is called the *Tikhonov functional*. A functional maps functions (defined in some suitable function space) onto the real line. The minimizer of the Tikhonov functional $\mathcal{E}(F)$ (i.e., the solution to the regularization problem) is denoted by $F_\lambda(\mathbf{x})$.

In a sense, we may view the regularization parameter λ as an indicator of the sufficiency of the given data set as examples that specify the solution $F_\lambda(\mathbf{x})$. In particular, the limiting case $\lambda \rightarrow 0$ implies that the problem is unconstrained, with the solution $F_\lambda(\mathbf{x})$ being completely determined from the examples. The other limiting case, $\lambda \rightarrow \infty$, on the other hand, implies that the prior smoothness constraint imposed by the differential operator $\mathbf{D}$ is by itself sufficient to specify the solution $F_\lambda(\mathbf{x})$, which is another way of saying that the examples are unreliable. In practical applications, the regularization parameter λ is assigned a value somewhere between these two limiting conditions, so that both the sample data and the prior information contribute to the solution $F_\lambda(\mathbf{x})$. Thus the regularizing term $\mathcal{E}_c(F)$ represents a *model complexity-penalty function*, the influence of which on the final solution is controlled by the regularization parameter λ .

Another way of viewing regularization is that it provides a practical solution to the bias-variance dilemma that is discussed in Chapter 2. Specifically, the optimum choice of the regularization parameter λ is designed to steer the solution to the learning problem toward a satisfactory balance between model bias and model variance by incorporating the right amount of prior information into it.

Fréchet Differential of the Tikhonov Functional

The *principle of regularization* may now be stated as:

Find the function $F_\lambda(\mathbf{x})$ that minimizes the Tikhonov functional $\mathcal{E}(F)$, defined by

$$\mathcal{E}(F) = \mathcal{E}_s(F) + \lambda \mathcal{E}_c(F)$$

where $\mathcal{E}_s(F)$ is the standard error term, $\mathcal{E}_c(F)$ is the regularizing term, and λ is the regularization parameter.

To proceed with the minimization of the cost functional $\mathcal{E}(F)$, we need a rule for evaluating the differential of $\mathcal{E}(F)$. We can take care of this matter by using the *Fréchet differential*. In elementary calculus, the tangent to a curve is a straight line that gives the

best approximation of the curve in the neighborhood of the point of tangency. Similarly, the Fréchet differential of a functional may be interpreted as the best local linear approximation. Thus the Fréchet differential of the functional $\mathcal{E}(F)$ is formally defined by (Dorny, 1975; Debnath and Mikusiński, 1990; de Figueiredo and Chen, 1993):

$$d\mathcal{E}(F, h) = \left[\frac{d}{d\beta} \mathcal{E}(F + \beta h) \right]_{\beta=0} \quad (5.24)$$

where $h(\mathbf{x})$ is a fixed function of the vector $\mathbf{x}$. In Eq. (5.24), the ordinary rules of differentiation are used. A necessary condition for the function $F(\mathbf{x})$ to be a relative extremum of the functional $\mathcal{E}(F)$ is that the Fréchet differential $d\mathcal{E}(F, h)$ must be zero at $F(\mathbf{x})$ for all $h \in \mathcal{H}$, as shown by

$$d\mathcal{E}(F, h) = d\mathcal{E}_s(F, h) + \lambda d\mathcal{E}_c(F, h) = 0 \quad (5.25)$$

where $d\mathcal{E}_s(F, h)$ and $d\mathcal{E}_c(F, h)$ are the Fréchet differentials of the functionals $\mathcal{E}_s(F)$ and $\mathcal{E}_c(F)$, respectively.

Evaluating the Fréchet differential of the standard error term $\mathcal{E}_s(F, h)$ of Eq. (5.21), we have

$$\begin{aligned}d\mathcal{E}_s(F, h) &= \left[\frac{d}{d\beta} \mathcal{E}_s(F + \beta h) \right]_{\beta=0} \\ &= \left[\frac{1}{2} \frac{d}{d\beta} \sum_{i=1}^N [d_i - F(\mathbf{x}_i) - \beta h(\mathbf{x}_i)]^2 \right]_{\beta=0} \\ &= - \sum_{i=1}^N [d_i - F(\mathbf{x}_i) - \beta h(\mathbf{x}_i)] h(\mathbf{x}_i) \Big|_{\beta=0} \\ &= - \sum_{i=1}^N [d_i - F(\mathbf{x}_i)] h(\mathbf{x}_i)\end{aligned}\quad (5.26)$$

At this point in the discussion, we find it instructive to invoke the *Riesz representation theorem*, which may be stated as follows (Debnath and Mikusiński, 1990; Kirsch, 1996):

Let f be a bounded linear functional in a Hilbert space (i.e., an inner product space that is complete)⁸ denoted by $\mathcal{H}$. There exists one $h_0 \in \mathcal{H}$ such that

$$f = (h, h_0)_{\mathcal{H}} \quad \text{for all } h \in \mathcal{H}$$

Moreover, we have

$$\|f\|_{\mathcal{H}^*} = \|h_0\|_{\mathcal{H}}$$

where $\mathcal{H}^*$ is the dual or conjugate of the Hilbert space $\mathcal{H}$.

The symbol $(\cdot, \cdot)_{\mathcal{H}}$ used here stands for the *inner (scalar) product* of two functions in $\mathcal{H}$ space. Hence, in light of the Riesz representation theorem, we may rewrite the Fréchet differential $d\mathcal{E}_s(F, h)$ of Eq. (5.26) in the equivalent form

$$d\mathcal{E}_s(F, h) = - \left(h, \sum_{i=1}^N (d_i - F) \delta_{\mathbf{x}_i} \right)_{\mathcal{H}} \quad (5.27)$$

where $\delta_{\mathbf{x}_i}$ denotes the *Dirac delta distribution* of $\mathbf{x}$, centered at $\mathbf{x}_i$; that is,

$$\delta_{\mathbf{x}_i}(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_i) \quad (5.28)$$

Consider next the evaluation of the Fréchet differential of the regularizing term $\mathcal{E}_c(F)$ of Eq. (5.22). Proceeding in a manner similar to that just described, we have

$$\begin{aligned} d\mathcal{E}_c(F, h) &= \frac{d}{d\beta} \mathcal{E}_c(F + \beta h)|_{\beta=0} \\ &= \frac{1}{2} \frac{d}{d\beta} \int_{\mathbb{R}^m} (\mathbf{D}[F + \beta h])^2 d\mathbf{x} |_{\beta=0} \\ &= \int_{\mathbb{R}^m} \mathbf{D}[F + \beta h] \mathbf{D}h d\mathbf{x} |_{\beta=0} \\ &= \int_{\mathbb{R}^m} \mathbf{D}F \mathbf{D}h d\mathbf{x} \\ &= (\mathbf{D}h, \mathbf{D}F)_{\mathcal{H}} \end{aligned} \quad (5.29)$$

where $(\mathbf{D}h, \mathbf{D}F)_{\mathcal{H}}$ is the inner product of the two functions $\mathbf{D}h(\mathbf{x})$ and $\mathbf{D}F(\mathbf{x})$ that result from the action of the differential operator $\mathbf{D}$ on $h(\mathbf{x})$ and $F(\mathbf{x})$, respectively.

Euler-Lagrange Equation

Given a linear differential operator $\mathbf{D}$, we can find a uniquely determined *adjoint operator*, denoted by $\tilde{\mathbf{D}}$, such that for any pair of functions $u(\mathbf{x})$ and $v(\mathbf{x})$ which are sufficiently differentiable and which satisfy proper boundary conditions, we can write (Lanczos, 1964)

$$\int_{\mathbb{R}^m} u(\mathbf{x}) \mathbf{D}v(\mathbf{x}) d\mathbf{x} = \int_{\mathbb{R}^m} v(\mathbf{x}) \tilde{\mathbf{D}}u(\mathbf{x}) d\mathbf{x} \quad (5.30)$$

Equation (5.30) is called *Green's identity*; it provides a mathematical basis for defining the adjoint operator $\tilde{\mathbf{D}}$ in terms of the given differential $\mathbf{D}$. Viewing $\mathbf{D}$ as a matrix, the adjoint operator $\tilde{\mathbf{D}}$ plays a role similar to that of a matrix transpose.

Comparing the left-hand side of Eq. (5.30) with the fourth line of Eq. (5.29), we may make the following identifications:

$$u(\mathbf{x}) = \mathbf{D}F(\mathbf{x})$$

$$\mathbf{D}v(\mathbf{x}) = \mathbf{D}h(\mathbf{x})$$

Using Green's identity, we may rewrite Eq. (5.29) in the equivalent form

$$\begin{aligned} d\mathcal{E}_c(F, h) &= \int_{\mathbb{R}^m} h(\mathbf{x}) \tilde{\mathbf{D}}\mathbf{D}F(\mathbf{x}) d\mathbf{x} \\ &= (h, \tilde{\mathbf{D}}\mathbf{D}F)_{\mathcal{H}} \end{aligned} \quad (5.31)$$

where $\tilde{\mathbf{D}}$ is the adjoint of $\mathbf{D}$.

Returning to the extremum condition described in Eq. (5.25) and substituting the Fréchet differentials of Eqs. (5.27) and (5.31) in that equation, we may now express the Fréchet differential $d\mathcal{E}(F, h)$ as

$$d\mathcal{E}(F, h) = \left(h, \left[\tilde{\mathbf{D}}\mathbf{D}F - \frac{1}{\lambda} \sum_{i=1}^N (d_i - F) \delta_{\mathbf{x}_i} \right] \right)_{\mathcal{H}} \quad (5.32)$$

Since the regularization parameter λ is ordinarily assigned a value somewhere in the open interval $(0, \infty)$, the Fréchet differential $d\mathcal{E}(F, h)$ is zero for every $h(\mathbf{x})$ in $\mathcal{H}$ space if and only if the following condition is satisfied in the distributional sense:

$$\tilde{\mathbf{D}}\mathbf{D}F_{\lambda} - \frac{1}{\lambda} \sum_{i=1}^N (d_i - F) \delta_{\mathbf{x}_i} = 0$$

or equivalently,

$$\tilde{\mathbf{D}}\mathbf{D}F_{\lambda}(\mathbf{x}) = \frac{1}{\lambda} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)] \delta(\mathbf{x} - \mathbf{x}_i) \quad (5.33)$$

Equation (5.33) is the *Euler-Lagrange equation* for the Tikhonov functional $\mathcal{E}(F)$; it defines a necessary condition for the Tikhonov functional $\mathcal{E}(F)$ to have an extremum at $F_{\lambda}(\mathbf{x})$ (Debnath and Mikusiński, 1990).

Green's Function

Equation (5.33) represents a partial differential equation in the approximating function F . The solution of this equation is known to consist of the integral transformation of the right-hand side of the equation.

Let $G(\mathbf{x}, \xi)$ denote a function in which both vectors $\mathbf{x}$ and ξ appear on equal footing but for different purposes: $\mathbf{x}$ as a parameter and ξ as an argument. For a given linear differential operator $\mathbf{L}$, we stipulate that the function $G(\mathbf{x}, \xi)$ satisfies the following conditions (Courant and Hilbert, 1970):

1. For a fixed ξ , $G(\mathbf{x}, \xi)$ is a function of $\mathbf{x}$ and satisfies the prescribed boundary conditions.
2. Except at the point $\mathbf{x} = \xi$, the derivatives of $G(\mathbf{x}, \xi)$ with respect to $\mathbf{x}$ are all continuous; the number of derivatives is determined by the order of the operator $\mathbf{L}$.
3. With $G(\mathbf{x}, \xi)$ considered as a function of $\mathbf{x}$, it satisfies the partial differential equation

$$\mathbf{L}G(\mathbf{x}, \xi) = 0 \quad (5.34)$$

everywhere except at the point $\mathbf{x} = \xi$, where it has a singularity. That is, the function $G(\mathbf{x}, \xi)$ satisfies the following partial differential equation (taken in the sense of distributions)

$$\mathbf{L}G(\mathbf{x}, \xi) = \delta(\mathbf{x} - \xi) \quad (5.35)$$

where, as defined previously, $\delta(\mathbf{x} - \xi)$ is the Dirac delta function positioned at the point $\mathbf{x} = \xi$.

The function $G(\mathbf{x}, \xi)$ thus described is called the *Green's function* for the differential operator $\mathbf{L}$. The Green's function plays a role for a linear differential operator that is similar to that for the inverse matrix for a matrix equation.

Let $\varphi(\mathbf{x})$ denote a continuous or piecewise continuous function of $\mathbf{x} \in \mathbb{R}^m$. Then the function

$$F(\mathbf{x}) = \int_{\mathbb{R}^m} G(\mathbf{x}, \xi) \varphi(\xi) d\xi \quad (5.36)$$

is a solution of the differential equation

$$\mathbf{L}F(\mathbf{x}) = \varphi(\mathbf{x}) \quad (5.37)$$

where $G(\mathbf{x}, \xi)$ is the Green's function for the linear differential operator $\mathbf{L}$ (Courant and Hilbert, 1970).

To prove the validity of $F(\mathbf{x})$ as a solution of Eq. (5.37), apply the differential operator $\mathbf{L}$ to Eq. (5.36), obtaining

$$\begin{aligned} \mathbf{L}F(\mathbf{x}) &= \mathbf{L} \int_{\mathbf{R}^m} G(\mathbf{x}, \xi) \varphi(\xi) d\xi \\ &= \int_{\mathbf{R}^m} \mathbf{L}G(\mathbf{x}, \xi) \varphi(\xi) d\xi \end{aligned} \quad (5.38)$$

The differential operator $\mathbf{L}$ treats ξ as a constant, acting on the kernel $G(\mathbf{x}, \xi)$ only as a function of $\mathbf{x}$. By using Eq. (5.35) in (5.38) we get

$$\mathbf{L}F(\mathbf{x}) = \int_{\mathbf{R}^m} \delta(\mathbf{x} - \xi) \varphi(\xi) d\xi$$

Finally, using the *sifting property* of the Dirac delta function, namely

$$\int_{\mathbf{R}^m} \varphi(\xi) \delta(\mathbf{x} - \xi) d\xi = \varphi(\mathbf{x})$$

we obtain $\mathbf{L}F(\mathbf{x}) = \varphi(\mathbf{x})$, as described in Eq. (5.37).

Solution to the Regularization Problem

Returning to the issue at hand, namely, that of solving the Euler-Lagrange equation (5.33), set

$$\mathbf{L} = \tilde{\mathbf{D}}\mathbf{D} \quad (5.39)$$

and

$$\varphi(\xi) = \frac{1}{\lambda} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)] \delta(\xi - \mathbf{x}_i) \quad (5.40)$$

Then we may use Eq. (5.36) to write

$$\begin{aligned} F_\lambda(\mathbf{x}) &= \int_{\mathbf{R}^m} G(\mathbf{x}, \xi) \left\{ \frac{1}{\lambda} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)] \delta(\xi - \mathbf{x}_i) \right\} d\xi \\ &= \frac{1}{\lambda} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)] \int_{\mathbf{R}^m} G(\mathbf{x}, \xi) \delta(\xi - \mathbf{x}_i) d\xi \end{aligned}$$

where in the last line we have interchanged the order of integration and summation. Finally, using the sifting property of the Dirac delta function, we get the desired solution to the Euler-Lagrange equation (5.33) as follows:

$$F_\lambda(\mathbf{x}) = \frac{1}{\lambda} \sum_{i=1}^N [d_i - F(\mathbf{x}_i)] G(\mathbf{x}, \mathbf{x}_i) \quad (5.41)$$

Equation (5.41) states that the minimizing solution $F_\lambda(\mathbf{x})$ to the regularization problem is a linear superposition of N Green's functions. The $\mathbf{x}_i$ represent the *centers of the expansion*, and the weights $[d_i - F(\mathbf{x}_i)]/\lambda$ represent the *coefficients of the expansion*. In other words, the solution to the regularization problem lies in an N -dimensional subspace of the space of smooth functions, and the set of Green's functions $\{G(\mathbf{x}, \mathbf{x}_i)\}$ centered at $\mathbf{x}_i$, $i = 1, 2, \dots, N$, constitutes a basis for this subspace. (Poggio and Girosi, 1990a). Note that the coefficients of expansion in Eq. (5.41) are first, *linear* in the estimation error defined as the difference between the desired response d_i and the corresponding output $F(\mathbf{x}_i)$ computed by the network, and second, inversely proportional to the regularization parameter λ .

Determination of the Expansion Coefficients

The next issue to be resolved is the determination of the unknown coefficients in the expansion of Eq. (5.41). Let

$$w_i = \frac{1}{\lambda} [d_i - F(\mathbf{x}_i)], \quad i = 1, 2, \dots, N \quad (5.42)$$

We may recast the minimizing solution of Eq. (5.41) simply as follows:

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^N w_i G(\mathbf{x}, \mathbf{x}_i) \quad (5.43)$$

Evaluating Eq. (5.43) at $\mathbf{x}_j$, $j = 1, 2, \dots, N$, we get

$$F_\lambda(\mathbf{x}_j) = \sum_{i=1}^N w_i G(\mathbf{x}_j, \mathbf{x}_i), \quad j = 1, 2, \dots, N \quad (5.44)$$

We now introduce the following definitions:

$$\mathbf{F}_\lambda = [F_\lambda(\mathbf{x}_1), F_\lambda(\mathbf{x}_2), \dots, F_\lambda(\mathbf{x}_N)]^T \quad (5.45)$$

$$\mathbf{d} = [d_1, d_2, \dots, d_N]^T \quad (5.46)$$

$$\mathbf{G} = \begin{bmatrix} G(\mathbf{x}_1, \mathbf{x}_1) & G(\mathbf{x}_1, \mathbf{x}_2) & \cdots & G(\mathbf{x}_1, \mathbf{x}_N) \\ G(\mathbf{x}_2, \mathbf{x}_1) & G(\mathbf{x}_2, \mathbf{x}_2) & \cdots & G(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ G(\mathbf{x}_N, \mathbf{x}_1) & G(\mathbf{x}_N, \mathbf{x}_2) & \cdots & G(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \quad (5.47)$$

$$\mathbf{w} = [w_1, w_2, \dots, w_N]^T \quad (5.48)$$

Then we may rewrite Eqs. (5.42) and (5.44) in matrix form as follows, respectively:

$$\mathbf{w} = \frac{1}{\lambda} (\mathbf{d} - \mathbf{F}_\lambda) \quad (5.49)$$

and

$$\mathbf{F}_\lambda = \mathbf{G}\mathbf{w} \quad (5.50)$$

Eliminating F_λ between Eqs. (5.49) and (5.50) and rearranging terms, we get

$$(\mathbf{G} + \lambda \mathbf{I})\mathbf{w} = \mathbf{d} \quad (5.51)$$

where $\mathbf{I}$ is the N -by- N identity matrix. We call matrix $\mathbf{G}$ the *Green's matrix*.

The differential operator $\mathbf{L}$ defined in Eq. (5.39) is *self-adjoint*, in the sense that its adjoint is equal to operator $\mathbf{L}$ itself. It follows therefore that the associated Green's function $G(\mathbf{x}, \mathbf{x}_i)$ is a *symmetric function*, as shown by

$$G(\mathbf{x}_i, \mathbf{x}_j) = G(\mathbf{x}_j, \mathbf{x}_i) \quad \text{for all } i \text{ and } j \quad (5.52)$$

Equation (5.52) states that the positions of the two points $\mathbf{x}$ and $\mathbf{x}_i$ can be interchanged without affecting the value of the Green's function $G(\mathbf{x}, \mathbf{x}_i)$. Equivalently, the Green's matrix $\mathbf{G}$ defined in Eq. (5.47) is a *symmetric matrix*; that is,

$$\mathbf{G}^T = \mathbf{G} \quad (5.53)$$

We now invoke the interpolation theorem, which is described in Section 5.3 in the context of the interpolation matrix Φ . We first note that Green's matrix $\mathbf{G}$ plays a role in regularization theory similar to that of Φ in RBF interpolation theory. Both $\mathbf{G}$ and Φ are N -by- N symmetric matrices. Accordingly, we may state that the matrix $\mathbf{G}$, for certain classes of Green's functions, is positive definite provided that the data points $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$ are distinct. The classes of Green's functions covered by Micchelli's theorem include inverse multiquadrics and Gaussian functions, but not multiquadrics. In practice, we may always choose λ sufficiently large to ensure that $\mathbf{G} + \lambda \mathbf{I}$ is positive definite and therefore invertible. This, in turn, means that the linear system of equations (5.51) will have a unique solution given by (Poggio and Girosi, 1990a)

$$\mathbf{w} = (\mathbf{G} + \lambda \mathbf{I})^{-1} \mathbf{d} \quad (5.54)$$

Thus, having selected the differential operator $\mathbf{D}$ and therefore having identified the associated Green's function $G(\mathbf{x}, \mathbf{x}_i)$, where $i = 1, 2, \dots, N$, we may use Eq. (5.54) to obtain the weight vector $\mathbf{w}$ for a specified desired response vector $\mathbf{d}$ and an appropriate value of regularization parameter λ .

In conclusion, we may state that the solution to the regularization problem is given by the expansion

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^N w_i G(\mathbf{x}, \mathbf{x}_i) \quad (5.55)$$

where $G(\mathbf{x}, \mathbf{x}_i)$ is the Green's function for the self-adjoint differential operator $\mathbf{L} = \mathbf{D}\mathbf{D}$, and w_i is the i th element of the weight vector $\mathbf{w}$; these two quantities are themselves defined by Eq. (5.35) and (5.54), respectively. Equation (5.55) states the following (Poggio and Girosi, 1990a):

- The regularization approach is equivalent to the expansion of the solution in terms of a set of Green's functions, whose characterization depends only on the form adopted for the stabilizer $\mathbf{D}$ and the associated boundary conditions.
- The number of Green's functions used in the expansion is equal to the number of examples used in the training process.

It should be noted, however, that the solution of the regularization problem given in Eq. (5.55) is incomplete, as it represents a solution *modulo* term $g(\mathbf{x})$ that lies in the

null space of the operator $\mathbf{D}$ (Poggio and Girosi, 1990a). We say this because all the functions that lie in the null space of $\mathbf{D}$ are "invisible" to the smoothing term $\|\mathbf{D}F\|^2$ in the cost functional $\mathcal{E}(F)$ of Eq. (5.23); by the *null space* of $\mathbf{D}$, we mean the set of all functions $g(\mathbf{x})$ for which $\mathbf{D}g$ is zero. The exact form of the additional term $g(\mathbf{x})$ is problem-dependent in the sense that it depends on the stabilizer chosen and the boundary conditions of the problem at hand. For example, it is not needed in the case of a stabilizer $\mathbf{D}$ corresponding to a bell-shaped Green's function such as a Gaussian or inverse multiquadric. For this reason, and since its inclusion does not modify the main conclusions, we will disregard it in the sequel.

The characterization of the Green's function $G(\mathbf{x}, \mathbf{x}_i)$, for a specified center $\mathbf{x}_i$, depends only on the form of the stabilizer $\mathbf{D}$, that is, on the *a priori* assumption made concerning the input-output mapping. If the stabilizer $\mathbf{D}$ is *translationally invariant*, the Green's function $G(\mathbf{x}, \mathbf{x}_i)$ centered at $\mathbf{x}_i$ will depend only on the difference between the arguments $\mathbf{x}$ and $\mathbf{x}_i$; that is,

$$G(\mathbf{x}, \mathbf{x}_i) = G(\mathbf{x} - \mathbf{x}_i) \quad (5.56)$$

If the stabilizer $\mathbf{D}$ is both *translationally* and *rotationally invariant*, the Green's function $G(\mathbf{x}, \mathbf{x}_i)$ will depend only on the *Euclidean norm* of the difference vector $\mathbf{x} - \mathbf{x}_i$, as shown by

$$G(\mathbf{x}, \mathbf{x}_i) = G(\|\mathbf{x} - \mathbf{x}_i\|) \quad (5.57)$$

Under these conditions, the Green's function must be a *radial-basis function*. In such a case, the regularized solution of Eq. (5.55) takes on the following special form (Poggio and Girosi, 1990a):

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^N w_i G(\|\mathbf{x} - \mathbf{x}_i\|) \quad (5.58)$$

The solution described in Eq. (5.58) constructs a linear function space that depends on the known data points according to the Euclidean distance measure.

The solution described by Eq. (5.58) is termed *strict interpolation*, since all the N data points available for training are used to generate the interpolating function $F(\mathbf{x})$. It is important, however, to realize that this solution differs from that of Eq. (5.11) in a fundamental respect: The solution of Eq. (5.58) is *regularized* by virtue of the definition given in Eq. (5.54) for the weight vector $\mathbf{w}$. It is only when we set the regularization parameter λ equal to zero that the two solutions may become one and the same.

Multivariate Gaussian Functions

The Green's function $G(\mathbf{x}, \mathbf{x}_i)$, whose linear differential operator $\mathbf{D}$ is both translationally and rotationally invariant and which satisfies the condition of Eq. (5.57), is of particular interest in practice. An example of such a Green's function is the *multivariate Gaussian function* defined by

$$G(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\frac{1}{2\sigma_i^2} \|\mathbf{x} - \mathbf{x}_i\|^2\right) \quad (5.59)$$

where $\mathbf{x}_i$ denotes the center of the function and σ_i denotes its width. The self-adjoint operator $\mathbf{L} = \tilde{\mathbf{D}}\mathbf{D}$ defining the Green's function of Eq. (5.59) is given by (Poggio and Girosi, 1990a)

$$\mathbf{L} = \sum_{n=0}^{\infty} (-1)^n \alpha_n \nabla^{2n} \quad (5.60)$$

where

$$\alpha_n = \frac{\sigma_i^{2n}}{n!2^n} \quad (5.61)$$

and ∇^{2n} is the iterated Laplace operator in m_0 dimensions, with

$$\nabla^2 = \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \cdots + \frac{\partial^2}{\partial x_{m_0}^2} \quad (5.62)$$

With the number of terms permitted to go to infinity in Eq. (5.60), $\mathbf{L}$ ceases to be a differential operator in the standard sense. For this reason, the operator $\mathbf{L}$ in Eq. (5.60) is referred to as a *pseudo-differential operator*.

Since by definition, $\mathbf{L} = \tilde{\mathbf{D}}\mathbf{D}$, we deduce from Eq. (5.60) that the operator $\mathbf{D}$ and its adjoint $\tilde{\mathbf{D}}$ are as follows, respectively (see note 10):

$$\begin{aligned} \mathbf{D} &= \sum_n \alpha_n^{1/2} \left(\frac{\partial}{\partial x_1} + \frac{\partial}{\partial x_2} + \cdots + \frac{\partial}{\partial x_{m_0}} \right)^n \\ &= \sum_{a+b+\cdots+k=n} \alpha_n^{1/2} \frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k}, \end{aligned} \quad (5.63)$$

and

$$\begin{aligned} \tilde{\mathbf{D}} &= \sum_n (-1)^n \alpha_n^{1/2} \left(\frac{\partial}{\partial x_1} + \frac{\partial}{\partial x_2} + \cdots + \frac{\partial}{\partial x_{m_0}} \right)^n \\ &= \sum_{a+b+\cdots+k=n} (-1)^n \alpha_n^{1/2} \frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k} \end{aligned} \quad (5.64)$$

Thus the regularized solution described in Eq. (5.58) is attained by using a stabilizer that includes all of its possible partial derivatives.

Using Eqs. (5.59) to (5.61) in (5.35) with ξ set equal to $\mathbf{x}_i$, we may write

$$\sum_n (-1)^n \frac{\sigma_i^{2n}}{n!2^n} \nabla^{2n} \exp\left(-\frac{1}{2\sigma_i^2} \|\mathbf{x} - \mathbf{x}_i\|^2\right) = \delta(\mathbf{x} - \mathbf{x}_i) \quad (5.65)$$

With the Green's function $G(\mathbf{x}, \mathbf{x}_i)$ defined by the special form of Eq. (5.59), the regularized solution given in Eq. (5.55) takes the form of a linear superposition of multivariate Gaussian functions as follows:

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^N w_i \exp\left(-\frac{1}{2\sigma_i^2} \|\mathbf{x} - \mathbf{x}_i\|^2\right) \quad (5.66)$$

where the linear weights, w_i , are themselves defined by (5.42).

In Eq. (5.66), the individual Gaussian members of the sum defining the approximating function $F(\mathbf{x})$ are assigned different variances. To simplify matters, the condition $\sigma_i = \sigma$ for all i is often imposed on $F(\mathbf{x})$. Even though RBF networks thus designed are of a somewhat restricted kind, they are still universal approximators (Park and Sandberg, 1991).

5.6 REGULARIZATION NETWORKS

The expansion of the regularized approximating function $F_\lambda(\mathbf{x})$ given in Eq. (5.55) in terms of the Green's function $G(\mathbf{x}, \mathbf{x}_i)$ centered at $\mathbf{x}_i$ suggests the network structure shown in Fig. 5.4 as a method for its implementation. For obvious reasons, this network is called a *regularization network* (Poggio and Girosi, 1990a). As with the network described in section 5.1, it consists of three layers. The first layer is composed of input (source) nodes whose number is equal to the dimension m_0 of the input vector $\mathbf{x}$ (i.e., the number of independent variables of the problem). The second layer is a hidden layer, composed of nonlinear units that are connected *directly* to all of the nodes in the input layer. There is one hidden unit for each data point $\mathbf{x}_i$, $i = 1, 2, \dots, N$, where N is the size of the training sample. The activation functions of the individual hidden units are defined by the Green's functions. Accordingly, the output of the i th hidden unit is $G(\mathbf{x}, \mathbf{x}_i)$. The output layer consists of a single linear unit, being fully connected to the hidden layer. By "linearity" we mean that the output of the network is a linearly weighted sum of the outputs of the hidden units. The weights of the output layer are the unknown coefficients of the expansion, defined in terms of the Green's functions $G(\mathbf{x}, \mathbf{x}_i)$ and the regularization parameter λ as seen in Eq. (5.54). Figure 5.4 depicts the architecture of the regularization network for a single output. Clearly, such an architecture can be readily extended to accommodate any number of network outputs desired.

The regularization network shown in Fig. 5.5 assumes that the Green's function $G(\mathbf{x}, \mathbf{x}_i)$ is *positive definite* for all i . Provided that this condition is satisfied, which it is in the case of the $G(\mathbf{x}, \mathbf{x}_i)$ having the Gaussian form given in Eq. (5.59), for example, then

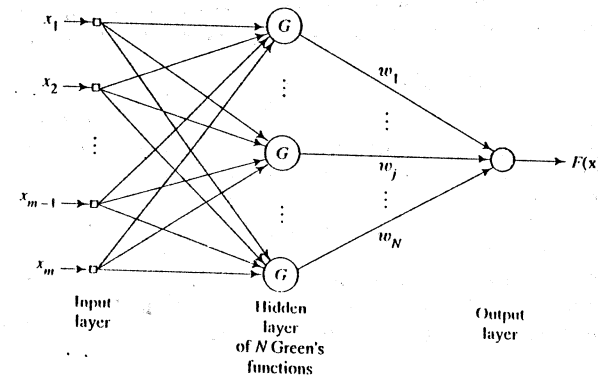


FIGURE 5.4 Regularization network.

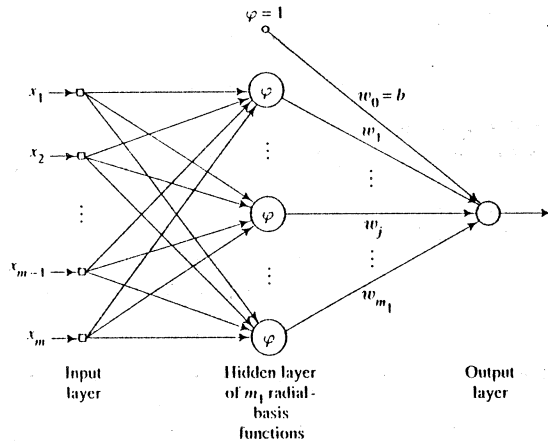


FIGURE 5.5 Radial-basis function network.

the solution produced by this network will be an “optimal” interpolant in the sense that it minimizes the functional $\mathcal{E}(F)$. Moreover, from the viewpoint of approximation theory, the regularization network has three desirable properties (Poggio and Girosi, 1990a):

1. The regularization network is a *universal approximator* in that it can approximate arbitrarily well any multivariate continuous function on a compact subset of $\mathbb{R}^{m_0}$, given a sufficiently large number of hidden units.
2. Since the approximation scheme derived from regularization theory is linear in the unknown coefficients, it follows that the regularization network has the *best-approximation property*. This means that given an unknown nonlinear function f , there always exists a choice of coefficients that approximates f better than all other possible choices.
3. The solution computed by the regularization network is *optimal*. Optimality here means that the regularization network minimizes a functional that measures how much the solution deviates from its true value as represented by the training data.

7 GENERALIZED RADIAL-BASIS FUNCTION NETWORKS

The one-to-one correspondence between the training input data $\mathbf{x}_i$ and the Green's function $G(\mathbf{x}, \mathbf{x}_i)$ for $i = 1, 2, \dots, N$ produces a regularization network that may sometimes be considered prohibitively expensive to implement in computational terms for large N . Specifically, the computation of the linear weights of the network (i.e., the coefficients of the expansion in Eq. (5.55) requires the inversion of an N -by- N matrix, which therefore grows polynomially with N (roughly as N^3). Furthermore, the likelihood of *ill conditioning* is higher for larger matrices; the *condition number* of a matrix

is defined as the ratio of the largest eigenvalue to the smallest eigenvalue of the matrix. To overcome these computational difficulties, the complexity of the network would have to be reduced, which requires an approximation to the regularized solution.

The approach taken involves searching for a suboptimal solution in a lower-dimensional space that approximates the regularized solution of Eq. (5.55). This is done by using a standard technique known in variational problems as *Galerkin's method*. According to this technique, the approximated solution $F^*(\mathbf{x})$ is expanded on a finite basis, as shown by (Poggio and Girosi, 1990a)

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i \phi_i(\mathbf{x}) \quad (5.67)$$

where $\{\phi_i(\mathbf{x}) | i = 1, 2, \dots, m_1\}$ is a new set of basis functions that we assume to be linearly independent without loss of generality. Typically, the number of basis functions is less than the number of data points (i.e., $m_1 \leq N$), and the w_i constitute a new set of weights. With radial-basis functions in mind, we set

$$\phi_i(\mathbf{x}) = G(\|\mathbf{x} - \mathbf{t}_i\|), \quad i = 1, 2, \dots, m_1 \quad (5.68)$$

where the set of centers $\{\mathbf{t}_i | i = 1, 2, \dots, m_1\}$ is to be determined. This particular choice of basis functions is the only one that guarantees that in the case of $m_1 = N$, and

$$\mathbf{t}_i = \mathbf{x}_i, \quad i = 1, 2, \dots, N$$

the correct solution of Eq. (5.58) is consistently recovered. Thus, using Eq. (5.68) in (5.67), we may redefine $F^*(\mathbf{x})$ as

$$\begin{aligned} F^*(\mathbf{x}) &= \sum_{i=1}^{m_1} w_i G(\mathbf{x}, \mathbf{t}_i) \\ &= \sum_{i=1}^{m_1} w_i G(\|\mathbf{x} - \mathbf{t}_i\|) \end{aligned} \quad (5.69)$$

Given the expansion of Eq. (5.69) for the approximating function $F^*(\mathbf{x})$, the problem we now address is the determination of the new set of weights $\{w_i | i = 1, 2, \dots, m_1\}$ so as to minimize the new cost functional $\mathcal{E}(F^*)$ defined by

$$\mathcal{E}(F^*) = \sum_{i=1}^N \left(d_i - \sum_{j=1}^{m_1} w_j G(\|\mathbf{x}_i - \mathbf{t}_j\|) \right)^2 + \lambda \|D F^*\|^2 \quad (5.70)$$

The first term on the right-hand side of Eq. (5.70) may be expressed as the squared Euclidean norm $\|\mathbf{d} - G\mathbf{w}\|^2$, where

$$\mathbf{d} = [d_1, d_2, \dots, d_N]^T \quad (5.71)$$

$$G = \begin{bmatrix} G(\mathbf{x}_1, \mathbf{t}_1) & G(\mathbf{x}_1, \mathbf{t}_2) & \cdots & G(\mathbf{x}_1, \mathbf{t}_{m_1}) \\ G(\mathbf{x}_2, \mathbf{t}_1) & G(\mathbf{x}_2, \mathbf{t}_2) & \cdots & G(\mathbf{x}_2, \mathbf{t}_{m_1}) \\ \vdots & \vdots & \ddots & \vdots \\ G(\mathbf{x}_N, \mathbf{t}_1) & G(\mathbf{x}_N, \mathbf{t}_2) & \cdots & G(\mathbf{x}_N, \mathbf{t}_{m_1}) \end{bmatrix} \quad (5.72)$$

$$\mathbf{w} = [w_1, w_2, \dots, w_{m_1}]^T \quad (5.73)$$

The desired response vector $\mathbf{d}$ is N -dimensional as before. However, the matrix $\mathbf{G}$ of Green's functions and the weight vector $\mathbf{w}$ have different dimensions; the matrix $\mathbf{G}$ is now N -by- m_1 and therefore no longer symmetric, and the vector $\mathbf{w}$ is m_1 -by-1. From Eq. (5.69) we note that the approximating function F^* is a linear combination of the Green's functions for the stabilizer $\mathbf{D}$. Accordingly, we may express the second term on the right-hand side of Eq. (5.70) as

$$\begin{aligned}\|\mathbf{D}F^*\|^2 &= (\mathbf{D}F^*, \mathbf{D}F^*)_H \\ &= \left[\sum_{i=1}^{m_1} w_i G(\mathbf{x}, \mathbf{t}_i), \tilde{\mathbf{D}} \mathbf{D} \sum_{i=1}^{m_1} w_i G(\mathbf{x}, \mathbf{t}_i) \right]_H \\ &= \left[\sum_{i=1}^{m_1} w_i G(\mathbf{x}, \mathbf{t}_i), \sum_{i=1}^{m_1} w_i \delta \mathbf{t}_i \right]_H \\ &= \sum_{i=1}^{m_1} \sum_{j=1}^{m_1} w_i w_j G(\mathbf{t}_i, \mathbf{t}_j) \\ &= \mathbf{w}^T \mathbf{G}_0 \mathbf{w}\end{aligned}\quad (5.74)$$

where in the second and third lines we made use of the definition of an adjoint operator and Eq. (5.35), respectively. The matrix $\mathbf{G}_0$ is a symmetric m_1 -by- m_1 matrix, defined by

$$\mathbf{G}_0 = \begin{bmatrix} G(\mathbf{t}_1, \mathbf{t}_1) & G(\mathbf{t}_1, \mathbf{t}_2) & \cdots & G(\mathbf{t}_1, \mathbf{t}_{m_1}) \\ G(\mathbf{t}_2, \mathbf{t}_1) & G(\mathbf{t}_2, \mathbf{t}_2) & \cdots & G(\mathbf{t}_2, \mathbf{t}_{m_1}) \\ \vdots & \vdots & \ddots & \vdots \\ G(\mathbf{t}_{m_1}, \mathbf{t}_1) & G(\mathbf{t}_{m_1}, \mathbf{t}_2) & \cdots & G(\mathbf{t}_{m_1}, \mathbf{t}_{m_1}) \end{bmatrix} \quad (5.75)$$

Thus the minimization of Eq. (5.70) with respect to the weight vector $\mathbf{w}$ yields the result (see Problem 5.5)

$$(\mathbf{G}^T \mathbf{G} + \lambda \mathbf{G}_0) \mathbf{w} = \mathbf{G}^T \mathbf{d} \quad (5.76)$$

As the regularization parameter λ approaches zero, the weight vector $\mathbf{w}$ converges to the pseudoinverse (minimum-norm) solution to the overdetermined least-squares data-fitting problem for $m_1 < N$, as shown by (Broomhead and Lowe, 1988)

$$\mathbf{w} = \mathbf{G}^+ \mathbf{d}, \quad \lambda = 0 \quad (5.77)$$

where $\mathbf{G}^+$ is the pseudoinverse of matrix $\mathbf{G}$; that is,

$$\mathbf{G}^+ = (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{G}^T \quad (5.78)$$

Weighted Norm

The norm in the approximate solution of Eq. (5.69) is ordinarily intended to be a Euclidean norm. When, however, the individual elements of the input vector $\mathbf{x}$ belong to different classes, it is more appropriate to consider a general *weighted norm*, the squared form of which is defined by (Poggio and Girosi, 1990a)

$$\begin{aligned}\|\mathbf{x}\|_C^2 &= (\mathbf{C}\mathbf{x})^T (\mathbf{C}\mathbf{x}) \\ &= \mathbf{x}^T \mathbf{C}^T \mathbf{C} \mathbf{x}\end{aligned}\quad (5.79)$$

where $\mathbf{C}$ is an m_0 -by- m_0 norm weighting matrix, and m_0 is the dimension of the input vector $\mathbf{x}$.

Using the definition of weighted norm, we may now rewrite the approximation to the regularized solution given in Eq. (5.69) in the more generalized form (Lowe, 1989; Poggio and Girosi, 1990a)

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G(\|\mathbf{x} - \mathbf{t}_i\|_C) \quad (5.80)$$

The use of a weighted norm may be interpreted in two ways. We may simply view it as applying an *affine transformation* to the original input space. In principle, allowing for such a transformation cannot degrade results from the default case, since it actually corresponds to an identity norm-weighting matrix. On the other hand, the weighted norm follows directly from a slight generalization of the m_0 -dimensional Laplacian in the definition of the pseudo-differential operator $\mathbf{D}$ in Eq. (5.63); see Problem 5.6. The use of a weighted norm may also be justified in the context of Gaussian radial-basis functions on the following grounds. A Gaussian radial-based function $G(\|\mathbf{x} - \mathbf{t}_i\|_C)$ centered at $\mathbf{t}_i$ and with norm weighting matrix $\mathbf{C}$ may be expressed as

$$\begin{aligned}G(\|\mathbf{x} - \mathbf{t}_i\|_C) &= \exp[-(\mathbf{x} - \mathbf{t}_i)^T \mathbf{C}^T \mathbf{C} (\mathbf{x} - \mathbf{t}_i)] \\ &= \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{t}_i)^T \Sigma^{-1} (\mathbf{x} - \mathbf{t}_i)\right]\end{aligned}\quad (5.81)$$

where the inverse matrix Σ^{-1} is defined by

$$\frac{1}{2} \Sigma^{-1} = \mathbf{C}^T \mathbf{C} \quad (5.82)$$

Equation (5.81) represents a multivariate Gaussian distribution with mean vector $\mathbf{t}_i$ and covariance matrix Σ . As such, it represents a generalization of the distribution described in Eq. (5.59).

The solution to the approximation problem given in Eq. (5.70) provides the framework for the *generalized radial-basis function (RBF) network* having the structure shown in Fig. 5.5. In this network, provision is made for a bias (i.e., data-independent variable) applied to the output unit. This is done simply by setting one of the linear weights in the output layer of the network equal to the bias and treating the associated radial-basis function as a constant equal to +1.

In structural terms, the generalized RBF network of Fig. 5.5 is similar to the regularization RBF network of Fig. 5.4. However, they differ from each other in two important ways:

1. The number of nodes in the hidden layer of the generalized RBF network of Fig. 5.5 is m_1 , where m_1 is ordinarily smaller than the number N of examples available for training. On the other hand, the number of hidden nodes in the regularization RBF network of Fig. 5.4 is exactly N .
2. In the generalized RBF network of Fig. 5.5, the linear weights associated with the output layer, and the positions of the centers of the radial-basis functions and the

norm weighting matrix associated with the hidden layer, are all unknown parameters that have to be learned. However, the activation functions of the hidden layer in the regularization RBF network of Fig. 5.4 are known, being defined by a set of Green's functions centered at the training data points; the linear weights of the output layer are the only unknown parameters of the network.

Receptive Field

The covariance matrix Σ determines the receptive field of the Gaussian radial-basis function $G(\|x - t_i\|_C)$ given in Eq. (5.81). For a prescribed center t_i , the *receptive field* of $G(\|x - t_i\|_C)$ is formally defined as the support of the function

$$\Psi(x) = G(\|x - t_i\|_C) - a \quad (5.83)$$

where a is some positive constant (Xu et al., 1994). In other words, the receptive field of $G(\|x - t_i\|_C)$ is that particular subset of the domain of the input vector x for which $G(\|x - t_i\|_C)$ takes sufficiently large values, greater than the prescribed level a .

In a manner corresponding to the way in which the norm-weighting matrix C was defined, we may identify three different scenarios pertaining to the covariance matrix Σ and its influence on the shape, size, and orientation of the receptive field:

1. $\Sigma = \sigma^2 I$, where I is the identity matrix and σ^2 is a common variance. In this case, the receptive field of $G(\|x - t_i\|_C)$ consists of a hypersphere centered at t_i and with a radius determined by σ .
2. $\Sigma = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_{m_0}^2)$, where σ_j^2 is the variance of the j th element of the input vector x and $j = 1, 2, \dots, m_0$. In this second case, the receptive field of $G(\|x - t_i\|_C)$ consists of a hyperellipse whose individual axes coincide with those of the input space, and with its extension along the j th axis being determined by σ_j .
3. Σ is a nondiagonal matrix. By definition, Σ is a positive definite matrix. We may therefore use the similarity transformation of matrix algebra to decompose Σ as follows:

$$\Sigma = Q^T \Lambda Q \quad (5.84)$$

where Λ is a diagonal matrix and Q is an orthonormal rotation matrix. The matrix Λ determines the shape and size of the receptive field, while the matrix Q determines its orientation.

XOR PROBLEM (REVISITED)

Consider again the XOR (Exclusive OR) problem, which we solved in Chapter 4 using a multilayer perceptron with a single hidden layer. Here we are going to present a solution to this same problem by using an RBF network.

The RBF network to be investigated consists of a pair of Gaussian functions, defined as:

$$G(\|x - t_i\|) = \exp(-\|x - t_i\|^2), \quad i = 1, 2 \quad (5.85)$$

where the centers t_1 and t_2 are

$$t_1 = [1, 1]^T$$

$$t_2 = [0, 0]^T$$

For the characterization of the output unit, we assume the following:

1. The output unit uses *weight-sharing*, which is justified by virtue of the symmetry of the problem; this is a form of prior information being built into the design of the network. With only two hidden units, we therefore only have a single weight w to be determined.
2. The output unit includes a bias b (i.e., data-independent variable). The significance of this bias is that the desired output values of the XOR function have nonzero mean.

Thus the structure of the RBF network proposed for solving the XOR problem is as shown in Fig. 5.6. The input-output relation of the network is defined by

$$y(x) = \sum_{i=1}^2 w G(\|x - t_i\|) + b \quad (5.86)$$

To fit the training data of Table 5.2, we require that

$$y(x_j) = d_j, \quad j = 1, 2, 3, 4 \quad (5.87)$$

where x_j is an input vector and d_j is the corresponding value of the desired output. Let

$$g_{ji} = G(\|x_j - t_i\|), \quad j = 1, 2, 3, 4; i = 1, 2 \quad (5.88)$$

Then, using the values of Table 5.2 in Eq. (5.88), we get the following set of equations written in matrix form:

$$Gw = d \quad (5.89)$$

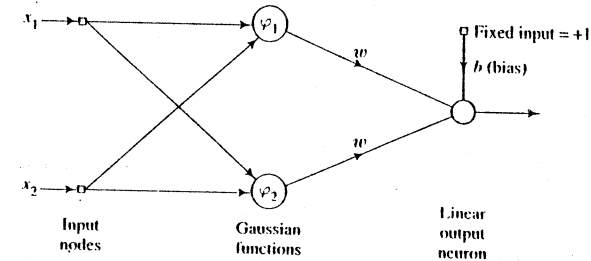


FIGURE 5.6 RBF network for solving the XOR problem.

TABLE 5.2 Input-Output Transformation
Computed for XOR Problem

Data Point, j	Input Pattern, $\mathbf{x}_j$	Desired Output, d_j
1	(1, 1)	0
2	(0, 1)	1
3	(0, 0)	0
4	(1, 0)	1

where

$$\mathbf{G} = \begin{bmatrix} 1 & 0.1353 & 1 \\ 0.3678 & 0.3678 & 1 \\ 0.1353 & 1 & 1 \\ 0.3678 & 0.3678 & 1 \end{bmatrix} \quad (5.90)$$

$$\mathbf{d} = [0 \quad 1 \quad 0 \quad 1]^T \quad (5.91)$$

$$\mathbf{w} = [w \quad w \quad b]^T \quad (5.92)$$

The problem described here is *overdetermined in the sense that we have more data points than free parameters*. This explains why the matrix $\mathbf{G}$ is not square. Consequently, no unique inverse exists for the matrix $\mathbf{G}$. To overcome this difficulty, we use the *minimum-norm* solution of Eq. (5.78), and so write

$$\begin{aligned} \mathbf{w} &= \mathbf{G}^+ \mathbf{d} \\ &= (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{G}^T \mathbf{d} \end{aligned} \quad (5.93)$$

Note that $\mathbf{G}^T \mathbf{G}$ is a square matrix with a unique inverse of its own. Substituting Eq. (5.90) in (5.93), we get

$$\mathbf{G}^+ = \begin{bmatrix} 1.8292 & -1.2509 & 0.6727 & -1.2509 \\ 0.6727 & -1.2509 & 1.8292 & -1.2509 \\ -0.9202 & 1.4202 & -0.9202 & 1.4202 \end{bmatrix} \quad (5.94)$$

Finally, substituting Eqs. (5.91) and (5.94) in (5.93), we get

$$\mathbf{w} = \begin{bmatrix} -2.5018 \\ -2.5018 \\ +2.8404 \end{bmatrix}$$

which completes the specification of the RBF network.

9 ESTIMATION OF THE REGULARIZATION PARAMETER

The regularization parameter λ plays a central role in the regularization theory of radial-basis function networks presented in Sections 5.5 through 5.7. To derive the full benefit of this theory we need an equally principled approach to the estimation of λ .

To fix ideas, consider a *nonlinear regression problem*, described by a model whose observable output y_i at time step i in response to an input vector $\mathbf{x}_i$ is defined by

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \quad i = 1, 2, \dots, N \quad (5.95)$$

where $f(\mathbf{x}_i)$ is a "smooth curve," and ϵ_i is a sample drawn from a white noise process of zero mean and variance σ^2 . That is

$$E[\epsilon_i] = 0 \quad \text{for all } i \quad (5.96)$$

and

$$E[\epsilon_i \epsilon_k] = \begin{cases} \sigma^2 & \text{for } k = i \\ 0 & \text{otherwise} \end{cases} \quad (5.97)$$

The problem is to reconstruct the underlying function of the model, $f(\mathbf{x}_i)$, given the training sample $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$.

Let $F_\lambda(\mathbf{x})$ be the regularized estimate of $f(\mathbf{x})$ for some value of the regularization parameter λ . That is, $F_\lambda(\mathbf{x})$ is the minimizer of the Tikhonov functional formulated for the nonlinear regression problem as:

$$\mathcal{E}(F) = \frac{1}{2} \sum_{i=1}^N [y_i - F(\mathbf{x}_i)]^2 + \frac{\lambda}{2} \|\mathbf{D}F(\mathbf{x})\|^2 \quad (5.98)$$

It is a nontrivial matter to choose a suitable value for λ , which controls the tradeoff between two conflicting issues:

- "Roughness" of the solution as measured by the term $\|\mathbf{D}F(\mathbf{x})\|^2$
- "Infidelity" of the data, as measured by the term $\sum_{i=1}^N [y_i - F(\mathbf{x}_i)]^2$

A good choice for the regularization parameter λ is the subject matter of this section.

Average Squared Error

Let $R(\lambda)$ denote the "average squared error over a given data set" between two functions: the regression function $f(\mathbf{x})$ pertaining to the model and the approximating function $F_\lambda(\mathbf{x})$ representing the solution for some λ , evaluated over the entire data set. That is,

$$R(\lambda) = \frac{1}{N} \sum_{i=1}^N [f(\mathbf{x}_i) - F_\lambda(\mathbf{x}_i)]^2 \quad (5.99)$$

The *optimum* λ is the particular value of λ that minimizes $R(\lambda)$.

Let $F_\lambda(\mathbf{x}_k)$ be expressed as a linear combination of the given set of observables as follows:

$$F_\lambda(\mathbf{x}_k) = \sum_{i=1}^N a_{ki}(\lambda) y_i \quad (5.100)$$

In matrix form, we may equivalently write

$$\mathbf{F}_\lambda = \mathbf{A}(\lambda) \mathbf{y} \quad (5.101)$$

where

$$\mathbf{F}_\lambda = [F_\lambda(\mathbf{x}_1), F_\lambda(\mathbf{x}_2), \dots, F_\lambda(\mathbf{x}_N)]^T$$

$$\mathbf{y} = [y_1, y_2, \dots, y_N]^T$$

and

$$\mathbf{A}(\lambda) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \quad (5.102)$$

The N -by- N matrix $\mathbf{A}(\lambda)$ is called the *influence matrix*.

Using this matrix notation, we may rewrite Eq. (5.99) in the form

$$R(\lambda) = \frac{1}{N} \|\mathbf{f} - \mathbf{F}_\lambda\|^2$$

$$= \frac{1}{N} \|\mathbf{f} - \mathbf{A}(\lambda)\mathbf{y}\|^2 \quad (5.103)$$

where the N -by-1 vector $\mathbf{f}$ is

$$\mathbf{f} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)]^T$$

We can go one step further in our matrix formulation by rewriting Eq. (5.95) in the form

$$\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon} \quad (5.104)$$

where

$$\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, \dots, \epsilon_N]^T$$

Hence, using Eq. (5.104) in (5.103) and then expanding terms, we obtain

$$R(\lambda) = \frac{1}{N} \|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} - \mathbf{A}(\lambda)\boldsymbol{\epsilon}\|^2$$

$$= \frac{1}{N} \|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f}\|^2 - \frac{2}{N} \boldsymbol{\epsilon}^T \mathbf{A}(\lambda)(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} \quad (5.105)$$

$$+ \frac{1}{N} \|\mathbf{A}(\lambda)\boldsymbol{\epsilon}\|^2$$

where $\mathbf{I}$ is the N -by- N identity matrix. To find the expected value of $R(\lambda)$, note the following points:

- The first term on the right-hand side of Eq. (5.105) is a constant and is therefore unaffected by the expectation operator.
- The expectation of the second term is zero by virtue of Eq. (5.96)

- The expectation of the scalar $\|\mathbf{A}(\lambda)\boldsymbol{\epsilon}\|^2$ is

$$E[\|\mathbf{A}(\lambda)\boldsymbol{\epsilon}\|^2] = E[\boldsymbol{\epsilon}^T \mathbf{A}^T(\lambda) \mathbf{A}(\lambda) \boldsymbol{\epsilon}]$$

$$= \text{tr}\{E[\boldsymbol{\epsilon}^T \mathbf{A}^T(\lambda) \mathbf{A}(\lambda) \boldsymbol{\epsilon}]\}$$

$$= E[\text{tr}\{\boldsymbol{\epsilon}^T \mathbf{A}^T(\lambda) \mathbf{A}(\lambda) \boldsymbol{\epsilon}\}] \quad (5.106)$$

where we first used the fact that the trace of a scalar is the same as the scalar itself, and then interchanged the order of expectation and trace operation.

Next we may use this rule in matrix algebra: Given two matrices $\mathbf{B}$ and $\mathbf{C}$ of compatible dimensions, the trace of $\mathbf{BC}$ is equal to the trace of $\mathbf{CB}$. Thus, setting $\mathbf{B} = \boldsymbol{\epsilon}^T$ and $\mathbf{C} = \mathbf{A}^T(\lambda) \mathbf{A}(\lambda) \boldsymbol{\epsilon}$, we may rewrite Eq. (5.106) in the equivalent form

$$E[\|\mathbf{A}(\lambda)\boldsymbol{\epsilon}\|^2] = E[\text{tr}\{\mathbf{A}^T(\lambda) \mathbf{A}(\lambda) \boldsymbol{\epsilon} \boldsymbol{\epsilon}^T\}]$$

$$= \sigma^2 \text{tr}[\mathbf{A}^T(\lambda) \mathbf{A}(\lambda)] \quad (5.107)$$

where in the last line we have made use of Eq. (5.97). Finally, noting that the trace of $\mathbf{A}^T(\lambda) \mathbf{A}(\lambda)$ is the same as the trace of $\mathbf{A}^2(\lambda)$, we may write

$$E[\|\mathbf{A}(\lambda)\boldsymbol{\epsilon}\|^2] = \sigma^2 \text{tr}[\mathbf{A}^2(\lambda)] \quad (5.108)$$

Putting these three results together, we may express the expected value of $R(\lambda)$ as

$$E[R(\lambda)] = \frac{1}{N} \|\mathbf{I} - \mathbf{A}(\lambda)\mathbf{f}\|^2 + \frac{\sigma^2}{N} \text{tr}[\mathbf{A}^2(\lambda)] \quad (5.109)$$

The average squared error over a given data set, $R(\lambda)$, however, is not a practical measure because it requires knowledge of the regression function $f(\mathbf{x})$, the function that is to be reconstructed. As an estimate of $E[R(\lambda)]$, we introduce the following definition (Craven and Wahba, 1979)

$$\hat{R}(\lambda) = \frac{1}{N} \|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2 + \frac{\sigma^2}{N} \text{tr}[\mathbf{A}^2(\lambda)] - \frac{\sigma^2}{N} \text{tr}[(\mathbf{I} - \mathbf{A}(\lambda))^2] \quad (5.110)$$

This estimate is *unbiased*, in that (following a procedure similar to that described for deriving Eq. (5.109)), we may show that

$$E[\hat{R}(\lambda)] = E[R(\lambda)] \quad (5.111)$$

Accordingly, the minimizer of the estimate $\hat{R}(\lambda)$ can be taken as a good choice for the regularization parameter λ .

Generalized Cross-Validation

A drawback of the estimate $\hat{R}(\lambda)$ is that it requires knowledge of the noise variance σ^2 . In situations encountered in practice, σ^2 is usually not known. To deal with situations of this kind, we may use the concept of generalized cross-validation that was originated by Craven and Wahba (1979).

We begin by adapting the ordinary leave-one-out form of cross-validation (described in Chapter 4) to the problem at hand. Specifically, let $F_{\lambda}^{[k]}(\mathbf{x})$ be the minimizer of the functional

$$\mathcal{E}(F) = \frac{1}{2} \sum_{i=1}^N [y_i - F_{\lambda}(\mathbf{x}_i)]^2 + \frac{\lambda}{2} \|DF(\mathbf{x})\|^2 \quad (5.112)$$

where the k th term $[y_k - F_{\lambda}(\mathbf{x}_k)]$ has been left out of the standard error term. By leaving out this term, we may take the ability of $F_{\lambda}^{[k]}(\mathbf{x})$ to "predict" the missing data point y_k as a measure of the goodness of λ . Accordingly, we may introduce the following measure of goodness

$$V_0(\lambda) = \frac{1}{N} \sum_{k=1}^N [y_k - F_{\lambda}^{[k]}(\mathbf{x}_k)]^2 \quad (5.113)$$

which depends on the data alone. The *ordinary cross-validation estimate* of λ is thus defined to be the minimizer of $V_0(\lambda)$ (Wahba, 1990).

A useful property of $F_{\lambda}^{[k]}(\mathbf{x}_k)$ is that if the data point y_k is replaced by the prediction $F_{\lambda}^{[k]}(\mathbf{x}_k)$, and the original Tikhonov functional $\mathcal{E}(F)$ of Eq. (5.98) is minimized using the data points $y_1, y_2, \dots, y_{k-1}, y_k, y_{k+1}, \dots, y_N$, we get $F_{\lambda}^{[k]}(\mathbf{x}_k)$ for the solution. This property, together with the fact that for each input vector $\mathbf{x}$ the minimizer $F_{\lambda}(\mathbf{x})$ of $\mathcal{E}(F)$ depends linearly on y_k , allows us to write:

$$F_{\lambda}^{[k]}(\mathbf{x}_k) = F_{\lambda}(\mathbf{x}_k) + (F_{\lambda}^{[k]}(\mathbf{x}_k) - y_k) \frac{\partial F_{\lambda}(\mathbf{x}_k)}{\partial y_k} \quad (5.114)$$

From Eq. (5.100), defining the entries of the influence matrix $\mathbf{A}(\lambda)$, we readily see that

$$\frac{\partial F_{\lambda}(\mathbf{x}_k)}{\partial y_k} = a_{kk}(\lambda) \quad (5.115)$$

where $a_{kk}(\lambda)$ is the k th diagonal element of $\mathbf{A}(\lambda)$. Hence, using Eq. (5.115) in (5.114), and solving the resulting equation for $F_{\lambda}^{[k]}(\mathbf{x}_k)$, we obtain

$$\begin{aligned} F_{\lambda}^{[k]}(\mathbf{x}_k) &= \frac{F_{\lambda}(\mathbf{x}_k) - a_{kk}(\lambda)y_k}{1 - a_{kk}(\lambda)} \\ &= \frac{F_{\lambda}(\mathbf{x}_k) - y_k}{1 - a_{kk}(\lambda)} + y_k \end{aligned} \quad (5.116)$$

Substituting Eq. (5.116) in (5.113), we may redefine $V_0(\lambda)$ as

$$V_0(\lambda) = \frac{1}{N} \sum_{k=1}^N \left[\frac{y_k - F_{\lambda}(\mathbf{x}_k)}{1 - a_{kk}(\lambda)} \right]^2 \quad (5.117)$$

Typically, $a_{kk}(\lambda)$ is different for different k , which means that the data points in $V_0(\lambda)$ are not treated equally. To circumvent this undesirable feature of ordinary cross-validation, Craven and Wahba (1979) introduced the *generalized cross-validation* (GCV),

using a rotation of coordinates.¹¹ Specifically, the ordinary cross-validation function $V_0(\lambda)$ of Eq. (5.117) is modified as:

$$V(\lambda) = \frac{1}{N} \sum_{k=1}^N \omega_k \left[\frac{y_k - F_{\lambda}(\mathbf{x}_k)}{1 - a_{kk}(\lambda)} \right]^2 \quad (5.118)$$

where the weights, ω_k , are themselves defined by

$$\omega_k = \left[\frac{1 - a_{kk}(\lambda)}{\frac{1}{N} \text{tr}[\mathbf{I} - \mathbf{A}(\lambda)]} \right]^2 \quad (5.119)$$

Then, the generalized cross-validation function $V(\lambda)$ becomes

$$V(\lambda) = \frac{\frac{1}{N} \sum_{k=1}^N [y_k - F_{\lambda}(\mathbf{x}_k)]^2}{\left[\frac{1}{N} \text{tr}[\mathbf{I} - \mathbf{A}(\lambda)] \right]^2} \quad (5.120)$$

Finally, using Eq. (5.100) in (5.120) yields

$$V(\lambda) = \frac{\frac{1}{N} \|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2}{\left[\frac{1}{N} \text{tr}[\mathbf{I} - \mathbf{A}(\lambda)] \right]^2} \quad (5.121)$$

which relies solely on quantities related to the data for its computation.

An Optimal Property of the Generalized Cross-Validation Function $V(\lambda)$

Let λ denote the minimizer of the expected value of the generalized cross-validation function $V(\lambda)$. The *expectation inefficiency* of the method of generalized cross-validation is defined by

$$I^* = \frac{E[R(\lambda)]}{\min_{\lambda} E[R(\lambda)]} \quad (5.122)$$

where $R(\lambda)$ is the average squared error over the data set given in Eq. (5.99). Naturally, the asymptotic value of I^* satisfies the condition

$$\lim_{N \rightarrow \infty} I^* = 1 \quad (5.123)$$

In other words, for large N , the average squared error $R(\lambda)$ with λ estimated by minimizing the generalized cross-validation function $V(\lambda)$ should be close to the minimum possible value of $R(\lambda)$, which makes $V(\lambda)$ a good method for estimating λ .

Summarizing Comments

The general idea is to choose the regularization parameter λ so as to minimize the average squared error over the data set, $R(\lambda)$. Unfortunately, this cannot be accomplished directly, since $R(\lambda)$ involves the unknown regression function $f(\mathbf{x})$. With this being so, there are two possibilities that may be pursued in practice:

- If the noise variance σ^2 is known, we may use the minimizer of the estimate $\hat{R}(\lambda)$ of Eq. (5.110) as the optimum choice of λ , optimum in the sense that it also minimizes $R(\lambda)$.
- If σ^2 is not known, we may use the minimizer of the generalized cross-validation function $V(\lambda)$ of Eq. (5.121) as a good choice of λ , which produces an expected mean square error that approaches the minimum possible expected mean square error as $N \rightarrow \infty$.

The important point to note here is that the theory justifying the use of generalized cross-validation for estimating λ is an asymptotic one. Good results can therefore be expected only when the available data set is long enough for the signal to be distinguishable from noise.

Practical experience with generalized cross-validation appears to show that it is robust against nonhomogeneity of variances and non-Gaussian noise (Wahba, 1990). However, the method is quite likely to produce unsatisfactory estimates of the regularization parameter λ if the noise process is highly correlated.

Finally, some comments pertaining to the computation of the generalized cross-validation function $V(\lambda)$ are in order. For given trial values of the regularization parameter λ , finding the denominator term $[\text{tr}\{\mathbf{I} - \mathbf{A}(\lambda)\}/N]^2$ in the formula of Eq. (5.121) is the most expensive part of the work involved in computing $V(\lambda)$. The "randomized trace method" described in Wahba et al. (1995) may be used to compute $\text{tr}\{\mathbf{A}(\lambda)\}$; it is feasible to apply this method to very large systems.

10. APPROXIMATION PROPERTIES OF RBF NETWORKS

In Chapter 4 we discuss the approximation properties of multilayer perceptrons. Radial-basis function networks exhibit good approximation properties of their own, paralleling those of multilayer perceptrons. The family of RBF networks is broad enough to uniformly approximate any continuous function on a compact set.¹²

Universal Approximation Theorem

Let $G: \mathbb{R}^{m_0} \rightarrow \mathbb{R}$ be an integrable bounded function such that G is continuous and

$$\int_{\mathbb{R}^{m_0}} G(\mathbf{x}) d\mathbf{x} \neq 0$$

Let $\mathcal{J}_G$ denote the family of RBF networks consisting of functions $F: \mathbb{R}^{m_0} \rightarrow \mathbb{R}$ represented by

$$F(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G\left(\frac{\mathbf{x} - \mathbf{t}_i}{\sigma}\right)$$

where $\sigma > 0$, $w_i \in \mathbb{R}$ and $\mathbf{t}_i \in \mathbb{R}^{m_0}$ for $i = 1, 2, \dots, m_1$. We may then state the *universal approximation theorem* for RBF networks (Park and Sandberg, 1991):

For any continuous input-output mapping function $f(\mathbf{x})$ there is an RBF network with a set of centers $\{\mathbf{t}_i\}_{i=1}^{m_1}$ and a common width $\sigma > 0$ such that the input-output mapping function $F(\mathbf{x})$ realized by the RBF network is close to $f(\mathbf{x})$ in the L_p norm, $p \in [1, \infty]$.

Note that in the universal approximation theorem as stated, the kernel $G: \mathbb{R}^{m_0} \rightarrow \mathbb{R}$ is not required to satisfy the property of radial symmetry. The theorem is therefore stronger than necessary for RBF networks. Most importantly, it provides the theoretical basis for the design of neural networks using radial basis functions for practical applications.

Curse of Dimensionality (Revisited)

In addition to the universal approximation property of RBF networks, there is the issue of the rate of approximation attainable by these networks that must be considered. From the discussion presented in Chapter 4, we recall that the *intrinsic complexity* of a class of approximating functions increases exponentially in the ratio m_0/s , where m_0 is the *input dimensionality* (i.e., dimension of the input space) and s is a *smoothness index* measuring the number of constraints imposed on an approximating function in that particular class. Bellman's *curse of dimensionality* tells us that, irrespective of the approximation technique employed, if the smoothness index s is maintained constant, the number of parameters needed for the approximating function to attain a prescribed degree of accuracy increases exponentially with the input dimensionality m_0 . The only way that we can achieve a rate of convergence independent of the input dimensionality m_0 , and therefore be immune to the curse of dimensionality, is for the smoothness index s to increase with the number of parameters in the approximating function so as to compensate for the increase in complexity. This point is illustrated in Table 5.3, adapted from Girosi and Anzellotti (1992). Table 5.3 summarizes the constraints on function

TABLE 5.3 Two Approximation Techniques and Corresponding Function Spaces with the Same Rate of Convergence $O(1/\sqrt{m_1})$, Where m_1 is the Size of the Hidden Space.

Function Space	Norm	Approximation Technique
$\int_{\mathbb{R}^{m_0}} \ s\ \tilde{F}(s) ds < \infty$	$L_2(\Omega)$	(a) multilayer perceptrons
where $\tilde{F}(s)$ is the multidimensional Fourier transform of the approximating function $F(\mathbf{x})$		$F(\mathbf{x}) = \sum_{i=1}^{m_1} a_i \varphi(w_i^T \mathbf{x} + b_i)$ where $\varphi(\cdot)$ is the sigmoid activation function
Sobolev space of functions whose derivatives up to order $2m > m_0$ are integrable	$L_2(\mathbb{R}^2)$	(b) RBF networks:
		$F(\mathbf{x}) = \sum_{i=1}^{m_1} a_i \exp\left(-\frac{\ \mathbf{x} - \mathbf{t}_i\ ^2}{2\sigma^2}\right)$

space that have to be satisfied by two approximating techniques, multilayer perceptrons and RBF networks, for the rate of convergence to be independent of the input dimensionality m_0 . Naturally, the constraints imposed on these two approximating techniques are different, reflecting the different paths followed in their formulations. In the case of RBF networks, the result holds in the *Sobolev space*¹³ of functions whose derivations up to order $2m > m_0$ are integrable. In other words, the number of derivatives of the approximating function that are integrable is required to increase with the input dimensionality m_0 in order to make the rate of convergence independent of m_0 . As explained in Chapter 4, a similar constraint applies to multilayer perceptrons, but in a rather deceptive way. The conclusion to be drawn from Table 5.3 may therefore be stated as:

The space of approximating functions attainable with multilayer perceptrons and RBF networks becomes increasingly constrained as the input dimensionality m_0 is increased.

The net result is that the curse of dimensionality can be broken neither by neural networks whether they are multilayer perceptrons or RBF networks, nor by any other nonlinear technique of a similar nature.

Relationship between Sample Complexity, Computational Complexity, and Generalization Performance

A discussion of the approximation problem would be incomplete without some consideration being given to the fact that, in practice, we do not have an infinite amount of data, but rather a training sample of some finite size. By the same token, we do not have a neural network with infinite computational complexity, but rather a finite one. Accordingly, there are two components to the generalization error of a neural network trained on a data set of finite size and tested on data not seen before, as discussed in Chapter 2. One component, called the *approximation error*, results from the limited capacity of the network to represent a target function of interest. The other component, called the *estimation error*, results from the limited amount of information contained in the training sample about the target function. Using this form of decomposition, Niyogi and Girosi (1996) have derived a *bound* on the generalization error produced by a Gaussian RBF network, expressed in terms of the size of the hidden layer and the size of the training sample. The derivation is for the case of learning a regression function in a model of the kind described in Eq. (5.95); the regression function belongs to a certain Sobolev space.

This bound, formulated in the terminology of PAC learning described in Chapter 2, may be stated as follows (Niyogi and Girosi, 1996):

Let G denote the class of Gaussian RBF networks with m_0 input (source) nodes and m_1 hidden units. Let $f(\mathbf{x})$ denote a regression function that belongs to a certain Sobolev space. Assume that the training sample $\mathcal{T} = \{(\mathbf{x}_i, d_i)\}_{i=1}^N$ is obtained by random sampling of the regressive model based on $f(\mathbf{x})$. Then, for any confidence parameter $\delta \in (0, 1]$, the generalization error produced by the network is bounded from above by

$$O\left(\frac{1}{m_1}\right) + O\left(\frac{m_0 m_1}{N} \log(m_1 N) + \frac{1}{N} \log\left(\frac{1}{\delta}\right)\right)^{1/2} \quad (5.124)$$

with probability greater than $1 - \delta$.

From the bound of Eq. (5.124), we may make the following deductions:

- The generalization error converges to zero only if the number of hidden units, m_1 , increases more slowly than the size N of the training sample.
- For a given size N of training sample, the optimum number of hidden units, m_1^* , behaves as (see Problem 5.11)

$$m_1^* \propto N^{1/3} \quad (5.125)$$

- The RBF network exhibits a rate of approximation $O(1/m_1)$ that is similar to that derived by Barron (1993) for the case of a multilayer perceptron with sigmoid activation functions; see the discussion in Section 4.12.

5.11 COMPARISON OF RBF NETWORKS AND MULTILAYER PERCEPTRONS

Radial-basis function (RBF) networks and multilayer perceptrons are examples of nonlinear layered feedforward networks. They are both universal approximators. It is therefore not surprising to find that there always exists an RBF network capable of accurately mimicking a specified MLP, or vice versa. However, these two networks differ from each other in several important respects.

1. An RBF network (in its most basic form) has a single hidden layer, whereas an MLP may have one or more hidden layers.
2. Typically the computation nodes of an MLP, located in a hidden or an output layer, share a common neuronal model. On the other hand, the computation nodes in the hidden layer of an RBF network are quite different and serve a different purpose from those in the output layer of the network.
3. The hidden layer of an RBF network is nonlinear, whereas the output layer is linear. However, the hidden and output layers of an MLP used as a pattern classifier are usually all nonlinear. When the MLP is used to solve nonlinear regression problems, a linear layer for the output is usually the preferred choice.
4. The argument of the activation function of each hidden unit in an RBF network computes the *Euclidean norm (distance)* between the input vector and the center of that unit. Meanwhile, the activation function of each hidden unit in an MLP computes the *inner product* of the input vector and the synaptic weight vector of that unit.
5. MLPs construct *global* approximations to nonlinear input-output mapping. On the other hand, RBF networks using exponentially decaying localized nonlinearities (e.g., Gaussian functions) construct *local* approximations to nonlinear input-output mappings.

This in turn means that for the approximation of a nonlinear input-output mapping, the MLP may require a smaller number of parameters than the RBF network for the same degree of accuracy.

The linear characteristics of the output layer of the RBF network mean that such a network is more closely related to Rosenblatt's perceptron than to the multilayer perceptron. However, the RBF network differs from the perceptron in that it is capable

of implementing arbitrary nonlinear transformations of the input space. This is well illustrated by the XOR problem, which cannot be solved by any linear perceptron but can be solved by an RBF network.

2 KERNEL REGRESSION AND ITS RELATION TO RBF NETWORKS

The theory of RBF networks presented so far has built on the notion of interpolation. In this section we take another viewpoint, namely, *kernel regression* building on the notion of density estimation.

To be specific, consider again the nonlinear regression model of Eq. (5.95), reproduced here for convenience of presentation:

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, 2, \dots, N$$

As a reasonable estimate of the unknown regression function $f(x)$, we may take the mean of observables (i.e., values of the model output y) near a point x . For this approach to be successful, however, the local average should be confined to observations in a small neighborhood (i.e., receptive field) around the point x , because in general, observations corresponding to points away from x will have different mean values. More precisely, we recall from the discussion presented in Chapter 2 that $f(x)$ is equal to the conditional mean of y given x (i.e., the regression of y on x), as shown by

$$f(x) = E[y|x]$$

Using the formula for the expectation of a random variable, we may write

$$f(x) = \int_{-\infty}^{\infty} y f_Y(y|x) dy \quad (5.126)$$

where $f_Y(y|x)$ is the conditional probability density function (pdf) of Y , given x . From probability theory, we have

$$f_Y(y|x) = \frac{f_{X,Y}(x, y)}{f_X(x)} \quad (5.127)$$

where $f_X(x)$ is the pdf of x and $f_{X,Y}(x, y)$ is the joint pdf of x and y . Hence, using Eq. (5.127) in (5.126), we obtain the following formula for the regression function

$$f(x) = \frac{\int_{-\infty}^{\infty} y f_{X,Y}(x, y) dy}{f_X(x)} \quad (5.128)$$

Our particular interest is in a situation where the joint probability density function $f_{X,Y}(x, y)$ is unknown. All that we have available is the training sample, $\{(x_i, y_i)\}_{i=1}^N$. To estimate $f_{X,Y}(x, y)$ and therefore $f_X(x)$, we may use a nonparametric estimator known as the *Parzen-Rosenblatt density estimator* (Rosenblatt, 1956, 1970; Parzen, 1962). Basic to the formulation of this estimator is a *kernel*, denoted by $K(x)$, which has properties similar to those associated with a probability density function:

- The kernel $K(x)$ is a continuous, bounded, and real function of x , and symmetric about the origin where it attains its maximum value.
- The total volume under the surface of the kernel $K(x)$ is unity; that is, for an m -dimensional vector x ,

$$\int_{\mathbb{R}^m} K(x) dx = 1 \quad (5.129)$$

Assuming that $x_1, x_2, \dots, x_N$ are independent random vectors and identically distributed, we may formally define the Parzen-Rosenblatt density estimate of $f_X(x)$ as:

$$\hat{f}_X(x) = \frac{1}{Nh^{m_0}} \sum_{i=1}^N K\left(\frac{x - x_i}{h}\right) \quad \text{for } x \in \mathbb{R}^{m_0} \quad (5.130)$$

where the smoothing parameter h is a positive number called *bandwidth* or simply *width*; h controls the size of the kernel. (The parameter h used here should not be confused with the h used to define the Fréchet derivative in Section 5.5.) An important property of the Parzen-Rosenblatt density estimator is that it is a *consistent estimator*¹⁴ (i.e., asymptotically unbiased) in the sense that if $h = h(N)$ is chosen as a function of N such that

$$\lim_{N \rightarrow \infty} h(N) = 0,$$

then

$$\lim_{N \rightarrow \infty} E[\hat{f}_X(x)] = f_X(x)$$

For this latter equation to hold, x should be a point of continuity for $f_X(x)$.

In a manner similar to that described in Eq. (5.130), we may formulate the Parzen-Rosenblatt density estimate of the joint probability density function $f_{X,Y}(x, y)$ as follows:

$$\hat{f}_{X,Y}(x, y) = \frac{1}{Nh^{m_0+1}} \sum_{i=1}^N K\left(\frac{x - x_i}{h}\right) K\left(\frac{y - y_i}{h}\right) \quad \text{for } x \in \mathbb{R}^{m_0} \text{ and } y \in \mathbb{R} \quad (5.131)$$

Integrating $\hat{f}_{X,Y}(x, y)$ with respect to y , we get the $f_X(x)$ of Eq. (5.130), and so we should. Moreover,

$$\int_{-\infty}^{\infty} y \hat{f}_{X,Y}(x, y) dy = \frac{1}{Nh^{m_0+1}} \sum_{i=1}^N K\left(\frac{x - x_i}{h}\right) \int_{-\infty}^{\infty} y K\left(\frac{y - y_i}{h}\right) dy$$

Changing the variable of integration by putting $z = (y - y_i)/h$, and using the symmetric property of the kernel $K(\cdot)$, we obtain the result

$$\int_{-\infty}^{\infty} y \hat{f}_{X,Y}(x, y) dy = \frac{1}{Nh^{m_0}} \sum_{i=1}^N y_i K\left(\frac{x - x_i}{h}\right) \quad (5.132)$$

Thus, using Eqs. (5.132) and (5.130) as estimates of the quantities in the numerator and denominator of Eq. (5.128), respectively, we obtain the following estimate of the regression function $f(x)$, after canceling common terms:

$$F(\mathbf{x}) = \hat{f}(\mathbf{x}) = \frac{\sum_{i=1}^N y_i K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)}{\sum_{i=1}^N K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)} \quad (5.133)$$

where in the denominator, for clarity of presentation, we have used j instead of i as the index of summation. As with an ordinary RBF network, the kernel regression estimator $F(\mathbf{x})$ defined in Eq. (5.133) is a universal approximator.

There are two ways in which the approximating function $F(\mathbf{x})$ may be viewed:

1. *Nadaraya-Watson regression estimator.* Define the *normalized weighting function*

$$W_{N,i}(\mathbf{x}) = \frac{K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)}{\sum_{j=1}^N K\left(\frac{\mathbf{x} - \mathbf{x}_j}{h}\right)}, \quad i = 1, 2, \dots, N \quad (5.134)$$

with

$$\sum_{i=1}^N W_{N,i}(\mathbf{x}) = 1 \quad \text{for all } \mathbf{x} \quad (5.135)$$

We may then rewrite the kernel regression estimator of Eq. (5.133) in the simplified form

$$F(\mathbf{x}) = \sum_{i=1}^N W_{N,i}(\mathbf{x}) y_i \quad (5.136)$$

which describes $F(\mathbf{x})$ as a *weighted average* of the y -observables. The particular form of weighting function $W_{N,i}(\mathbf{x})$ given in Eq. (5.136) was originally proposed by Nadaraya (1964) and Watson (1964). Accordingly, the approximating function of Eq. (5.136) is often called the *Nadaraya-Watson regression estimator (NWRE)*.¹⁵

2. *Normalized RBF network.* For the second viewpoint, we assume *spherical symmetry* of the kernel $K(\mathbf{x})$, in which case we may set (Krzyzak et al., 1996)

$$K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) = K\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|}{h}\right) \quad \text{for all } i \quad (5.137)$$

where $\|\cdot\|$ denotes the Euclidean norm of the enclosed vector. Correspondingly, we define the *normalized radial basis function*

$$\psi_N(\mathbf{x}, \mathbf{x}_i) = \frac{K\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|}{h}\right)}{\sum_{j=1}^N K\left(\frac{\|\mathbf{x} - \mathbf{x}_j\|}{h}\right)}, \quad i = 1, 2, \dots, N \quad (5.138)$$

with

$$\sum_{i=1}^N \psi_N(\mathbf{x}, \mathbf{x}_i) = 1 \quad \text{for all } \mathbf{x} \quad (5.139)$$

The subscript N in $\psi_N(\mathbf{x}, \mathbf{x}_i)$ signifies the use of *normalization*.

For the regression problem considered here, we recognize that the “linear weights,” w_i , applied to the basic functions $\psi_N(\mathbf{x}, \mathbf{x}_i)$ are simply the observables, y_i , of the regression model for the input data $\mathbf{x}_i$. Thus, letting

$$y_i = w_i, \quad i = 1, 2, \dots, N$$

we may reformulate the approximating function of Eq. (5.133) in the general form

$$F(\mathbf{x}) = \sum_{i=1}^N w_i \psi_N(\mathbf{x}, \mathbf{x}_i) \quad (5.140)$$

Equation (5.140) represents the input-output mapping of a *normalized radial-basis function (RBF) network* (Moody and Darken, 1989; Xu et al., 1994). Note that

$$0 \leq \psi_N(\mathbf{x}, \mathbf{x}_i) \leq 1 \quad \text{for all } \mathbf{x} \text{ and } \mathbf{x}_i \quad (5.141)$$

Accordingly, $\psi_N(\mathbf{x}, \mathbf{x}_i)$ may be interpreted as the probability of an event described by the input vector $\mathbf{x}$, conditional on $\mathbf{x}_i$.

The basic difference between the normalized radial-basis function $\psi_N(\mathbf{x}, \mathbf{x}_i)$ of Eq. (5.138) and an ordinary radial-basis function is a denominator term that constitutes the *normalization factor*. This normalization factor is an estimate of the underlying pdf of the input vector $\mathbf{x}$. Consequently, the basis functions $\psi_N(\mathbf{x}, \mathbf{x}_i)$ for $i = 1, 2, \dots, N$ sum to unity for all $\mathbf{x}$, as described in Eq. (5.139). In contrast, there is no guarantee that this condition is satisfied by the basis (Green's) functions of the ordinary RBF network of Eq. (5.57).

The derivation of the input-output mapping $F(\mathbf{x})$ described in Eq. (5.138) was presented here by using the notion of density estimation. Like the hypersurface reconstruction problem, density estimation is an ill-posed problem. To make it well-posed, some form of regularization has to be used. The Parzen-Rosenblatt density estimator, and therefore the Nadaraya-Watson regression estimator, can be derived within the framework of regularization theory (Vapnik, 1982). Naturally, the cost functional to be minimized for density estimation is different from the deterministic Tikhonov functional of Eq. (5.23). The cost functional for density estimation consists of the sum of two terms: a square error term involving the unknown probability density function, and an appropriate form of stabilizing functional.

Multivariate Gaussian Distribution

A variety of kernel functions is possible in general. However, both theoretical and practical considerations limit the choice. As with the Green's function, a widely used kernel is the multivariate Gaussian distribution:

$$K(\mathbf{x}) = \frac{1}{(2\pi)^{m/2}} \exp\left(-\frac{\|\mathbf{x}\|^2}{2}\right) \quad (5.142)$$

where m_i is the dimension of the input vector $\mathbf{x}$. The spherical symmetry of the kernel $K(\mathbf{x})$ is clearly apparent in Eq. (5.142). Assuming the use of a common width (spread) σ that plays the role of smoothing parameter h for a Gaussian distribution, and centering the kernel on a data point $\mathbf{x}_i$, we may write

$$K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) = \frac{1}{(2\pi\sigma^2)^{m_i/2}} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right), \quad i = 1, 2, \dots, N \quad (5.143)$$

Thus, using Eq. (5.143), the Nadaraya–Watson regression estimator takes the following form (Specht, 1991):

$$F(\mathbf{x}) = \frac{\sum_{i=1}^N y_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right)}{\sum_{i=1}^N \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right)} \quad (5.144)$$

where the denominator term representing the Parzen–Rosenblatt density estimator, consists of the sum of N multivariate Gaussian distributions centered on the data points $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$.

Correspondingly, using Eq. (5.143) in (5.138) and then (5.140), the input–output mapping function of the normalized RBF network takes the following form:

$$F(\mathbf{x}) = \frac{\sum_{i=1}^N w_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right)}{\sum_{i=1}^N \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right)} \quad (5.145)$$

In Eqs. (5.144) and (5.145) the centers of the normalized radial-basis functions coincide with the data points $\{\mathbf{x}_i\}_{i=1}^N$. As with ordinary radial-basis functions, a smaller number of normalized radial-basis functions can be used, with their centers treated as free parameters to be chosen according to some heuristic (Moody and Darken, 1989) or determined in a principled manner (Poggio and Girosi, 1990a).

5.13 LEARNING STRATEGIES

The learning process undertaken by a radial-basis function (RBF) network irrespective of its theoretical background, may be visualized as follows. The linear weights associated with the output unit(s) of the network tend to evolve on a different “time scale” compared to the nonlinear activation functions of the hidden units. Thus, as the hidden layer’s activation functions evolve slowly in accordance with some *nonlinear* optimization strategy, the output layer’s weights adjust themselves rapidly through a *linear* optimization strategy. The important point is that the different layers of an RBF network perform different tasks, and so it is reasonable to separate the optimization of the hidden and output layers of the network by using different techniques, and perhaps by operating on different time scales (Lowe, 1991a).

There are different learning strategies that we can follow in the design of an RBF network, depending on how the centers of the radial-basis functions of the network are specified. Here we identify four approaches. The first three design strategies pertain to an RBF network whose formulation is based on interpolation theory. The last design strategy combines elements of regularization theory and kernel regression estimation theory.

1. Fixed Centers Selected at Random

The simplest approach is to assume *fixed* radial-basis functions defining the activation functions of the hidden units. The locations of the centers may be chosen *randomly* from the training data set. This is considered to be a “sensible” approach, provided that the training data are distributed in a representative manner for the problem at hand (Lowe, 1989). For the radial-basis functions themselves, we may employ an *isotropic* Gaussian function whose standard deviation is fixed according to the spread of the centers. Specifically, a (normalized) radial-basis function centered at $\mathbf{t}_i$ is defined as

$$G(\|\mathbf{x} - \mathbf{t}_i\|^2) = \exp\left(-\frac{m_i}{d_{\max}^2} \|\mathbf{x} - \mathbf{t}_i\|^2\right), \quad i = 1, 2, \dots, m_1 \quad (5.146)$$

where m_i is the number of centers and $d_{\max}$ is the maximum distance between the chosen centers. In effect, the standard deviation (i.e., width) of all the Gaussian radial-basis functions is fixed at

$$\sigma = \frac{d_{\max}}{\sqrt{2m_1}} \quad (5.147)$$

This formula ensures that the individual radial-basis functions are not too peaked or too flat; both of these two extreme conditions should be avoided. As an alternative to Eq. (5.147) we may use individually scaled centers with broader widths in areas of lower data density, which requires experimentation with the training data.

The only parameters that would need to be learned in this approach are the linear weights in the output layer of the network. A straightforward procedure for doing this is to use the *pseudoinverse method* (Broomhead and Lowe, 1988). Specifically, we have (see also Eqs. (5.77) and (5.78))

$$\mathbf{w} = \mathbf{G}^+ \mathbf{d} \quad (5.148)$$

where $\mathbf{d}$ is the desired response vector in the training set. The matrix $\mathbf{G}^+$ is the pseudoinverse of the matrix $\mathbf{G}$, which is itself defined as

$$\mathbf{G} = \{g_{ji}\} \quad (5.149)$$

where

$$g_{ji} = \exp\left(-\frac{m_i}{d^2} \|\mathbf{x}_j - \mathbf{t}_i\|^2\right), \quad j = 1, 2, \dots, N; \quad i = 1, 2, \dots, m_1 \quad (5.150)$$

where $\mathbf{x}_j$ is the j th input vector of the training sample.

Basic to all algorithms for the computation of a pseudoinverse of a matrix is the *singular-value decomposition (SVD)* (Golub and Van Loan, 1996):

If G is a real N -by- M matrix, there exist orthogonal matrices

$$U = \{u_1, u_2, \dots, u_N\}$$

and

$$V = \{v_1, v_2, \dots, v_M\}$$

such that

$$U^T G V = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_K), \quad K = \min(M, N) \quad (5.151)$$

where

$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_K > 0$$

The column vectors of the matrix U are called the *left singular vectors* of G , and the column vectors of the matrix V are called its *right singular vectors*. The $\sigma_1, \sigma_2, \dots, \sigma_K$ are called the *singular values* of the matrix G . According to the singular value decomposition theorem, the M -by- N pseudoinverse of matrix G is defined by

$$G^+ = V \Sigma^+ U^T \quad (5.152)$$

where Σ^+ is itself an N -by- N matrix defined in terms of the singular values of G by

$$\Sigma^+ = \text{diag}\left(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \dots, \frac{1}{\sigma_K}, 0, \dots, 0\right) \quad (5.153)$$

Efficient algorithms for the computation of a pseudoinverse matrix are discussed in Golub and Van Loan (1996).

It is interesting that experience with the random selection of centers shows that this method is relatively insensitive to the use of regularization; see Problem 5.14 for a computer experiment on pattern classification using this method. This kind of performance suggests that the random selection of centers as a method for the design of RBF networks from a large training set of fixed size is perhaps a regularization method in its own right.

2. Self-Organized Selection of Centers

The main problem with the method of fixed centers just described is the fact that it may require a large training set for a satisfactory level of performance. One way of overcoming this limitation is to use a *hybrid learning process*, consisting of two different stages (Moody and Darken, 1989; Lippmann, 1989b; Chen et al., 1992):

- *Self-organized learning* stage, the purpose of which is to estimate appropriate locations for the centers of the radial basis functions in the hidden layer.
- *Supervised learning* stage, which completes the design of the network by estimating the linear weights of the output layer.

Although batch processing can be used to implement these two stages of learning, it is preferable to take an adaptive (iterative) approach.

For the self-organized learning process we need a *clustering* algorithm that partitions the given set of data points into subgroups, each of which should be as homogeneous as possible. One such algorithm is the *k-means clustering algorithm* (Duda and Hart, 1973), which places the centers of the radial-basis functions in only those regions of the input space $\mathcal{X}$ where significant data are present. Let m_1 denote the number of radial-basis functions; the determination of a suitable value for m_1 may require experimentation. Let $\{t_k(n)\}_{k=1}^{m_1}$ denote the centers of the radial-basis functions at iteration n of the algorithm. Then, the *k-means clustering* algorithm proceeds as follows:

1. *Initialization.* Choose random values for the initial centers $t_k(0)$; the only restriction is that these initial values be different. It may also be desirable to keep the Euclidean norm of the centers small.
2. *Sampling.* Draw a sample vector x from the input space $\mathcal{X}$ with a certain probability. The vector x is input into the algorithm at iteration n .
3. *Similarity matching.* Let $k(x)$ denote the index of the best-matching (winning) center for input vector x . Find $k(x)$ at iteration n by using the minimum-distance Euclidean criterion:

$$k(x) = \arg \min_k \|x(n) - t_k(n)\|, \quad k = 1, 2, \dots, m_1 \quad (5.154)$$

where $t_k(n)$ is the center of the k th radial-basis function at iteration n .

4. *Updating.* Adjust the centers of the radial-basis functions, using the update rule:

$$t_k(n+1) = \begin{cases} t_k(n) + \eta[x(n) - t_k(n)], & k = k(x) \\ t_k(n), & \text{otherwise} \end{cases} \quad (5.155)$$

where η is a *learning-rate parameter* that lies in the range $0 < \eta < 1$.

5. *Continuation.* Increment n by 1, go back to step 2, and continue the procedure until no noticeable changes are observed in the centers t_k .

The *k-means clustering* algorithm just described is, in fact, a special case of a competitive (winner-takes-all) learning process known as the *self-organizing map*, which is discussed in Chapter 9. This latter algorithm also lends itself for implementing the self-organized learning stage.

A limitation of the *k-means clustering* algorithm is that it can only achieve a local optimum solution that depends on the initial choice of cluster centers. Consequently, computing resources may be wasted in that some initial centers get stuck in regions of the input space $\mathcal{X}$ with a scarcity of data points and may therefore never have the chance to move to new locations where they are needed. The net result is possibly an unnecessarily large network. To overcome this limitation of the conventional *k-means clustering* algorithm, Chen (1995) proposes the use of an *enhanced k-means clustering algorithm* due to Chinunrueng and Séquin (1994), which is based on a cluster variation-weighted measure that enables the algorithm to converge to an optimum or near-optimum configuration, independent of the initial center locations.

Having identified the individual centers of the Gaussian radial-basis functions and their common-width using the *k-means clustering* algorithm or its enhanced version, the next and final stage of the hybrid learning process is to estimate the weights of the output layer. A simple method for this estimation is the least-mean-square (LMS)

algorithm described in Chapter 3. The vector of output signals produced by the hidden units constitutes the input vector to the LMS algorithm. Note also that the k -means clustering algorithm for the hidden units and the LMS algorithm for the output unit(s) may proceed with their own individual computations in a concurrent fashion, thereby accelerating the training process.

3. Supervised Selection of Centers

In the third approach, the centers of the radial-basis functions and all other free parameters of the network undergo a supervised learning process; in other words, the RBF network takes on its most generalized form. A natural candidate for such a process is error-correction learning, which is most conveniently implemented using a gradient-descent procedure that represents a generalization of the LMS algorithm.

The first step in the development of such a learning procedure is to define the instantaneous value of the cost function.

$$\mathcal{E} = \frac{1}{2} \sum_{j=1}^N e_j^2 \quad (5.156)$$

where N is the size of the training sample used to do the learning, and e_j is the error signal defined by

$$\begin{aligned} e_j &= d_j - F^*(x_j) \\ &= d_j - \sum_{i=1}^M w_i G(\|x_j - t_i\|_{C_i}) \end{aligned} \quad (5.157)$$

The requirement is to find the free parameters w_i , t_i , and Σ_i^{-1} (the latter being related to the norm-weighting matrix C_i) so as to minimize $\mathcal{E}$. The results of this minimization are summarized in Table 5.4; the derivations of these results are presented as an exercise to the reader in Problem 5.13. The following points are noteworthy in Table 5.4.

- The cost function $\mathcal{E}$ is convex with respect to the linear parameters w_i , but non-convex with respect to the centers t_i and matrix Σ_i^{-1} ; in the latter case, the search for the optimum values of t_i and Σ_i^{-1} may get stuck at a local minimum in parameter space.
- The update equations for w_i , t_i , and Σ_i^{-1} are (in general) assigned different learning-rate parameters η_1 , η_2 , and η_3 , respectively.
- Unlike the back-propagation algorithm, the gradient-descent procedure described in Table 5.4 for an RBF network does *not* involve error back-propagation.
- The gradient vector $\partial\mathcal{E}/\partial t_i$ has an effect similar to a *clustering effect* that is task-dependent (Poggio and Girosi, 1990a).

For the *initialization* of the gradient-descent procedure, it is often desirable to begin the search in parameter space from a *structured* initial condition that limits the region of parameter space to be searched to an already known useful area, which may be achieved by implementing a standard pattern-classification method (Lowe, 1991a). In so doing, the likelihood of converging to an undesirable local minimum in weight space is reduced. For example, we may begin with a *Gaussian classifier*, which assumes

TABLE 5.4 Adaptation Formulas for the Linear Weights and the Positions and Spreads of Centers for RBF Network^a

1. Linear weights (output layer)	
	$\frac{\partial\mathcal{E}(n)}{\partial w_i(n)} = \sum_{j=1}^N e_j(n) G(\ x_j - t_i\ _{C_i})$
	$w_i(n+1) = w_i(n) - \eta_1 \frac{\partial\mathcal{E}(n)}{\partial w_i(n)}, \quad i = 1, 2, \dots, m_1$
2. Positions of centers (hidden layer)	
	$\frac{\partial\mathcal{E}(n)}{\partial t_i(n)} = 2w_i(n) \sum_{j=1}^N e_j(n) G'(\ x_j - t_i\ _{C_i}) \Sigma_i^{-1} [x_j - t_i(n)]$
	$t_i(n+1) = t_i(n) - \eta_2 \frac{\partial\mathcal{E}(n)}{\partial t_i(n)}, \quad i = 1, 2, \dots, m_1$
3. Spreads of centers (hidden layer)	
	$\frac{\partial\mathcal{E}(n)}{\partial \Sigma_i^{-1}(n)} = -w_i(n) \sum_{j=1}^N e_j(n) G'(\ x_j - t_i(n)\ _{C_i}) Q_{\mu}(n)$
	$Q_{\mu}(n) = [x_j - t_i(n)][x_j - t_i(n)]^T$
	$\Sigma_i^{-1}(n+1) = \Sigma_i^{-1}(n) - \eta_3 \frac{\partial\mathcal{E}(n)}{\partial \Sigma_i^{-1}(n)}$

^aThe term $e_j(n)$ is the error signal of output unit j at time n . The term $G'(\cdot)$ is the first derivative of the Green's function $G(\cdot)$ with respect to its argument.

that each pattern in each class is drawn from a Gaussian distribution; this special form of pattern classifier based on the Bayes hypothesis testing procedure is discussed in Chapter 3.

A question that arises at this stage of the discussion is: What can be gained by adapting the positions of the centers of the radial-basis functions? The answer to this question naturally depends on the application of interest. Nevertheless, on the basis of some results reported in the literature, there is practical merit to the idea of allowing the centers to move. Work done by Lowe (1989) on speech recognition using RBF networks indicates that nonlinear optimization of the parameters that define the activation functions of the hidden layer is beneficial when a minimal network configuration is required. However, according to Lowe, the same performance on generalization may be achieved by using a larger RBF network; that is, a network with a larger number of fixed centers in the hidden layer, and only adapting the output layer of the network by linear optimization.

Wettschereck and Dietterich (1992) have compared the performance of (Gaussian) radial-basis function networks with fixed centers to that of generalized radial-basis function networks with adjustable centers; in the latter case, the positions of the centers are determined by supervised learning. The performance comparison was made for the NETalk task. The original NETalk experiment was carried out by Sejnowski and Rosenberg (1987) using a multilayer perceptron trained with the back-propagation algorithm; it is described in Chapter 13. The purpose of the experiment performed by Wettschereck and Dietterich was to understand how a neural network could

learn to map English spelling into its phonetic pronunciation. The experimental study by Wettchereck and Dietterich in the NETalk domain may be summarized as follows:

- RBF networks (with unsupervised learning of the centers' locations and supervised learning of the output-layer weights) did *not* generalize nearly as well as multilayer perceptrons trained with the back-propagation algorithm.
- Generalized RBF networks (with supervised learning of the centers' locations as well as the output-layer weights) were able to exceed substantially the generalization performance of multilayer perceptrons.

4. Strict Interpolation with Regularization

A method for designing RBF networks that combines elements of the regularization theory of Section 5.5 and the kernel regression estimation theory described in Section 5.12 in a principled way is described in Yee (1998). The method involves the combined use of the following four ingredients:

1. Radial-basis function, G , admissible (possibly with some scaling) as the kernel of a (mean-square) consistent Nadaraya–Watson regression estimate (NWRE).
2. Diagonal input norm-weighting matrix, Σ^{-1} , common to all centers with entries

$$\Sigma = \text{diag}(h_1, h_2, \dots, h_{m_0}) \quad (5.156)$$

where $h_1, h_2, \dots, h_{m_0}$ are the per dimension *bandwidths* of a consistent NWRE with (scaled) kernel G , as previously set forth, and m_0 is the dimensionality of the input space. For example, we may set $h_i = \alpha_i \sigma_i^2$, $i = 1, 2, \dots, m_0$, where σ_i^2 is the sample variance of the i th input variable estimated from the available training input data. The positive input *scale factors* $\alpha_1, \alpha_2, \dots, \alpha_{m_0}$ can then be determined by using a suitable cross-validation (CV) procedure, as explained in Section 5.9.

3. Regularized strict interpolation, which involves training for the linear weights according to Eq. (5.54).
4. Selection of the regularization parameter λ and the input scale factors $\alpha_1, \alpha_2, \dots, \alpha_{m_0}$, which is accomplished via an *asymptotically optimal* method such as the leave-out-one CV method defined in Eq. (5.117) or the GCV method defined in Eq. (5.121). The selected parameters may be interpreted as follows:
 - The larger the value of λ selected, the larger is the noise corrupting the measurement of parameters.
 - When the radial-basis function G is a unimodal kernel (e.g., the Gaussian kernel), the smaller the value of a particular α_i , the more “sensitive” the overall network output is to the associated input dimension. Conversely, the larger the value of a particular α_i , the less “relevant” the associated input dimension is to explaining the variation of the overall network output with respect to changes in the input. Hence, we can use the selected α_i to rank the relative *significance* of the input variables and thereby indicate which input variables are suitable candidates for dimensionality reduction, if necessary.

The justification for this design procedure is discussed in detail in Yee (1998). For our purposes here, we can motivate these design choices as follows. It can be shown that the NWRE corresponds to a special class of regularized RBF networks, in the

sense that any NWRE may be approximated with vanishing mean-square and absolute error by a suitably constructed sequence of regularized RBF networks for which the regularization parameter sequence $\{\lambda_N\}$ is allowed to grow (at an appropriate rate) to infinity with N , the size of training sample. On the other hand, as $N \rightarrow \infty$, we have (under some mild conditions) the convergence of the risk defined in Eq. (5.99) to the (global) mean-squared error. If we use an asymptotically optimal parameter selection procedure for the regularization parameter sequence, then, by construction, the resulting sequence of RBF networks must have (asymptotically) minimum mean-squared error over all possible choices of regularization parameter sequence, including the one corresponding to the NWRE. If conditions then hold such that the NWRE is known to be mean-squared error consistent, the same must also be true for the regularized RBF network designed according to the same procedure. In other words, regularized RBF networks designed according to this procedure can *inherit* the consistency properties of the NWRE. This consequence allows us to leverage the known consistency results of the NWRE in areas such as *time series regression*, where *dependent* and *nonstationary* processes are often encountered and where the usual neural network assumptions of i.i.d. training data and stationary processes are invalid. In summary, by *synthesizing* elements of both regularization theory and kernel regression estimation theory, the design procedure outlined here offers a practical prescription for theoretically supported regularized RBF network design and application.

5.14 COMPUTER EXPERIMENT: PATTERN CLASSIFICATION

In this section we use a computer experiment to illustrate the design of a regularized RBF network based on the use of strict interpolation. The computer experiment involves a binary classification problem based on data drawn from two equiprobable overlapping two-dimensional Gaussian distributions corresponding to classes $\mathcal{C}_1$ and $\mathcal{C}_2$. Details of the Gaussian distributions are the same as those described in Section 4.8. Class $\mathcal{C}_1$ is characterized by mean vector $[0, 0]^T$ and common variance 1, whereas class $\mathcal{C}_2$ is characterized by mean vector $[0, 2]^T$ and common variance 4. The experiment described in this section may thus be viewed as the regularization RBF counterpart to the back-propagation learning experiment of Section 4.8.

With two classes $\mathcal{C}_1$ and $\mathcal{C}_2$, the regularized RBF network is constructed to have two output functions, one for each class. Also, binary-valued class indicator outputs are used as the desired output values, as shown by

$$d_k^{(p)} = \begin{cases} 1 & \text{if pattern } p \text{ belongs to class } \mathcal{C}_k \\ 0 & \text{otherwise} \end{cases}$$

where $k = 1, 2$.

Before we proceed with the experiment, however, we must resolve the issue of an output decision rule for performing the pattern classification. In Yee (1998) it is shown that the outputs of a regularized RBF network classifier provide estimates of the posterior class probabilities. This is true only under the condition that the network is trained with the binary-valued class indicator vector type of desired outputs. We may now proceed to apply the decision rule of Eq. (4.55) for this class of networks:

Select the class corresponding to the maximum output function.

The method of strict interpolation for the selection of centers is tested with different values of regularization parameter λ . For a prescribed λ , Eq. (5.54) is used to compute the weight vector of the output layer in the RBF network, as shown by

$$\mathbf{w} = (\mathbf{G} + \lambda \mathbf{I})^{-1} \mathbf{d}$$

where $\mathbf{G}$ is an N -by- N Green's matrix whose ji -th element is equal to the radially symmetric Green's function $G(\mathbf{x}_j, \mathbf{x}_i)$, N is the sample size, and $\mathbf{d}$ is the desired response vector.

For each regularization parameter λ , the ensemble comprises 50 independent networks, each of which is tested against the same reference set of 1000 patterns.

Table 5.5 presents the ensemble statistics for the probability of correct classification P_c , computed for the case of $m_1 = 20$ centers. The ensemble statistics are computed for different values of the regularization parameter λ . Table 5.6 presents the corresponding results computed for the case of a larger regularized RBF network with $m_1 = 100$ centers.

Figure 5.7 displays the decision boundaries formed by the network outputs for a regularization parameter $\lambda = 10$, for which we have the best statistics. The two parts of Fig. 5.7 correspond to the best- and worst-performing network within the ensemble under test; both parts of the figure are for the case of 100 units.

TABLE 5.5 Size of Hidden Layer $m_1 = 20$ Centers: Details of Probability of Correct Classification, $P_c(\%)$ for Varying Regularization Parameter

Ensemble Statistic	Regularization Parameter, λ					
	0	0.1	1	10	100	1000
Mean	57.49	72.42	74.42	73.80	72.46	72.14
Std. dev. ^a	7.47	4.11	3.51	4.17	4.98	5.09
Minimum	44.20	61.60	65.80	63.10	60.90	60.50
Maximum	72.70	78.30	78.90	79.20	79.40	79.40

^aStd. dev.: Standard deviation.

TABLE 5.6 Size of Hidden Layer $m_1 = 100$ Centers: Details of Probability of Correct Classification, $P_c(\%)$ for Varying Regularization Parameter

Ensemble Statistic	Regularization Parameter, λ					
	0	0.1	1	10	100	1000
Mean	50.58	77.03	77.72	77.87	76.47	75.33
Std. dev. ^a	4.70	1.45	0.94	0.91	1.62	2.25
Minimum	41.00	70.60	75.10	75.10	72.10	70.10
Maximum	61.30	79.20	79.80	79.40	78.70	78.20

^aStd. dev.: Standard deviation.

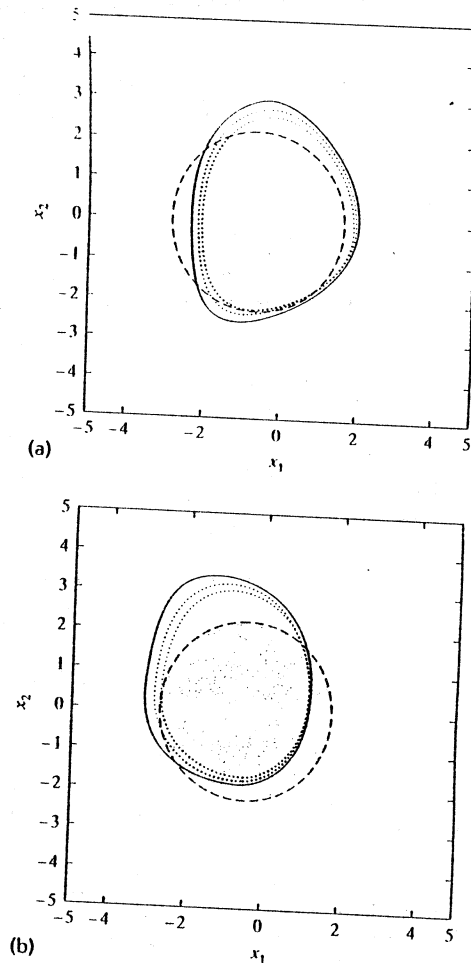


FIGURE 5.7 Results of experiment on pattern classification using strict interpolation with regularized RBF networks. (a) Best solution. (b) Worst solution. Dashed (shaded) circle represents the optimum Bayes solution.

Comparing Tables 5.5 and 5.6, we make the following observations:

1. For both $m_1 = 20$ and $m_1 = 100$ centers, the classification performance of the network for $\lambda = 0$ is relatively poor.
2. The use of regularization has a dramatic influence on the classification performance of the RBF network.
3. For $\lambda \geq 0.1$, the classification performance of the network is somewhat insensitive to an increase in the regularization parameter λ . For $m_1 = 20$ centers, best performance is obtained at $\lambda = 1$, and for $m_1 = 100$ centers, it is obtained at $\lambda = 10$.
4. Increasing the number of centers from $m_1 = 20$ to $m_1 = 100$ improves the classification performance by about 4.5 percent.

SUMMARY AND DISCUSSION

The structure of an RBF network is unusual in that the constitution of its hidden units is entirely different from that of its output units. With radial-basis functions providing the foundation for the design of the hidden units, the theory of RBF networks is linked closely with that of radial-basis functions, which is one of the main fields of study in numerical analysis (Singh, 1992). Another interesting point is that with linear weights of the output layer providing a set of adjustable parameters, much can be gained by ploughing through the extensive literature on linear adaptive filters (Haykin, 1996).

Unlike multilayer perceptrons trained with the back-propagation algorithm, the design of RBF networks follows a principled approach. In particular, Tikhonov's regularization theory presented in Section 5.5 provides a sound mathematical basis for the formulation of RBF networks. The Green's function $G(\mathbf{x}, \xi)$ plays a central role in this theory. The form of the Green's function as the basis function of the network is determined by the form of the *smoothing constraint* imposed in the application of regularization theory. The smoothing constraint specified by the differential operator $\mathbf{D}$ of Eq. (5.63) leads to the formulation of a multivariate Gaussian function for the Green's function. By formulating a different composition for the differential operator $\mathbf{D}$, we naturally end up with a different form of Green's function. Keep in mind that once the requirement of fewer basis functions than data points is relaxed, the reduction in computational complexity becomes an important factor in determining the smoothing regularizer. Here is another possible reason for the use of some other function (e.g., the thin-plate-spline function described in Problem 5.1) as the basis function in the design of the regularized RBF network in Fig. 5.5. Whatever the choice of basis functions, to derive the full benefits of regularization theory applied to the design of RBF networks, we need a principled approach for estimating the regularization parameter λ . The generalized cross-validation described in Section 5.9 fills this need. The theory justifying the use of generalized cross-validation is an asymptotic one, which stipulates that the training set must be large enough for a good estimate of λ to be attainable.

Another principled approach for the design of RBF networks is via kernel regression. This approach involves the use of density estimation, for which the radial-basis functions sum to unity exactly. Multivariate Gaussian distributions provide a convenient method for satisfying this requirement.

In conclusion, the input-output mapping function of a Gaussian RBF network bears a close resemblance to that realized by a mixture of experts. This latter model is discussed in Chapter 7.

NOTES AND REFERENCES

1. Radial-basis functions were first introduced in the solution of the real multivariate interpolation problem. The early work on this subject is surveyed in Powell (1985). It is now one of the main fields of research in numerical analysis.

Broomhead and Lowe (1988) were the first to exploit the use of radial-basis functions in the design of neural networks. Another major contribution to the theory and design of radial-basis function networks is due to Poggio and Girosi (1990a). This latter paper emphasizes the use of regularization theory applied to this class of networks as a method for improved generalization to new data.

2. The proof of Cover's theorem follows from these considerations (Cover, 1965):
 - *Schlafl's theorem* or *function-counting theorem*, which states that the number of homogeneously linearly separable dichotomies of N vectors in general position in Euclidean- m_1 space is equal to

$$C(N, m_1) = 2 \sum_{m=0}^{m_1-1} \binom{N-1}{m}$$

A set of vectors $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ is said to be in *general position* in Euclidean- m_1 space if every subset of m_1 or fewer vectors is linearly independent.

- *Reflection invariance* of the joint probability distribution of $\mathcal{X}$, which implies that the probability (conditional on $\mathcal{X}$) that a random dichotomy be separable is equal to the unconditional probability that a particular dichotomy of $\mathcal{X}$ (all N vectors in one category) be separable.

The function-counting theorem has been independently proved in different forms and applied to specific configurations of perceptrons (i.e., linear threshold units) by Cameron (1960), Joseph (1960), and Winder (1961). In Cover (1968), this theorem is applied to evaluate the capacity of a network of perceptrons in terms of the total number of adjustable parameters, which is shown to be lower-bounded by $N/(1 + \log_2 N)$ where N is the number of input patterns.

3. Another approach for regularization by incorporating prior information into an input-output mapping is through the use of Bayesian interpolation; for a detailed exposition of this approach see MacKay (1992a,b) and Neal (1995).
4. Regularization theory is usually credited to Tikhonov (1963). A similar approach was described in Phillips (1962). For this reason, the theory is sometimes referred to as *Tikhonov-Phillips regularization*.

A form of regularization in the actuarial literature was considered in Whittaker (1923); the smoothing process considered therein was referred to as *graduation* or *adjustment* of the observations.

For a discussion of regularization theory in book form, see Tikhonov and Arsenin (1977), Mozerov (1993), and Kirch (1996).

5. The concept "function space" evolved in consequence of Hilbert's fundamental investigation of a certain class of integral equations. While Fredholm, the originator of Fredholm integrals, formulated the problem in essentially algebraic language, Hilbert recognized the close relation of the problem with the analytic geometry of second-order surfaces in a Euclidean space of many dimensions (Lanczos, 1964).
6. A *normed space* is a linear vector space in which a real-valued function $\|\mathbf{x}\|$, called the *norm* of $\mathbf{x}$, is defined. The norm $\|\mathbf{x}\|$ has the following properties:

$$\|\mathbf{x}\| > 0 \text{ for } \mathbf{x} \neq \mathbf{0}$$

$$\|\mathbf{0}\| = 0$$

$$\|a\mathbf{x}\| = |a| \cdot \|\mathbf{x}\|, \quad a = \text{constant}$$

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$$

The norm $\|\mathbf{x}\|$ plays the role of "length" of $\mathbf{x}$.

7. Strictly speaking, we require the function $f(\mathbf{x})$ responsible for the data generation to be a member of a *reproducing kernel Hilbert space* (RKHS) with a reproducing kernel in the form of the Dirac delta distribution δ (Tapia and Thompson, 1978). We do this because

we require that the Dirac delta distribution δ be in the *decreasing, infinitely continuously differentiable functions*, that is, the classical space $\mathcal{S}$ of test functions for the *Schwarz theory of distributions*, with finite $\mathbf{D}$ -induced norm, as shown by

$$H_p = \{f \in \mathcal{S} : \|\mathbf{D}f\| < \infty\}$$

Generally speaking, engineers usually think of only the L_2 space whenever Hilbert space is mentioned, perhaps on the grounds that L_2 space is isomorphic to any Hilbert space. But the *norm* is the most important feature of a Hilbert space, and *isometrics* (i.e., norm-preserving isomorphism) are more important than simply additive isomorphism (Kailath, 1974). The theory of RKHS shows that there are many other different and quite useful Hilbert spaces besides the L_2 space. For a tutorial review of RKHS, see Kailath (1971).

8. An *inner product space* is a linear vector space in which the inner product of $\mathbf{u}$ and $\mathbf{v}$, denoted by $(\mathbf{u}, \mathbf{v})$, is induced with the following properties:

$$\begin{aligned} (\mathbf{u}, \mathbf{v}) &= (\mathbf{v}, \mathbf{u}) \\ (a\mathbf{u}, \mathbf{v}) &= a(\mathbf{u}, \mathbf{v}), \quad a = \text{constant} \\ (\mathbf{u} + \mathbf{v}, \mathbf{w}) &= (\mathbf{u}, \mathbf{w}) + (\mathbf{v}, \mathbf{w}) \\ (\mathbf{u}, \mathbf{u}) &> 0 \quad \text{for } \mathbf{u} \neq \mathbf{0} \end{aligned}$$

An inner product space $\mathcal{H}$ is said to be *complete*, and referred to as a Hilbert space, if every Cauchy sequence picked from $\mathcal{H}$ converges in norm to a limit in $\mathcal{H}$. A sequence of vectors $\{\mathbf{x}_n\}$ is called a *Cauchy sequence* if for every $\epsilon > 0$ there exists a number M such that (Debnath and Mikusiński 1990)

$$\|\mathbf{x}_m - \mathbf{x}_n\| < \epsilon \quad \text{for all } (m, n) > M$$

9. In Girosi et al. (1995), a different method for deriving Eq. (5.55) is presented by relating the regularizing term $\mathcal{E}_c(F)$ directly to the smoothness of the approximating function $F(\mathbf{x})$.

Smoothness is viewed as a measure of the oscillatory nature of a function. In particular, a function is said to be smoother than another function if it is less oscillatory. In other words, the smoother a function is, the smaller its high-frequency content will be. With this measure of smoothness in mind, let $F(\mathbf{s})$ be the multidimensional Fourier transform of $F(\mathbf{x})$, with $\mathbf{s}$ denoting a multidimensional transform variable. Let $H(\mathbf{s})$ denote a positive function that tends to zero as $\|\mathbf{s}\|$ approaches infinity, that is $1/H(\mathbf{s})$ represents the action of a "high-pass filter." Then, according to Girosi et al. (1995), we may define a smoothness functional representing the regularizing term as:

$$\mathcal{E}_c(F) = \frac{1}{2} \int_{\mathbb{R}^m} \frac{|F(\mathbf{s})|^2}{H(\mathbf{s})} d\mathbf{s}$$

where m_0 is the dimension of $\mathbf{x}$. By virtue of Parseval's theorem of Fourier theory, this functional is a measure of the power contained in the output of the high-pass filter $1/H(\mathbf{s})$. Thus, by casting the regularization problem in the Fourier domain and using properties of the Fourier transform, the solution of Eq. (5.55) is derived.

10. The most general form of a linear differential operator is

$$\mathbf{D} = p(x_1, x_2, \dots, x_{m_0}) \frac{\partial^n}{\partial x_1^a \partial x_2^b \dots \partial x_{m_0}^k}, \quad a + b + \dots + k = n$$

where $x_1, x_2, \dots, x_{m_0}$ are the elements of vector $\mathbf{x}$, and $p(x_1, x_2, \dots, x_{m_0})$ is some function of these elements. The adjoint operator of $\mathbf{D}$ is (Morse and Feshbach, 1953)

$$\tilde{\mathbf{D}} = (-1)^n \frac{\partial^n}{\partial x_1^a \partial x_2^b \dots \partial x_{m_0}^k} [p(x_1, x_2, \dots, x_{m_0})], \quad a + b + \dots + k = n$$

11. To obtain generalized cross-validation from ordinary cross-validation, we may consider a *ridge regression problem* described in Wahba (1990):

$$\mathbf{y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\epsilon} \quad (1)$$

where $\mathbf{X}$ is an N -by- N matrix of inputs, and the noise vector $\boldsymbol{\epsilon}$ has a mean vector of zero and a covariance matrix equal to $\sigma^2 \mathbf{I}$. Using the singular value decomposition of $\mathbf{X}$, we may write

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and $\mathbf{D}$ is a diagonal matrix. Let

$$\tilde{\mathbf{y}} = \mathbf{U}^T \mathbf{y}$$

$$\boldsymbol{\beta} = \mathbf{V}^T \boldsymbol{\alpha}$$

and

$$\tilde{\boldsymbol{\epsilon}} = \mathbf{U}^T \boldsymbol{\epsilon}$$

We may then use $\mathbf{U}$ and $\mathbf{V}$ to transform Eq. (1) into

$$\tilde{\mathbf{y}} = \mathbf{D}\boldsymbol{\beta} + \tilde{\boldsymbol{\epsilon}} \quad (2)$$

The diagonal matrix $\mathbf{D}$ (not to be confused with a differential operator) is chosen to have its singular values come in pairs. Then there is an orthogonal matrix $\mathbf{W}$ for which $\mathbf{W}\mathbf{D}\mathbf{W}^T$ is a *circulant matrix*; that is,

$$\mathbf{A} = \mathbf{W}\mathbf{D}\mathbf{W}^T$$

$$= \begin{bmatrix} a_0 & a_1 & \dots & a_{N-1} \\ a_{N-1} & a_0 & \dots & a_{N-2} \\ a_{N-2} & a_{N-1} & \dots & a_{N-3} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \dots & a_0 \end{bmatrix}$$

which is constant down the diagonal. Let

$$\mathbf{z} = \mathbf{W}\tilde{\mathbf{y}}$$

$$\boldsymbol{\gamma} = \mathbf{W}\boldsymbol{\beta}$$

and

$$\boldsymbol{\xi} = \mathbf{W}\tilde{\boldsymbol{\epsilon}}$$

We may then use $\mathbf{W}$ to transform Eq. (2) into

$$\mathbf{z} = \mathbf{A}\boldsymbol{\gamma} + \boldsymbol{\xi} \quad (3)$$

The diagonal matrix $\mathbf{D}$ has "maximally uncoupled" rows, while the circulant matrix $\mathbf{A}$ has "maximally coupled" rows.

With these transformations at hand, we may now state that generalized cross-validation is equivalent to transforming the ridge regression problem of Eq. (1) into the maximally coupled form Eq. (3), then doing ordinary cross-validation on $\mathbf{z}$, and finally transforming back to the original coordinate system (Wahba, 1990).

12. In an appendix to a chapter contribution in Powell (1992) that is based on a lecture presented in 1990, credit is given to a result due to A.C. Brown. The result, apparently obtained in 1981, states that an RBF network can map an arbitrary function from a closed domain in $\mathbb{R}^m$ to $\mathbb{R}$.

Hartman et al. (1990) consider Gaussian functions and approximations on compact subsets of $\mathbb{R}^m$ that are convex; therein it is shown that RBF networks with a single hidden layer of Gaussian units are universal approximators. However, the most rigorous proof of the universal approximation property of RBF networks is presented in Park and Sandberg (1991); this latter work was completed before the publication of the paper by Hartman et al.

13. Let Ω be a bounded domain in $\mathbb{R}^n$ with boundary Γ . Consider the set $\mathcal{F}$ of real-valued functions that are continuous and have a continuous gradient on $\Omega = \Omega \cup \Gamma$. The bilinear form

$$\int_{\Omega} (\text{grad } u; \text{grad } v + uv) dx$$

is clearly an admissible inner product on $\mathcal{F}$. The completion of $\mathcal{F}$ in the norm generated by this inner product is known as the *Sobolev space* (Debnath and Mikusiński, 1990). Sobolev spaces play an important role in the theory of partial differential equations and are therefore important examples of Hilbert spaces.

14. For a proof of the asymptotically unbiased property of the Parzen–Rosenblatt density estimator, see Parzen (1962) and Cacoullos (1966).
15. The Nadaraya–Watson regression estimator has been the subject of extensive study in statistics literature. In a broader context, nonparametric functional estimation occupies a central place in statistics; see Härdle (1990), and the collection of papers in Roussas (1991).

PROBLEMS

Radial-basis functions

- 5.1 The *thin-plate-spline* function is described by

$$\varphi(r) = \left(\frac{r}{\sigma}\right)^2 \log\left(\frac{r}{\sigma}\right) \text{ for some } \sigma > 0 \text{ and } r \in \mathbb{R}$$

Justify the use of this function as a translationally and rotationally invariant Green's function.

- 5.2 The set of values given in Section 5.8 for the weight vector $\mathbf{w}$ of the RBF network of Fig. 5.6 presents one possible solution for the XOR problem. Investigate another set of values for the weight vector $\mathbf{w}$ for solving this problem.
- 5.3 In Section 5.8 we presented a solution of the XOR problem using an RBF network with two hidden units. In this problem we consider an exact solution of the XOR problem using an RBF network with four hidden units, with each radial-basis function center

being determined by each piece of input data. The four possible input patterns are defined by (0, 0), (0, 1), (1, 1), (1, 0), which represent the cyclically ordered corners of a square.

- (a) Construct the interpolation matrix Φ for the resulting RBF network. Hence, compute the inverse matrix Φ^{-1} .
- (b) Calculate the linear weights of the output layer of the network.
- 5.4 The Gaussian function is the only radial-basis function that is factorizable. Using this property of the Gaussian function, show that a Green's function $G(\mathbf{x}, \mathbf{t})$ defined as a multivariate Gaussian distribution may be factorized as follows:

$$G(\mathbf{x}, \mathbf{t}) = \prod_{i=1}^m G(x_i, t_i)$$

where x_i and t_i are the i th elements of the m -by-1 vectors $\mathbf{x}$ and $\mathbf{t}$.

Regularized networks

- 5.5 Consider the cost functional

$$\mathcal{E}(F^*) = \sum_{i=1}^N \left[d_i - \sum_{j=1}^{m_1} w_j G(\|\mathbf{x}_j - \mathbf{t}_i\|) \right]^2 + \lambda \|\mathbf{D} F^*\|^2$$

which refers to the approximating function

$$F^*(\mathbf{x}) = \sum_{j=1}^{m_1} w_j G(\|\mathbf{x} - \mathbf{t}_j\|)$$

Using the Fréchet differential, show that the cost functional $\mathcal{E}(F^*)$ is minimized when

$$(\mathbf{G}^T \mathbf{G} + \lambda \mathbf{G}_0) \mathbf{w} = \mathbf{G}^T \mathbf{d}$$

where the N -by- m_1 matrix $\mathbf{G}$, the m_1 -by- m_1 matrix $\mathbf{G}_0$, the m_1 -by-1 vector $\mathbf{w}$, and the N -by-1 vector $\mathbf{d}$ are defined by Eqs. (5.72), (5.75), (5.73), and (5.46), respectively.

- 5.6 Suppose that we define

$$(\tilde{\mathbf{D}} \mathbf{D})_{ij} = \sum_{k=0}^{\infty} (-1)^k \frac{\nabla_{ij}^{2k}}{k! 2^k}$$

where

$$\nabla_{ij}^2 = \sum_{\mu=1}^{m_0} \sum_{\nu=1}^{m_0} u_{\mu\nu} \frac{\partial^2}{\partial x_{\mu} \partial x_{\nu}}$$

The m_0 -by- m_0 matrix $\mathbf{U}$, with its ji -th element denoted by u_{ji} , is symmetric and positive definite. Hence the inverse matrix $\mathbf{U}^{-1}$ exists, and so it permits the following decomposition via the similarity transformation:

$$\begin{aligned} \mathbf{U}^{-1} &= \mathbf{V}^T \Sigma \mathbf{V} \\ &= \mathbf{V}^T \Sigma^{1/2} \Sigma^{1/2} \mathbf{V} \\ &= \mathbf{C}^T \mathbf{C} \end{aligned}$$

where $\mathbf{V}$ is an orthogonal matrix, Σ is a diagonal matrix, $\Sigma^{1/2}$ is the square root of Σ , and the matrix $\mathbf{C}$ is defined by

$$\mathbf{C} = \Sigma^{1/2} \mathbf{V}$$

The problem is to solve for the Green's function $G(\mathbf{x}, \mathbf{t})$ that satisfies the following condition (in the distributional sense):

$$(\tilde{\mathbf{D}}\mathbf{D})_0 G(\mathbf{x}, \mathbf{t}) = \delta(\mathbf{x} - \mathbf{t})$$

Using the multidimensional Fourier transform to solve this equation for $G(\mathbf{x}, \mathbf{t})$, show that

$$G(\mathbf{x}, \mathbf{t}) = \exp\left(-\frac{1}{2}\|\mathbf{x} - \mathbf{t}\|_C^2\right)$$

where

$$\|\mathbf{x}\|_C^2 = \mathbf{x}^T \mathbf{C}^T \mathbf{C} \mathbf{x}$$

5.7 Consider a regularizing term defined by

$$\int_{\mathbb{R}^m} \|\mathbf{D}F(\mathbf{x})\|^2 d\mathbf{x} = \sum_{k=0}^{\infty} a_k \int_{\mathbb{R}^m} \|D^k F(\mathbf{x})\|^2 d\mathbf{x}$$

where

$$a_k = \frac{\sigma^{2k}}{k!2^k}$$

and the linear differential operator D is defined in terms of the gradient operator ∇ and the Laplacian operator ∇^2 as follows:

$$D^{2k} = (\nabla^2)^k$$

and

$$D^{2k+1} = \nabla(\nabla^2)^k$$

Show that

$$\mathbf{D}F(\mathbf{x}) = \sum_{k=0}^{\infty} \frac{\sigma^{2k}}{k!2^k} \nabla^{2k} F(\mathbf{x})$$

5.8 In Section 5.5 we derived the approximating function $F_k(\mathbf{x})$ of Eq. (5.66) by using the relationship of Eq. (5.65). In this problem we wish to start with the relationship of Eq. (5.65) and use the multidimensional Fourier transformation to derive Eq. (5.66). Perform this derivation by using the following definition of the multidimensional Fourier transform of the Green's function $G(\mathbf{x})$:

$$G(\mathbf{s}) = \int_{\mathbb{R}^m} G(\mathbf{x}) \exp(-i \mathbf{s}^T \mathbf{x}) d\mathbf{x}$$

where $i = \sqrt{-1}$ and $\mathbf{s}$ is the m_0 -dimensional transform variable.

5.9 Consider the nonlinear regression problem described in Eq. (5.95). Let a_{ik} denote the ik -th element of the inverse matrix $(\mathbf{G} + \lambda \mathbf{I})^{-1}$. Hence, starting with Eq. (5.58), show that the estimate of the regression function $f(\mathbf{x})$ may be expressed as

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^N \psi(\mathbf{x}, \mathbf{x}_k) y_k$$

where y_k is the model output for the input $\mathbf{x}_k$, and

$$\psi(\mathbf{x}, \mathbf{x}_k) = \sum_{i=1}^N G(\|\mathbf{x} - \mathbf{x}_i\|) a_{ik}, \quad k = 1, 2, \dots, N$$

where $G(\|\cdot\|)$ is the Green's function.

5.10 *Spline functions* are examples of piecewise polynomial approximators (Schumaker, 1981). The basic idea behind the method of splines is as follows. An approximation region of interest is broken up into a finite number of subregions via the use of *knots*; the knots can be fixed, in which case the approximators are *linearly* parameterized, or they can be variable, in which case the approximators are *nonlinearly* parameterized. In both cases, in each region of the approximation a polynomial of degree at most n is used, with the additional requirement that the overall function be $n-1$ times differentiable. Polynomial splines are relatively smooth functions that are easy to store, manipulate, and evaluate on a computer.

Among spline functions used in practice, *cubic splines* are perhaps the most popular. The cost functional for a cubic spline, pertaining to a one-dimensional input, is defined by

$$\mathcal{E}(F) = \frac{1}{2} \sum_{i=1}^N [y_i - f(x_i)]^2 + \frac{\lambda}{2} \int_{x_1}^{x_N} \left[\frac{d^2 f(x)}{dx^2} \right]^2 dx$$

where, in the language of splines, λ denotes a *smoothing* parameter.

(a) Justify the following properties of the solution $f_\lambda(x)$ to this problem:

(1) $f_\lambda(x)$ is a cubic polynomial between two successive values of x .

(2) $f_\lambda(x)$ and its first two derivatives are all continuous, except at the boundary points where the second derivative of $f_\lambda(x)$ is zero.

(b) Since $\mathcal{E}(f)$ has a unique minimum, we must have

$$\mathcal{E}(f_\lambda + \alpha g) \geq \mathcal{E}(f_\lambda)$$

for any g drawn from the same class of twice-differentiable functions as f_λ and for any real-valued constant α . This means that $\mathcal{E}(f_\lambda + \alpha g)$, interpreted as a function of α , must have a local minimum at $\alpha = 0$. Hence, show that

$$\int_{x_1}^{x_N} \left(\frac{d^2 f_\lambda(x)}{dx^2} \right) \left(\frac{d^2 g(x)}{dx^2} \right) dx = \frac{1}{2} \sum_{i=1}^N [y_i - f_\lambda(x_i)] g(x_i)$$

which is the Euler-Lagrange equation for the cubic spline problem.

Rate of approximation

5.11 Equation (5.124) defines the upper bound on the generalization error of a Gaussian RBF network designed to learn a regression function that belongs to a certain Sobolev space. Using this bound, derive the formula of Eq. (5.125) for the optimum size of this network for a specified size of training sample.

Kernel estimation

5.12 Suppose that you are given a "noiseless" training sample $\{f(\mathbf{x}_i)\}_{i=1}^N$, and that the requirement is to design a network that generalizes to data samples that are corrupted by additive noise and therefore not included in the training set. Let $F(\mathbf{x})$ denote the approximating function realized by such a network, which is chosen so that the expected squared error

$$J(F) = \frac{1}{2} \sum_{i=1}^N \int_{\mathbb{R}^m} [f(\mathbf{x}_i) - F(\mathbf{x}_i, \xi)]^2 f_\xi(\xi) d\xi$$

is minimum, where $f_\xi(\mathbf{x})$ is the probability density function of a noise distribution in the input space $\mathbb{R}^m$. Show that the solution of this least-squares problem is given by (Webb, 1994)

$$F(\mathbf{x}) = \frac{\sum_{i=1}^N f(\mathbf{x}_i) f_\xi(\mathbf{x} - \mathbf{x}_i)}{\sum_{i=1}^N f_\xi(\mathbf{x} - \mathbf{x}_i)}$$

Compare this estimator to the Nadaraya–Watson regression estimator.

Supervised selection of centers

5.13 Consider the cost functional

$$\mathcal{E} = \frac{1}{2} \sum_{j=1}^N e_j^2$$

where

$$\begin{aligned} e_j &= d_j - F^*(x_j) \\ &= d_j - \sum_{i=1}^{m_1} w_i G(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i}) \end{aligned}$$

The free parameters are the linear weights w_i , the centers $\mathbf{t}_i$ of the Green's functions, and the inverse covariance matrix $\Sigma_i^{-1} = \mathbf{C}_i^T \mathbf{C}_i$, where $\mathbf{C}_i$ is the norm weighting matrix. The problem is to find the values of these free parameters that minimize the cost functional $\mathcal{E}$. Derive the following partial derivatives:

- $\frac{\partial \mathcal{E}}{\partial w_i} = \sum_{j=1}^N e_j G(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i})$
- $\frac{\partial \mathcal{E}}{\partial \mathbf{t}_i} = 2w_i \sum_{j=1}^N e_j G'(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i}) \Sigma_i^{-1} (\mathbf{x}_j - \mathbf{t}_i)$
- $\frac{\partial \mathcal{E}}{\partial \Sigma_i^{-1}} = -w_i \sum_{j=1}^N e_j G'(\|\mathbf{x}_j - \mathbf{t}_i\|_{C_i}) \mathbf{Q}_{ji}$

where $G'(\cdot)$ is the derivative of $G(\cdot)$ with respect to its argument, and

$$\mathbf{Q}_{ji} = (\mathbf{x}_j - \mathbf{t}_i)(\mathbf{x}_j - \mathbf{t}_i)^T$$

For the rules for differentiating a scalar with respect to a vector, see note 2 of Chapter 3.

Computer Experiments

5.14 In this problem we continue with the computer experiment in Section 5.13 to study the random selection of centers for the design of an RBF network used as a binary pattern classifier. The purpose of the experiment is to demonstrate that the generalization performance of the network so trained is relatively good.

The network is intended to solve the binary pattern-classification problem described in Section 5.13, where the requirement is to classify data drawn from a mixture model consisting of two equiprobable overlapping two-dimensional Gaussian distributions. One distribution has a mean vector $[0, 0]^T$ and common variance 1, whereas the other distribution has a mean vector $[0, 2]^T$ and common variance 4. The “select class with maximum function output” decision rule is used for the classification.

- Consider a random selection of centers using $m_1 = 20$ centers. Compute the mean, standard deviation, and minimum and maximum values of the probability of correct

classification P_c for different values of regularization parameter $\lambda = 0, 0.1, 1, 10, 100, 1000$. For the computation of ensemble statistics, use 50 independent network trials per ensemble, with each one tested against a fixed reference set of 1000 patterns.

- Construct the decision boundary computed for the configuration described in part (a) for regularization parameter $\lambda = 1$.
- Repeat the computations described in part (a) for $m_1 = 10$ centers (selected at random).
- In light of your results, discuss the merit of random selection of centers as a method for the design of RBF networks, and the role of regularization in the performance of the network as a pattern classifier.
- Compare your results with those presented in Section 5.13 that were computed using the method of strict interpolation. In particular, confirm that the random selection of centers is relatively insensitive to the regularization parameter.

5.15 It may be argued that in the case of the experiment described in Section 5.13 involving the classification of a pair of Gaussian-distributed classes, the RBF network considered there performed well since it uses Gaussian radial-basis functions to approximate the underlying Gaussian class conditional distributions. In this problem we use a computer experiment to explore the design of a strict-interpolation Gaussian RBF network for distinctly discontinuous class conditional distributions. Specifically, consider two equiprobable classes $\mathcal{C}_1$ and $\mathcal{C}_2$ whose distributions

- $U(\mathcal{C}_1)$, where $\mathcal{C}_1 \triangleq \Omega_1$ is a circle of radius $r = 2.34$ centered at $\mathbf{x}_c = [-2, 30]^T$
- $U(\mathcal{C}_2)$, where $\mathcal{C}_2 \subset \mathbb{R}^2$ is a square region centered at $\mathbf{x}_c$ with side length $r = \sqrt{2\pi}$

Here $U(\Omega)$ denotes a uniform distribution over $\Omega \subset \mathbb{R}^2$. These parameters are chosen so that the decision region for class $\mathcal{C}_1$ is the same as in the Gaussian-distributed case considered in Section 5.13. Investigate the use of regularization as a means of improving the classification performance of a Gaussian RBF network using strict interpolation.

