Radial Basis Function (RBF) Networks

Function approximation

- MLP is both a pattern classifier and a function approximator
- As a function approximator, MLP is nonlinear, semiparametric, and universal

Function approximation background

- Weierstrass theorem: any continuous real function in an interval can be approximated arbitrarily well by a set of polynomials
- Taylor expansion approximates any differentiable function by polynomials in a neighborhood of a point
- Fourier series gives a way of approximating any periodic function by a sum of sine's

Linear projection

- Approximate function f (x) by a linear combination of simpler functions

$$F(\mathbf{x}) = \sum_j w_j \varphi_j(\mathbf{x})$$

- If w_j's can be chosen so that approximation error is arbitrarily small for any function f (**x**) over the domain of interest, $\{\varphi_j\}$ has the property of universal approximation, or $\{\varphi_j\}$ is complete

Example bases

- $\mathrm{sinc}\, x = \frac{\sin(x)}{x}$

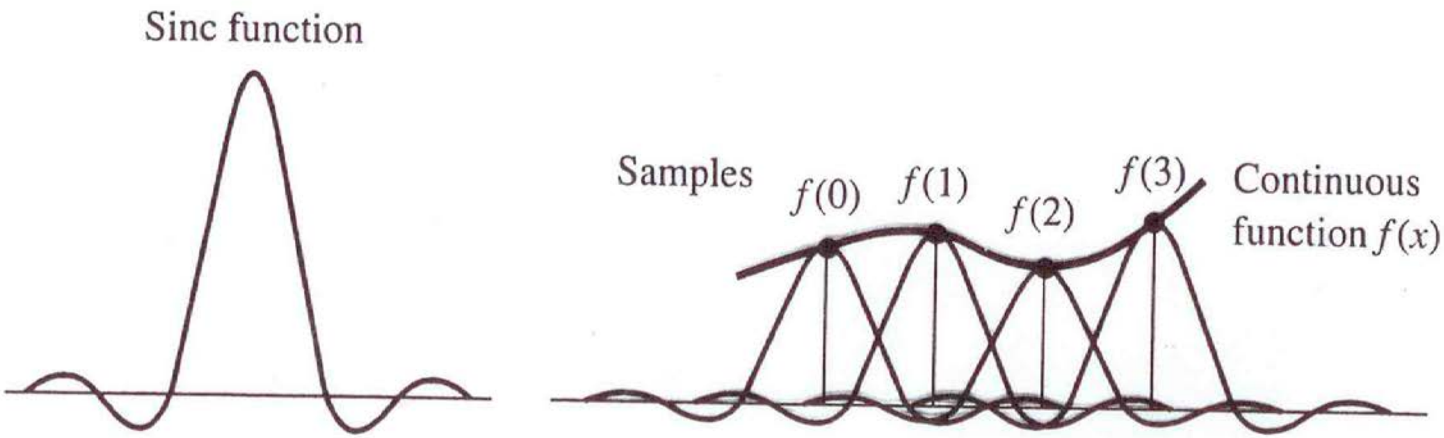


FIGURE 5-4 Decomposition by sinc functions

Example bases (cont.)

- sine function

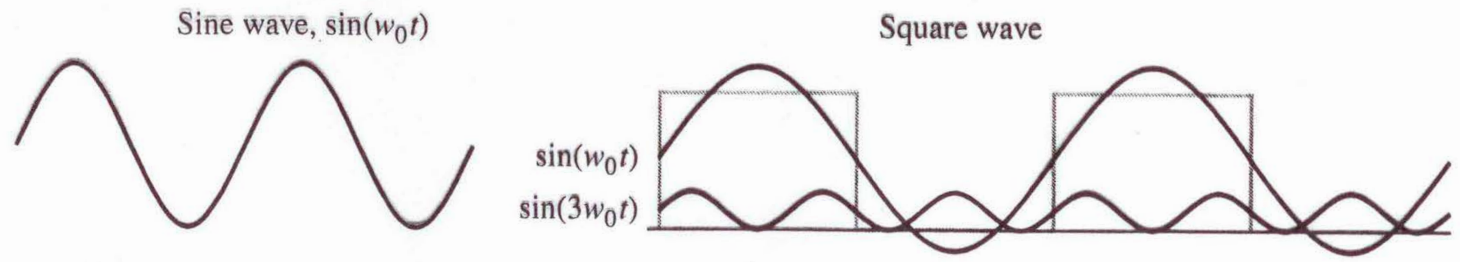


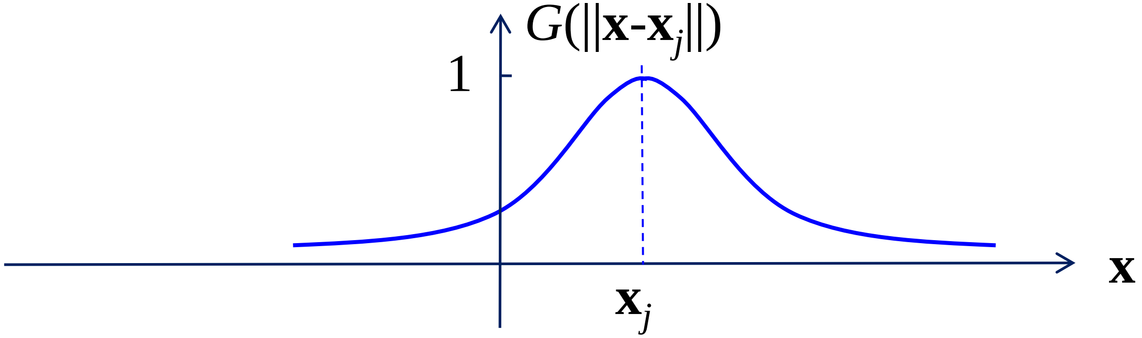
FIGURE 5-5 Decomposition by sine waves

Radial basis functions

- Consider

$$\varphi_j(\mathbf{x}) = \exp\left(-\frac{1}{2\sigma^2}||\mathbf{x}-\mathbf{x}_j||^2\right)$$
$$= G(||\mathbf{x}-\mathbf{x}_j||)$$

- A Gaussian is a local basis function, falling off exponentially from the center

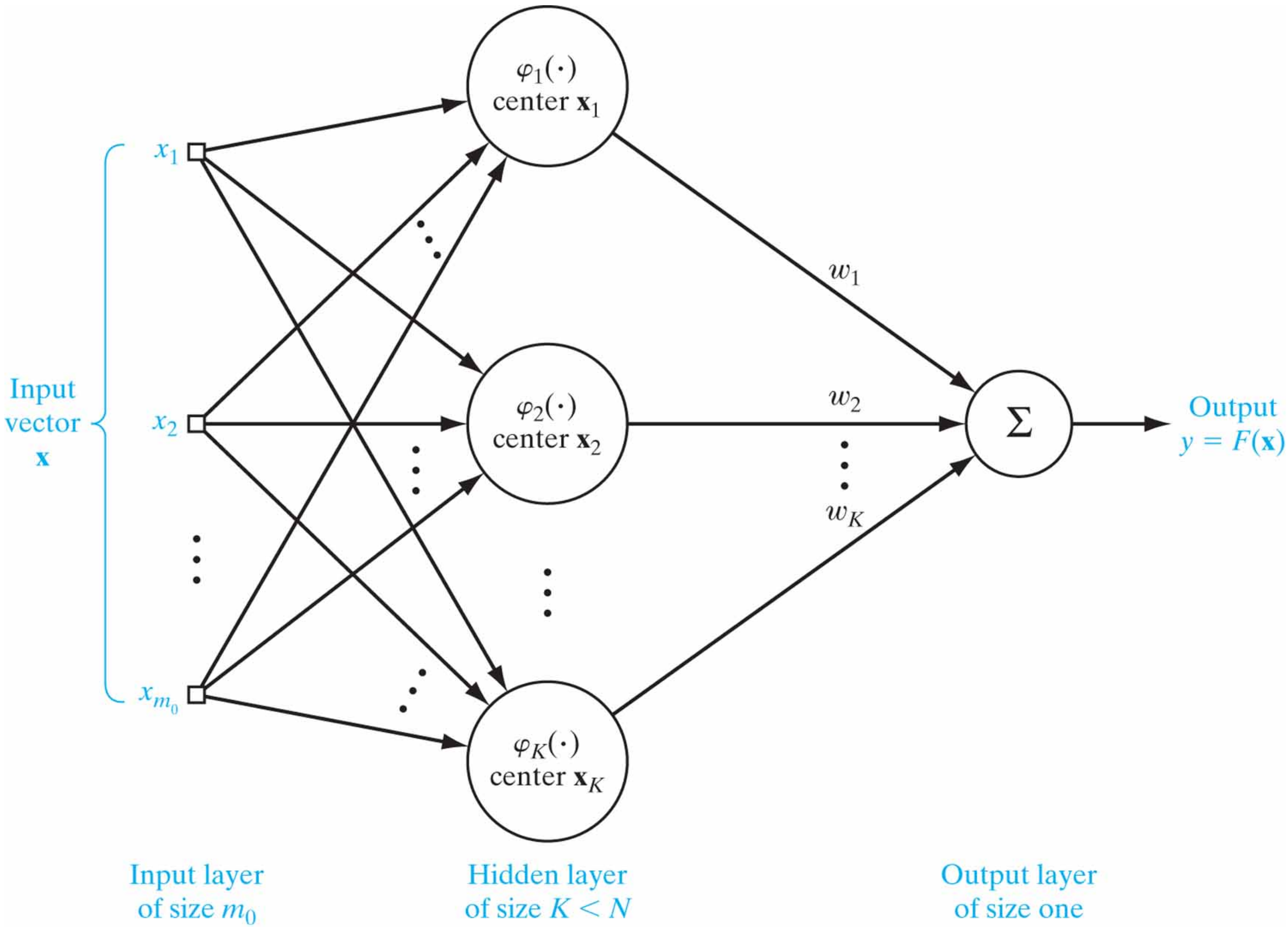


Radial basis functions (cont.)

- Thus approximation by RBF becomes

$$F(\mathbf{x}) = \sum_j w_j G(||\mathbf{x} - \mathbf{x}_j||)$$

RBF approximation illustration



Remarks

- Gaussians are universal approximators

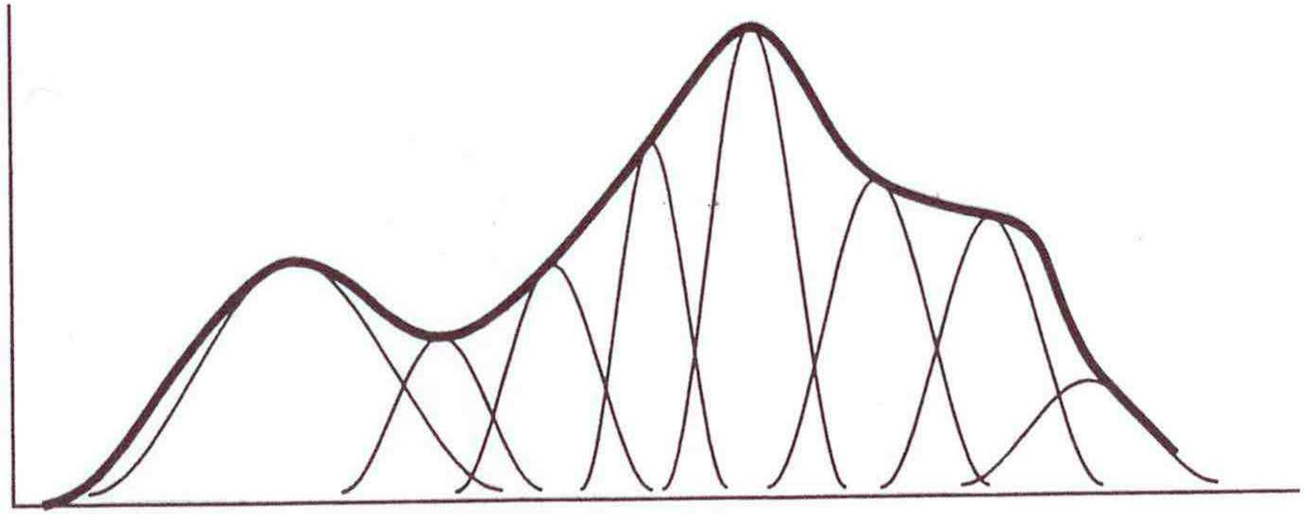


FIGURE 5-8 Approximation by RBFs in one dimension

Remarks (cont.)

- Such a radial basis function is called a kernel, a term from statistics
  - As a result, RBF nets are a kind of kernel methods
- Other RBFs exist, such as multiquadrics (see textbook)

Four questions to answer for RBF nets

- How to identify Gaussian centers?
- How to determine Gaussian widths?
- How to choose weights w_j's?
- How to select the number of bases?

Gaussian centers

- Identify Gaussian centers via unsupervised clustering: *K*-means algorithm
- Goal of the *K*-means algorithm: Divide *N* input patterns into *K* clusters so as to minimize the final variance. In other words, partition patterns into *K* clusters C_j's to minimize the following cost function

$$J = \sum_{j=1}^{K} \sum_{i \in C_j} ||\mathbf{x}_i - \mathbf{u}_j||^2$$

where $\mathbf{u}_j = \frac{1}{||C_j||} \sum_{i \in C_j} \mathbf{x}_i$ is the mean (center) of cluster *j*

K-means algorithm

1. Choose a set of K cluster centers randomly from the input patterns
2. Assign the N input patterns to the K clusters using the squared Euclidean distance rule:

$$\mathbf{x} \text{ is assigned to } C_j \text{ if } ||\mathbf{x}-\mathbf{u}_j||^2 \leq ||\mathbf{x}-\mathbf{u}_i||^2 \text{ for } i \neq j$$

K-means algorithm (cont.)

3. Update cluster centers

$$\mathbf{u}_j = \frac{1}{||C_j||} \sum_{i \in C_j} \mathbf{x}_i$$

4. If any cluster center changes, go to step 2; otherwise stop

- **Remark:** The *K*-means algorithm always converges, but the global minimum is not assured

Calculating Gaussian widths

- Once cluster centers are determined, the variance within each cluster can be set to

$$\sigma_j^2 = \frac{1}{||C_j||} \sum_{i \in C_j} ||\mathbf{u}_j - \mathbf{x}_i||^2$$

  - **Remark**: to simplify the RBF net design, different clusters often assume the same Gaussian width:

$$\sigma = \frac{d_{\max}}{\sqrt{2K}}$$

where d_{max} is the maximum distance between cluster centers

Weight update

- With the hidden layer decided, weight training can be treated as a linear regression problem, and the LMS algorithm is an efficient way for weight update
  - Note that a bias term needs to be included

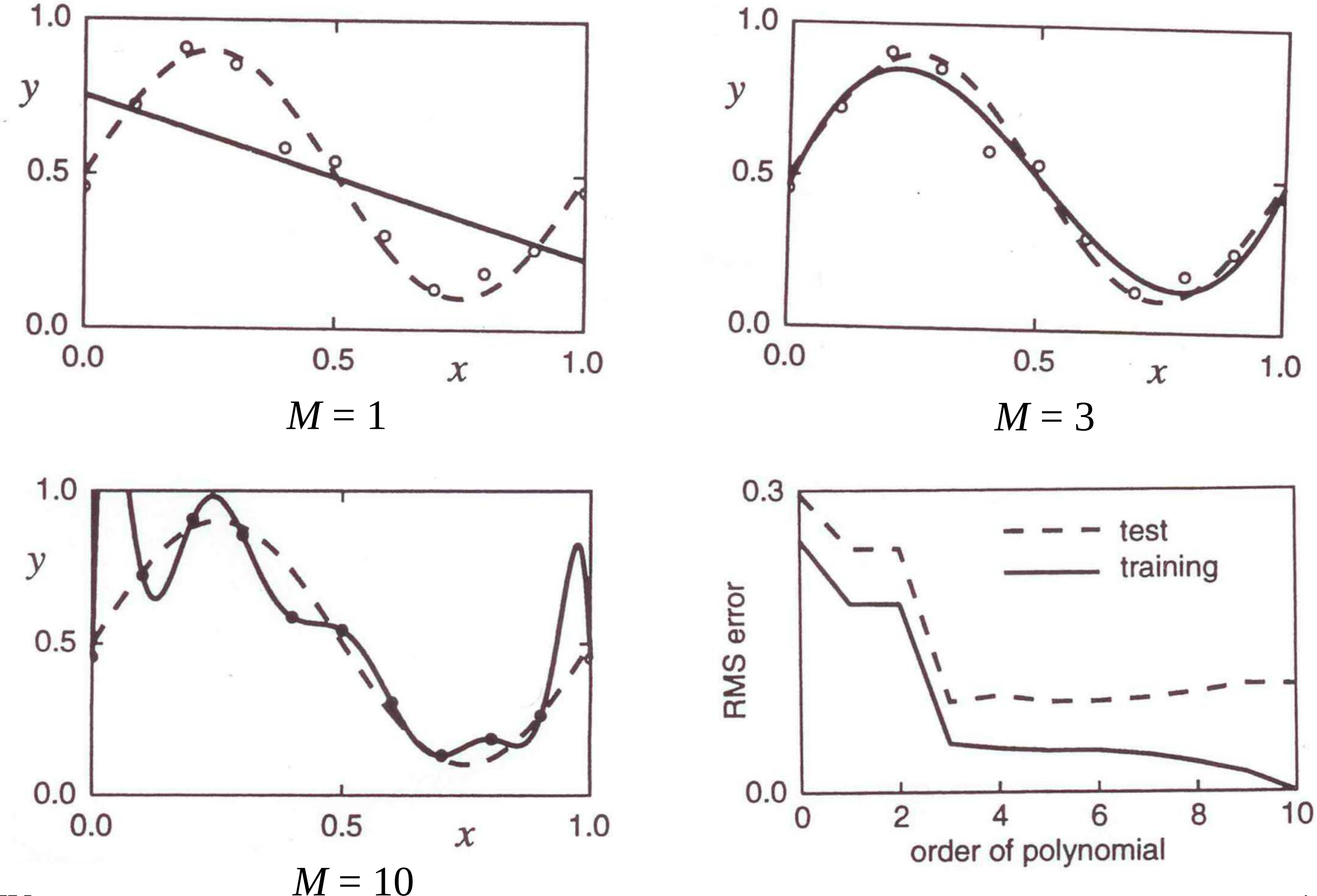
Selection of the number of bases: bias-variance dilemma

- The same problem as that of selecting the size of an MLP for classification
- The problem of overfitting
  - Example: Consider polynomial curve fitting

$$F(x) = \sum_{j=0}^{M} w_j x^j$$

for $f(x) = 0.5 + 0.4\sin(2\pi x)$ by an M-order F

Overfitting: $F(x) = \sum_{j=0}^{M} w_j x^j$



Occam's razor

- The best scientific model is the simplest that is consistent with the data
  - In our case, it translates to the principle that a learning machine should be large enough to approximate the data well, but not larger
- Occam's razor is a general principle governing supervised learning and generalization

Bias and variance

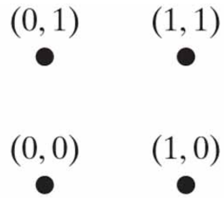
- Bias: training error – difference between desired output and actual output for a particular training sample
- Variance: generalization error – difference between the learned function from a particular training sample and the function derived from all training samples
  - Example: two extreme cases: zero bias and zero variance

Bias and variance (cont.)

- The optimal size of a learning machine is thus a compromise between the bias and the variance of a model
  - In other words, a good-sized model is the one where both bias and variance are low
- For RBF nets, in practice, cross validation can be applied to select the number of bases, where validation error is measured for a series of numbers of bases

XOR problem, again

- RBF nets can also be applied to pattern classification problems
  - XOR problem revisited



(a)

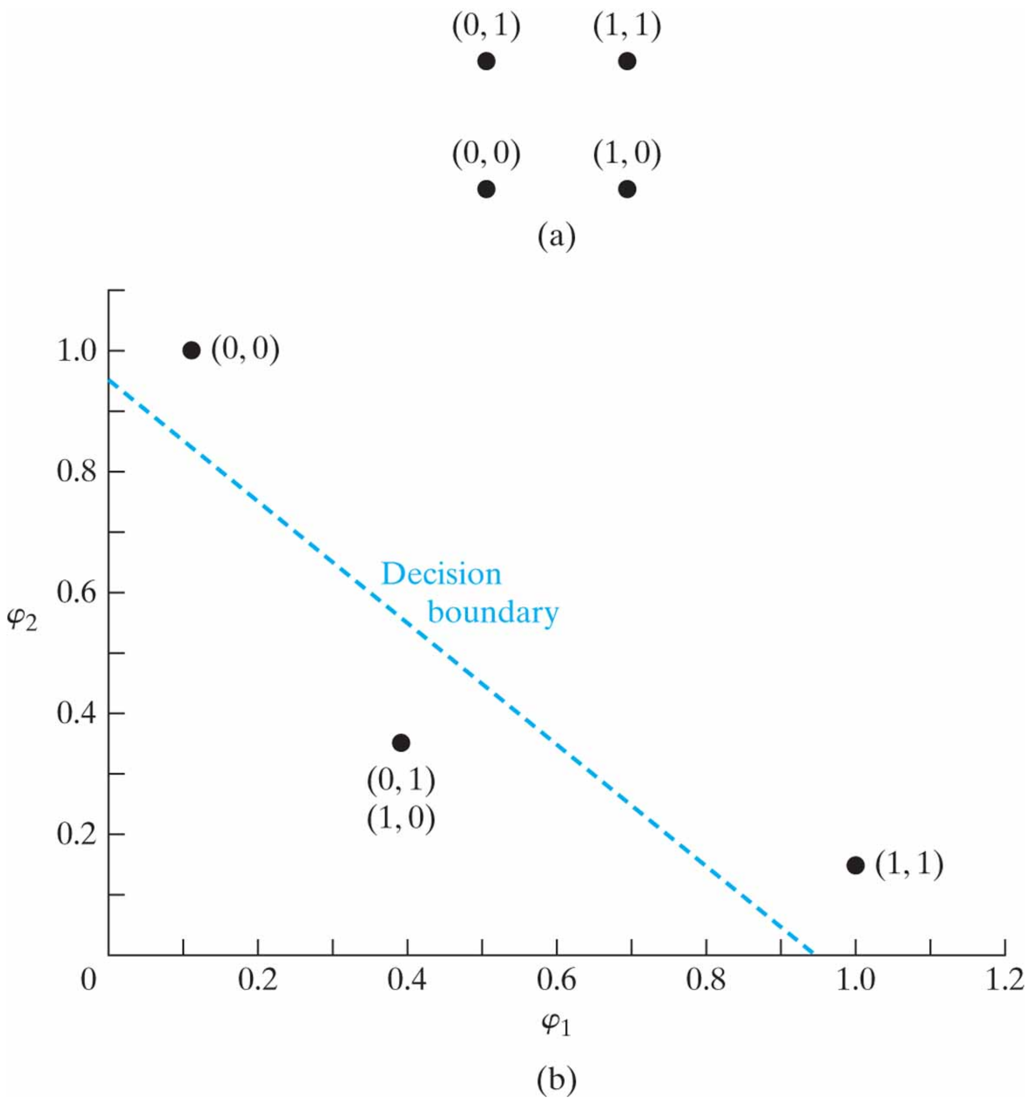
XOR problem (cont.)

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

XOR problem, again

- RBF nets can also be applied to pattern classification problems
  - XOR problem revisited



Comparison between RBF and MLP

- For RBF nets, bases are local, while for MLP, “bases” are global
- Generally, more bases are needed than hidden units in MLP
- Training is more efficient for RBF nets