CHAPTER 3

FUZZY LOGIC FUNDAMENTALS

3.1 INTRODUCTION

The past few years have witnessed a rapid growth in the number and variety of applications of fuzzy logic (FL). FL techniques have been used in image-understanding applications such as detection of edges, feature extraction, classification, and clustering. Fuzzy logic poses the ability to mimic the human mind to effectively employ modes of reasoning that are approximate rather than exact. In traditional hard computing, decisions or actions are based on precision, certainty, and vigor. Precision and certainty carry a cost. In soft computing, tolerance and impression are explored in decision making. The exploration of the tolerance for imprecision and uncertainty underlies the remarkable human ability to understand distorted speech, decipher sloppy handwriting, comprehend nuances of natural language, summarize text, and recognize and classify images. With FL, we can specify mapping rules in terms of words rather than numbers. Computing with the words explores imprecision and tolerance. Another basic concept in FL is the fuzzy if–then rule. Although rule-based systems have a long history of use in artificial intelligence, what is missing in such systems is machinery for dealing with fuzzy consequents or fuzzy antecedents. In most applications, an FL solution is a translation of a human solution. Thirdly, FL can model nonlinear functions of arbitrary complexity to a desired degree of accuracy. FL is a convenient way to map an input space to an output space. FL is one of the tools used to model a multiinput, multioutput system.

Soft computing includes fuzzy logic, neural networks, probabilistic reasoning, and genetic algorithms. Today, techniques or a combination of techniques from all these areas are used to design an intelligence system. Neural networks provide algorithms for learning, classification, and optimization, whereas fuzzy logic deals with issues such as forming impressions and reasoning on a semantic or linguistic level. Probabilistic reasoning deals with uncertainty. Although there are substantial areas of overlap between neural networks, FL, and probabilistic reasoning, in general they are complementary rather than competitive. Recently, many intelligent systems called neuro fuzzy systems have been used. There are many ways to combine neural networks and FL techniques. Before doing so, however, it is necessary to understand basic ideas in the design of FL techniques. In this chapter, we will introduce FL concepts such as fuzzy sets and their properties, FL operators, hedges, fuzzy proposition and rule-based systems, fuzzy maps and inference engine, defuzzification methods, and the design of an FL decision system.

3.2 FUZZY SETS AND MEMBERSHIP FUNCTIONS

Zadeh introduced the term *fuzzy logic* in his seminal work “Fuzzy sets,” which described the mathematics of fuzzy set theory (1965). Plato laid the foundation for what would become fuzzy logic, indicating that there was a third region beyond True and False. It was Lukasiewicz who first proposed a systematic alternative to the bivalued logic of Aristotle. The third value Lukasiewicz proposed can be best translated as “possible,” and he assigned it a numeric value between True and False. Later he explored four-valued logic and five-valued logic, and then he declared that, in principle, there was nothing to prevent the derivation of infinite-valued logic. FL provides the opportunity for modeling conditions that are inherently imprecisely defined. Fuzzy techniques in the form of approximate reasoning provide decision support and expert systems with powerful reasoning capabilities. The permissiveness of fuzziness in the human thought process suggests that much of the logic behind thought processing is not traditional two-valued logic or even multivalued logic, but logic with fuzzy truths, fuzzy connectiveness, and fuzzy rules of inference. A fuzzy set is an extension of a crisp set. Crisp sets allow only full membership or no membership at all, whereas fuzzy sets allow partial membership. In a crisp set, membership or nonmembership of element x in set A is described by a characteristic function $\mu_A(x)$, where $\mu_A(x) = 1$ if $x \in A$ and $\mu_A(x) = 0$ if $x \notin A$. Fuzzy set theory extends this concept by defining partial membership. A fuzzy set A on a universe of discourse U is characterized by a membership function $\mu_A(x)$ that takes values in the interval $[0,1]$. Fuzzy sets represent commonsense linguistic labels like *slow, fast, small, large, heavy, low, medium, high, tall,* etc. A given element can be a member of more than one fuzzy set at a time. A fuzzy set A in U may be represented as a set of ordered pairs. Each pair consists of a generic element x and its grade of membership function; that is, i $A = \{(x, \mu_A(x)) \mid x \in U\}$, x is called a support value if $\mu_A(x) > 0$. A linguistic variable x in the universe of discourse U is characterized by $T(x) = \{T_x^{\,1}, T_x^{\,2}, \ldots, T_x^{\,k}\}$ and $\mu(x) = \{\mu_x^{\,1}, \mu_x^{\,2}, .., \mu_x^{\,k}\}$, where $T(x)$ is the term set of x — that is, the set of names of linguistic values of x, with each T_x^i being a fuzzy number with membership function μ_x^i defined on U. For example, if x indicates height, then $T(x)$ may refer to sets such as *short, medium,* or *tall*. A membership function is essentially a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. As an example, consider a fuzzy set *tall*. Let the universe of discourse be heights from 40 inches to 90 inches. With a crisp set, all people with height 72 or more inches are considered tall, and all people with height of less than 72 inches are considered not tall. The crisp set membership function for set *tall* is shown in Figure 3.1. The corresponding fuzzy set with a smooth membership function is shown in Figure 3.2. The curve defines the transition from not tall and shows the degree of mem-

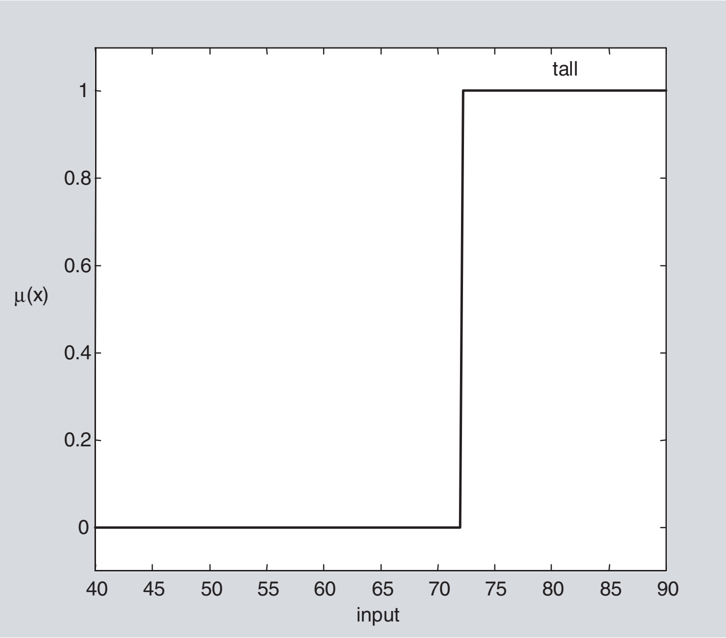

Figure 3.1 Crisp membership function.

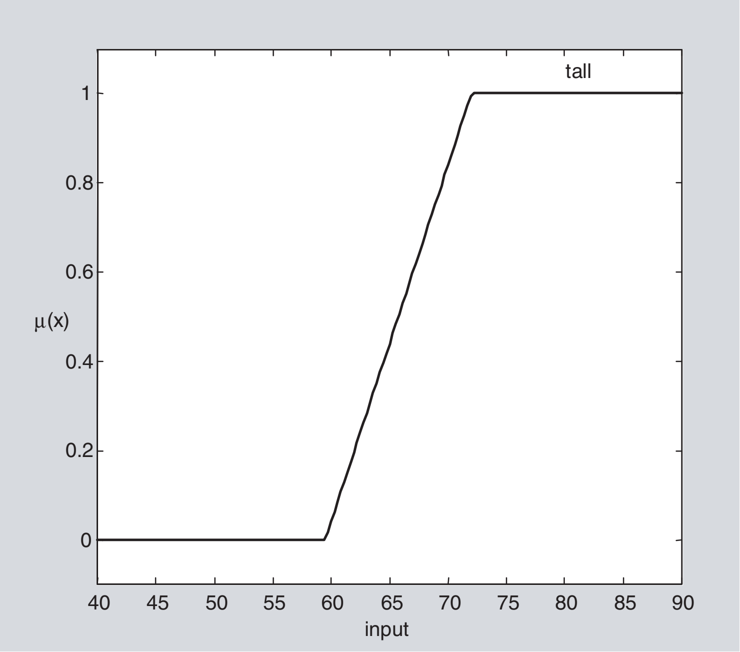

Figure 3.2 An example of a fuzzy membership function.

bership for a given height. We can extend this concept to multiple sets. If we consider a universe of discourse from 40 inches to 90 inches, then, to describe height, we can use three term values such as *short, average,* and *tall*. In practice, the terms *short, medium,* and *tall* are not used in the strict sense. Instead, they imply a smooth transition. Fuzzy membership functions representing these sets are shown in Figure 3.3. The Figure shows that a person with height 65 inches will have membership value 1 for set medium, whereas a person with height 60 inches may be a member of the set short and also a member of the set medium; only the degree of membership varies with these sets. Various types of membership functions are used, including triangular, trapezoidal, generalized bell shaped, Gaussian curves, polynomial curves, and sigmoid functions. Figure 3.3 shows trapezoidal membership functions. Triangular curves depend on three parameters *a, b,* and *c* and are given by

$$f(x;a,b,c) = \begin{cases} 0 & \text{for } x < a \\ \dfrac{x-a}{b-a} & \text{for } a \le x < b \\ \dfrac{c-x}{c-b} & \text{for } b \le x \le c \\ 0 & \text{for } x > c \end{cases} \tag{3.1}$$

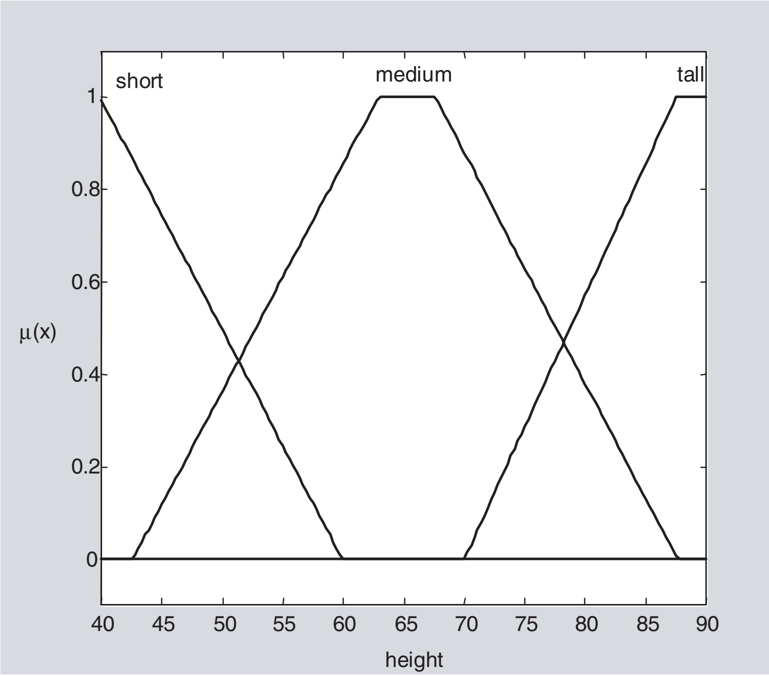

Figure 3.3 Trapezoidal membership functions.

Trapezoidal curves depend on four parameters and are given by

$$f(x;a,b,c,d) = \begin{cases} 0 & \text{for } x < a \\ \dfrac{x-a}{b-a} & \text{for } a \le x < b \\ 1 & \text{for } b \le x < c \\ \dfrac{d-x}{d-c} & \text{for } c \le x < d \\ 0 & \text{for } d \le x \end{cases} \tag{3.2}$$

The π-shaped membership functions are given by (Giarratano and Riley, 1993)

$$f(x;b,c) = \begin{cases} S\left(x; c-b, c-b/2, c\right) & \text{for } x \le c \\ 1 - S\left(x; c, c+b/2, c+b\right) & \text{for } x > c \end{cases} \tag{3.3}$$

where $S\left(x;a,b,c\right)$ represents a membership function defined as

$$S(x;a,b,c) = \begin{cases} 0 & \text{for } x < a \\ \dfrac{2\left(x-a\right)^2}{\left(c-a\right)^2} & \text{for } a \le x < b \\ 1 - \dfrac{2\left(x-c\right)^2}{\left(c-a\right)^2} & \text{for } b \le x \le c \\ 1 & \text{for } x > c \end{cases} \tag{3.4}$$

In Equation (3.4), *a, b*, and *c* are the parameters that are adjusted to fit the desired membership data. The parameter *b*? is the half width of the curve at the crossover point. The Gaussian and π-shaped membership functions are shown in Figures 3.4 and 3.5, respectively. Gaussian curves depend on two parameters σ and c and are represented by

$$f(x;\sigma,c) = \exp\left[\frac{-(x-c)^2}{2\sigma^2}\right] \tag{3.5}$$

In designing a fuzzy inference system, membership functions are associated with term sets that appear in the antecedent or consequent of rules.

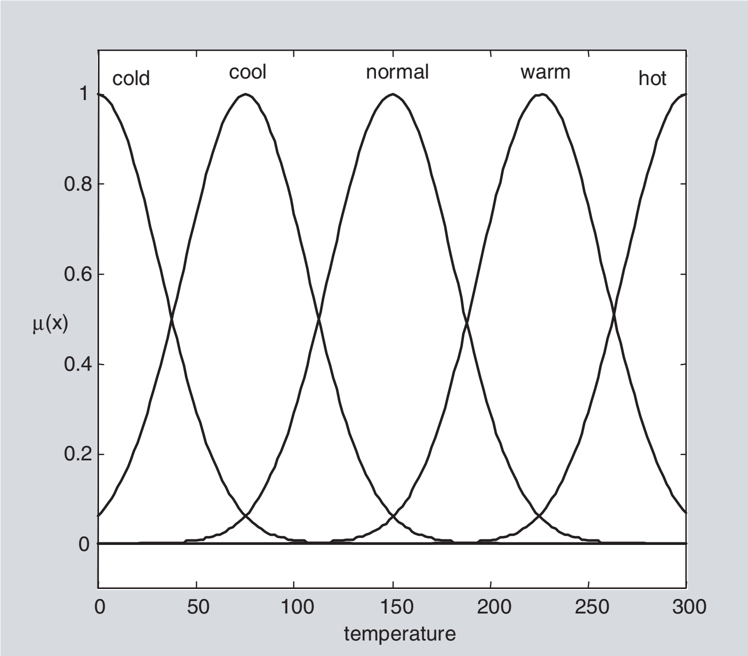

Figure 3.4 Gaussian membership functions.

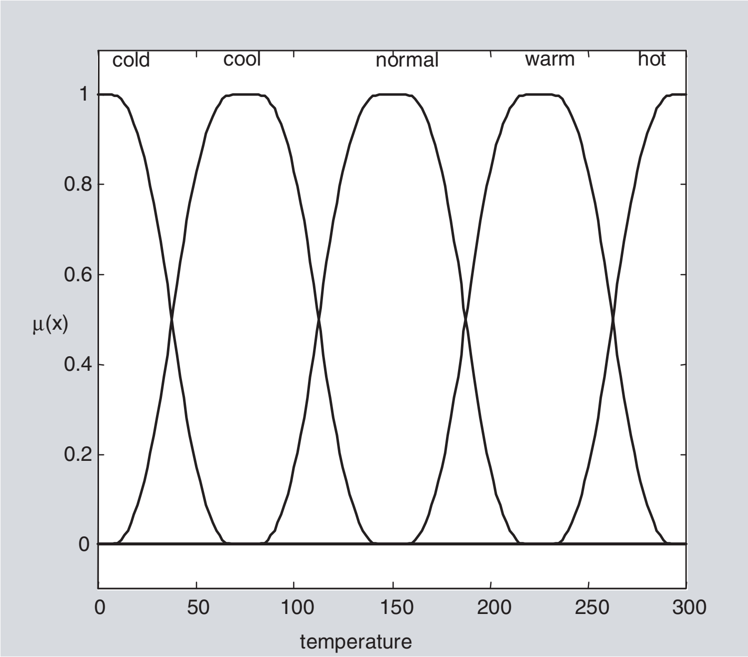

Figure 3.5 π-shaped membership functions.

3.3 LOGICAL OPERATIONS AND IF–THEN RULES

Fuzzy set operations are analogous to crisp set operations. The important thing in defining fuzzy set logical operators is that if we keep fuzzy values to the extremes 1 (True) or 0 (False), the standard logical operations should hold. In order to define fuzzy set logical operators, let us first consider crisp set operators. The most elementary crisp set operations are union, intersection, and complement, which essentially correspond to *OR, AND,* and *NOT* operators, respectively. Let A and B be two subsets of U. The union of A and B, denoted $A \cup B$, contains all elements in either A or B; that is, $\mu_{A\cup B}(x) = 1$ if $x \in A$ or $x \in B$. The intersection of A and B, denoted $A \cap B$, contains all the elements that are simultaneously in A and B; that is, $\mu_{A\cap B}(x) = 1$ if $x \in A$ and $x \in B$. The complement of A is denoted by $\bar{A}$, and it contains all elements that are not in A; that is $\mu_{\bar{A}}(x) = 1$ if $x \notin A$, and $\mu_{\bar{A}}(x) = 0$ if $x \in A$. The truth tables for these operators are shown in Figure 3.6.

In FL, the truth of any statement is a matter of degree. In order to define FL operators, we have to find the corresponding operators that preserve the results of using *AND, OR,* and *NOT* operators. The answer is *min, max,* and *complement* operations. These operators are defined, respectively, as

$$\begin{aligned} \mu_{A\cup B}(x) &= \max\left[\mu_A(x), \mu_B(x)\right] \\ \mu_{A\cap B}(x) &= \min\left[\mu_A(x), \mu_B(x)\right] \\ \mu_{\bar{\mathrm{A}}}(x) &= 1 - \mu_{\mathrm{A}}(x) \end{aligned} \tag{3.6}$$

The formulas for *AND, OR,* and *NOT* operators in Equation (3.6) are useful for proving other mathematical properties about sets; however, *min* and *max* are not the only ways to describe the intersection and union of two sets. Zadeh (1965) defined fuzzy union and fuzzy intersection as

$$\begin{aligned} \mu_{A\cup B}(x) &= \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x) \\ \mu_{A\cap B}(x) &= \mu_A(x)\mu_B(x) \end{aligned} \tag{3.7}$$

AND			OR			NOT	
A	B	A∪B	A	B	A∪B	A	$\bar{A}$
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Figure 3.6 Truth tables for *AND*, *OR*, and NOT operators.

In more general terms, the intersection of two fuzzy sets A and B is specified by a binary mapping T that aggregates two membership functions as

$$\mu_{A\cap B}(x) = T(\mu_A(x), \mu_B(x)) \tag{3.8}$$

For example, the binary operator T may represent the multiplication of $\mu_A(x), \mu_B(x)$. These fuzzy intersection operators are referred to as T-*norm* (triangular norm) operators, and they meet the following basic requirements:

$$\begin{aligned} \text{boundary:}\ & T(0,0)=0,\ T(a,1)=T(1,a)=a \\ \text{monotonicity:}\ & T(a,b)\le T(c,d) \text{ if } a\le c \text{ and } b\le d \\ \text{commutativity:}\ & T(a,b)=T(b,a) \\ \text{associativity:}\ & T(a,T(b,c))=T(T(a,b),c) \end{aligned} \tag{3.9}$$

The first requirement ensures the correct generalization of crisp sets. The second requirement implies that a decrease in the membership values in A and B cannot produce an increase in the membership value of the intersection of sets A and B. The third requirement specifies that the operation is insensitive to the order in which fuzzy sets are combined, and the fourth requirement enables us to take the intersection of any number of fuzzy sets and any order of pairwise groupings. Similar to fuzzy intersection, the fuzzy union operator is specified by the following binary mapping S:

$$\mu_{A\cup B} = S(\mu_A(x), \mu_B(x)) \tag{3.10}$$

These fuzzy union operators are known as T-*conorm* or S-*norm* operators, and they satisfy the following requirements:

$$\begin{aligned} \text{boundary:}\ & S(1,1)=1,\ S(a,0)=S(0,a)=a \\ \text{monotonicity:}\ & S(a,b)\le S(c,d) \text{ if } a\le c \text{ and } b\le d \\ \text{commutativity:}\ & S(a,b)=S(b,a) \\ \text{associativity:}\ & S(a,S(b,c))=S(S(a,b),c) \end{aligned} \tag{3.11}$$

Several *T*-norms and *S*-norms have been suggested in the literature (Yager, 1980; Dubois and Prade, 1980; Schweizer and Sklar, 1963, Sugeno, 1977). One example of a pair of *S*-norm and *T*-norm operators is the bounded sum and bounded product:

$$\begin{aligned} x \oplus y &= \min\left[1, x+y\right] \\ x \otimes y &= \max\left[0, x+y-1\right] \end{aligned} \tag{3.12}$$

Most applications use *min* for fuzzy intersection, *max* for fuzzy union, and $1-\mu_A\left(x\right)$ for complementation. We have to remember that operators used in FL, such as union, intersection, and complement, reduce to their crisp logic counterparts when the membership functions are restricted to 0 or 1.

Fuzzy inference systems consist of if–then rules that specify a relationship between the input and output fuzzy sets. Fuzzy relations present a degree of presence or absence of association or interaction between the elements of two or more sets. Let U and V be two universes of discourse. A fuzzy relation $R\left(U,V\right)$ is a set in the product space $U \times V$ and is characterized by the membership function $\mu_R\left(x,y\right)$, where $x \in U$ and $y \in V$, and $\mu_R\left(x,y\right) \in \left[0,1\right]$. Fuzzy relations play an important role in fuzzy inference systems. FL uses notions from crisp logic. Concepts in crisp logic can be extended to FL by replacing 0 or 1 values with fuzzy membership values. A singleton fuzzy rule assumes the form "if *x* is *A*, then *y* is *B*," where $x \in U$ and $y \in V$, and has a membership function, $\mu_{A\to B}\left(x,y\right)$, where $\mu_{A\to B}\left(x,y\right) \in \left[0,1\right]$. The if part of the rule, "*x* is *A*," is called the *antecedent* or *premise*, while the then part of the rule, "*y* is *B*," is called the *consequent* or *conclusion*. Interpreting an if–then rule involves two distinct steps. The first step is to evaluate the antecedent, which involves fuzzifying the input and applying any necessary fuzzy operators. The second step is implication, or applying the result of the antecedent to the consequent, which essentially evaluates the membership function $\mu_{A\to B}\left(x,y\right)$. It can be seen that in crisp logic a rule is fired if the premise is exactly the same as the antecedent of the rule, and the result of such rule firing is the rule's actual consequent. In fuzzy logic, a rule is fired so long as there is a nonzero degree of similarity between the premise and the antecedent of the rule. For most applications, the fuzzy membership function $\mu_{A\to B}\left(x,y\right)$ for a given relation is obtained with the minimum or product implication, given, respectively, as follows:

$$\mu_{A\cap B}\left(x\right) = \mu_A\left(x\right)\mu_B\left(x\right) \tag{3.13}$$

$$\mu_{A\cap B}\left(x\right) = \min\left[\mu_A\left(x\right), \mu_B\left(x\right)\right] \tag{3.14}$$

It was Mamdani (1977) who first proposed the minimum implication, and later Larsen (1980) proposed the product implication. The minimum and product inferences have nothing to do with traditional prepositional logic; hence, they are collectively referred to as engineering implications. Details of implication methods can be found in the classic tutorial paper by Mendel (1995).

3.4 FUZZY INFERENCE SYSTEM

A fuzzy inference system (FIS) essentially defines a nonlinear mapping of the input data vector into a scalar output, using fuzzy rules. The mapping process involves input/output membership functions, FL operators, fuzzy if–then rules, aggregation of output sets, and defuzzification. An FIS with multiple outputs can be considered as a collection of independent multiinput, single-output systems. A general model of a fuzzy inference system (FIS) is shown in Figure 3.7. The FLS maps crisp inputs into crisp outputs. It can be seen from the figure that the FIS contains four components: the fuzzifier, inference engine, rule base, and defuzzifier. The rule base contains linguistic rules that are provided by experts. It is also possible to extract rules from numeric data. Once the rules have been established, the FIS can be viewed as a system that maps an input vector to an output vector. The fuzzifier maps input numbers into corresponding fuzzy memberships. This is required in order to activate rules that are in terms of linguistic variables. The fuzzifier takes input values and determines the degree to which they belong to each of the fuzzy sets via membership functions. The inference engine defines mapping from input fuzzy sets into output fuzzy sets. It determines the degree to which the antecedent is satisfied for each rule. If the antecedent of a given rule has more than one clause, fuzzy operators are applied to obtain one number that represents the result of the antecedent for that rule. It is possible that one or more rules may fire at the same time. Outputs for all rules are then aggregated. During aggregation, fuzzy sets that represent the output of each rule are combined into a single fuzzy set. Fuzzy rules are fired in parallel, which is one of the important aspects of an FIS. In an FIS, the order in which rules are fired does not affect the output. The defuzzifier maps output fuzzy sets into a crisp number. Given a fuzzy set that encompasses a range of output values, the defuzzifier

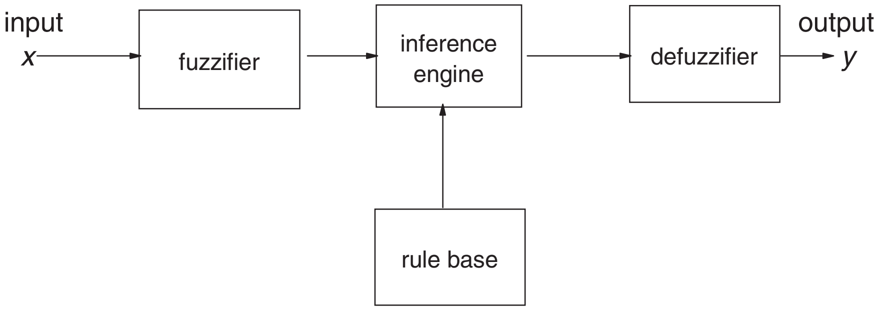

Figure 3.7 Block diagram of a fuzzy inference system.

returns one number, thereby moving from a fuzzy set to a crisp number. Several methods for defuzzification are used in practice, including the centroid, maximum, mean of maxima, height, and modified height defuzzifier. The most popular defuzzification method is the centroid, which calculates and returns the center of gravity of the aggregated fuzzy set. FISs employ rules. However, unlike rules in conventional expert systems, a fuzzy rule localizes a region of space along the function surface instead of isolating a point on the surface. For a given input, more than one rule may fire. Also, in an FIS, multiple regions are combined in the output space to produce a composite region. A general schematic of an FIS is shown in Figure 3.8.

Consider a multiinput, multioutput system. Let $\mathbf{x} = (x_1, x_2, . . ., x_n)^T$ be the input vector and $\mathbf{y} = (y_1, y_2, \ldots, y_m)^T$ be the output vector. The linguistic variable x_i in the universe of discourse U is characterized by $T(\mathbf{x}) = \{T_x^{\ 1}, T_x^{\ 2}, \ldots, T_x^{\ k}\}$ and $\mu(\mathbf{x}) = \{\mu_x^{\ 1}, \mu_x^{\ 2}, \ldots, \mu_x^{\ k}\}$ where $T(\mathbf{x})$ is a term set of **x**; that is, it is the set of names of linguistic values of **x**, with each $T_x^{\ i}$ being a fuzzy member and the membership function $\mu_x^{\ i}$ defined on$U$. As an illustration, we consider a fuzzy inference system with two inputs $(n = 2)$ and one output $(m = 1)$. Let the two inputs represent the number of years of education and the number of years of experience, and let the output of the system be *salary*. Let x_1 indicate the number of years of education, $T(x_1)$ represent its term set {*low, medium, high*}, and the universe of discourse be $[0-15]$. Let x_2 indicate the number of years of experience, the universe of discourse be $[0-30]$, and the corresponding term set be {*low, medium, high*}. Similarly, linguistic variable y in the universe of discourse V is characterized by $T(\mathbf{y}) = \{T_y^{\ 1}, T_y^{\ 2}, \ldots, T_y^{\ l}\}$, where $T(\mathbf{y})$ is a term set of **y**; that is, T *is* the set of names of linguistic values of **y**, with each $T_y^{\ i}$ being a fuzzy membership function $\mu_y^{\ i}$ defined on V. If the

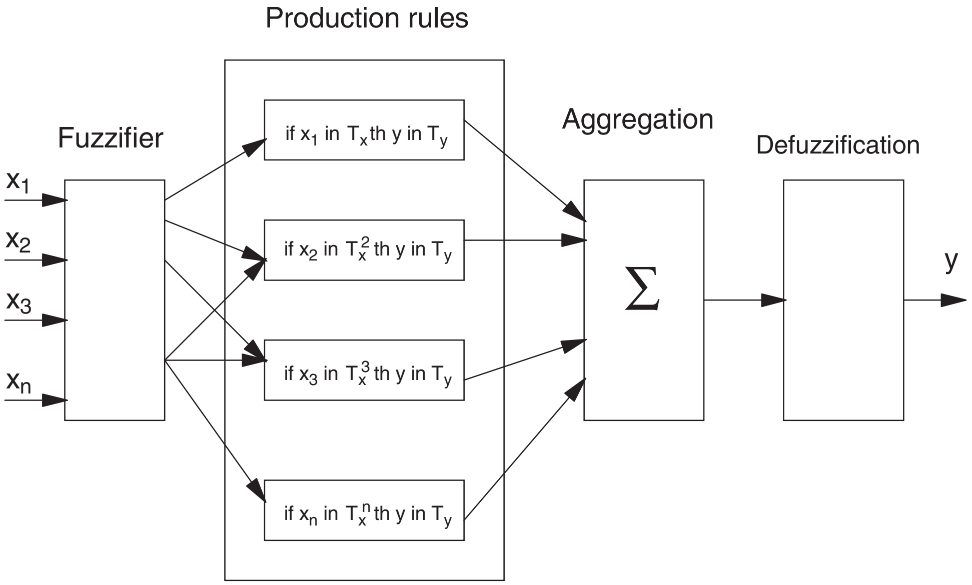

Figure 3.8 Schematic diagram of a fuzzy inference system.

variable y represents *salary*, then $T(y)$ represents a term set {*very low, low, medium, high, very high*}, and the universe of discourse is [20–200], which represents the minimum and maximum in thousands of dollars—that is, 20,000, and 200,000, respectively. In order to map input variables x_1 and x_2 to output y, it is necessary that we first define the corresponding fuzzy sets. The membership functions for the input and output variables are shown in Figure 3.9. The first step in evaluating the output of a FIS is to apply the inputs and determine the degree to which they belong to each of the fuzzy sets. The fuzzifier block performs the mapping from the input feature space to fuzzy sets in a certain universe of discourse. A specific value x_1 is then mapped to the fuzzy set $T_{x_i}{}^1$ with degree $\mu_{x_i}{}^1$, to fuzzy set $T_{x_i}{}^2$ with degree $\mu_{x_i}{}^2$, and so on. In order to perform this mapping, we can use fuzzy sets of any shape, such as triangular, Gaussian, π-shaped, etc.

A fuzzy rule base contains a set of fuzzy rules R. A single if–then rule assumes the form "if x is T_x then y is T_y." An example of a rule might be "if *education* is *high* and *experience* is *high*, then *salary* is *very high*." For a multiinput, multioutput system,

$$R = (R_1, R_2, \ldots, R_n) \tag{3.15}$$

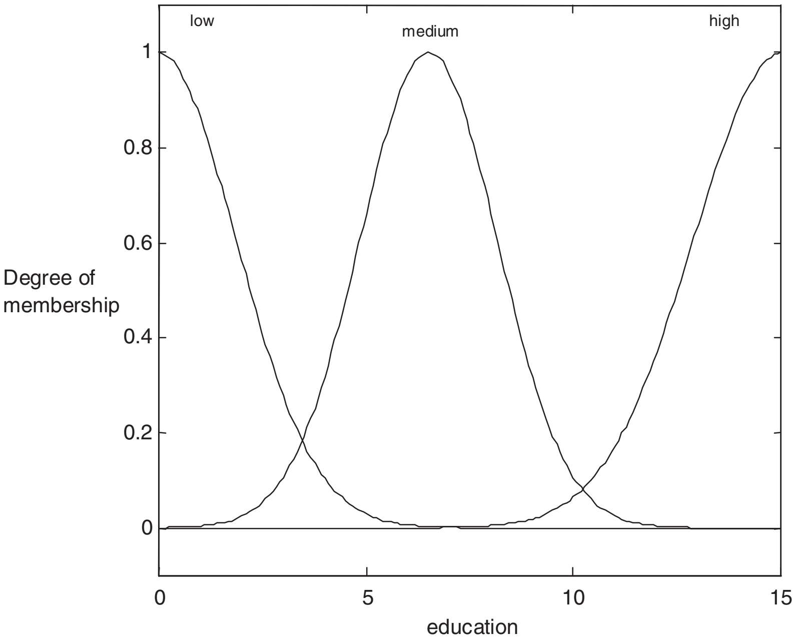

Figure 3.9a Fuzzy membership functions for input1.

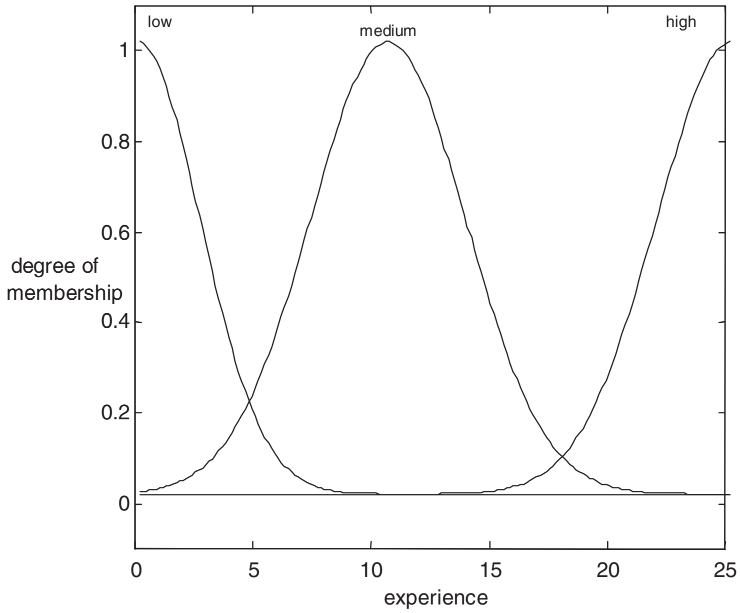

Figure 3.9b Fuzzy membership function for input2.

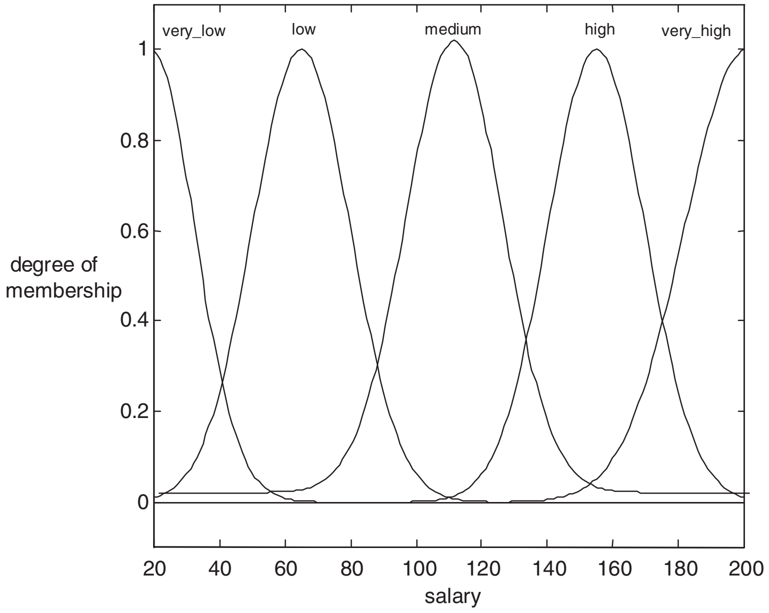

Figure 3.9c Fuzzy membership function for output1.

where the *i*th fuzzy rule is

$$R_i = \text{if}\left(x_1 \text{ is } T_{x_1}, \text{ and } \ldots, x_p \text{ is } T_{x_p}\right) \text{ then } \left(y_1 \text{ in } T_{y_1}, \text{ and } \ldots, Y_q \text{ in } T_{y_q}\right) \tag{3.16}$$

The p preconditions of R_i form a fuzzy set $T_{x_1} \times T_{x_2} \times \cdots \times T_{x_p}$, and the consequent is the union of q independent outputs. If we consider a multiinput, single-output system, then the consequent reduces to $\left(y_1 \text{ is } T_{y1}\right)$. For the given example, the rules are stated as

R_1: *if education is low and experience is low, then salary is very low*
R_2: *if education is low and experience is medium, then salary is low*
R_3: *if education is low and experience is high, then salary is medium*
R_4: *if education is medium and experience is low, then salary is low*
R_5: *if education is medium and experience is medium, then salary is medium*
R_6: *if education is medium and experience is high, then salary is high*
R_7: *if education is high and experience is low, then salary is medium*
R_8: *if education is high and experience is medium, then salary is high*
R_9: *if education is high and experience is high, then salary is very high*

Interpreting an if–then rule is a three part process: (a) Resolve all fuzzy statements in the antecedent to a degree of membership between 0 and 1; (b) if there are multiple parts to the antecedent, apply fuzzy logic operators and resolve the antecedent to a single number between 0 and 1, is the result being the degree of support for the rule; and (c) apply the implication method, using the degree of support for the entire rule to shape the output fuzzy set. If the rule has more than one antecedent, the fuzzy operator is applied to obtain one number that represents the result of applying that rule. For example, consider an *i*th rule

$$R_i\text{: if } x_1 \text{ is } T_{x_i} \text{ and } x_2 \text{ is } T_2^{\,i} \text{ then } y \text{ is } T_y^{\,i} \tag{3.17}$$

Then the firing strength or membership of the rule can be defined as

$$\alpha_i = \min\left(\mu_{x_1}{}^{i}\left(x_1\right), \mu_{x_2}{}^{i}\left(x_2\right)\right)$$

or (3.18)

$$\alpha_i = \mu_{x_1}{}^{i}\left(x_1\right)\mu_{x_2}{}^{i}\left(x_2\right)$$

Equation (3.18) represents fuzzy intersection with the minimum or product operators. Each fuzzy rule yields a single number that represents the firing strength of that rule. The firing strength is then used to shape the output fuzzy set that represents the consequent part of the rule. The implication method is defined as the shaping of the consequent (the output fuzzy set), based on the antecedent. The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. Two methods are commonly used: the minimum and the product methods, represented, respectively, by Equations (3.19) and (3.20), respectively.

$$\mu_y^{\,i}(w)' = \min\left(\alpha_i, \mu_y^{\,i}(w)\right) \tag{3.19}$$

$$\mu_y^{\,i}(w)' = \alpha_i \mu_y^{\,i}(w) \tag{3.20}$$

where *w* is the variable that represents the support value of the membership function. For the given example, if we assume that *education* equals 10 years and *experience* equals 18.6 years, then it can be seen in Figure 3.10 that rules R_5 and R_6 fire.

After we obtain firing strengths of the rules, we need to combine the corresponding output fuzzy sets into one composite fuzzy set. The process of combining output fuzzy sets into a single set is called *aggregation*, a processthat unifies the outputs of all the rules. Essentially, aggregation takes all fuzzy sets that represent the output for each rule and combines them into a single fuzzy set that is used as the input to the defuzzification process. Aggregation occurs only once for each output variable. The inputs to the aggregation process are truncated or modified output fuzzy sets obtained as the output of the implication process. The output of the aggregation process is a single fuzzy set that represents the output variable. Since the aggregation method is commutative, the order in which the rules are executed is not important. The commonly used aggregation method is the *max* method. If we have two rules with output fuzzy sets represented by two fuzzy sets $\mu_y^{i}(w)$ and $\mu_y^{2}(w)$, then, combining the two sets, we obtain the output decision

$$\mu_y(w) = \max\left(\mu_y^{\,1}(w)\mu_y^{\,2}(w)\right) \tag{3.21}$$

Notice that the last result is a membership curve. The output of aggregation of fuzzy sets in our example is shown in Figure 3.10. In order to get a crisp value for output *y,* we need a defuzzification process. The input to the defuzzification process is a fuzzy set (the aggregate output fuzzy set), and the output of the defuzzification process is a single crisp number. The most commonly used defuzzification method is the centroid calculation. Methods of defuzzification are discussed in the next section.

The fuzzy inference process defines the mapping surface $y = f\left(x_1, x_2\right)$, which is illustrated in Figure 3.11. The inference process can be described completely in the five steps shown in Figure 3.12.

Step 1: Fuzzy Inputs

The first step is to take inputs and determine the degree to which they belong to each of the appropriate fuzzy sets via membership functions.

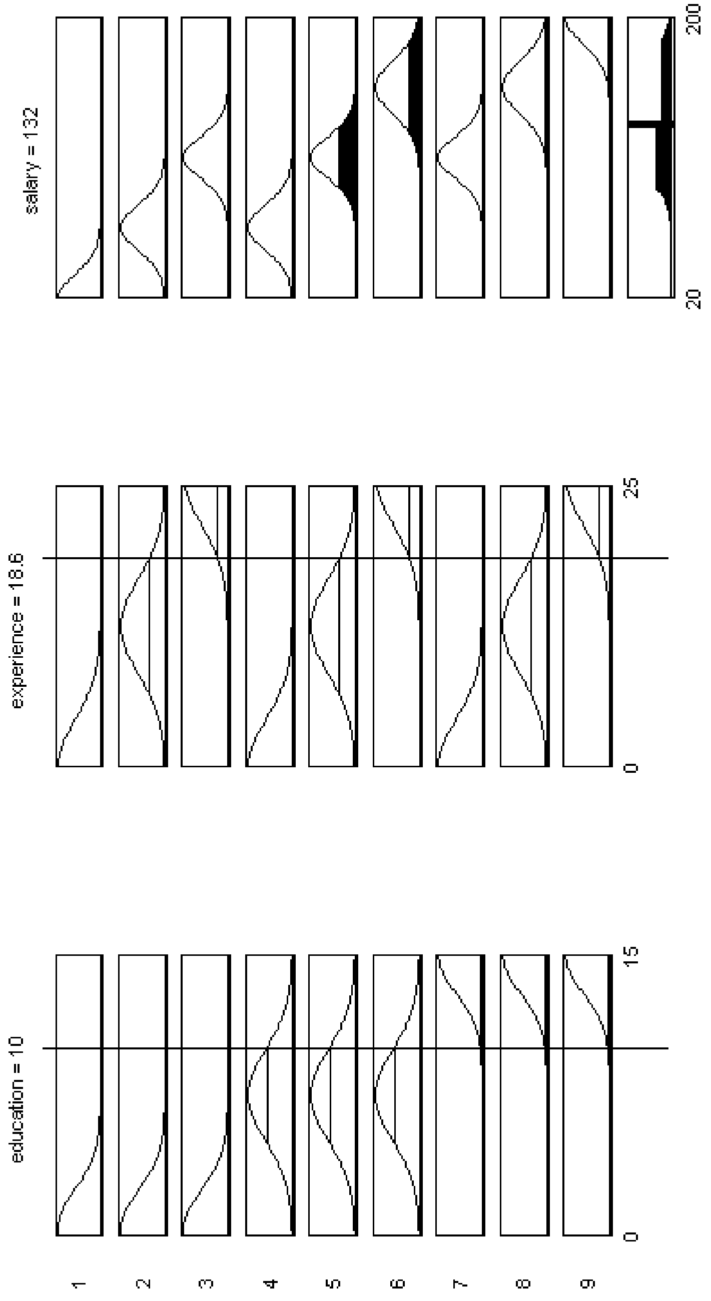

Figure 3.10 Fuzzy rules.

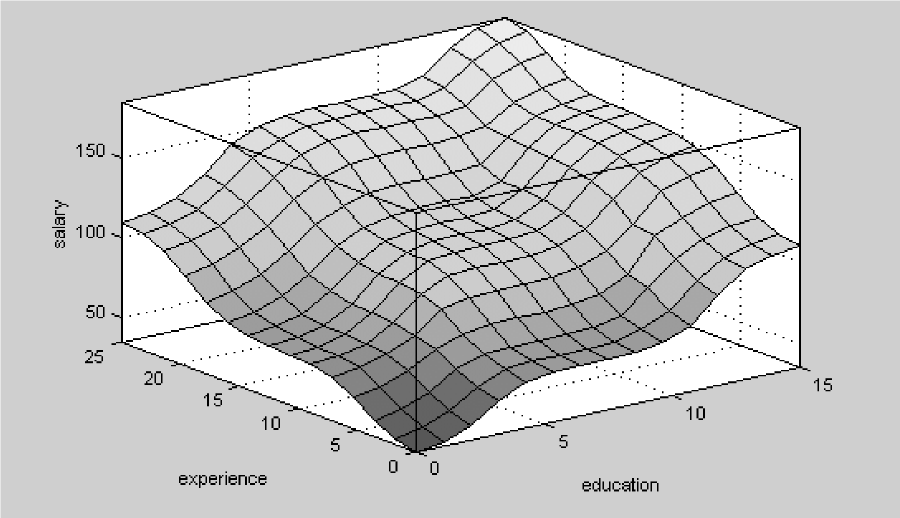

Figure 3.11 Mapping surface.

Step 2: Apply Fuzzy Operators

Once the inputs have been fuzzified, we know the degree to which each part of the antecedent has been satisfied for each rule. If a given rule has more than one part, the fuzzy logical operators are applied to evaluate the composite firing strength of the rule.

Step 3: Apply the Implication Method

The implication method is defined as the shaping of the output membership functions on the basis of the firing strength of the rule. The input for the implication process is a single number given by the antecedent, and the output is a fuzzy set. Two commonly used methods of implication are the minimum and the product.

Step 4: Aggregate all Outputs

Aggregation is a process whereby the outputs of each rule are unified. Aggregation occurs only once for each output variable. The input to the aggregation process is the truncated output fuzzy sets returned by the implication process for each rule. The output of the aggregation process is the combined output fuzzy set.

Step 5: Defuzzify

The input for the defuzzification process is a fuzzy set (the aggregated output fuzzy set), and the output of the defuzzification process is a crisp value obtained by using some defuzzification method such as the centroid, height, or maximum. As an example, we consider a system that determines dinner in a restaurant on the basis of the service received. We consider input membership functions with different degrees of overlap. Here, the input x denotes the quality of the

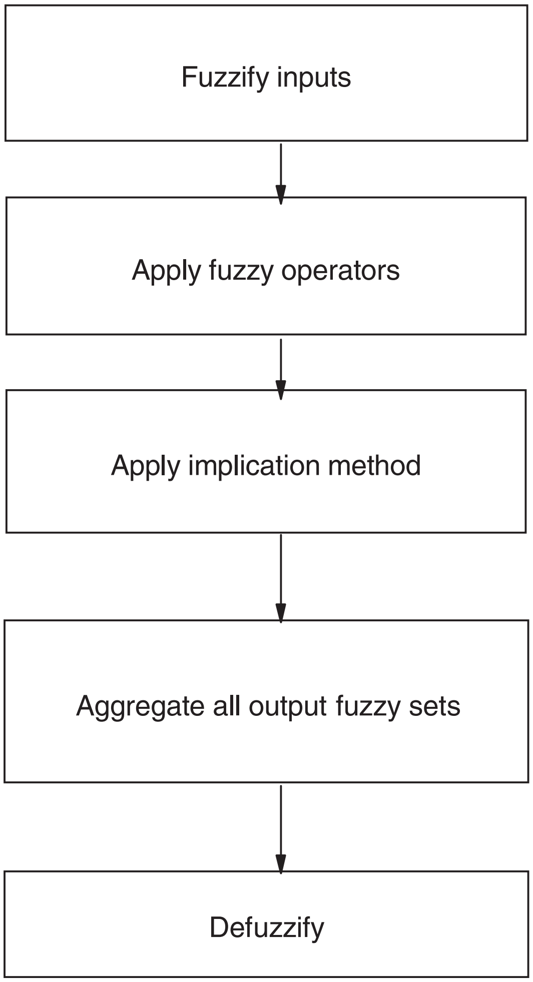

Figure 3.12 Fuzzy inference process.

service, which is represented by a number between 0 and 20, where 20 designates *very good* and 0 *very poor*. The input x is represented by the term set {*very poor, poor, average, good, very good*}. The output y represents the *tip*, which varies between 5 and 30 percent, and is given by the term set {*very cheap, cheap, average, generous, very generous*}. The input–output fuzzy sets for this example are shown in Figures 3.13 and 3.14, respectively. We have assumed the following five rules that define mapping $y = f(x)$:

R_1: *if service is very poor, then tip is very cheap*
R_2: *if service is poor, then tip is cheap*
R_3: *if service is average, then tip is average*
R_4: *if service is good, then tip is generous*
R_5: *if service is very good, then tip is very generous*

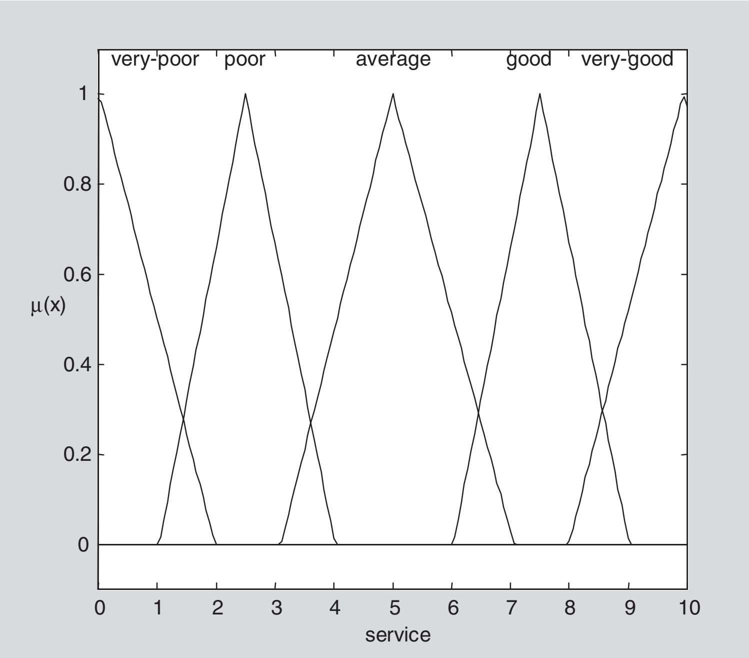

Figure 3.13 Input fuzzy sets.

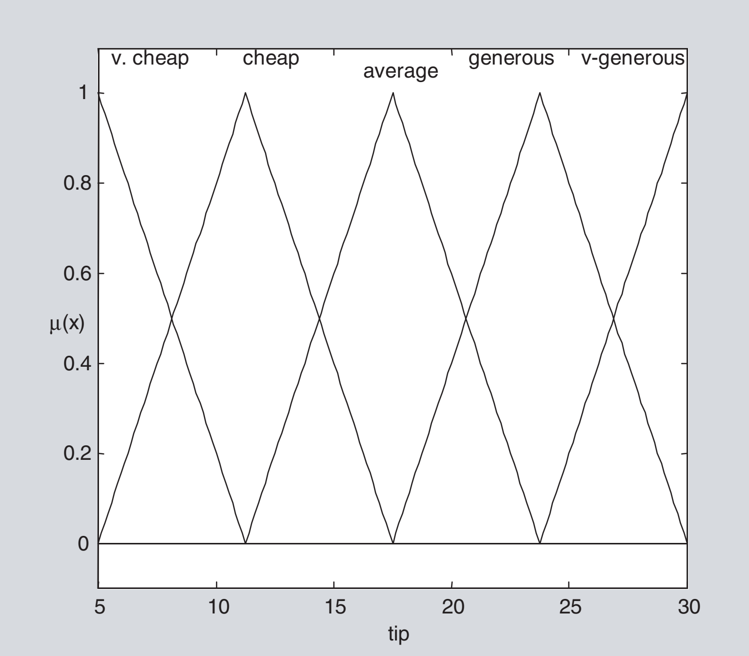

Figure 3.14 Output fuzzy sets.

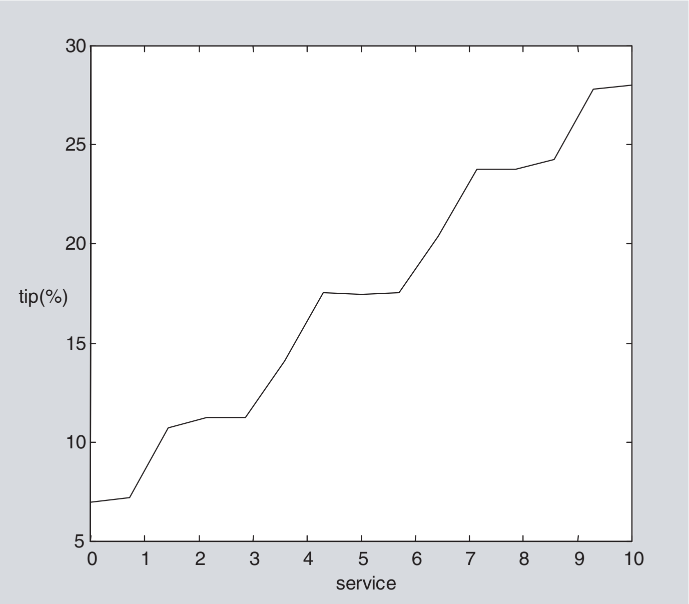

Figure 3.15 Mapping surface.

We have implemented the FIS with these rules. The mapping function for the system is shown in Figure 3.15. In order to see the effect of overlap between input fuzzy sets on the mapping functions, we used another set of fuzzy input–output membership functions, illustrated in Figures 3.16 and 3.17, respectively. The corresponding mapping function is shown in Figure 3.18. The mapping function $y = f\left(x\right)$ depends on the number of input fuzzy sets and their shapes. It can be seen from the preceding example that, as we increase the overlap between the input fuzzy sets, the mapping function becomes smoother.

3.5 DEFUZZIFICATION

A fuzzy inference system maps an input vector to a crisp output value. In order to obtain a crisp output, we need a defuzzification process. The input to the defuzzification process is a fuzzy set (the aggregated output fuzzy set), and the output of the defuzzification process is a single number. Many defuzzification techniques have been proposed in the literature. The most commonly used method is the centroid. Other methods include the maximum, the means of maxima, height, and modified height method. The five methods may be described as follows:

(a) Centroid defuzzification method: In this method, the defuzzifier determines the center of gravity (centroid) $y_i\,'$ of B and uses that value as the output of the FLS. For a continuous aggregated fuzzy set, the centroid is given by

$$y' = \frac{\int\limits_s y_i \mu_B\left(y\right)dy}{\int\limits_s \mu_B\left(y\right)dy} \tag{3.22}$$

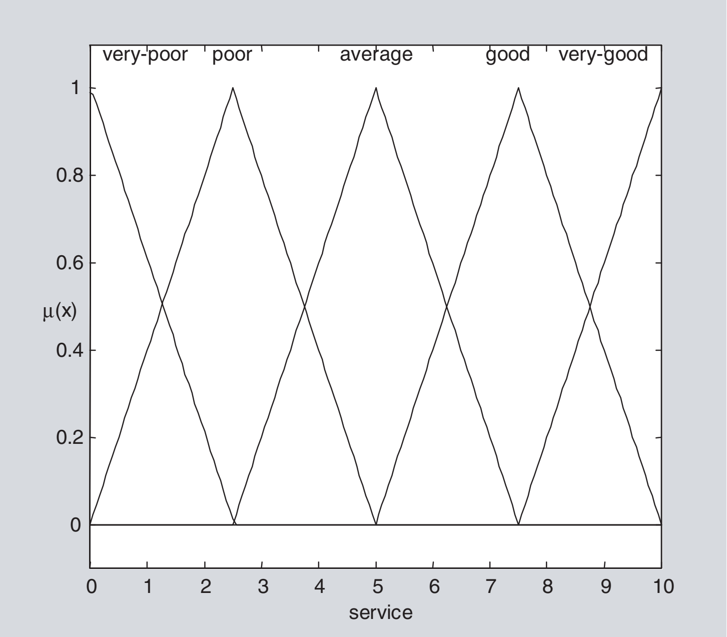

Figure 3.16 Input fuzzy sets.

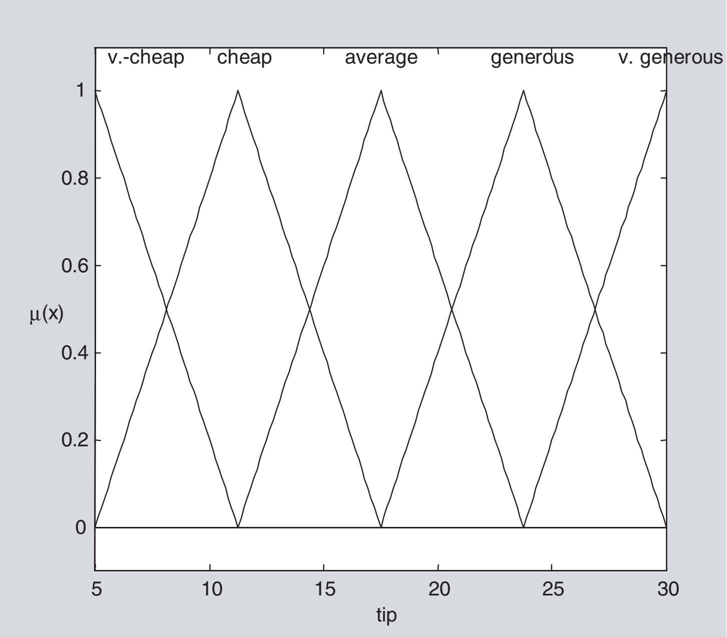

Figure 3.17 Output fuzzy sets.

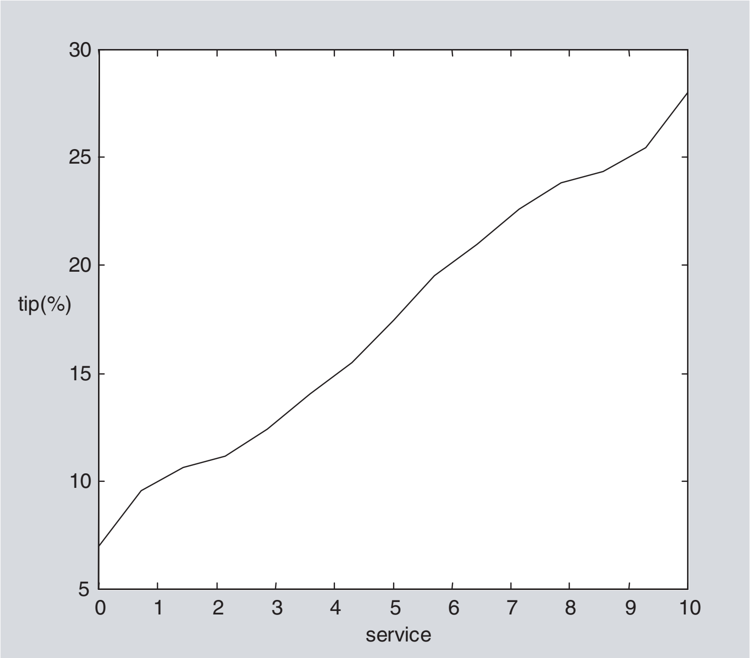

Figure 3.18 Mapping surface.

where S denotes the support of $\mu_B(y)$. Often, discretized variables are used so that y' can be approximated as shown in Equation (3.23), which uses summations instead of integration.

$$y' = \frac{\sum_{i=1}^{n} y_i \mu_B(y_i)}{\sum_{i=1}^{n} \mu_B(y_i)} \tag{3.23}$$

The centroid defuzzification method finds the “balance” point of the solution fuzzy region by calculating the weighted mean of the output fuzzy region. It is the most widely used technique because, when it is used, the defuzzified values tend to move smoothly around the output fuzzy region. The technique is unique, however, and not easy to implement computationally. The method of centroid defuzzification is depicted in Figure 3.19.

(b) Maximum-decomposition method: In this method, the defuzzifier examines the aggregated fuzzy set and chooses that output y for which $\mu_B(y)$ is the maximum, as shown in Figure 3.20. Unlike the centroid method, the maximum-decomposition

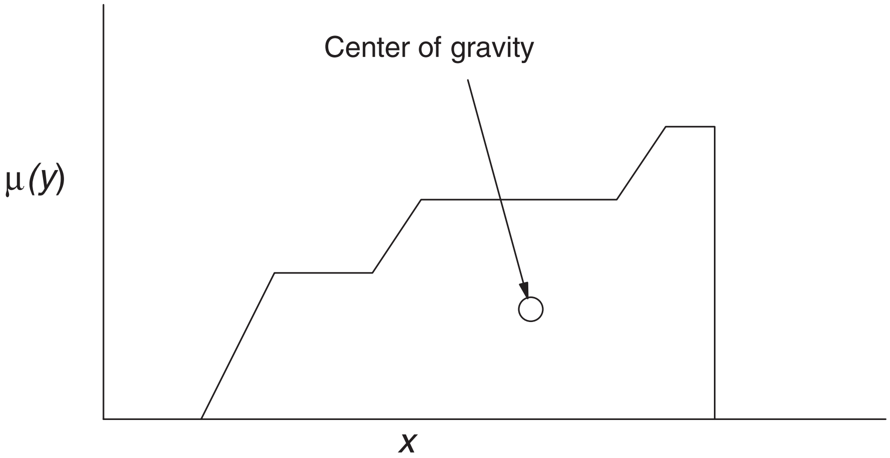

Figure 3.19 Centroid defuzzification method.

method has some properties that are applicable to a narrower class of problems. The output value for this method is sensitive to a single rule that dominates the fuzzy rule set. Also, the output value tends to jump from one frame to the next as the shape of the fuzzy region changes.

(c) Center of maxima: In a multimode fuzzy region, the center-of-maxima technique finds the highest plateau and then the next highest plateau. The midpoint between the centers of these plateaus is selected as shown in Figure 3.21.

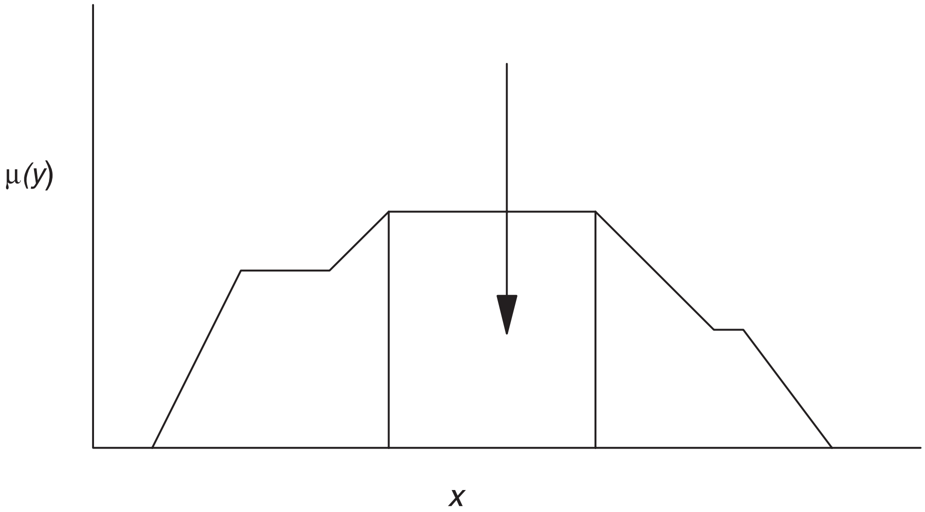

Figure 3.20 Defuzzification (maximum-decomposition method).

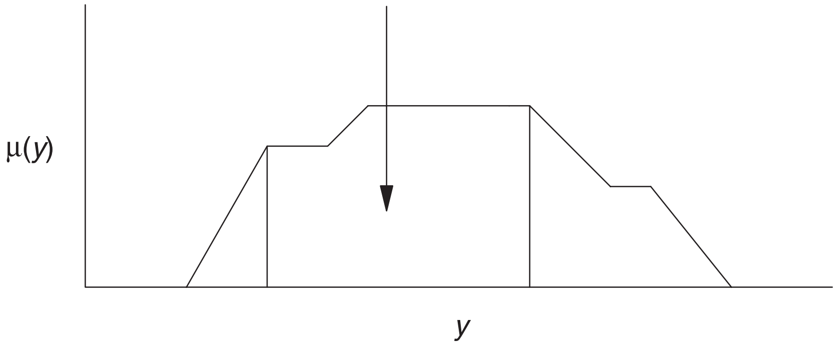

Figure 3.21 Defuzzification (average of maximums).

(d) Height defuzzification: In this method, the defuzzifier first evaluates $\mu_{B_i}(y)$ at y'_1 and then computes the output of the FLS, where denotes the center of gravity of fuzzy sets B_i . Output y_h in this case is given by

$$y_h = \frac{\sum_{i=1}^{m} y_i{}' \mu_B(y_i)}{\sum_{i=1}^{m} \mu_B(y_i)} \tag{3.24}$$

where m represents the number of output fuzzy sets obtained after implication and y_i' represents the centroid of fuzzy region i. This technique is easy to use because the centers of gravity of commonly used membership functions are known ahead of time. Regardless of whether minimum or product inference is used, the fuzzy inference process essentially defines the mapping of the given vector of crisp values to an output crisp value using fuzzy rules stored in the knowledge base.

The fuzzy inference process just discussed is known as *Mamdani's fuzzy inference method.* Sugeno (1977) suggested a fuzzy inference method that is similar to Mamdani's. In Sugeno's method, the first two parts, namely, mapping inputs to fuzzy membership functions and applying fuzzy operators, are the same as in Mamdani's method. The main difference between the two is in the evaluation of the output membership functions. In Sugeno's method, the output membership function is a constant or a linear function. A fuzzy rule for the zero-order Sugeno method is of the form if x is A and y is B then $C = K$, where A and B are fuzzy sets in the antecedent and K is a constant. The first-order Sugeno model has rules of the form if x is A and y is B then $C = px + qy + r$, where A and B are fuzzy sets in the antecedent and $p, q,$ and r are constants.

3.6 FUZZY SET REPRESENTATION WITH A CUBE

Kosko (1997) provided a geometric representation of fuzzy sets. The geometry of fuzzy sets involves both the domain $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, and the range of mapping $\mu_A : x \rightarrow [0,1]$. The geometry of fuzzy sets aids us in describing fuzziness and defining fuzzy concepts. Kosko represents the fuzzy power set $F(2^x)$, the set of all fuzzy subsets of **x**, by a cube. A point in the cube represents a fuzzy set. The set of all fuzzy subsets equals the unit hypercube $I^n = [0,1]^n$. Vertices of the cube, I^n, define crisp sets. A two-dimensional hypercube is shown in Figure 3.22. We may view the fuzzy subset $P(0.6, 0.3)$ as a point in the unit hypercube. With this formulation, we define fuzzy set intersection by the pairwise minimum, union by the pairwise maximum, and complement by a reversal of order, as follows:

$$\begin{aligned}
\mu_{A\cap B} &= \min(\mu_A, \mu_B) \\
\mu_{A\cup B} &= \max(\mu_A, \mu_B) \\
{\mu_A}^C &= 1 - \mu_A
\end{aligned} \tag{3.25}$$

Consider two fuzzy sets, $A = (0.7 \quad 0.1 \quad 1.0 \quad 0.8)$ and $B = (0.9 \quad 0.6 \quad 0.2 \quad 0.2)$ in a four-dimensional cube. The following are various intersections, unions, and complements of these sets:

$$\begin{aligned}
A \cap B &= (0.7 \quad 0.1 \quad 0.2 \quad 0.2) \\
A \cup B &= (0.9 \quad 0.6 \quad 1.0 \quad 0.8) \\
A^C &= (0.3 \quad 0.9 \quad 0.0 \quad 0.2) \\
A \cap A^C &= (0.3 \quad 0.1 \quad 0.0 \quad 0.2) \\
A \cup A^C &= (0.7 \quad 0.9 \quad 1.0 \quad 0.8)
\end{aligned} \tag{3.26}$$

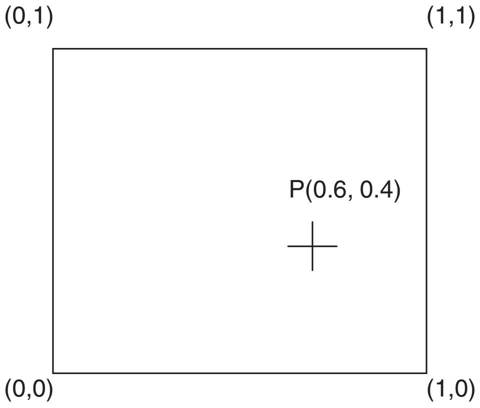

Figure 3.22 Unit cube representing a fuzzy set.

The overlap-fit vector $A \cap A^C$ in this example is not equal to all zeros, and the overlap-fit vector $A \cup A^C$ is not equal to the vector of all ones. A distance between two fuzzy sets is often required to calculate fuzziness or entropy. The distance of order l between two points *A* and *B* can be defined as

$$d^l\left(A;B\right) = \left(\sum_{i=1}^{n}\left|\mu_A\left(x_i\right) - \mu_B\left(x_i\right)\right|^l\right)^{\frac{1}{l}} \tag{3.27}$$

where *n* represents the dimension of the unit cube, *l* is the order of the distance measure, x_i represents *i*th variable, $\mu_A\left(x_i\right)$ represents membership values for fuzzy set *A*, and $\mu_B\left(x_i\right)$ denotes membership values for fuzzy set *B*. For *l* = 1, Equation (3.27) represents the Hamming distance, whereas for *l* = 2, it represents the Euclidean distance. This leads us to a simple definition of the size $S\left(A\right)$ of set *A* as the sum of its components or the fuzzy Hamming distance between the origin and the point in the hypercube representing the set:

$$S\left(A\right) = \sum_{i=1}^{n} A_i \tag{3.28}$$

The position of the fuzzy set in the unit hypercube determines the set's fuzziness. Since entropy is a measure of uncertainty, the term *fuzzy entropy* is defined to quantify fuzziness. The fuzzy entropy of fuzzy set *A*, $E\left(A\right)$, varies from 0 to 1 on the unit hypercube. The vertices of the cube have zero entropy, while the midpoint has the maximum, or unity, entropy. Kosko (1997) defined entropy as the ratio of the distance between the point defining the fuzzy subset and the nearest vertex to the distance between the point and the farthest vertex:

$$E\left(A\right) = \frac{d^l\left(A, A_{\text{near}}\right)}{d^l\left(A, A_{\text{far}}\right)} \tag{3.29}$$

The fuzzy entropy theorem states that the entropy can be written as

$$E\left(A\right) = \frac{S\left(A \cap A^C\right)}{S\left(A \cup A^C\right)} \tag{3.30}$$

Kosko (1986, 1987) provided a geometric proof of the fuzzy entropy theorem. With this representation, the fuzzy inference process can be represented as the mapping of a fuzzy set from one cube to another.

3.7 HEDGES

A *linguistic hedge* or modifier is an operation that modifies the meaning of a term or a fuzzy set. For example, if *hot* is a fuzzy set, then *very hot, more or less hot*, and *extremely hot* are examples of hedges that are applied to that fuzzy set. Hedges can be viewed as operators that act upon a fuzzy set's membership function to modify it. Hedges play the same role in fuzzy production rules that adjectives and adverbs play in English sentences. There are hedges that intensify the characteristics of a fuzzy set (*very, extremely*), that dilute the membership curve (*somewhat, rather, quite*), that form the complement (*not*), and that approximate a scalar to a fuzzy set (*about, close to, approximately*). The mechanics underlying the hedge operation is generally heuristic in nature. For example, $\mu_A^{1.3}(x)$ is used frequently to implement the hedge *slightly.* Zadeh's original definition of the hedge *very* intensifies the fuzzy region by squaring the membership function at each point in the set $\mu_A^{2}(x)$. On the other hand, the hedge *somewhat* dilutes the fuzzy region by taking the square root of the membership function at each point along the set $\mu_A^{0.5}(x)$. A generalization of the concentrator hedge is

$$\mu_{\mathrm{con}(A)}(x) = \mu_A^{\,n}(x) \tag{3.31}$$

where $n \geq 1$. This hedge simply replaces the exponent of the intensification function with a real positive number greater than unity. Figure 3.23 shows the hedge that uses the general concentrator form with $n = 3$. The complement of *very* is a hedge group represented by *somewhat, rather,* and *quite*. These hedges basically dilute the force of a fuzzy set membership function. A gener-

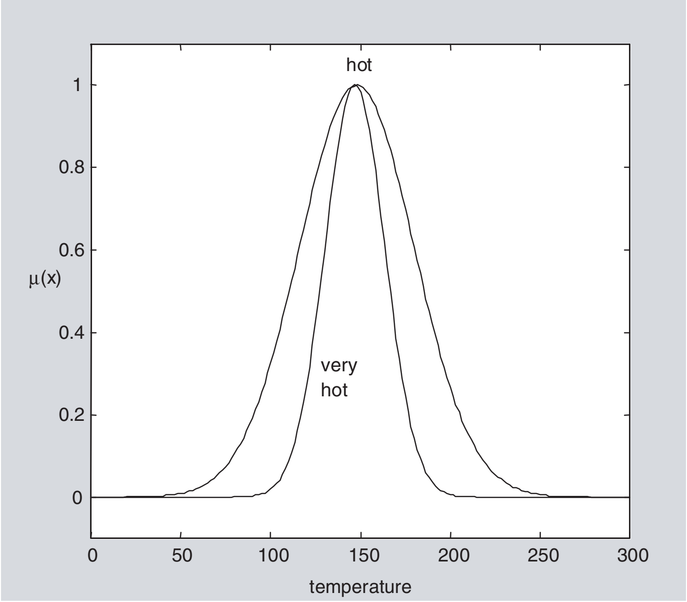

Figure 3.23 Concentrator hedge.

alization of the dilator hedge simply replaces the exponent of the intensification function with a real positive number less than unity, expressed as a fraction $(1/n)$. The generalized dilator edge is defined as

$$\mu_{dil(A)}(x) = \mu_{A^{\frac{1}{n}}}(x) \tag{3.32}$$

where $n \geq 1$. Generalized dilator hedges are shown in Figure 3.24. The contrast hedges change the nature of fuzzy regions by making the region either less fuzzy (intensification) or more fuzzy (diffusion). Hedges such as *positively, absolutely,* and *definitely* are contrast hedges, changing a fuzzy set by raising the truth values above (0.5) and decreasing all the truth values below (0.5), thus reducing the overall fuzziness of the region (Figure 3.25). These hedges are represented by

$$\mu_{inf}(A) = \begin{cases} 2\left(\mu_A^{\ 2}(A)\right) & \text{if } \mu_A(A) \geq 0.5 \\ 1-2\left(\mu_A^{\ n}(A)\right) & \text{otherwise} \end{cases} \tag{3.33}$$

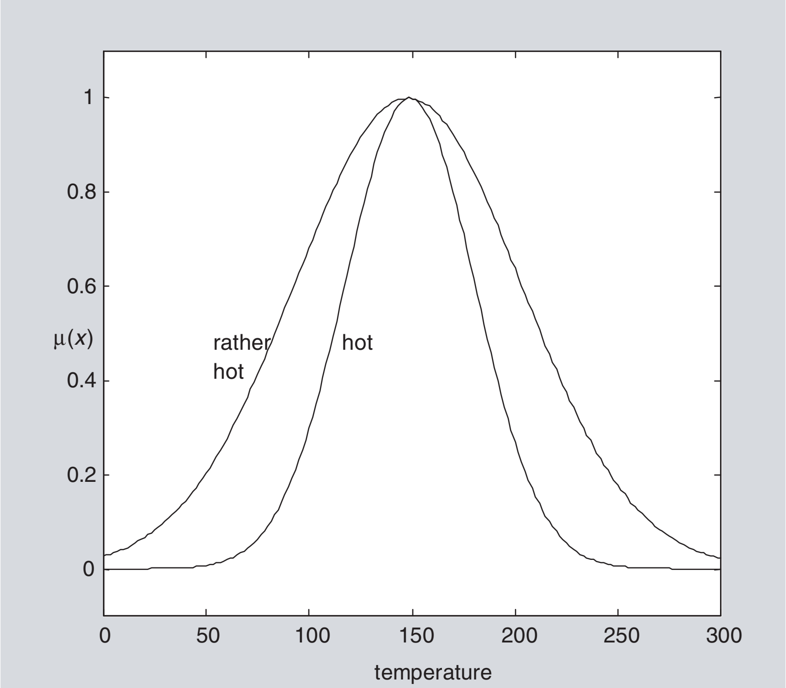

Figure 3.24 Dilator hedge.

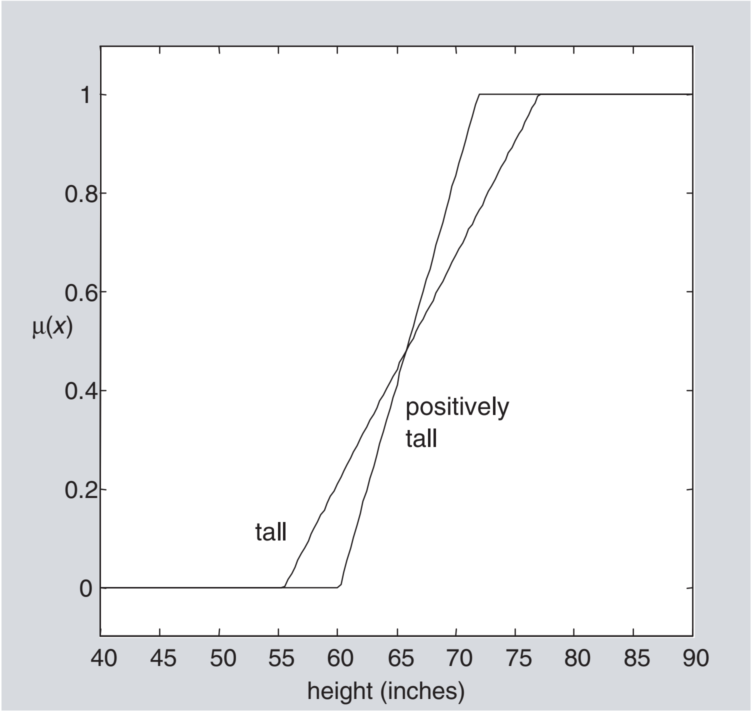

Figure 3.25 Intensification hedge.

Similarly, a hedge such as *generally* changes the fuzzy surface by reducing all truth values above (0.5) and increasing all truth values below (0.5), as shown in Figure 3.26. These hedges are represented by

$$\mu_{def}(A) = \begin{cases} \dfrac{1}{2}\left(\mu_A^{\frac{1}{2}}(A)\right) & \text{if } \mu_A(A) \geq 0.5 \\ 1 - \dfrac{1}{2}\left(\mu_A^{\frac{1}{2}}(A)\right) & \text{otherwise} \end{cases} \tag{3.34}$$

Since a hedge is linguistic in nature, multiple hedges can be applied to a single fuzzy region. The approximation hedges are an important class of transformers. They not only broaden or restrict existing bell-shaped fuzzy regions, but also convert scalar values into bell-shaped fuzzy regions. The most often used approximate hedge is the *about* hedge, which creates a space that is proportional to the height and width of the generated fuzzy space.

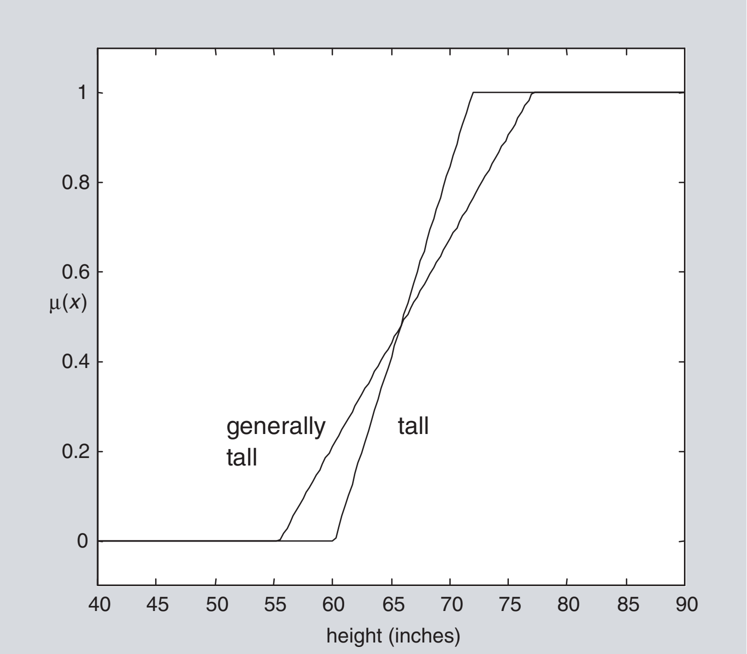

Figure 3.26 Diffusion hedge.

3.8 FUZZY SYSTEMS AS FUNCTION APPROXIMATORS

A fuzzy system can be used to approximate a function. Kosko (1997) described a class of additive fuzzy systems. An additive fuzzy system approximates a functioon by covering its graph with fuzzy patches. The approximation improves as the fuzzy patches grow in number. Additive fuzzy systems have a feed-forward architecture that resembles the feed-forward multilayer neural systems used to approximate functions. Additive fuzzy systems are different from conventional fuzzy inference systems. Additive fuzzy systems add the then parts of fired if-then rules, whereas conventional fuzzy inference systems combine the then part with pairwise maxima. The fuzzy mapping function for the adaptive fuzzy system $F : X \rightarrow Y$ that approximates a function $f : x \rightarrow y$ is shown in Figures 3.27 and 3.28. Fuzzy patches approximate the function. The approximation improves with the number of patches. However, the computational cost also increases with the number of patches. A schematic diagram for an additive fuzzy system is presented in Figure 3.29. A Cartesian product space for the system is shown in Figure 3.30. Rules for the additive system are of the form "If X is A, then Y is B." Additive fuzzy systems fire all rules in parallel and average the scaled then part sets. The class of additive fuzzy systems represents a large number of additive systems. The most commonly used model for an additive fuzzy

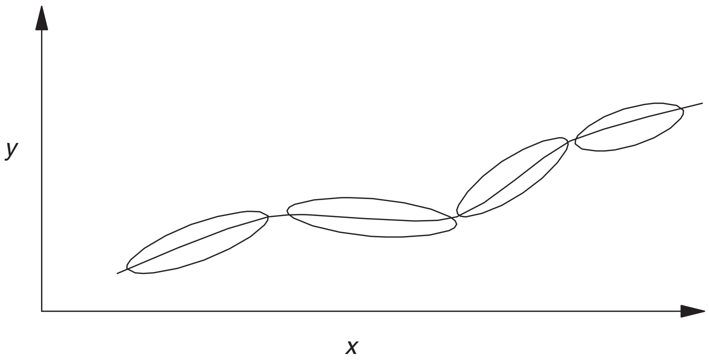

Figure 3.27 Function approximation with four fuzzy patches.

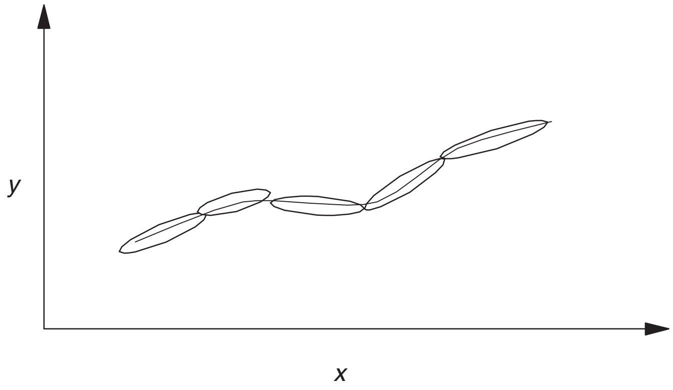

Figure 3.28 Function approximation with five fuzzy patches.

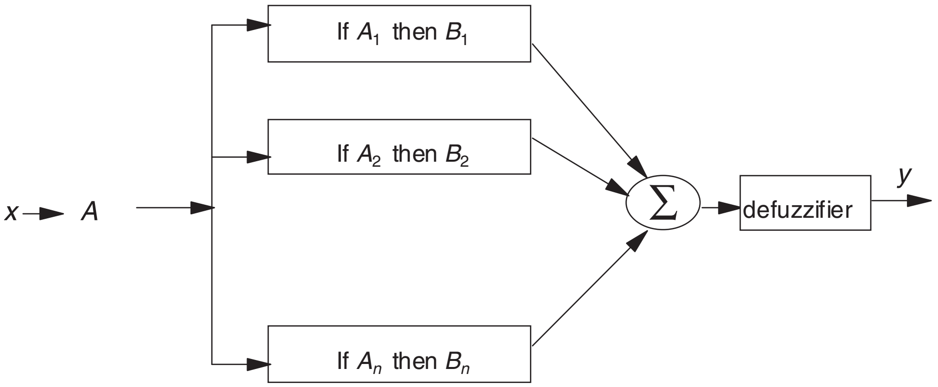

Figure 3.29 Additive fuzzy interference system.

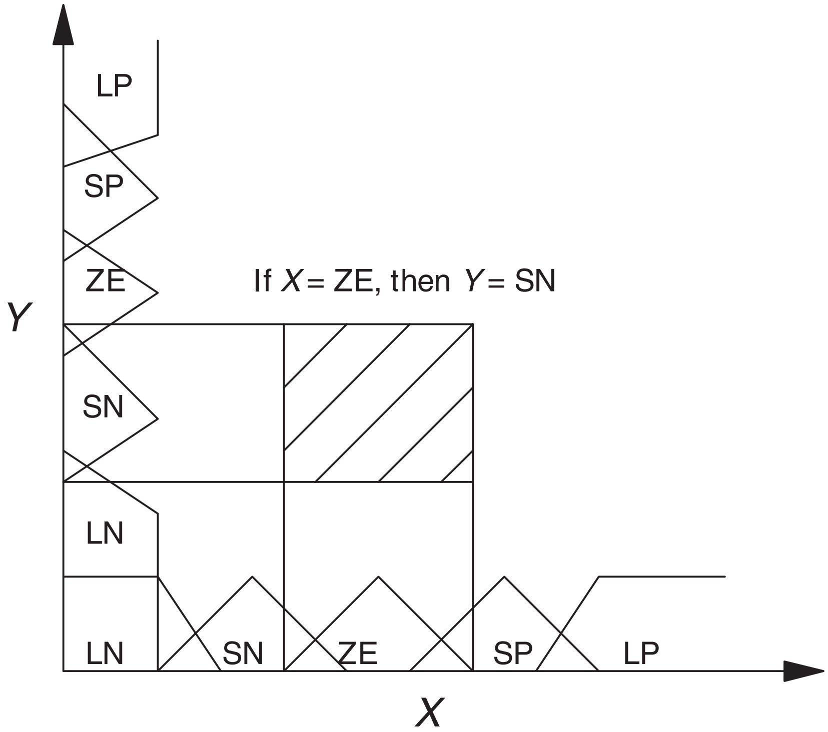

Figure 3.30 Fuzzy rule as a state patch.

system is the standard additive model (SAM), which defines a function given by Kosko (1992) , viz.,

$$\begin{aligned} y = F(x) &= \frac{\int y\, b(y)\, dy}{\int b(y)\, dy} \\ &= \frac{\sum_{j=1}^{m} w_j\, V_j\, a_j(x)\, c_{y_j}}{\sum_{j=1}^{m} w_j\, V_j\, a_j(x)} \end{aligned} \tag{3.35}$$

where V_j is the volume of the jth then-part set B_j and w_j is the weight of the jth rule (often, $w_j = 1$). The term c_y is the centroid of the jth output set. Fit valu $a_i(x)$ scales the then-part set B_j, and m is the number of rules. The SAM equation departs from the linguistic context of earlier fuzzy models. The complexity of a SAM system depends on the complexity of the if-part fuzzy sets A_j and the dimensionality of the problem. Simple sets such as trapezoids and bell curves lead to efficient approximation.

3.9 EXTRACTION OF RULES FROM SAMPLE DATA POINTS

Earlier, we saw that in order to design a FIS, we need a rule base that contains fuzzy rules. Usually, these rules are obtained from expert knowledge. However, on many occasions, we may not know rules, but we may have sample data points or training samples in the input/output spaces. In situations like this, it is possible to generate fuzzy rules that define the mapping surface. The generated rules then can be used to design an FIS that performs the desired mapping. Wang and Mendel (1991) suggested a systematic method for extracting fuzzy rules from sample data points. The method consists of five steps. They have also shown that the mapping surface can approximate any real continuous function on a compact set to a desired degree of accuracy. In this method, it is possible to combine information of two kinds: numeric and linguistic.

Consider a function $y = f(x_1, x_2)$. We can design an FIS with two inputs and one output to approximate the function, using the following steps:

Step 1. Divide the input/output space into fuzzy regions: Divide each domain interval into $2N + 1$ regions. Let the regions be denoted as S_N (small N), . . . , S_1 (small 1), CE (central), B_1 (big 1), . . . , B_N (big N). The number of regions can be different for each variable. Assign each region a fuzzy membership function. Fuzzy membership functions for x_1, x_2, and y are shown in Figure 3.31a–c.

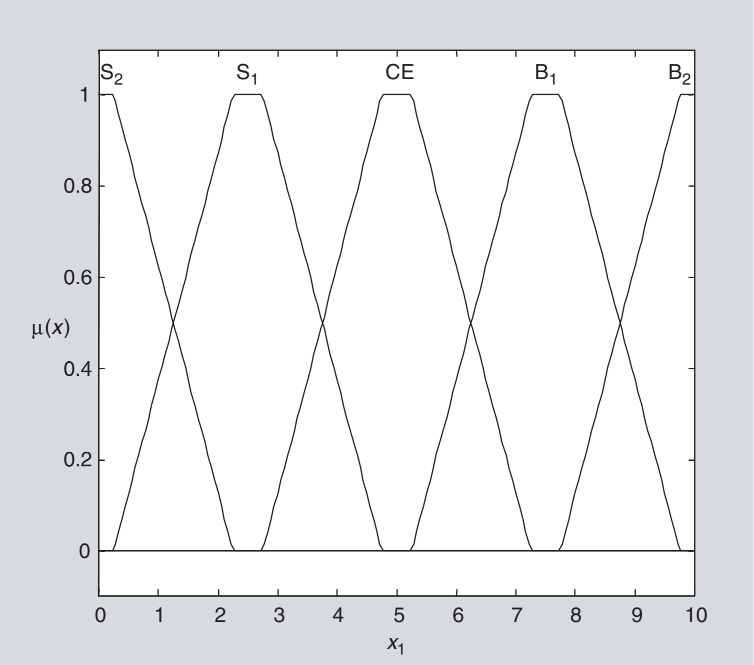

Figure 3.31a Fuzzy membership function.

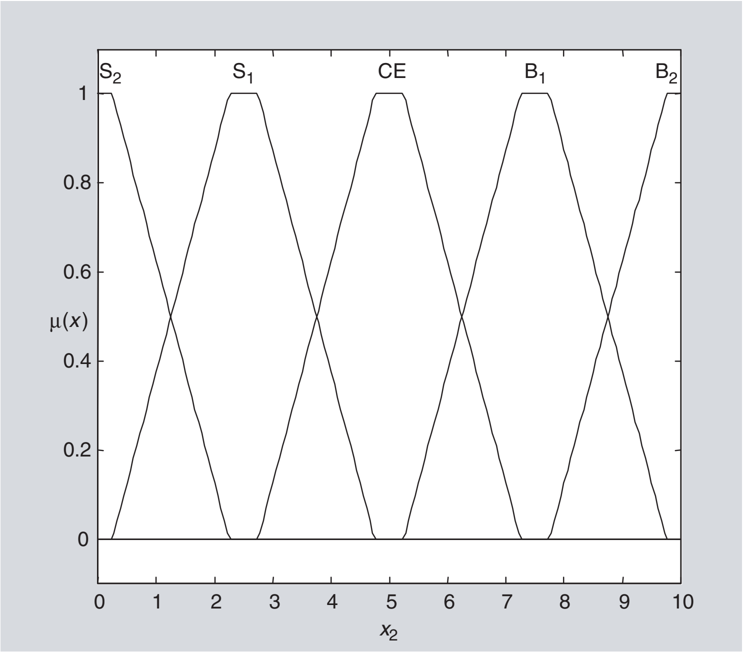

Figure 3.31b Fuzzy membership function.

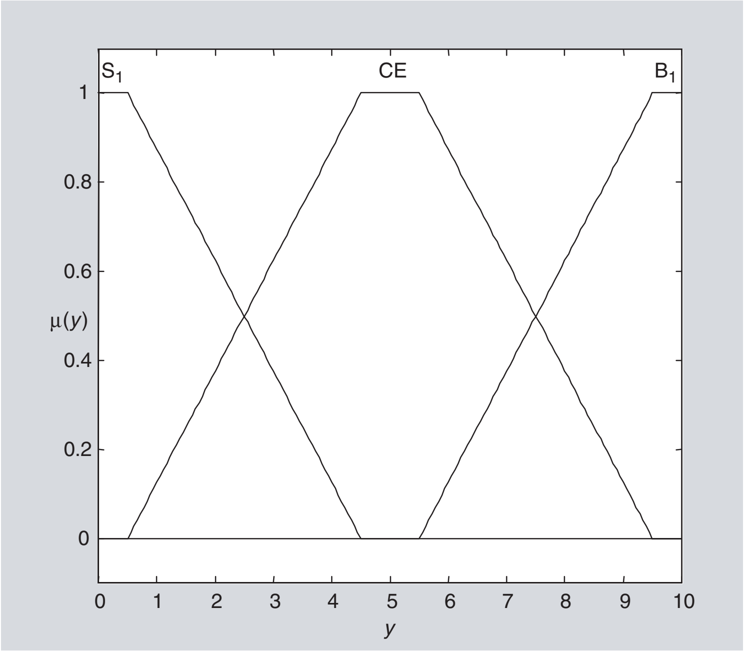

Figure 3.31c Fuzzy membership function.

Step 2. Generate fuzzy rules from given data points: First determine the degree of the given data point $x_1(i), x_2(i),$ and $y(i)$ for each membership function, and assign the point to the region with the maximum degree of membership. Then obtain a rule from the given data point. The rule may look like the following:

R_l: if x_1 is B_1 and x_2 is CE, then y is B_1

Step 3. Assign a degree to each rule: There are many data points, each generating a rule. Therefore, some rules may conflict. The degree of a rule can be defined as follows: For rule R_i, viz., if x_1 is A and x_2 is B, then y is C, the degree of the rule is defined as

$$D(R_i) = \mu_A(x_1)\mu_B(x_2)\mu_c(y) \tag{3.36}$$

where $\mu_A(x_1), \mu_B(x_2),$ and $\mu_C(y)$ represent membership values in fuzzy sets A, B, and C, respectively. In practice, we may have a priori information about data points. We therefore assign a degree to each data point that represents expert belief in the rule. The degree of the rule then can be written as

$$D(R_i) = \mu_A(x_1)\mu_B(x_2)\mu_c(y)\mu_i \tag{3.37}$$

Step 4. Create a combined FAM bank, shown in Figure 3.32. Cells in the FAM bank are to be filled with fuzzy rules, by assignment. If there is more than one rule in any cell, then the rule with the maximum degree is used. If there is a rule with an *OR* operator, it fills all the cells in that row or column.

Step 5. Determine a mapping based on the combined FAM bank. We used the following defuzzification to determine output y for given inputs (x_1, x_2): The output membership region o_1 is given by

$$\mu_{io} = \mu_{1i}(x_1)\mu_{2i}(x_2) \tag{3.38}$$

The centroid defuzzification method can be used to obtain the crisp output *y.*

We can view the preceding five-step procedure as a block. Input to the block consists of examples and expert rules, and the output of the block is a mapping from input to output space. Wang and Mendel (1991) used the five-step process for a chaotic time-series prediction problem. A chaotic time series which is sufficiently complicated that it appears to be “random” can be

	B_2					
	B_1					
X_2	CE					
	S_1					
	S_2					
		S_2	S_1	CE	B_1	B_2
				x_1		

Figure 3.32 The form of a FAM bank.

generated from a deterministic nonlinear system. The time series Wang and Mendel used is generated from the differential equation

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{3.39}$$

where $t > 17$. Those two researchers generated and plotted the time series with 1,000 points, using the first 700 points to train the system and the last 300 points to test the system. They approximated the time series using both an FLS and a neural network and compared the results. They showed that the approximation could be greatly improved by dividing the domain into finer intervals. An FLS can approximate any arbitrary continuous function to any desired degree of accuracy. The ability to approximate an arbitrary function is an important property that can be used in many image-processing applications. Given discrete data points, we can generate a function passing through those points. The function can then be used in applications such image interpolation or predictive image coding.

3.10 FUZZY BASIS FUNCTIONS

From the previous section, we know that a FIS is a nonlinear system that maps a crisp input vector **x** to a scalar *y*. The mapping can be represented as $y = f(\mathbf{x})$. Discussions in the previous sections offer a geometric interpretation of a FIS. Kim and Mendel (1995) provided the mathematical formula that defines the mapping function for a FIS. Their expansion for the FIS mapping function is

$$y = f\left(x\right) = \frac{\sum_{l=1}^{M} y'_1 \prod_{i=1}^{p} \mu_i{}'\left(x_i\right)}{\sum_{l=1}^{M} \prod_{i=1}^{p} \mu_i{}'\left(x_i\right)} \tag{3.40}$$

where M denotes the number of rules, y_i' represents the center of gravity of the output fuzzy set that is associated with the rule R_i, p is the dimension of vector $\mathbf{x}$, and $\mu_i{}'\left(x_i\right)$ represents input membership functions. This expansion is valid only when we choose singleton fuzzification functions, product inference, maximum-product composition, and height defuzzification (Mendel, 1995). Equation (3.40) can be rewritten as

$$y = f\left(x\right) = \sum_{l=1}^{M} y' \phi_i\left(x\right) \tag{3.41}$$

were the $\phi_i\left(x\right)$ are called *fuzzy basis functions* (FBFs) and are given by

$$\phi_i\left(x\right) = \frac{\prod_{i=1}^{p} \mu_i{}'\left(x_i\right)}{\sum_{l=1}^{m} \prod_{i=1}^{p} \mu_i{}'\left(x_i\right)} \tag{3.42}$$

Equation (3.42) is valid for singleton fuzzification. The representation in Equation (3.40) is referred to as the *fuzzy basis function expansion*. Relationships between the fuzzy function and other basis functions (such as trigonometric functions) were studied extensively by Kim and Mendel (1993). It can be seen that the FBF expansion is essentially a sum over M rules, each of which generates an FBF. The rules in the FBF expansion can be obtained from numeric data as well as from expert' knowledge. It is therefore convenient to decom $f\left(x\right)$ as

$$y = f\left(x\right) = f_N\left(x\right) + f_L\left(x\right) \tag{3.43}$$

Equation (3.43) can be rewritten as

$$y = f\left(x\right) = \sum_{i=1}^{Mn} y'_{Ni}\, \phi_{Ni}\left(x\right) + \sum_{k=1}^{M_l} y' \phi_{L_k}\left(x\right) \tag{3.44}$$

where $M = M_N + M_L$ and M_N FBFs are associated with numeric data, and M_L fuzzy basis functions are associated with linguistic information. These FBFs are given by

$$\phi_{Ni}(x) = \frac{\prod_{s=1}^{p} \mu_s{}'(x_s)}{\sum_{j=1}^{M_N} \prod_{s=1}^{p} \mu_s{}'(x_s)} \tag{3.45}$$

$$\phi_{L_k}(x) = \frac{\prod_{s=1}^{p} \mu_s{}'(x_s)}{\sum_{j=1}^{M_L} \prod_{s=1}^{p} \mu_s{}'(x_s)} \tag{3.46}$$

It can be seen from Equations (3.45) and (3.46) that each FBF is normalized by information that is associated with both numeric and linguistic information.

Because the decomposition of an FBF depends on M, where $M = M_N + M_L$, a mapping function associated with an FLS can be expressed as a summation over FBFs. Unlike other classical basis functions (e.g., trigonometric functions), which are inherently orthogonal, FBFs are not orthogonal. They are, however, important and unique, because of the fact that they are the only basis functionsthat can include numeric, as well as linguistic, information.

3.11 DESIGN AND IMPLEMENTATION OF A FUZZY INFERENCE SYSTEM

FISs map vectors from input space to output space. A number of other methods, including neural networks, mathematical functions, and conventional control systems, can perform a similar mapping. For any given problem, if a simpler solution is already available, we must try that first. Alternative methods may be used instead of an FLS. However, among the many advantages FISs have are that they are flexible, they have the ability to model any nonlinear function to any arbitrary degree of accuracy, they are based on rules that can be specified with a natural language, they can be built from expert knowledge, and they are tolerant to imprecise data. Fuzzy logic techniques can be used to complement other techniques, such as neural networks or genetic algorithms. In order to design an FIS, we must first decide inputs, outputs, each of their domains, and fuzzy inference rules. Mapping rules can be obtained from numeric data or from expert knowledge. We also need to decide input and output membership functions, overlap between these functions, implication and aggregation methods, and the defuzzification method. With the choice of these parameters, we can design a variety of FISs. In order to implement an FIS, we can select any method and algorithm and any programming environment. Today, many software and hardware tools are available for designing and implementing fuzzy logic systems. Most software tools provide extensive debugging and optimization features, as well as a graphical user interface (GUI) environment that makes fuzzy logic system design simple and easy. One such package is the MATLAB fuzzy logic toolbox, which relies heavily on GUI tools that provide an environment for FIS design, analysis, and implementation. Examples shown in earlier sections may be worked out with the use of this toolbox. MATLAB provides five primary GUI tools for building, editing, and observing an FIS.

3.12 SUMMARY

The chapter deals with fuzzy logic fundamentals. Zadeh (1965) introduced the term fuzzy logic and described the mathematics of fuzzy set theory. Fuzzy set allows partial memberships. Fuzzy sets represent linguistic labels or term sets such as slow, fast, low, medium, high, etc. Fuzzy membership functions represent term sets. Commonly used membership functions are triangular, trapezoidal, bell shaped, and Gaussian curves. In fuzzy logic, the truth of any statement is a matter of degree. In fuzzy logic, operators such as AND, OR, and NOT are implemented by intersection, union, and complement operators. There are various ways to define these operators. Commonly, AND, OR, and NOT operators are implemented by the *min, max,* and *complement* operators. A fuzzy inference system (FIS) maps crisp inputs to crisp outputs. An FIS consists of four components: the fuzzifier, inference engine, rule base, and defuzzifier. The fuzzifier maps input numbers into corresponding fuzzy membership values. The inference engine defines mapping from input fuzzy sets to output fuzzy sets. It determines the degree to which the antecedent part is satisfied for each rule. If the antecedent part of the rule has more than one clause, fuzzy operators are applied to obtain a number that represents the result of the antecedent part for that rule. Outputs of all rules are then aggregated. The defuzzifier maps the output fuzzy sets into a crisp number. The commonly used defuzzification method is the centriod method. The chaper also provides an illustrative example for a FIS.

Kosko (1997) has provided geometric representation of fuzzy sets. He represents fuzzy sets with a unit hypercube in a multidimensional space. The corners of the cube represent crisp sets. The position of a fuzzy set in the unit hypercube determines its fuzziness. Kosko (1997) defines fuzzy entropy of a fuzzy set. Kosko has also shown that a fuzzy system can be used for function approximation, and that an additive fuzzy system approximates a function by covering its graph with fuzzy patches. The approximation improves as the fuzzy patches grow in number.

The chapter also describes hedges. A linguistic hedge is an operation that modifies the meaning of the term or fuzzy set. Hedges play the same role in fuzzy production rules that adjectives and adverbs play in English sentences. There are hedges that intensify characteristics of a fuzzy set (*very, extremely*), dilute fuzzy sets (*somewhat, rather*) or form a complement (*not*). Zadeh's original definition of the hedge very intensifies the fuzzy region by squaring the membership function at each point in the set, that is $\mu_A^2(x)$. On the other hand the hedge somewhat dilutes the fuzzy region and it is given b $\mu_A^{0.5}(x)$. A few other hedges are described in the chapter.

In designing a FIS we need fuzzy rules. Often, these rules are obtained from experts knowledge. However, it is also possible to generate fuzzy rules from sample data points. Wang and Mendel (1991) have suggested a systematic method for extracting fuzzy rules from sample data points. The method consists of five steps. The chapter basically describes fundamentals of fuzzy logic, fuzzy membership functions, how to design an FIS, and how to extract fuzzy rules form sample data points.

REFERENCES

Bezdek, J. C., 1992. Computing with uncertainty. *IEEE Communications Magazine*, vol. 30, pp. 24–36.

Bezdek, J. C., 1993. Fuzzy models—what are they and why—editorial. *IEEE Transactions on Fuzzy Systems,* vol .1, pp. 1–5.

Cox, E., 1994. *The Fuzzy Systems Handbook*. Cambridge, MA: Academic Press.

Dubois, D. and Prade, H., 1980. *Fuzzy Sets and Systems: Theory and Applications*. New York: Academic Press.

Giarratano., J.and Riley, G., 1993. *Expert systems: Principles and Programming*. Boston, MA: PWS-Kent.

Kaufmann, A. and Gupta, M. M., 1985. *Introduction to Fuzzy Arithmetic*. New York: Van Nostrand.

Kim, Y. M. and Mendel, J. M., 1995. Fuzzy basis functions: comparison with other basis functions. *USC SIPI Report,* no. 299.

Kosko, B., 1991. *Neural Networks and Fuzzy Systems*. Englewood Cliffs, NJ: Prentice Hall.

Kosko, B., 1992. Fuzzy systems as universal approximators. *Proceedings of the IEEE International Conference on Fuzzy Systems.* San Diego, CA, pp. 1151–1162.

Kosko, B., 1994. Fuzzy systems as universal approximators. *IEEE Transactions on Computers*, vol. 43, no. 11, pp. 1329–1333.

Kosko, B., 1997. *Fuzzy Engineering*. Upper Saddle River, NJ: Prentice Hall.

Larsen, P. M., 1980. Industrial applications of fuzzy logic control. *International Journal of Man-Machine Studies*, vol. 12, no. 1, pp. 3–10.

Mamdani, E. H., 1977. Applications of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, vol. 26, no. 12, pp. 1182–1191.

Mendel, J. M., 1995. Fuzzy logic systems for engineering: A tutorial. *Proceedings of the IEEE*, vol. 83, no. 3, pp. 345–377.

Schweizer, B. Sklar, A., 1963. Associative functions and abstract semi-groups. Publ. Math. Debrecen, vol. 10, pp 69−81.

Sugeno, M., 1977. Fuzzy measures and fuzzy integrals: Asurvey. In *Fuzzy Automata and Decision Processes*, M. M. Gupta, et al. eds., New York: North-Holland.

Wang, Li-Xin and Mendel, J. M., 1991. Generating fuzzy rules by learning from examples. *Proceedings of the IEEE International Symposium on Intelligent Control*, Arlington, VA, pp. 263–268.

Yager, R., 1980. On a general class of fuzzy connectives. *International Journal of Fuzzy Sets and Systems*, vol 4, pp 235−242.

Zadeh, L. A., 1965. Fuzzy sets. *Information and Control*, vol, 8, pp, 338–353.

Zadeh, L. A., 1973. Outline of a new approach to analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, pp. 28–44.

Zadeh, L. A., 1978. PRUF—A meaning representation language for natural languages. *International Journal of Man-Machine Studies,* vol. 10, pp. 395–460.

Zadeh, L. A., 1994. Fuzzy logic, neural networks and soft computing. *Communications of the ACM*, vol. 37, pp. 77–84.

EXERCISES

3.1 Design a Sugeno-style FIS tipper with two inputs and one output. Let the first input be food and the second input be service.The output is the tip. Let the universe of discourse for both inputs be 1–10 and for the output 5–30. Assume the following three rules:

If service is poor or food is rancid, then tip is cheap

If service is good, then tip is average

If service is excellent or food is delicious, then tip is generous

Assume that the input variable*food* is represented by the term set {*rancid, average, delicious*} and the input variable *service* is represented by the term set {*poor, good, excellent*}. You may use triangular or trapezoidal membership functions. If *service* and *food* are rated as 4 and 9, respectively, what will be the *tip*, with your system?

3.2 Design a fuzzy inference system that can predict *profit*. Inputs to the system are *cost*, *sales*, and *market share*. Assume the following:

	Variable	**Minimum**	**Maximum**	**Term set**	**Remark**
1	*sales*	$0	$3,000,000	*LN, SN, ZR, SP, LP*	Input1
2	*cost per unit*	$0	$200	*LN, SN, ZR, SP, LP*	Input2
3	*market share*	0%	50%	*LOW, MEDIUM, HIGH*	Input3
4	*profit*	0%	$500,000	*LN, SN, ZR, SP, LP*	Output

Assume the following rules:

(a) If *sales* are *SN*, then *profit* is *SN*

(b) If *market share* is *HIGH*, then *profit* is *SP*

(c) If *cost* is *ZR*, then *profit* is *SP*

(d) If *cost* is *SN* and *market share* is *HIGH*, then *profit* is *LP*

Use trapezoidal membership functions and show the mapping surface. What will be the *profit* if *sales* are $2,100,000, *cost* per unit is $75, and the market share is 33%?

3.3 Repeat Exercise 2, using Gaussian membership functions. Also, add the following three rules to the rule base and plot the mapping surface:

(a) If *sales* are *LP*, then *profit* is *SP*

(b) If *sales* are *LP* and *market share* is *HIGH*, then *profit* is *LP*

(c) If *market share* is *LOW* and *sales* are *LN*, then *profit* is *LN*

3.4 Find the *union* and *intersection* of the following fuzzy sets:

$$\mu_A(x) = \begin{cases} 0 & \text{for } x \le 2 \\ \dfrac{x}{2} + 1 & \text{for } -2 \le x \le 0 \\ \dfrac{-x}{6} + 1 & \text{for } 0 \le x \le 6 \\ 0 & \text{for } x \ge 6 \end{cases} \tag{3.47}$$

$$\mu_B(x) = \begin{cases} 0 & \text{for } x \le -4 \\ \dfrac{x}{7} + \dfrac{4}{7} & \text{for } -4 \le x \le 3 \\ \dfrac{-x}{2} + \dfrac{5}{2} & \text{for } 4 \le x \le 5 \\ 0 & \text{for } x \ge 5 \end{cases} \tag{3.48}$$

3.5 Consider the following fuzzy sets in a five-dimensional unit cube:

$$\begin{aligned} X &= (1.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0) \\ O &= (0.0 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0) \\ A &= (0.2 \quad 0.6 \quad 0.7 \quad 0.9 \quad 0.0) \\ B &= (0.3 \quad 0.5 \quad 0.2 \quad 0.8 \quad 0.1) \\ C &= (0.5 \quad 0.8 \quad 0.7 \quad 1.0 \quad 0.5) \\ D &= (1.0 \quad 0.0 \quad 1.0 \quad 0.0 \quad 1.0) \end{aligned} \tag{3.49}$$

(a) Find $(A \cup B) \cap (A \cup C)$.

(b) Find $d_1(A, A_{\text{near}})$ and $d_2(A, A_{\text{far}})d_1(A, A_{\text{near}})$, where d_1 and d_2 represent the first- and the second-order distances, respectively.

(c) Find $E(A), E(D)$, where $E(.)$ represents entropy.

3.6 Plot the following fuzzy set for integer values of *x* between −4 and 5. Also, plot the corresponding generalized concentrator and the dilator hedges.

3.7 **(a)** Explain the Wang and Mendel algorithm for extracting fuzzy rules from data samples.
(b) Consider the following samples and show outputs for various steps in the algorithm. What rules can you generate from these data samples?

x_1	x_2	x_3	Class
27	5	1	Water
28	29	29	Forest