APPLICATIONS OF FUZZY SYSTEMS

Galip Cansever

E-mail: galip.cansever@altinbas.edu.tr

BASIC CONCEPTS OF FUZZY SETS

More formally, consider a variable with universe of discourse $\mathcal{X} \subseteq \Re$, and let x be a real number (i.e., $x \in \mathcal{X}$). Let M denote a fuzzy set defined on $\mathcal{X}$. A *membership function* $\mu^M(x)$ associated with M is a function that maps $\mathcal{X}$ into [0, 1] and gives the degree of membership of $\mathcal{X}$ is M. We say that the fuzzy set M *is characterized by* μ^M. Then the fuzzy set M is defined as

$$M = \{(x, \mu^M(x)) : x \in \mathcal{X}\}$$

This is a pairing of elements from $\mathcal{X}$ with their associated membership values.

For instance, the fuzzy set WARM (a linguistic value), when referring to OUTDOOR TEMPERATURE (a linguistic variable), may be characterized by the membership function of Figure 2.1.

Triangular Membership Functions

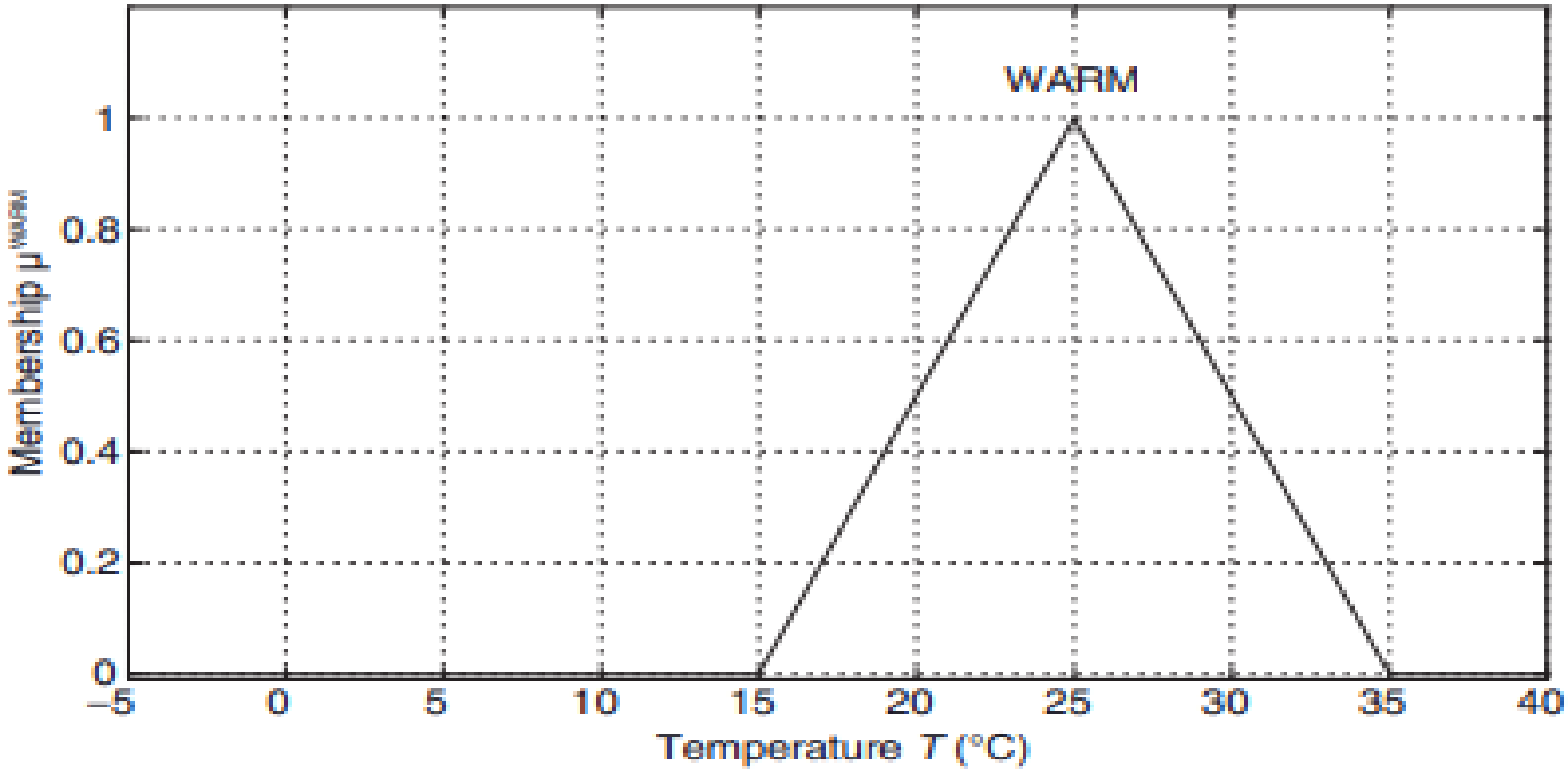


Figure 2.1. Triangular membership function.

This is called a *triangular* membership function, for obvious reasons. It is defined by the conditional function

$$\mu^{\text{WARM}}(T) = \begin{cases} 0.1T - 1.5, & 15 \le T < 25 \\ -0.1T + 3.5, & 25 \le T < 35 \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

where $\mu^{\text{WARM}}(T)$ is membership in the WARM fuzzy set and T is temperature.

Note that $\mu^{\text{WARM}}(T)$ is defined for all temperatures T even though it is zero for some T. The universe of discourse for TEMPERATURE is the entire set of possible temperatures $(-273, \infty)$°C, although there may be an effective universe of discourse of, say, $[-20, 50]$°C if it is known that the temperature will never be out of this range.

The membership function indicates that a temperature of 25°C (77°F) is definitely considered warm, temperatures >25°C are decreasingly considered warm as they increase from 25 to 35°C (95°F), and temperatures <25°C are decreasingly considered warm as they decrease from 25 to 15°C (59°F). According to the membership function, temperatures <15°C are not considered warm at all, and temperatures >35°C are also not considered warm at all. A temperature of 20°C is considered warm to a degree 0.5 because $\mu^{\text{WARM}}(20) = 0.5$, and a temperature of 32°C is considered warm to a degree 0.3 because $\mu^{\text{WARM}}(32) = 0.3$.

Another possibility for a membership function to characterize the WARM fuzzy set is the *Gaussian* membership function of Figure 2.2. The mathematical expression for a Gaussian function is

$$\mu(x) = \exp\left(-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2\right) \tag{2.2}$$

where c is the center of the function (i.e., the point at which the function attains its maximum of 1, and $\sigma > 0$ determines the spread, or width of the function).

The membership function characterizing the WARM fuzzy set shown in Figure 2.2 conveys similar information to the membership function of Figure 2.1,

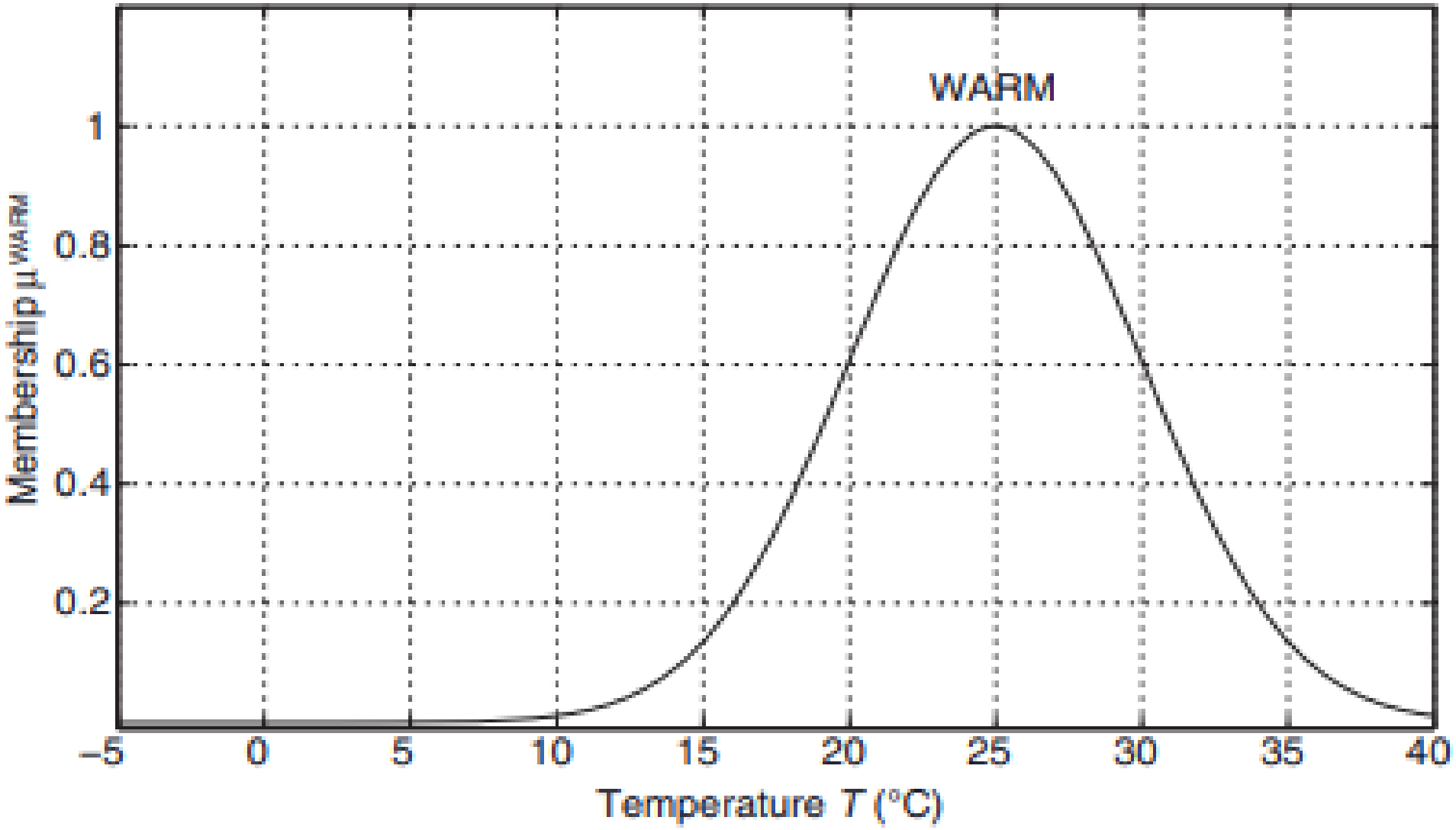


Figure 2.2. Gaussian membership function.

namely, that temperatures close to 25°C are considered warmer, while temperatures further away from 25°C in either direction are considered less warm. The mathematical expression for the membership function in Figure 2.2 is

$$\mu^{\text{WARM}}(T) = \exp\left(-\frac{1}{2}\left(\frac{T-25}{5}\right)^2\right) \tag{2.3}$$

The shape of membership functions is arbitrary. The only requirement is that the membership function make sense for the fuzzy set being defined. For instance, a membership function like that of Figure 2.3 would not make sense if we wanted to characterize the fuzzy set of warm temperatures. From this membership function, it appears that 20 and 30°C are both considered warmer than 25°C!

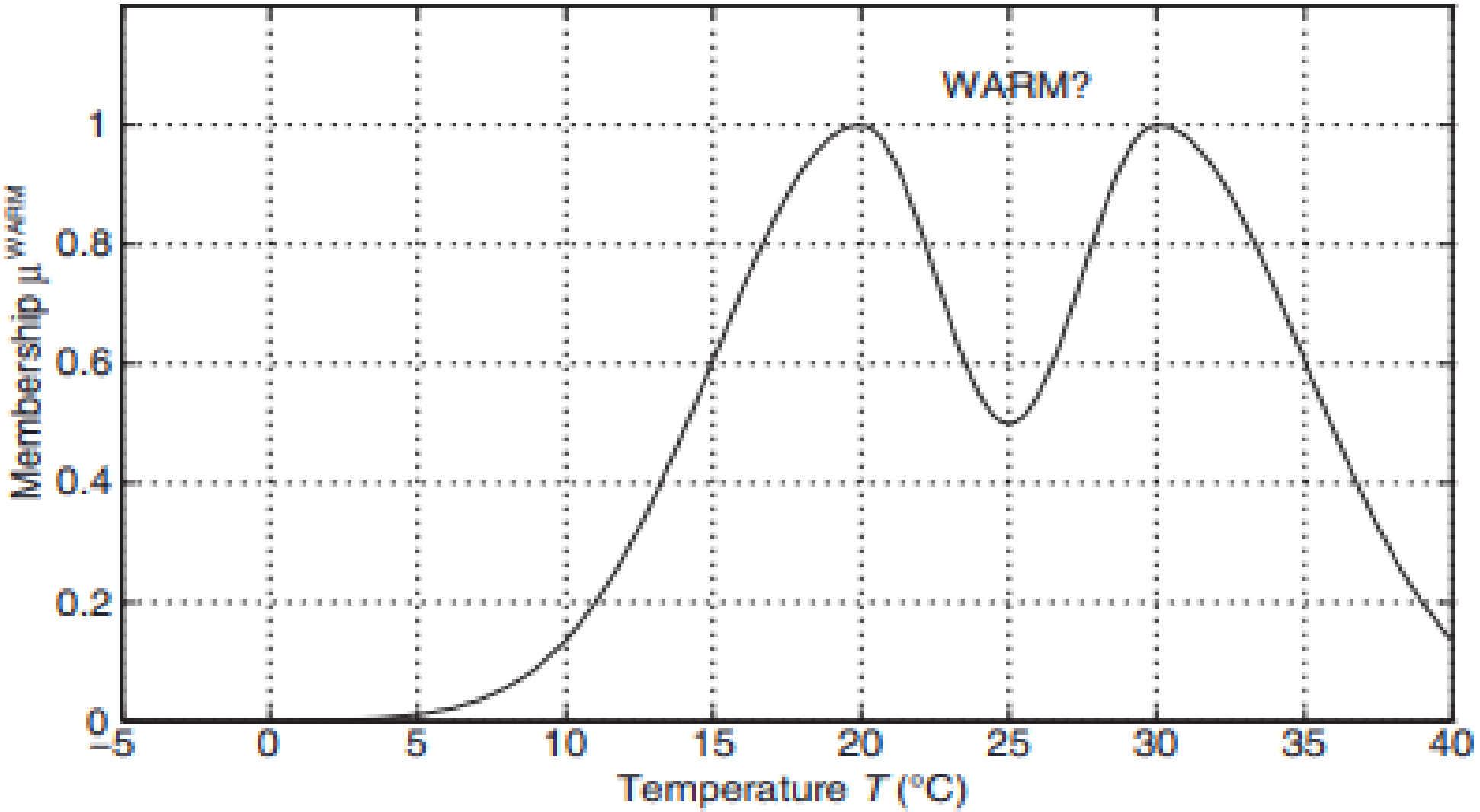


Figure 2.3. Illogical membership function to characterize WARM temperatures.

While any membership function shape is permissible, in most applications of identification and control either triangular or Gaussian membership functions suffice. Other shapes include trapezoidal, raised cosine, and so on. In this book, we use either triangular or Gaussian membership functions. An additional commonly used membership function shape is the *singleton*, to be introduced in Section 2.5. In some applications, especially identification and classification ones, the triangles or Gaussians may be asymmetrical, as in Figure 2.4.

Usually, several fuzzy sets are defined for the same variable. For instance, we could define four fuzzy sets characterizing COLD, COOL, WARM, and HOT temperatures (see Fig. 2.5).

Note that the COLD membership function saturates at 1 for all temperatures <5°C. This indicates that all temperatures <5°C are considered cold since there are no fuzzy sets defined for even lower temperatures (say, a FRIGID fuzzy set). Also note that, like crisp sets, an element may belong to more than one set. For example, a temperature of 15°C belongs to all four fuzzy sets to different degrees of certainty. This temperature is considered COOL with absolute certainty, and is also considered

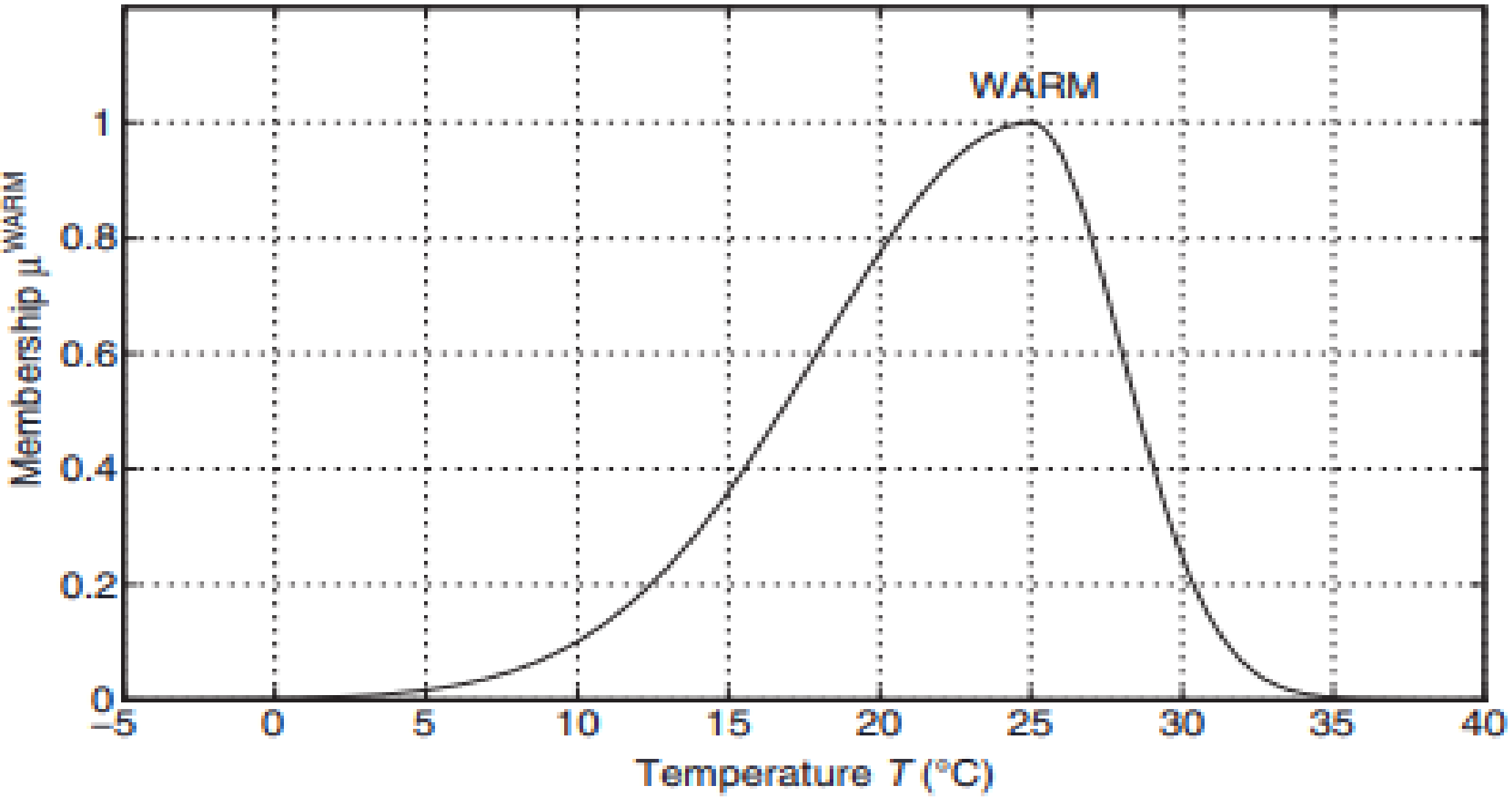


Figure 2.4. Asymmetrical Gaussian membership function.

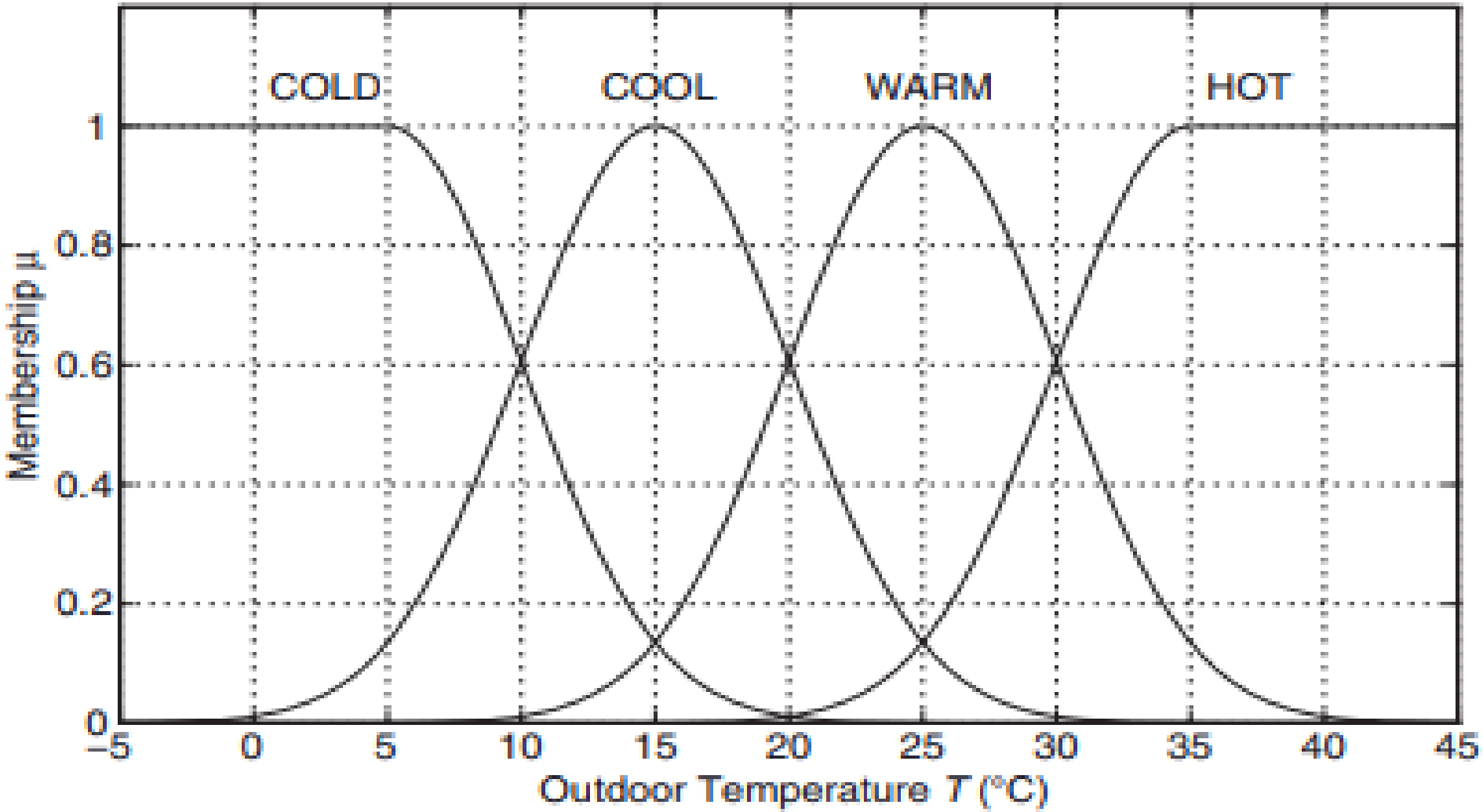


Figure 2.5. Four fuzzy sets defined for the TEMPERATURE variable.

WARM and COLD to lesser extents. This temperature is also considered HOT to a very small extent. In most applications, adjacent fuzzy sets overlap.

2.2 USEFUL CONCEPTS FOR FUZZY SETS

The following are some concepts for fuzzy sets that we will find useful in later studies:

- The ***support*** of a fuzzy set is the set of points in the universe of discourse for which the membership function is >0. For example, the support of the WARM

fuzzy set in Figure 2.1 is the open interval (15, 35)°C. The support of the WARM fuzzy set in Figure 2.2 is (−273, ∞)°C.

- An ***α*-cut** of a fuzzy set is the set of points in the universe of discourse where the membership function is $>\alpha$. It is the collection of all members of the set with a certain minimum degree of membership. For example, the 0.5-cut of the fuzzy set in Figure 2.1 is (20, 30)°C. The 0.5-cut of the fuzzy set in Figure 2.2 is (19.1129, 30.8871)°C.
- The ***height*** of a fuzzy set is the peak value reached by its membership function. This is usually 1.
- A ***normal*** fuzzy set is a fuzzy set whose membership function reaches 1 for at least one point in the universe of discourse.
- A ***convex*** fuzzy set is a fuzzy set whose membership function satisfies

$$\mu(\lambda x_1 + (1-\lambda)x_2) \geq \min(\mu(x_1), \mu(x_2)) \tag{2.4}$$

$\forall(x_1, x_2)$ and $\forall\lambda \in (0, 1]$. The concept of a convex fuzzy set is not to be confused with the concept of a convex function, although the definitions look similar. Many membership functions are not convex as functions, although they characterize convex fuzzy sets. The fuzzy sets of Figures 2.1 and 2.2 are convex, although the membership function of Figure 2.2 is not a convex function. The fuzzy set of Figure 2.3 is not convex. Nonconvex fuzzy sets are legal in fuzzy systems, as long as they accurately represent their corresponding quantity.

2.3 SOME SET-THEORETIC AND LOGICAL OPERATIONS ON FUZZY SETS

It is possible to define concepts (subset, compliment, intersection, union, etc.) for fuzzy sets similarly to crisp sets. Below we give some of the more often-used operations on fuzzy sets.

Fuzzy Subset

Let M^1 and M^2 be two fuzzy sets defined for a variable on the universe of discourse $\mathcal{X}$, and let their associated membership functions be $\mu^1(x)$ and $\mu^2(x)$, respectively. Then, M^1 is a *fuzzy subset* of M^2 (or $M^1 \subseteq M^2$) if $\mu^1(x) \leq \mu^2(x) \forall x \in \mathcal{X}$.

Fuzzy Compliment

Consider a fuzzy set M defined for a variable on the universe of discourse $\mathcal{X}$, and let M have associated membership function $\mu^M(x)$. The *fuzzy compliment* of M is a fuzzy set $\overline{M}$ characterized by membership function $\mu^{\overline{M}}(x) = 1 - \mu^M(x)$.

Fuzzy Intersection (AND)

Let M^1 and M^2 be two fuzzy sets defined for a variable on the universe of discourse $\mathcal{X}$, and let their associated membership functions be

$\mu^1(x)$ and $\mu^2(x)$, respectively. The *fuzzy intersection* of M^1 and M^2, denoted by $M^1 \cap M^2$, is a fuzzy set with membership function (1) $\mu^{M^1 \cap M^2}(x) = \min\{\mu^1(x), \mu^2(x) : x \in \mathcal{X}\}$ (*minimum*) or (2) $\mu^{M^1 \cap M^2}(x) = \{\mu^1(x)\mu^2(x) : x \in \mathcal{X}\}$ (*algebraic product*).

Notice that we give two possibilities to characterize the intersection of two fuzzy sets: *min* and *algebraic product*. There are other methods that can be used to represent intersection as well [8–13]. In general, we could use any operation on the two membership functions $\mu^1(x)$ and $\mu^2(x)$ that satisfies the common-sense requirements:

1. An element in the universe cannot belong to the intersection of two fuzzy sets to a greater degree than it belongs to either one of the fuzzy sets individually.
2. If an element does not belong to one of the fuzzy sets, then it cannot belong to the intersection of that fuzzy set and another fuzzy set.
3. If an element belongs to both fuzzy sets with absolute certainty, then it belongs to the intersection of the two fuzzy sets with absolute certainty.

Since the membership function values are between 0 and 1, the operations of *minimum* and *product* both satisfy the above three requirements. These two operations are found to suffice for most fuzzy systems.

If we use the notation * to represent the AND operation, then the membership function characterizing the fuzzy intersection of M^1 and M^2 can be generically written as $\mu^{M^1 \cap M^2}(x) = \mu^1(x) * \mu^2(x)$. The * operation (whether *min*, *product*, or other) is called a triangular norm or *T-norm*.

Fuzzy Union (Or)

Let M^1 and M^2 be two fuzzy sets defined for a variable on the universe of discourse $\mathcal{X}$, and let their associated membership functions be $\mu^1(x)$ and $\mu^2(x)$, respectively. The *fuzzy union* of M^1 and M^2, denoted by $M^1 \cup M^2$, is a fuzzy set with membership function (1) $\mu^{M^1 \cup M^2}(x) = \max\{\mu^1(x), \mu^2(x) : x \in \mathcal{X}\}$ (*maximum*) or (2) $\mu^{M^1 \cup M^2}(x) = \{\mu^1(x) + \mu^2(x) - \mu^1(x)\mu^2(x) : x \in \mathcal{X}\}$ (*algebraic sum*).

Notice that we give two possibilities to characterize the union of two fuzzy sets: *max* and *algebraic sum*. There are other methods that can be used to represent union as well (see the above references). In general, we could use any operation on the two membership functions $\mu^1(x)$ and $\mu^2(x)$ that satisfies the common-sense requirements:

1′. An element in the universe cannot belong to the union of two fuzzy sets to a lesser degree than it belongs to either one of the fuzzy sets individually.

2′. If an element belongs to one of the fuzzy sets, then it must belong to the union of that fuzzy set and another fuzzy set.

3′. If an element does not belong to either fuzzy set, then it cannot belong to the union of the two fuzzy sets.

Since the membership function values are between 0 and 1, the operations of *maximum* and *algebraic sum* both satisfy requirements 1′–3′. These two operations are found to suffice for most fuzzy systems.

If we use the notation $\oplus$ to represent the OR operation, then the membership function characterizing the fuzzy union of M^1 and M^2 can be written as $\mu^{M^1 \cup M^2}(x) = \mu^1(x) \oplus \mu^2(x)$. The $\oplus$ operation (whether *max*, *algebraic sum*, or other) is called a triangular conorm or *T-conorm*.

Fuzzy Cartesian Product

Cartesian product refers to combining fuzzy sets defined for *different* variables on *different* universes of discourse. If M_1^j, M_2^k, ... , M_n^l are fuzzy sets defined on the universes $\mathcal{X}_1$, $\mathcal{X}_2$, ... , $\mathcal{X}_n$, respectively, with fuzzy set M_i^m characterized by membership function μ_i^m, their Cartesian product is a fuzzy set, denoted by $M_1^j \times M_2^k \times \cdots \times M_n^l$, characterized by the membership function

$$\mu^{M_1^j \times M_2^k \times \cdots \times M_n^1}(x_1, x_2, \ldots, x_n) = \mu_1^j(x_1) * \mu_2^k(x_2) * \cdots * \mu_1^l(x_n) \qquad (2.5)$$

where the T-norm * is defined as above (*min*, *product*, etc.).

2.4 EXAMPLE

Let the COMFORTABLE ROOM TEMPERATURE (CRT) fuzzy set be characterized by the membership function

$$\mu^{CRT}(T) = \exp\left(-0.5\left(\frac{T-25}{2}\right)^2\right) \tag{2.6}$$

which is shown in Figure 2.6.

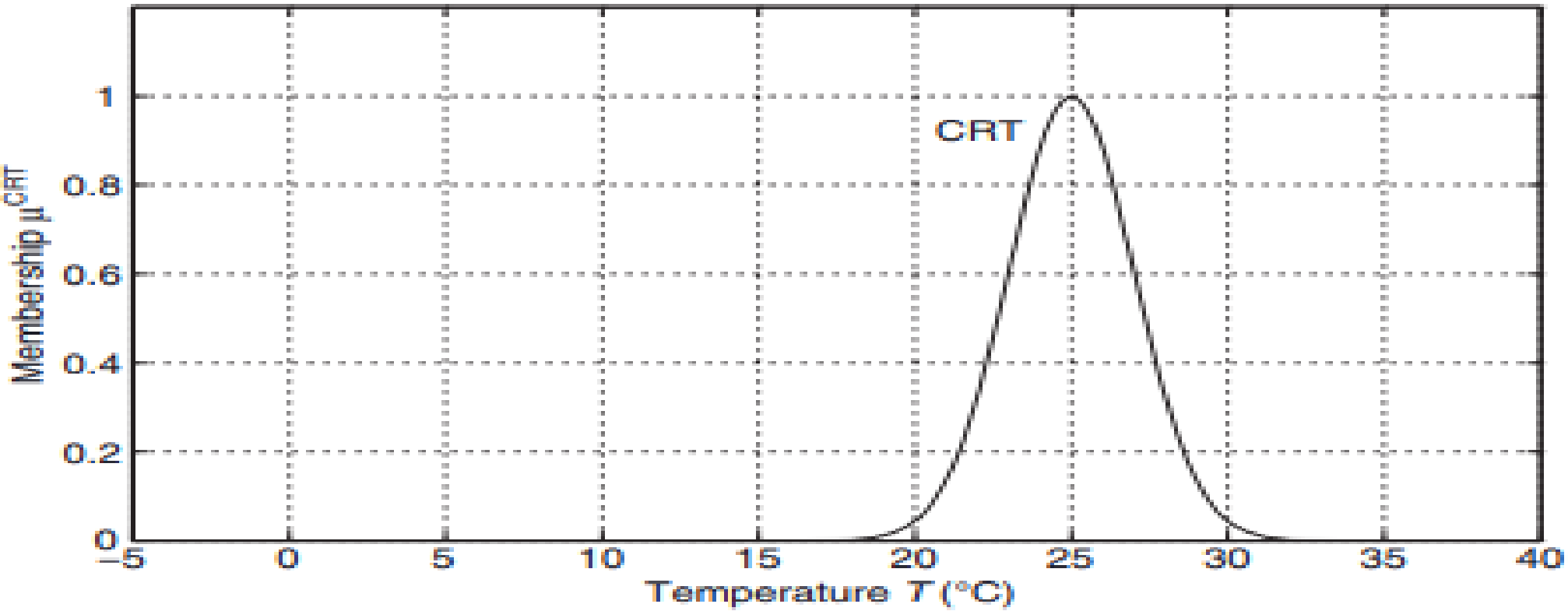


Figure 2.6. Membership function for CRT fuzzy set.

Now it is clear that the CRT fuzzy set is a subset of the WARM fuzzy set shown in Figure 2.2, because $\mu^{\text{CRT}}(T) \leq \mu^{\text{WARM}}(T) \forall T$. A plot of $\mu^{\text{CRT}}(T)$ and $\mu^{\text{WARM}}(T)$ together is shown in Figure 2.7.

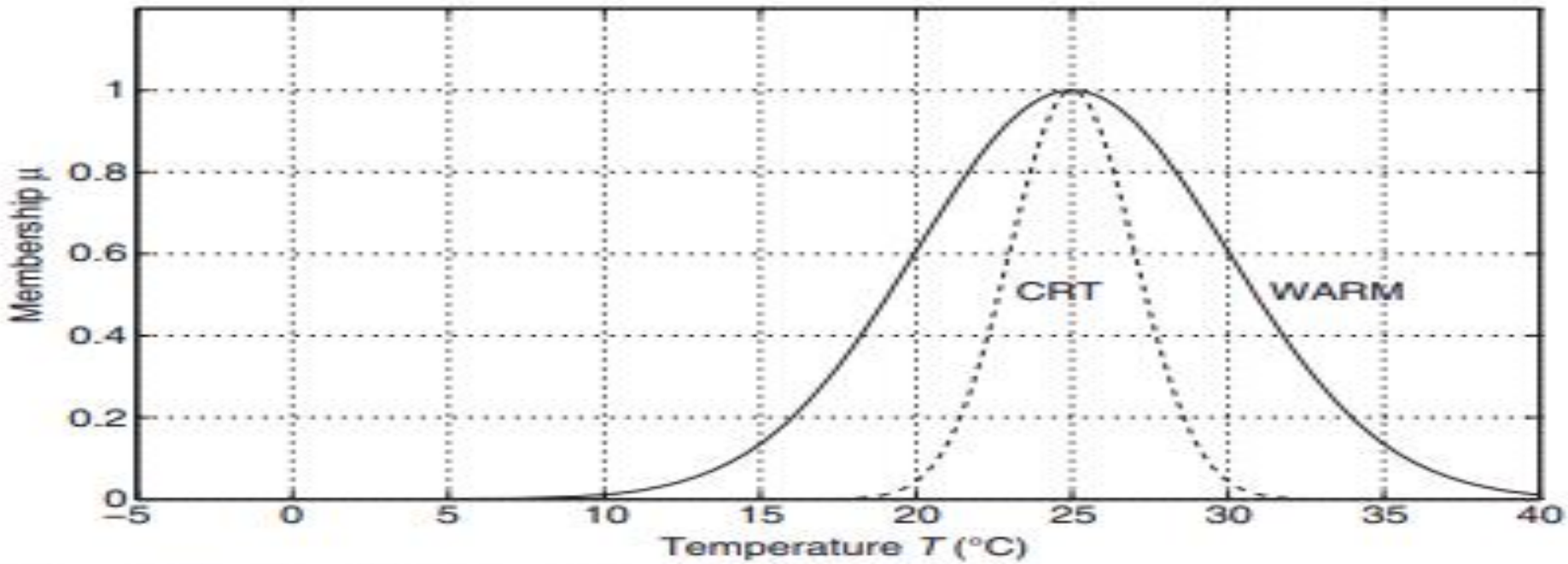


Figure 2.7. Graphical illustration of a fuzzy subset.

The compliment of the WARM fuzzy set (i.e., the NOT WARM) or $\overline{\text{WARM}}$ fuzzy set, is characterized by the membership function

$$\mu^{\overline{\text{WARM}}}(T) = 1 - \exp\left(-0.5\left(\frac{T-25}{5}\right)^2\right) \tag{2.7}$$

and shown in Figure 2.8.

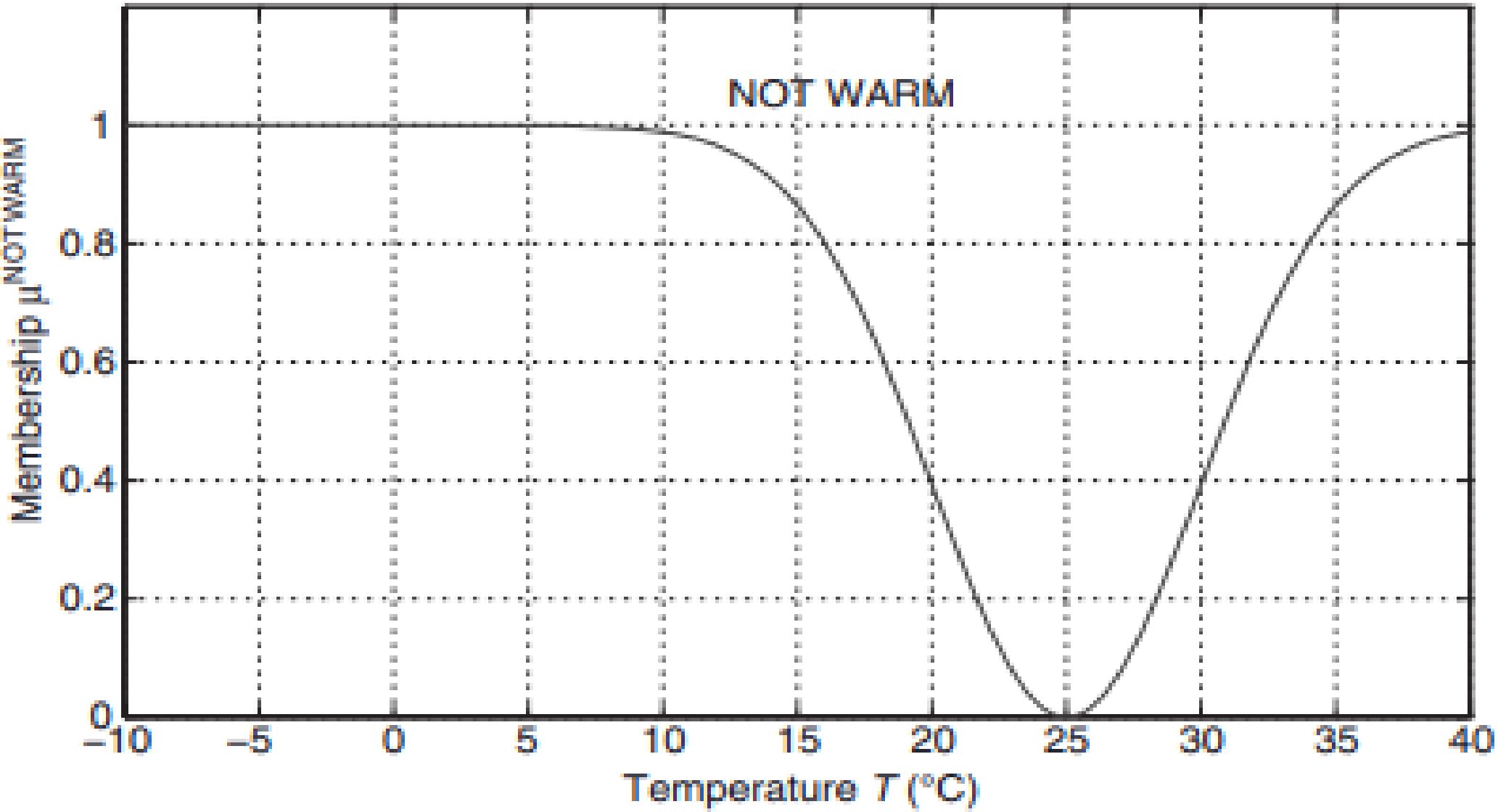


Figure 2.8. Membership function characterizing the NOT WARM fuzzy set.

The membership function characterizing the intersection of the COOL and WARM fuzzy sets using the *min* T-norm is given by

$$\begin{aligned}\mu^{\text{COOL}\cap\text{WARM}}(T) &= \min\left\{\mu^{COOL}(T), \mu^{WARM}(T)\right\} \\ &= \min\left\{\exp\left(-0.5\left(\frac{T-15}{5}\right)^2\right), \exp\left(-0.5\left(\frac{T-25}{5}\right)^2\right)\right\} \qquad (2.8)\end{aligned}$$

This membership function is shown in Figure 2.9.

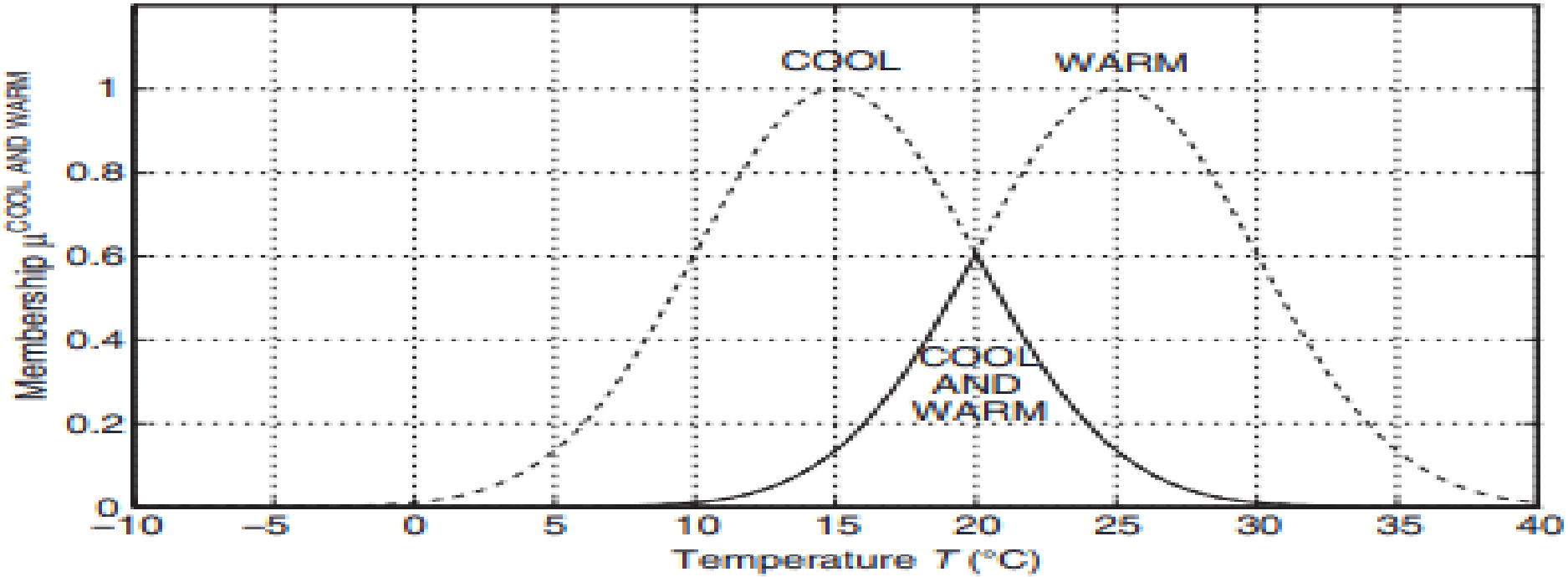


Figure 2.9. The membership function characterizing the fuzzy. intersection of COOL and WARM fuzzy sets.

The determination of μ using the *algebraic product* T-norm is left as an exercise.

The membership function characterizing the union of the COOL and WARM fuzzy sets using the *max* T-conorm is given by

$$\begin{aligned}\mu^{\text{COOL}\cup\text{WARM}}(T) &= \max\left\{\mu^{\text{COOL}}(T), \mu^{\text{WARM}}(T)\right\} \\ &= \max\left\{\exp\left(-0.5\left(\frac{T-15}{5}\right)^2\right), \exp\left(-0.5\left(\frac{T-25}{5}\right)^2\right)\right\} \quad (2.9)\end{aligned}$$

This membership function is shown in Figure 2.10.

The determination of $\mu^{\text{COOL}\cup\text{WARM}}$ using the *algebraic sum* T-norm is left as an exercise.

To illustrate fuzzy Cartesian product, suppose we also have defined three fuzzy sets, LOW, GENTLE, and HIGH for a variable called WIND SPEED, as shown in Figure 2.11.

Then we can characterize a fuzzy set for the conjunction WARM TEMPERATURE and GENTLE WIND by the Cartesian product WARM $\times$ GENTLE. The WARM $\times$ GENTLE fuzzy set is characterized by the membership function

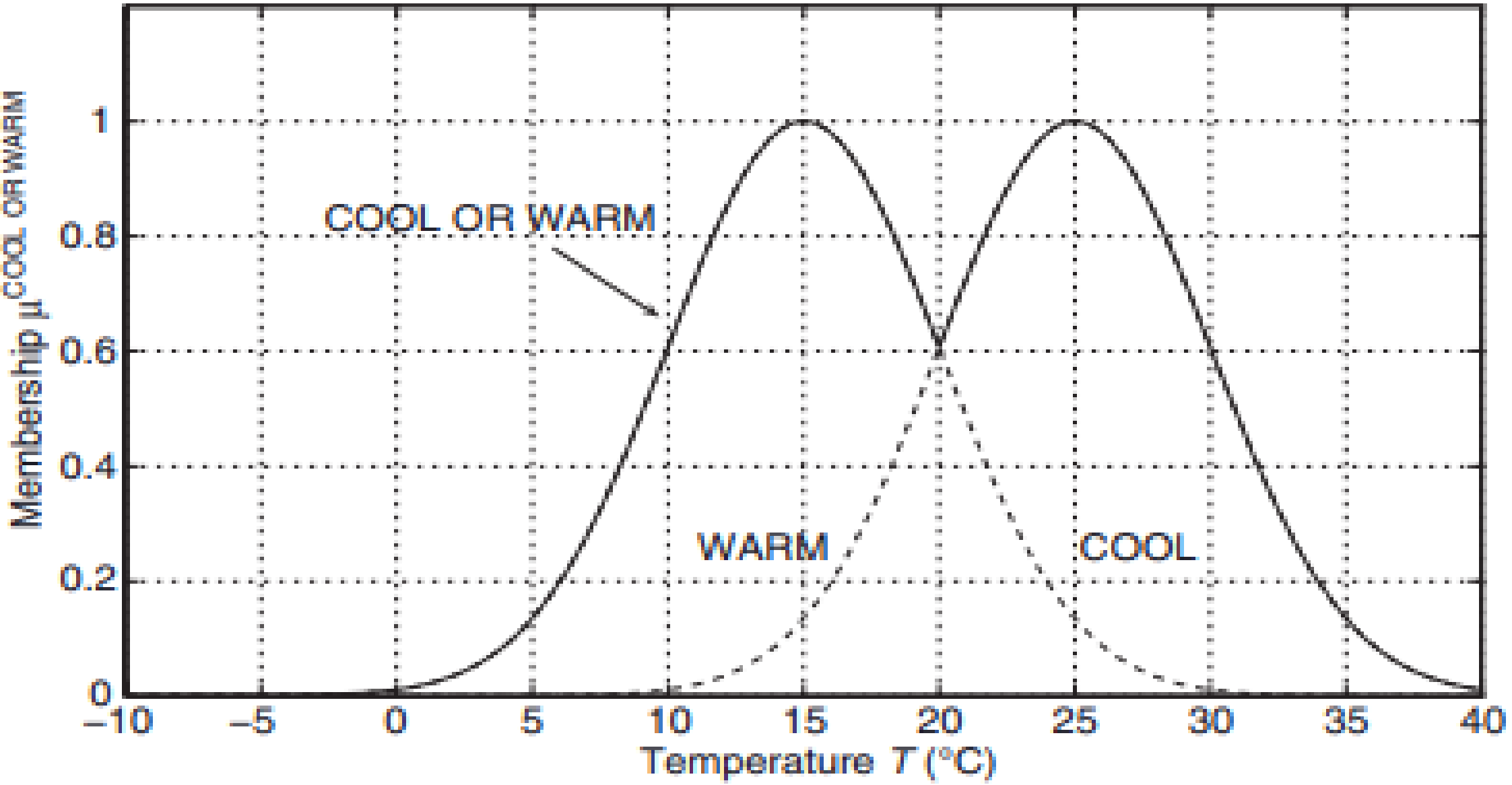


Figure 2.10. The membership function characterizing the fuzzy union of COOL and WARM fuzzy sets.

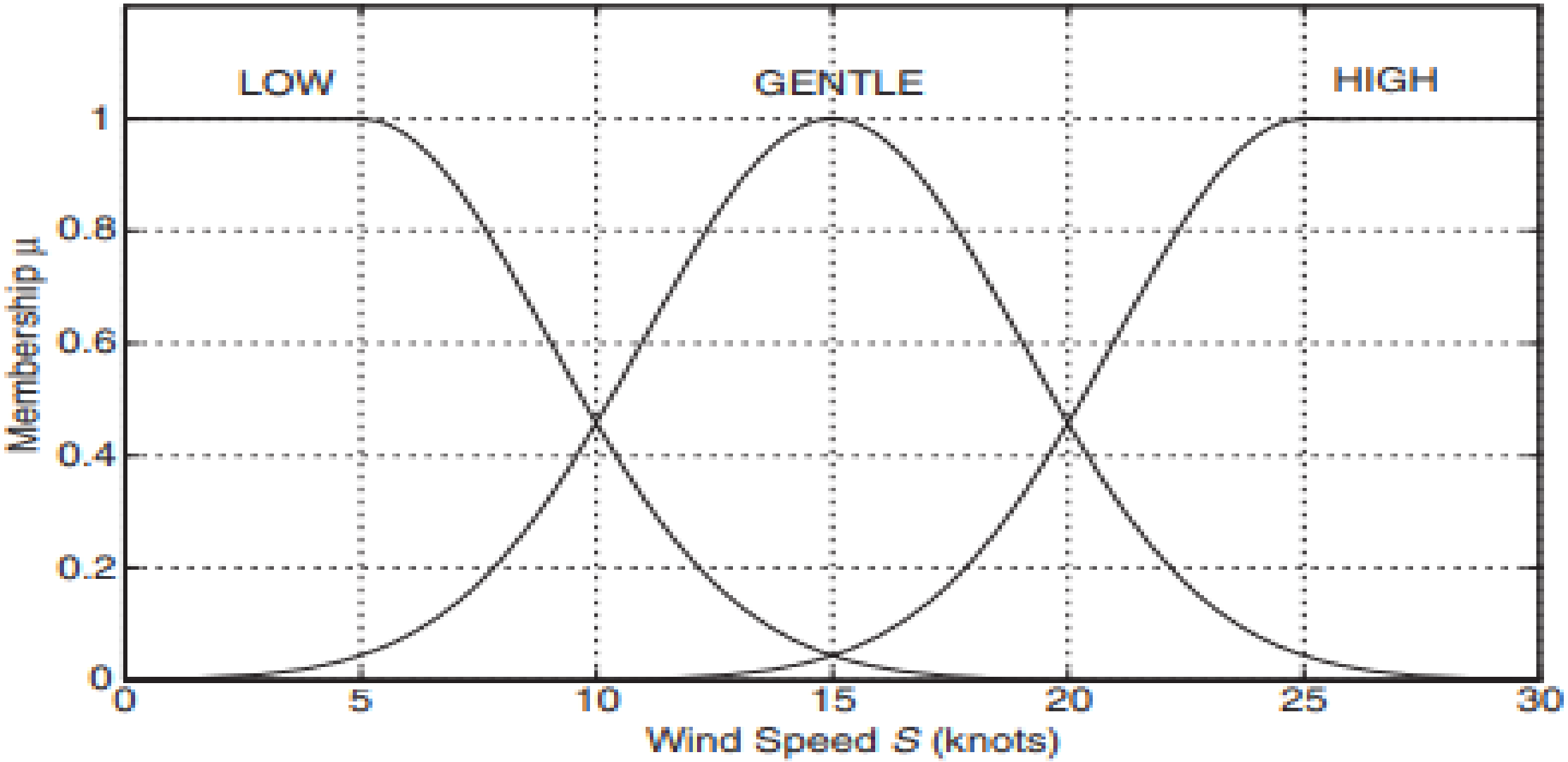


Figure 2.11. Fuzzy sets defined for WIND SPEED.

$$\mu^{\text{WARM and GENTLE}}(T, S) = \mu^{\text{WARM}}_{\text{TEMP}}(T) * \mu^{\text{GENTLE}}_{\text{WIND}}(S) \tag{2.10}$$

where subscripts denote the linguistic variable and superscripts denote the linguistic value.

Using the *product* T-norm, this gives

$$\mu^{\text{WARM and GENTLE}}(T, S) = \exp\left(-0.5\left(\frac{T-25}{5}\right)^2\right)\exp\left(-0.5\left(\frac{S-15}{4}\right)^2\right) \tag{2.11}$$

Note that $\mu^{\text{WARM and GENTLE}}(T, S)$ is a function of two variables (temperature T and wind speed S), so when plotted it will be a three-dimensional surface over the T–S plane (see Fig. 2.12).

Note that $\mu^{\text{WARM and GENTLE}}(T, S)$ is a function of two variables (temperature T and wind speed S), so when plotted it will be a three-dimensional surface over the T–S plane (see Fig. 2.12).

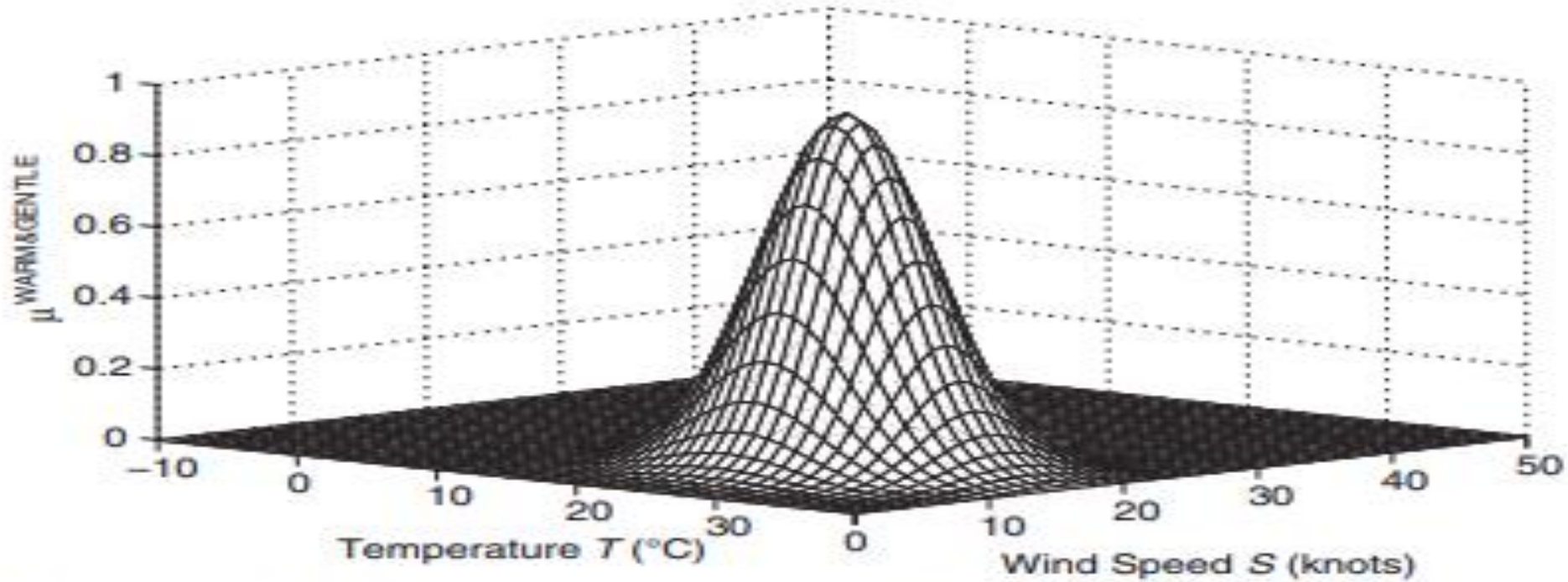


Figure 2.12. Membership function for WARM & GENTLE fuzzy set.

According to the membership function of Figure 2.12, a temperature of 25°C together with a wind speed of 15 knots is classified as WARM&GENTLE with certainty 1 (i.e., absolute certainty), a temperature of 20°C together with a wind speed of 20 knots is classified as WARM&GENTLE with certainty 0.2777, and a temperature of 28°C together with a wind speed of 13 knots is classified as WARM&GENTLE with certainty 0.7371.

2.5 SINGLETON FUZZY SETS

Another type of fuzzy set that is used extensively in fuzzy systems (see Chapter 3) is the *singleton* fuzzy set. This type of set contains only one member, and that member's degree of belongingness to the set is unity. Suppose we create five singleton fuzzy sets on the VOLTAGE universe of discourse such that a NEGATIVE LARGE (NL) voltage is defined as exactly −1 V, a NEGATIVE SMALL (NS) voltage is defined as exactly −0.5 V, a ZERO (Z) voltage is defined as exactly 0 V, a POSITIVE SMALL (PS) voltage is defined as exactly 0.5 V, and a POSITIVE LARGE (PL) voltage is defined as exactly 1 V. We can consider these five voltages as the only members of five singleton fuzzy sets characterized by the membership functions shown in Figure 2.13.

Although it seems unnecessarily complicated to do this, the characterization of certain quantities as members of singleton fuzzy sets is very useful for fuzzification and certain output fuzzy sets (see Chapter 3).

MAMDANI FUZZY SYSTEMS

A system is a combination of components that as a whole operate on a vector of input functions of time $x(t) \in \Re^n$ for each t to produce a vector of output functions of time $y(t) \in \Re^m$ for each t. A fuzzy system is a system that uses fuzzy logic to operate on the input $x(t)$ to produce the crisp output $y(t)$. It can be shown that a fuzzy system with n inputs and m outputs (i.e., a multi-input multi-output, or MIMO fuzzy system) is equivalent to m fuzzy systems, each with n inputs and one output (i.e., multi-input single-output, or MISO fuzzy systems). Therefore, we will study only MISO fuzzy systems in this book. All fuzzy controllers and identifiers are fuzzy systems.

3.1 IF-THEN RULES AND RULE BASE

Much of human decision making occurs in the form of "if-then" rules. The four forms of if-then rules in classical logic are called *modus ponendo ponens* (Latin for "mode that affirms by affirming") or simply *modus ponens*, *modus tollendo tollens* (Latin for "mode that denies by denying"), or simply *modus tollens*, *modus ponendo tollens* (Latin for "mode that denies by affirming"), and *modus tollendo ponens* (Latin for "mode that affirms by denying").

For examples of these, consider the problem of steering a vehicle toward a target in the presence of an obstacle. Possible rules for deciding which way to steer might be (this is not a complete set):

1. If the target direction is ahead, then the steering direction is straight (modus ponendo ponens).
2. If the obstacle direction is ahead then the steering direction is not straight (modus ponendo tollens).
3. If the obstacle direction is not ahead then the steering direction is straight (modus tollendo ponens).
4. If the target direction is not ahead then the steering direction is not straight (modus tollendo tollens).

In statement 1, the recommendation to steer straight is affirmed by affirming that the target is ahead. In statement 2, the recommendation to steer straight is denied by affirming that an obstacle is ahead. In statement 3, the recommendation to steer straight is affirmed by denying that the obstacle is ahead. In statement 4, the recommendation to steer straight is denied by denying that the target is ahead. Of course, a complete set of rules for steering to a target in the presence of obstacles would require more rules than the above. People reason in all of these ways. All of these modes of reasoning can be implemented with fuzzy logic (see [14] and [15] for practical examples of controllers employing *modus ponendo tollens* logic).

The vast majority of if-then rules used in fuzzy control and identification are of the *modus ponens* form. For example, consider the rule R_i:

$$R_i \qquad \text{If } \tilde{x} \text{ is } \tilde{P}, \text{ then } \tilde{y} \text{ is } \tilde{Q} \tag{3.1}$$

where $\tilde{x}$ is a linguistic variable defined on universe $\mathcal{X}$, $\tilde{P}$ is a linguistic value described by fuzzy set P defined on universe $\mathcal{X}$, $\tilde{y}$ is a linguistic variable defined on universe $\mathcal{Y}$, and $\tilde{Q}$ is a linguistic value described by fuzzy set Q defined on universe $\mathcal{Y}$. A tilde over a symbol indicates a linguistic variable or value.

The first part of the statement, "$\tilde{x}$ is $\tilde{P}$," is called the *premise* of the rule, and the second part of the statement, "$\tilde{y}$ is $\tilde{Q}$ " is called the *consequent* of the rule. The consequent of the rule is affirmed by affirming the premise—if the premise is true, the consequent is also true.

An example of an if-then rule in *modus ponens* form pertaining to stopping a car is, "If SPEED is FAST then BRAKE PRESSURE is HEAVY." In this rule, SPEED is FAST is the premise and BRAKE PRESSURE is HEAVY is the consequent. SPEED is the input linguistic variable, FAST is a linguistic value of SPEED and is a fuzzy set on the SPEED universe, BRAKE PRESSURE is the output linguistic variable, and HEAVY is a linguistic value of BRAKE PRESSURE and is a fuzzy set on the BRAKE PRESSURE universe.

Note that there may be more than one part to the premise, that is, we could have the rule R_j:

$$R_j \qquad \text{If } \tilde{x}_1 \text{ is } \tilde{P}_1^k \text{ and } \tilde{x}_2 \text{ is } \tilde{P}_2^l \text{ and } \cdots \text{ and } \tilde{x}_n \text{ is } \tilde{P}_n^m, \text{ then } \tilde{y} \text{ is } \tilde{Q}^j \tag{3.2}$$

In this rule, the premise is a conjunction of n conditions: $\tilde{x}_1$ is $\tilde{P}_1^k$ and $\tilde{x}_2$ is $\tilde{P}_2^l$ and … and $\tilde{x}_n$ is $\tilde{P}_n^m$. For example, another rule about stopping a car might be, "If SPEED is FAST and GRADE is DOWNHILL then BRAKE PRESSURE is VERY HEAVY."

In general, we use a number of rules to accomplish a task. Several rules are needed to specify actions to be taken under different conditions. The collection of rules as a whole is called a *Rule base*. A simple rule base for stopping a car might be

1. If SPEED is SLOW, then BRAKE PRESSURE is LIGHT.
2. If SPEED is MEDIUM, then BRAKE PRESSURE is MEDIUM.
3. If SPEED is FAST, then BRAKE PRESSURE is HEAVY.

To maintain notational simplicity, in the rest of this book we will dispense with the tildes over linguistic variables and values. The fact that these are linguistic quantities will be assumed. Thus, Rule j in (3.2) will simply be written

$$R_j \qquad \text{If } x_1 \text{ is } P_1^k \text{ and } x_2 \text{ is } P_2^l \text{ and } \cdots \text{ and } x_n \text{ is } P_n^m, \text{ then } y \text{ is } Q^j$$

where it will be understood that x_1 is a linguistic variable, P_1^k is the fuzzy set associated with linguistic value $\tilde{P}_1^k$, and so on.

3.2 FUZZY SYSTEMS

The fuzzy systems considered in this book have n inputs $x_i \in \mathcal{X}_i$, where $i = 1, 2, \ldots, n$ and $\mathcal{X}_i$ is the universe of discourse for x_i, and one output $y \in \mathcal{Y}$, where $\mathcal{Y}$ is the universe of discourse for y (as explained above, we assume a MISO fuzzy system). The fuzzy system has the following structure (Fig. 3.1):

Figure 3.1. Structure of fuzzy systems.

The inputs x and output y are *crisp* (i.e., they are real numbers, not fuzzy sets). The fuzzification block converts the crisp inputs into fuzzy sets. The inference mechanism uses the rules in the rule base to convert these fuzzy sets into other fuzzy sets that are representative of the recommendations of the various rules in the rule base. The defuzzification block combines these fuzzy recommendations to give a crisp output y.

3.3 FUZZIFICATION

The function of the fuzzification stage is to convert the measured quantities from the process (voltages, velocities, temperatures, etc.) into fuzzy sets to be used by the inference stage. For example, if there is process or measurement noise, we may want to account for this by creating fuzzy sets for the measured quantities rather than assuming they are accurate as measured. In this case, the measured quantities are not believed exactly as measured (because they contain noise), but are converted into fuzzy sets that reflect their degree of undependability.

In many cases, however, the measurements are believed as measured. If measurement and process noise is low and measured quantities can be taken as true, the fuzzification stage consists of creating singleton membership functions at the measured quantities (see Section 2.5). Singleton fuzzifiucation will be used throughout the remainder of this book.

3.4 INFERENCE

The first function of the inference stage is to determine the degree of firing of each rule in the rule base. Consider rule R_i (3.1), which has a single input x. Let fuzzy set P be characterized by the membership function $\mu^P(x)$, and fuzzy set Q be characterized by the membership function $\mu^Q(y)$. For a particular crisp input $x \in \mathcal{X}$, we say rule R_i is *fired*, or is *on* (i.e., it is taken as true) to the extent $\mu^P(x)$. As mentioned in Chapter 2, this is a real number in the interval [0, 1]. More generally, we call fuzzy set P the *premise fuzzy set for Rule i*, and μ^P, which we will call μ_i, the *premise membership function for Rule i*. Then, for a particular real input x, rule R_i is fired (or is *on*) to the extent $\mu_i(x)$.

Now consider rule R_j (3.2), which has n inputs $x_1, x_2, \ldots, x_n$. Let the fuzzy set $P_1^k \times P_2^l \times \cdots \times P_n^m$ be characterized by the membership function $\mu^{P_1^k \times P_2^l \times \cdots \times P_n^m}$. For a particular real input $\underline{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n$, we say rule R_j is *fired* to the extent $\mu^{P_1^k \times P_2^l \times \cdots \times P_n^m}(\underline{x}) = \mu_1^k(x_1) * \mu_2^l(x_2) * \cdots * \mu_n^m(x_n)$ [see (2.5)]. This is a real number in the interval [0, 1]. More generally, we call fuzzy set $P_1^k \times P_2^l \times \cdots \times P_n^m$ the *premise fuzzy set for Rule* R_j and $\mu^{P_1^k \times P_2^l \times \cdots \times P_n^m}$, which we will call μ_j, the *premise membership function for Rule j*. Then for a particular real input $\underline{x}$, rule R_j is fired (or is *on*) to the extent $\mu_j(\underline{x}) = \mu_1^k(x_1) * \mu_2^l(x_2) * \cdots * \mu_n^m(x_n)$.

The second function of the inference stage is to determine the degree to which each rule's recommendation is to be weighted in arriving at the final decision and to determine an implied fuzzy set corresponding to each rule. Consider rule R_j (3.2) with input $\underline{x} = (x_1, x_2, \ldots, x_n)$. From the discussion above, we know this rule is fired to the degree $\mu_j(\underline{x})$. Therefore we *attenuate* the recommendation of rule R_j, which is fuzzy set Q^j characterized by $\mu^{Q^j}(y)$, by $\mu_j(\underline{x})$. This produces an *implied fuzzy set* $\hat{Q}^j$ defined on $\mathcal{Y}$, characterized by the membership function

$$\mu^{\hat{Q}^j}(y) = \mu_j(\underline{x}) * \mu^{Q^j}(y) \tag{3.3}$$

$$\mu^{\hat{Q}^j}(y) = \mu_j(\underline{x}) * \mu^{Q^j}(y) \tag{3.3}$$

If there are R rules in the form of (3.2), each rule has its own premise membership function $\mu_j(\underline{x})$, $j = 1, 2, \ldots, R$. The R rules produce R implied fuzzy sets $\hat{Q}^j$, $j = 1, 2, \ldots, R$, each characterized by a membership function calculated as in (3.3). Note that $\mu^{Q^j}(y)$ and $\mu^{\hat{Q}^j}(y)$ are defined $\forall y \in \mathcal{Y}$. On the other hand, the degree of firing of rule j, $\mu_j(\underline{x})$, is a function of a particular real vector $\underline{x} \in \Re^n$, hence is a real number.

In summary, the function of the inference stage is twofold: (1) to determine the degree of firing of each rule in the rule base, and (2) to create an implied fuzzy set for each rule corresponding to the rule's degree of firing.

3.5 DEFUZZIFICATION

The function of the defuzzification stage is to convert the collection of recommendations of all rules into a crisp output. Consider a rule base consisting of R rules of the form (3.2). Then, we have R implied fuzzy sets, one from each rule, each recommending a particular outcome. In order to arrive at one crisp output y, we combine all of these recommendations by taking a weighted average of the various recommendations. There are several ways to do this, but perhaps the two most common

are *center of gravity* (COG) defuzzification, and *center average* (CA) defuzzification.

3.5.1 Center of Gravity (COG) Defuzzification

Suppose the consequent fuzzy set of Rule i is Q^i, characterized by membership $\mu^{Q^i}(y)$. Define the *center of area* of $\mu^{Q^i}(y)$ to be the point q_i in the universe $\mathcal{Y}$ with the property that

$$\int_{-\infty}^{q_i} \mu^{Q^i}(y)dy = \int_{q^i}^{\infty} \mu^{Q^i}(y)dy \tag{3.4}$$

Then, the crisp output of the fuzzy system is calculated using COG defuzzification as

$$y^{\text{crisp}} = \frac{\sum_{i=1}^{R} q_i \int \mu^{\hat{Q}^i}}{\sum_{i=1}^{R} \int \mu^{\hat{Q}^i}} \tag{3.5}$$

where $\int \mu^{\hat{Q}^i}$ is the area under $\mu^{\hat{Q}^i}$. The expression in (3.5) is a weighted average of the q_i's, $i = 1, \ldots, R$.

3.5.2 Center Average (CA) Defuzzification

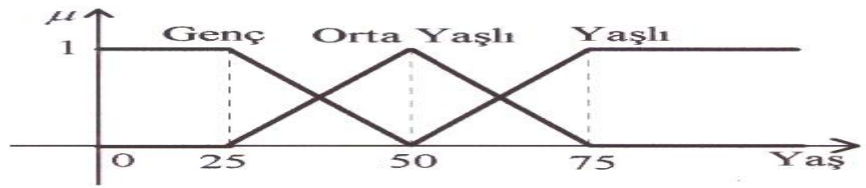
Suppose the consequent fuzzy set of Rule i is Q^i, characterized by membership $\mu^{Q^i}(y)$. Then the crisp output of the fuzzy system is calculated using CA defuzzification as

$$y^{\text{crisp}} = \frac{\sum_{i=1}^{R} q_i \max_y \left\{\mu^{Q^i}(y)\right\}}{\sum_{i=1}^{R} \max_y \left\{\mu^{Q^i}(y)\right\}} \tag{3.6}$$

Note that if Q^i is normal (it usually is), then $\max_y \left\{\mu^{Q^i}(y)\right\} = \mu_i(\underline{x})$ whether *min* or *prod* is used as the T-norm in (3.3). In that case, CA defuzzification gives

$$y^{\text{crisp}} = \frac{\sum_{i=1}^{R} q_i \mu_i(\underline{x})}{\sum_{i=1}^{R} \mu_i(\underline{x})} \tag{3.7}$$

Evrensel kümede birden fazla fuzzy küme mevcut ise bu durumda her kümeye ait bir üyelik fonksiyonu belirlenebilir,(Şekil 2.6).



Şekil 2.6. Evrensel kümede üyelik fonksiyonlarının belirlenmesi

Burada 'yaş' sözel bir değişkeni ifade eder. Evrensel kümeyi Y şeklinde gösterirsek bu kümeyi aşağıdaki gibi yazabiliriz.

Y (yaş)= { genç, orta yaşlı, yaşlı} (2.32)

Burada yaş değişkeni [0, 100] arasında değişir. "genç"., "orta yaşlı" ve "yaşlı" sözel ifadeler fuzzy kümelerin isimleridir. y tanım olarak yaş sözel değişkeni ise fuzzy alt kümeleri matematiksel olarak aşağıdaki gibi gösterilebilir.

$$\text{genç} = \int_0^{25} 1/y \; + \int_{25}^{50} [(50-y)/25]/y$$

$$\text{orta yaşlı} = \int_{25}^{50} [(y-25)/25]/y \; + \int_{50}^{75} [(75-y)/25]/y \qquad (2.33)$$

$$\text{yaşlı} = \int_{50}^{75} [(y-50)/25]/y \; + \int_{75}^{100} 1/y$$

2.5.2. Normal ve iç bükey üyelik fonksiyonu tanımı

Bir üyelik fonksiyonunda en büyük ağırlık 1 ise bu üyelik fonksiyonuna normal üyelik fonksiyonu denir.

$$\max_{x \in X} \mu_A(x) = 1 \qquad \text{(normal üyelik fonksiyonu)} \tag{2.34}$$

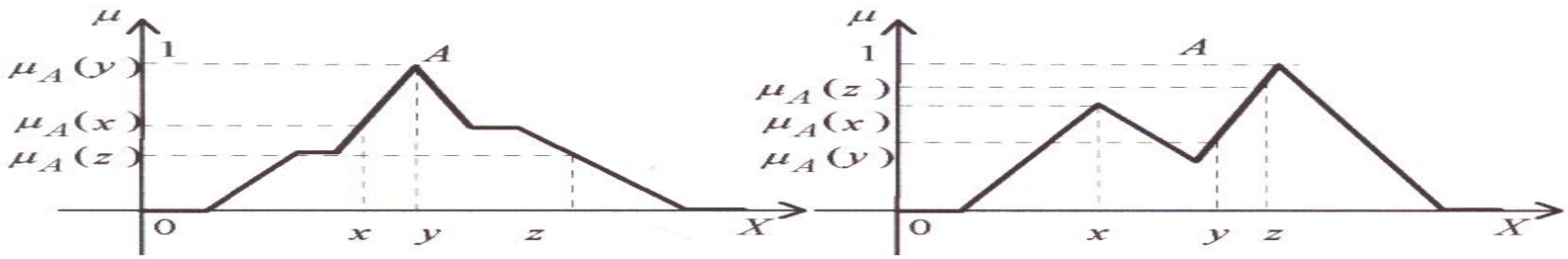
Eğer üyelik fonksiyonunda aşağıdaki tanımların biri geçerli ise bu üyelik fonksiyonu içbükey olur.

$$\mu_A(\lambda x_1 + (1-\lambda)x_2) \geq \min\left[\mu_A(x_1), \mu_A(x_2), x_1, x_2 \in U, \lambda \in [0,1]\right] \tag{2.35}$$

veya

$$\mu_A(y) \geq \min\left[\mu_A(x), \mu_A(z)\right] \qquad x < y < z \tag{2.36}$$

Bu durum şekil 2.7. 'de grafiksel olarak gösterilmektedir.

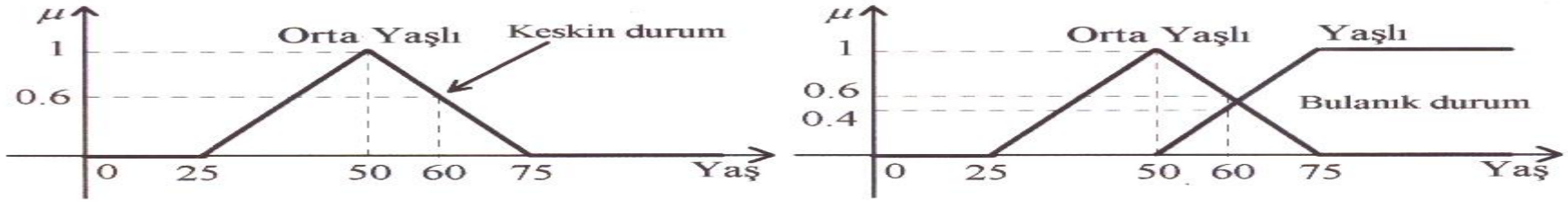


(a) İçbükey, normal (b) içbükey olmayan, normal

Şekil 2.7. İç bükey ve normal üyelik fonksiyonları

2.5.3. Bulanıklaştırma işlemi

Şekil 2.8.a 'da gösterilen üyelik fonksiyonunda yaş=65 için üyelik ağırlığı hesaplanırsa 0.6 olarak yanlız bir değer bulunur. Bu değer keskin değerdir.Çünkü bu şekilde yanlız "Orta Yaşlı" isimli bir



üyelik fonksiyonu vardır. Şayet Şekil 2.8.b. için aynı işlemler yapıldığında 0.6 ve 0.4 gibi iki üyelik ağırlık değeri bulunur. Çünkü bu şekil " Orta Yaşlı" ve "Yaşlı " isimli iki üyelik fonksiyonuna sahiptir. Dolayısıyla yapılan işlem bulanıklaştırmadır.

3. ÇIKARIM İŞLEMİ

Çıkarım işlemi Fuzzy Lojik Kontrol için en önemli kavramlardan birisidir. Algılayıcılardan alınan giriş bilgileri çıkarım işleminden geçirildikten sonra kontrol edilecek sisteme verilecek olan çıkış hesaplamak için kullanılır.

Fuzzy Lojik Kontrol (FLK) çıkarım sistemi genel olarak uzman sistemden daha kolaydır. Çünkü FLK önceki sonuçlardan bağımsız sadece sonraki olaylarla ilgilidir. FLK'nın kural tabanı uzman bilgiden yararlanılarak elde edilir. Kural tabanı Çok Girişli Çok Çıkışlı (ÇGÇÇ) sistem yapısına sahiptir,(Lee, 1990b) .

$$R = \{R^1_{\text{ÇGÇÇ}}, R^2_{\text{ÇGÇÇ}}, \ldots\ldots\ldots\ldots, R^n_{\text{ÇGÇÇ}}\} \qquad (3.1)$$

Burada $R^i_{\text{ÇGÇÇ}}$: if $(x = A_i \; and \ldots\ldots\ldots \text{and } y = B_i)$ then $(z_1 = C_i, \ldots\ldots\ldots, z_q = D_i)$ kuralını ifade eder. $Ux\ldots\ldots\ldots xV$ kartezyen çarpımdaki $A_i x\ldots\ldots\ldots xB_i$ bir fuzzy küme $R^i_{\text{ÇGÇÇ}}$ 'ın giriş kısmındaki ifadesini oluşturur. Sonuç q tane bağımsız kontrol olayının bileşimidir. *i*. $R^i_{\text{ÇGÇÇ}}$ bağıntısı bir bulanık anlam olarak ifade edilebilir.

$$R^i_{\text{ÇGÇÇ}} : (A_i x\ldots\ldots\ldots xB_i) \rightarrow (z_1 + \ldots\ldots\ldots + z_q) \qquad (3.2)$$

R bağıntısı aşağıdaki gibi bileşim işlemi olarak da ifade edilebilir.

$$R = \left\{ \bigcup_{i=1}^{n} R^i_{\text{ÇGÇÇ}} \right\}$$

$$= \left\{ \bigcup_{i=1}^{n} [(A_i x\ldots\ldots\ldots xB_i) \rightarrow (z_1 + \ldots\ldots\ldots + z_q)] \right\}$$

$$= \left\{ \bigcup_{i=1}^{n} [(A_i x \ldots\ldots\ldots xB_i) \rightarrow z_1], \bigcup_{i=1}^{n} [(A_i x \ldots\ldots\ldots xB_i) \rightarrow z_2], \ldots\ldots, \bigcup_{i=1}^{n} [(A_i x \ldots\ldots\ldots xB_i) \rightarrow z_q] \right\}$$

$$= \left\{ \bigcup_{k=1}^{q} \bigcup_{i=1}^{n} [(A_i x \ldots\ldots\ldots xB_i) \rightarrow z_k] \right\} \qquad (3.3)$$

$$R = \left\{ RB^1_{\text{ÇGTÇ}}, RB^2_{\text{ÇGTÇ}}, \ldots\ldots\ldots\ldots, RB^q_{\text{ÇGTÇ}} \right\}$$

Sonuçta FLK 'nın R bağıntısı Çok Girişli Tek Çıkışlı (ÇGTÇ) bağıntı ($RB^i_{\text{ÇGTÇ}}$) alt kural tabanının bir kümesi biçimine getirilmiş olur. Her bir $RB^i_{\text{ÇGTÇ}}$ alt kural tabanı bir kontrol çıkışı ve işlem giriş değişkenleri çarpımını içeren n tane fuzzy kontrol kuralından oluşur. ÇGÇÇ 'lı fuzzy sistemin kural yapısı ÇGTÇ 'lı bir sistem yapılarının toplamı olarak ifade edilebilir .

$$R = \left\{ RB^1_{\text{ÇGTÇ}}, RB^2_{\text{ÇGTÇ}}, \ldots\ldots\ldots\ldots, RB^q_{\text{ÇGTÇ}} \right\} \qquad (3.4)$$

Burada $R^k_{\text{ÇGTÇ}}$: if $(x = A_i$ and $\ldots\ldots\ldots$ and $y = B_i)$ then $(z_k = D_i)$ $i = 1, 2, \ldots\ldots, n$ ifadesini gösterir. ÇGTÇ 'lı fuzzy kontrol kurallarını iki girişli / tek çıkışlı fuzzy sistemde aşağıdaki gibi tanımlayalım.

giriş: $x = A'$ and $y = B'$
R_1: if $x = A_1$ and $y = B_1$ then $z = C_1$
also R_2: if $x = A_2$ and $y = B_2$ then $z = C_2$
$\ldots\ldots\ldots\ldots\ldots\ldots$
$\ldots\ldots\ldots\ldots\ldots\ldots$
also R_n: if $x = A_n$ and $y = B_n$ then $z = C_n$

$z = C'$ (3.5)

x ve y giriş, z çıkış değişkenidir. A_i, B_i ve C_i sırasıyla U, V ve W evrensel kümesinde x, y ve z sözel değişkenlerin sözel değerleridir.

Fuzzy lojik kontrol kuralı " if $(x = A_i$ and $y = B_i)$ then $(z = C_i)$ " R fuzzy bağıntı yerine geçer ve aşağıdaki gibi tanımlanır.

$$\mu_{R_i} \triangleq \mu_{(A_i \text{ and } B_i \to C_i)}(u, v, w) = [\mu_{A_i}(u) \text{ and } \mu_{B_i}(v)] \to \mu_{C_i}(w) \qquad (3.6)$$

Burada " A_i and B_i " UxV 'deki $A_i \times B_i$ fuzzy kümedir. $R_i \triangleq (A_i \text{ and } B_i) \to C_i$, $UxVxW$ 'daki bir fuzzy bağıntıdır ve "$\to$" sembolü bir fuzzy bağıntı kümesini ifade eder. C' sonucu " and " ve fuzzy bağıntıyı içeren çıkarım kurallarından max-min kompozisyon işlemi kullanılarak hesaplanır.

FLK çıkarım sisteminin bazı elverişli özellikleri vardır. Kuralların hepsi max-min ve " also " bağlayıcısı ile bir araya gelirler. Böylece fuzzy kontrol kümesinin hepsinden elde edilecek bir kontrol sonucu tek kontrol kuralından çıkan bir ağırlığa eşittir. Ayrıca aynı ağırlık sonucu sonradan anlatılacağı gibi max-çarpma kompoziyon işleminden de elde edilir. Aşağıdaki işlemlerin sonucu klasik lojik işlemlerinde geçerli değildir.

Teorem 1.: $(A', B') \circ \bigcup_{i=1}^{n} R_i = \bigcup_{i=1}^{n} (A', B') \circ R_i$ (3.7)

İspat :

$$C' = (A', B') \circ \bigcup_{i=1}^{n} R_i = (A', B') \circ \bigcup_{i=1}^{n} (A_i \text{ and } B_i \to C_i) \qquad (3.8)$$

Her $w \in W$ için C' fuzzy kümesinin $\mu_{C'}$ üyelik fonksiyonu belirlenir.

$$\mu_{C'}(w) = (\mu_{A'}(u), \mu_{B'}(v)) \circ \max_{u,v,w} \left(\mu_{R_1}(u,v,w), \mu_{R_2}(u,v,w), \ldots\ldots, \mu_{R_n}(u,v,w)\right)$$

$$= \max_{u,v} \left\{ \min \left[(\mu_{A'}(u), \mu_{B'}(v)) , \max_{u,v,w} \left(\mu_{R_1}(u,v,w), \mu_{R_2}(u,v,w), \ldots\ldots, \mu_{R_n}(u,v,w)\right) \right] \right\}$$

$$= \max_{u,v} \left\{ \max_{u,v,w} \left\{ \min \left[(\mu_{A'}(u), \mu_{B'}(v)) , \mu_{R_1}(u,v,w) \right], \ldots\ldots\ldots\ldots, \right. \right.$$

$$\min[(\mu_{A'}(u), \mu_{B'}(v)), \mu_{R_n}(u,v,w)]\}\} \tag{3.9}$$

$$= \max_{u,v,w}\{[(\mu_{A'}(u), \mu_{B'}(v)) \circ \mu_{R_1}(u,v,w)], \ldots\ldots\ldots, [(\mu_{A'}(u), \mu_{B'}(v)) \circ \mu_{R_n}(u,v,w)]\}$$

Bu durumda;

$$\begin{aligned} C' &= [(A', B') \circ R_1] \cup [(A', B') \circ R_2] \cup \ldots\ldots \cup [(A', B') \circ R_n] \\ &= \bigcup_{i=1}^{n} (A', B') \circ R_i \\ &= \bigcup_{i=1}^{n} (A', B') \circ (A_i \text{ and } B_i \rightarrow C_i) \\ &\triangleq \bigcup_{i=1}^{n} C_i' \end{aligned} \tag{3.10}$$

Teorem 2: R_c, R_p, R_{bp}, ve R_{dp} bağlayıcı ifadelerini bulmak için aşağı - daki işlemler . yazılabilir.

eğer $\mu_{A_i \times B_i} = \mu_{A_i} \wedge \mu_{B_i}$ ise;

$$(A', B') \circ (A_i \text{ and } B_i \rightarrow C_i) = [A' \circ (A_i \rightarrow C_i)] \cap [B' \circ (B_i \rightarrow C_i)] \tag{3.11}$$

eğer $\mu_{A_i \times B_i} = \mu_{A_i} \cdot \mu_{B_i}$ ise;

$$(A', B') \circ (A_i \text{ and } B_i \rightarrow C_i) = [A' \circ (A_i \rightarrow C_i)] \cdot [B' \circ (B_i \rightarrow C_i)]$$

İspat:

$$\begin{aligned} C_i' &= (A', B') \circ (A_i \text{ and } B_i \rightarrow C_i) \\ \mu_{C_i'} &= (\mu_{A'}, \mu_{B'}) \circ (\mu_{A_i \times B_i} \rightarrow \mu_{C_i}) \\ &= (\mu_{A'}, \mu_{B'}) \circ [\min(\mu_{A_i}, \mu_{B_i}) \rightarrow \mu_{C_i}] \\ &= (\mu_{A'}, \mu_{B'}) \circ \min[(\mu_{A_i} \rightarrow \mu_{C_i}), (\mu_{B_i} \rightarrow \mu_{C_i})] \\ &= \max_{u,v}\{\min[(\mu_{A'}, \mu_{B'}), \min[(\mu_{A_i} \rightarrow \mu_{C_i}), (\mu_{B_i} \rightarrow \mu_{C_i})]]\} \end{aligned} \tag{3.12}$$

$$= \max_{u,v}\left\{\min\left[\min\left[\mu_{A'}, \left(\mu_{A_i} \rightarrow \mu_{C_i}\right)\right], \min\left[\mu_{B'}, \left(\mu_{B_i} \rightarrow \mu_{C_i}\right)\right]\right]\right\}$$

$$= \min\left\{\left[\mu_{A'} \circ \left(\mu_{A_i} \rightarrow \mu_{C_i}\right)\right], \left[\mu_{B} \circ \left(\mu_{B_i} \rightarrow \mu_{C_i}\right)\right]\right\}$$

Böylece;

$$C_i' = \left[A' \circ (A_i \rightarrow C_i)\right] \cap \left[B' \circ (B_i \rightarrow C_i)\right] \tag{3.13}$$

ifadesi elde edilmiş olur.

Aşağıdaki iki özel teorem FLK ' nın anlaşılmasında önemli iki durumu ifade eder.

Teorem 3: Eğer $A' = u_0$, $B' = v_0$ olan teknokta girişleri ise R_c kuralı "min" işlemini ve R_p kuralıda çarpma işlemini gösterir ve aşağıdaki gibi ifade edilirler.

$$\begin{aligned} &1)\ R_c : \alpha_i^{\wedge} \wedge \mu_{C_i}(w) && 2)\ R_c : \alpha_i^{\cdot} \wedge \mu_{C_i}(w) \\ &\quad R_p : \alpha_i^{\wedge} \cdot \mu_{C_i}(w) && \quad R_p : \alpha_i^{\cdot} \cdot \mu_{C_i}(w) \end{aligned} \tag{3.14}$$

Burada $\alpha_i^{\wedge} = \mu_{A_i}(u_0) \wedge \mu_{B_i}(v_0)$ ve $\alpha_i^{\cdot} = \mu_{A_i}(u_0) \cdot \mu_{B_i}(v_0)$ ifadelerine eşittir.

İspat:

1) $C' = \left[A' \circ (A_i \rightarrow C_i)\right] \cap \left[B' \circ (B_i \rightarrow C_i)\right]$

$$\begin{aligned} \mu_{C_i'} &= \min\left\{\left[u_0 \circ \left(\mu_{A_i}(u) \rightarrow \mu_{C_i}(w)\right)\right], \left[v_0 \circ \left(\mu_{B_i}(v) \rightarrow \mu_{C_i}(w)\right)\right]\right\} \\ &= \min\left\{\left[\left(\mu_{A_i}(u_0) \rightarrow \mu_{C_i}(w)\right)\right], \left[\left(\mu_{B_i}(v_0) \rightarrow \mu_{C_i}(w)\right)\right]\right\} \end{aligned} \tag{3.15}$$

2) $C_i' = \left[A' \circ (A_i \rightarrow C_i)\right] \cdot \left[B' \circ (B_i \rightarrow C_i)\right]$

$$\begin{aligned} \mu_{C_i'} &= \left[u_0 \circ \left(\mu_{A_i}(u) \rightarrow \mu_{C_i}(w)\right)\right] \cdot \left[v_0 \circ \left(\mu_{B_i}(v) \rightarrow \mu_{C_i}(w)\right)\right] \\ &= \left[\left(\mu_{A_i}(u_0) \rightarrow \mu_{C_i}(w)\right)\right] \cdot \left[\left(\mu_{B_i}(v_0) \rightarrow \mu_{C_i}(w)\right)\right] \end{aligned} \tag{3.16}$$

Aşağıda anlatılacağı gibi son teorem hesaplama işlemlerini kolaylaştırdığı gibi FLK 'de fuzzy çıkarım işleminin grafik yoldan yapılmasınıda sağlar. max-çarpma işlemine tekrar bakılacak olursa;

Teorem 1': $(A', B') \cdot \bigcup_{i=1}^{n} R_i = \bigcup_{i=1}^{n} (A', B') \cdot R_i$ (3.17)

Teorem 2': R_c, R_p, R_{bp}, ve R_{dp} bağlayıcı ifadelerini bulmak için aşağıdaki işlemler . yazılabilir.

Eğer $\mu_{A_i \times B_i} = \mu_{A_i} \wedge \mu_{B_i}$ ise;

$$(A', B') \cdot (A_i \text{ and } B_i \rightarrow C_i) = [A' \cdot (A_i \rightarrow C_i)] \bigcap [B' \cdot (B_i \rightarrow C_i)] \quad (3.18)$$

Eğer $\mu_{A_i \times B_i} = \mu_{A_i} \cdot \mu_{B_i}$ ise;

$$(A', B') \cdot (A_i \text{ and } B_i \rightarrow C_i) = [A' \cdot (A_i \rightarrow C_i)] \cdot [B' \cdot (B_i \rightarrow C_i)] \quad (3.19)$$

Teorem 3': Eğer $A' = u_0$, $B' = v_0$ olan teknokta girişleri ise R_c kuralı "min" işlemini ve R_p kuralıda çarpma işlemini gösterir ve aşağıdaki gibi ifade edilirler.

$$\begin{array}{ll} 1)\ R_c : \alpha_i^{\wedge} \wedge \mu_{C_i}(w) & 2)\ R_c : \alpha_i \wedge \mu_{C_i}(w) \\ \quad R_p : \alpha_i^{\wedge} \cdot \mu_{C_i}(w) & \quad R_p : \alpha_i \cdot \mu_{C_i}(w) \end{array} \quad (3.20)$$

Burada $\alpha_i^{\wedge} = \mu_{A_i}(u_0) \wedge \mu_{B_i}(v_0)$ ve $\alpha_i = \mu_{A_i}(u_0) \cdot \mu_{B_i}(v_0)$ ifadelerine eşittir..

Böylece en son durum aşağıdaki gibi olur.

$$\begin{array}{l} R_c : \mu_{C'} = \bigcup_{i=1}^{n} \alpha_i \wedge \mu_{C_i} \\ R_p : \mu_{C'} = \bigcup_{i=1}^{n} \alpha_i \cdot \mu_{C_i} \end{array} \quad (3.21)$$

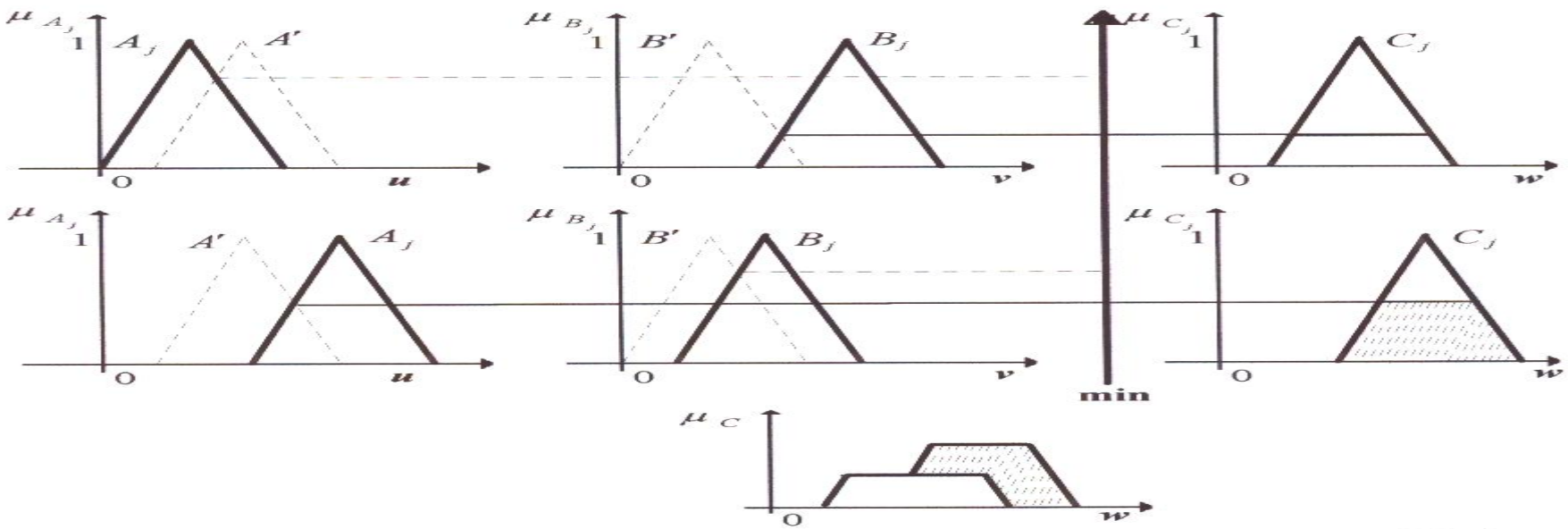
Burada α_i ağırlık faktörü i. kuralda fuzzy kontrol işleminin bir ölçüsüdür. Söz konusu ağırlık faktörü iki yöntem belirlenir. Birinci yöntem FLK uygulamalarında geniş ölçüde kullanılan kartezyen çarpımda bulunan "min" işlemi ve ikinci yöntem kartezyen çarpımdaki

cebirsel çarpma işlemidir.Giriş işlemlerindeki kolaylığı sağlamak bakımından birinci yöntem ikinci yöntemden daha etkindir. Bu yüzden uygulamalarda birinci yöntem daha çok kullanılmaktadır.

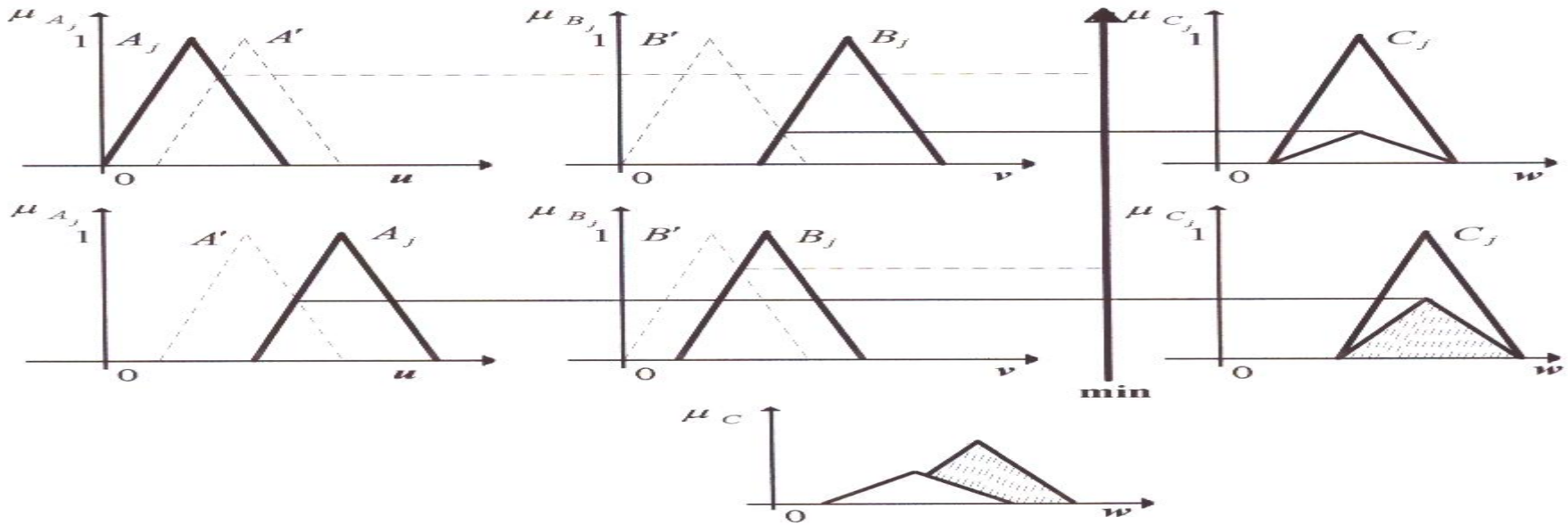
Basit olarak iki fuzzy kontrol kuralının aşağıdaki gibi kabul edildiğini kabul edersek ;

$$\begin{aligned} R_1&: \quad \text{if } x = A_1 \text{ and } y = B_1 \text{ then } z = C_1 \\ R_2&: \quad \text{if } x = A_2 \text{ and } y = B_2 \text{ then } z = C_2 \end{aligned} \tag{3.22}$$

Şekil 3.1. R_c ve α_i^ şartları altında Teorem 2 'nin grafik yorumunu gösterir. Şekil 3.2. R_p ve α_i' şartları altında Teorem 2 'nin grafik yorumunu gösterir.



Şekil 3.1. R_c ve α^ şartları için Teorem 2 'nin grafik gösterilimi



Şekil 3.2. R_p ve α' şartları için Teorem 2 'nin grafik gösterilimi

Girişler algılayıcılar tarafından ölçülür ve keskin değere sahiptirler. Giriş değerinin ölçekleme vasıtasıyla fuzzy kümelerdeki giriş bilgisine çevrilmesi daha uygun olur. Aynı zamanda bu değerler fuzzy teknoktaya karşılık gelirler. Birinci ve ikinci kuralların α_1 ve α_2 ağırlık faktörleri aşağıdaki gibi açıklanır.

$$\alpha_1 = \mu_{A_1}(x_0) \wedge \mu_{B_1}(y_0)$$
$$\alpha_2 = \mu_{A_2}(x_0) \wedge \mu_{B_2}(y_0) \qquad (3.23)$$

Burada $\mu_{A_1}(x_0)$ ve $\mu_{B_1}(y_0)$ kural tabanındaki bilgi ile kaynak bilgisi arasında aynı derecede etkiye sahiptir. Bu bağıntılar dört çeşit fuzzy karar vermede önem taşırlar.

3.1. Birinci tip fuzzy karar verme ; Fuzzy anlam ifadesi olarak "min" işlemi

Bu tür karar vermede *i*. kural kontrol kararına yön gösterir.

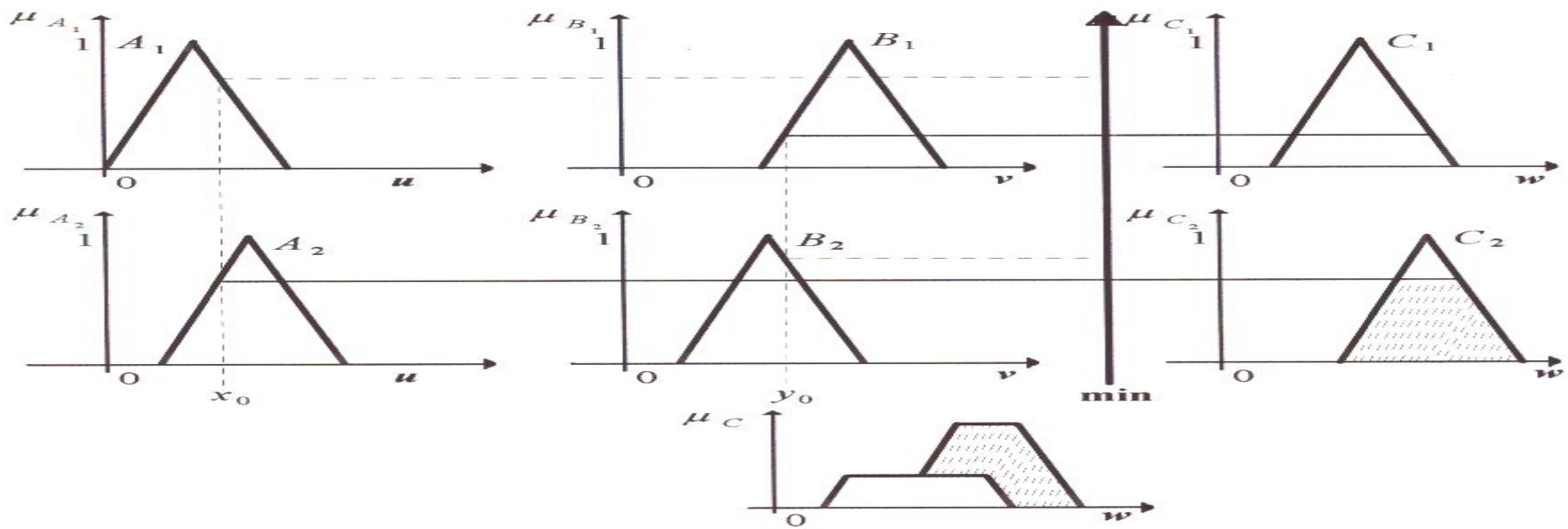
$$\mu_{C_i'}(w) = \alpha_i \wedge \mu_{C_i}(w) \tag{3.24}$$

Çıkarımı yapılan μ_C üyelik fonksiyonu aşağıdaki gibi hesaplanır.

$$\mu_C(w) = \mu_{C_1'} \vee \mu_{C_2'} \tag{3.25}$$

$$\mu_C(w) = [\alpha_1 \wedge \mu_{C_1}(w)] \vee [\alpha_2 \wedge \mu_{C_2}(w)]$$

Şekil 3.3. fuzzy karar verme işleminde R_c yontemi yoluyla teorem 3'ün grafiksel yorumunu gösterir.İleride anlatılacağı gibi kontrol değerini belirlemek için netleştirme işlemcilerinden herhangi biri kullanılır.



Şekil 3.3. Fuzzy karar verme 1 işleminin grafiksel gösterilimi

3.2. İkinci tip fuzzy karar verme; Fuzzy anlam ifadesi olarak "çarpma" işlemi

Bu tür karar vermede *i*. kural kontrol kararına yön gösterir.

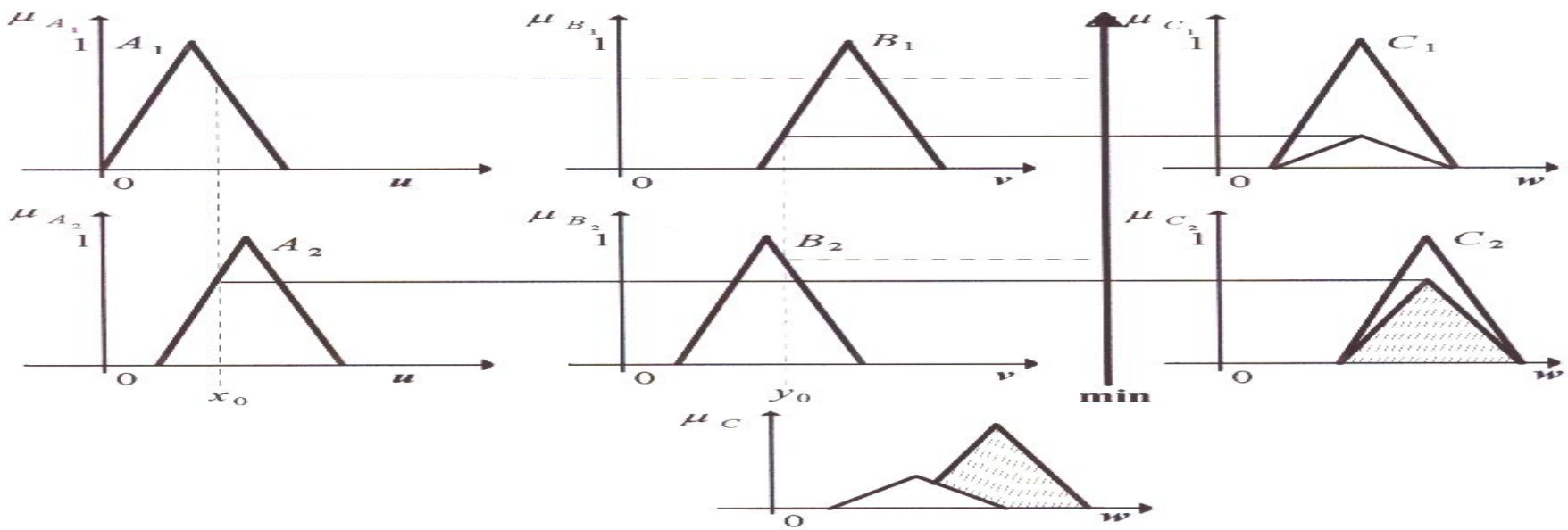
$$\mu_{C_i'}(w) = \alpha_i \cdot \mu_{C_i}(w) \tag{3.26}$$

Çıkarımı yapılan μ_C üyelik fonksiyonu aşağıdaki gibi hesaplanır.

$$\mu_C(w) = \mu_{C_1'} \vee \mu_{C_2'} \tag{3.27}$$

$$\mu_C(w) = [\alpha_1 \cdot \mu_{C_1}(w)] \vee [\alpha_2 \cdot \mu_{C_2}(w)]$$

Şekil 3.4. fuzzy karar verme işleminde R_p yöntemi yoluyla teorem 3'ün bir grafiksel yorumunu gösterir.



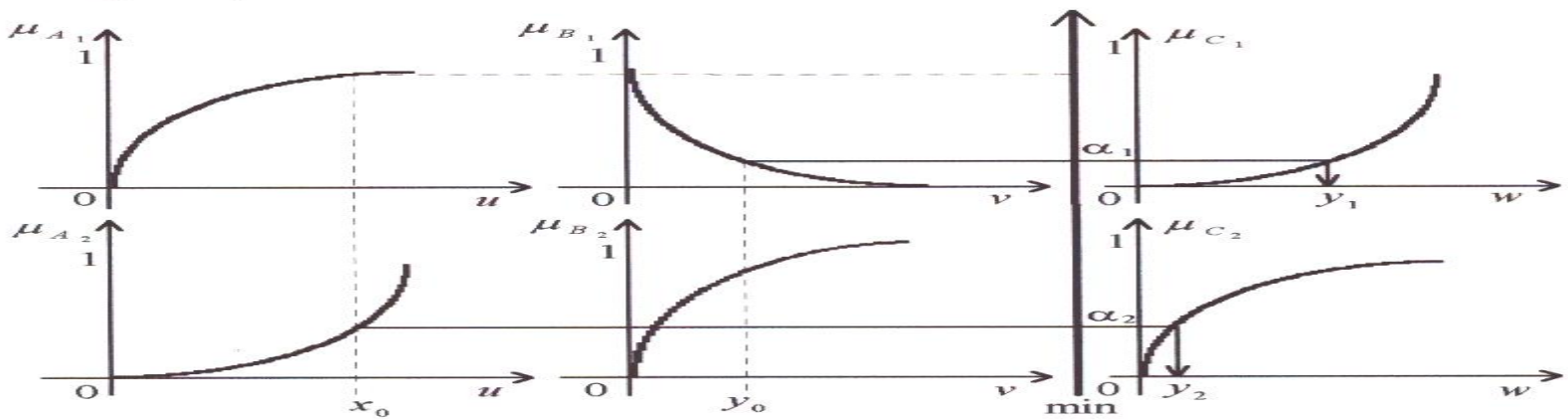
Şekil 3.4. Fuzzy karar verme 2 işleminin grafiksel gösterilimi

3.3. Üçüncü tip fuzzy karar verme ; Monotonik üyelik fonksiyonu olarak sözel terimli metod

A_i, B_i ve C_i fuzzy kümelerin üyelik fonksiyonu monotonik olduğu birinci tip karar verme ile aynı yolu kullanan basit bir yöntemdir. Bununla beraber işlemlerde A_i, B_i monotonik olmayabilir, fakat C_i monotonik olması gerekir.

Bu yöntemde birinci kuraldan elde edilen sonuç $\alpha_1 = C_1(y_1)$ ifadesindeki α_1'e ikinci kuraldan elde edilen sonuç $\alpha_2 = C_2(y_2)$ ifadesindeki α_2'ye eşittir. Keskin kontrol sonuç değeri aşağıdaki gibi ağırlıkların kombinasyonu olarak alınır.(Şekil 4.5)

$$z_0 = \frac{\alpha_1 y_1 + \alpha_2 y_2}{\alpha_1 + \alpha_2} \qquad (3.28)$$



Şekil 3.5. Fuzzy karar verme 3 işleminin grafiksel gösterilimi

3.4. Dördüncü tip fuzzy karar verme; Kuralın sonucu sözel giriş değişkenlerinin bir fonksiyonu

Dördüncü tip fuzzy karar verme sonuç durumu eşitlik fonksiyonu ile değiştirilmiş bir şeklidir.Karar vermenin bu metodunda *i*. kontrol kuralı aşağıdaki yapıdadır.

$$R_i : \text{if } (x = A_i \text{ and } \ldots\ldots\ldots \text{and } y = B_i) \text{ then } z = f_i(x,\ldots,y) \qquad (3.29)$$

Burada sırasıyla $x,\ldots,y$ giriş değişkenlerini ve z kontrol çıkış değerini gösteren sözel değişkenlerdir. $A_i,\ldots\ldots\ldots,B_i$ $U,\ldots\ldots\ldots\ldots,V$ evrensel kümesindeki $x,\ldots,y$ sözel değişkenlerinin sözel değerleridir. f_i ve $i = 1, 2,\ldots\ldots\ldots,n$ giriş alt uzayında tanımlanan $x,\ldots,y$ giriş değişkenlerinin bir fonksiyonudur.

Basit olarak çıkış değerini hesaplamak için aşağıdaki iki kuralı ele alalım ;

$$\begin{aligned} R_1&: \text{ if } x = A_1 \text{ and } y = B_1 \text{ then } z = f_1(x,y) \\ R_2&: \text{ if } x = A_2 \text{ and } y = B_2 \text{ then } z = f_2(x,y) \end{aligned} \qquad (3.30)$$

Birinci kuraldan kontrol olayının çıkarılan sonuç değeri $\alpha_1 f_1(x_0, y_0)$ ve ikinci kuraldan kontrol olayının çıkarılan sonuç değeri $\alpha_2 f_2(x_0, y_0)$ 'dır. Keskin çıkış kontrol sonuç değeri aşağıdaki ifade ile hesaplanır.

$$z_0 = \frac{\alpha_1 f_1(x_0, y_0) + \alpha_2 f_2(x_0, y_0)}{\alpha_1 + \alpha_2} \qquad (3.31)$$

Bu yöntem Tagaki ve Sugeno tarafından geliştirilmiş ve bir model arabayı kullanıcı olmaksızın uygun bir şekilde graja park etme işlemine kullanmışlardır, (Sugeno and Murakami, 1985).

3.5.Çıkarım işleminin bir uygulama üzerinde grafiksel ve cebirsel anlatımı

3.5.1. Grafiksel anlatım

FLD 'nin giriş değişkenleri hata (e), hatanın değişimi (ce) ve çıkış değişkeni (u) alınmıştır (Şekil3.6). Evrensel kümeler e, ce ve u sözel değişkenleri temsil eder. Bu duruma göre aşağıdaki kurallar isteğe göre belirlenmiş olsun, (Li and Lau, 1989).

$$\begin{aligned}
&\text{if } e = \text{SF} \text{ and } ce = \text{KP} \text{ then } u = \text{KN}\\
&\text{if } e = \text{SF} \text{ and } ce = \text{SF} \text{ then } u = \text{SF}\\
&\text{if } e = \text{KN} \text{ and } ce = \text{KN} \text{ then } u = \text{KP}\\
&\text{if } e = \text{KN} \text{ and } ce = \text{SF} \text{ then } u = \text{BP}
\end{aligned} \qquad (3.32)$$

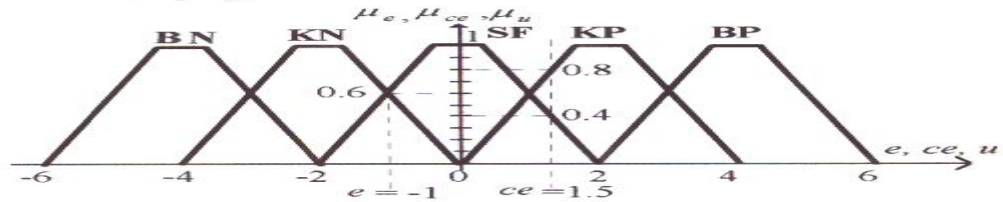
Şekil 3.6. FLK 'nin giriş ve çıkış değişkenleri

Burada e, ce ve u sözel değişkenli üç evrensel küme her biri fuzzy alt küme olan sözel etiketleri içerir.

- Büyük Pozitif (BP)
- Küçük Pozitif (KP)
- Sıfır (SF)
- Küçük Negatif (KN)
- Büyük Negatif (BN)

Bu sözel etiketlere ilişkin üyelik fonksiyonları şekil 3.7 'de gösterilmiştir. Bu şekilde üyelik fonksiyonları e, ce ve u sözel değişkenlerinin hepsi için aynı seçilmiştir. e, ce ve u sözel

değişkenleri için kontrol edilecek sistemin durumuna göre birbirinden farklı üyelik fonksiyonları ve başka sözel etiketler seçilebilir.

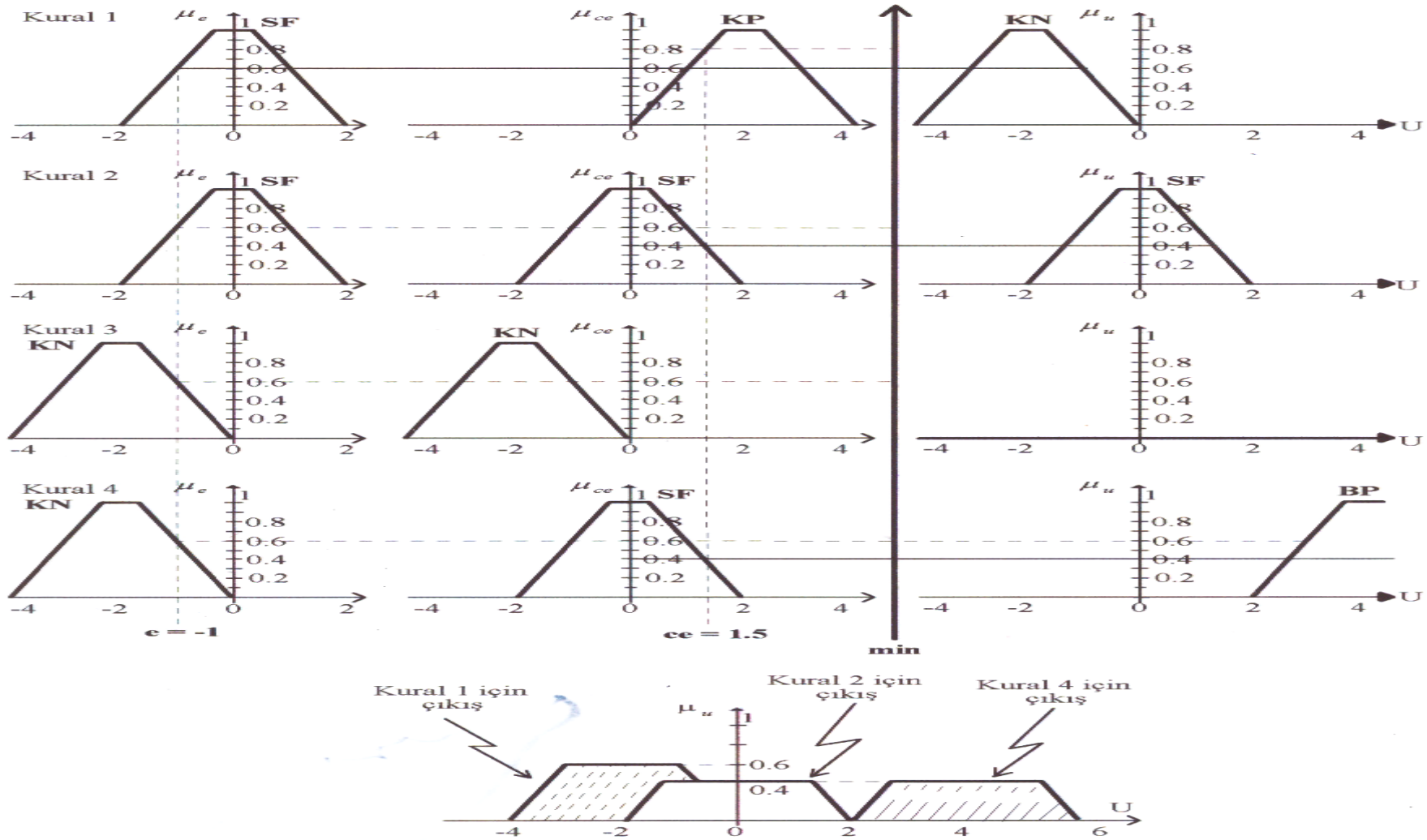


Şekil 3.7. *e*, *ce* ve *u* için üyelik fonksiyonları.

Tablo 3.1. Kuralların gösterilimi

e \ ce	BN	KN	SF	KP	BP
BN	X	X	X	X	X
KN	X	KP	BP	X	X
SF	X	X	SF	KN	X
KP	X	X	X	X	X
BP	X	X	X	X	X

Başta verilen kurallar tablo 3.1 'de gösterilmiştir. Bu tabloda *e* için BN, KP ve BP, *ce* için BN, BP ve *u* için BN sözel etiketli fuzzy alt kümeler kullanılmamıştır. Eğer sistemi daha iyi kontrol edebilmek için yeni kurallar ilave edilmesi istenilirse duruma göre bu sözel etiketler kullanılabilir.



Şekil 3.8. Bulanıklaştırma işlemi ve çıkış grafiğinin bulunması

Şekil 3.7 'de görüldüğü gibi $e = -1$ ve $ce = 1.5$ ise e için KN ve SF sözel etiketli ce için SF ve KP sözel etiketli fuzzy alt kümelerde bir ağırlığa sahip olurlar. Bu durumda e ve ce evrensel kümelerindeki fuzzy alt kümeleri arasında oluşturulacak kartezyen çarpımlar aşağıdaki gibi olur.

$$\begin{aligned} &1.\ e = \text{KN} \quad ce = \text{SF} \\ &2.\ e = \text{KN} \quad ce = \text{KP} \\ &3.\ e = \text{SF} \quad ce = \text{KP} \\ &4.\ e = \text{SF} \quad ce = \text{SF} \end{aligned} \tag{3.33}$$

Fakat yukarıdaki 1, 3, 4 nolu durumlar daha önce yazılan kurallarda mevcut durumdadır. 1 nolu durum kural 4 'e, 3 nolu durum kural 1 'e ve 4 nolu durum kural 2 'ye karşılık gelir. Bu kurallara ilişkin bulanıklaştırma işlemi ve şonuçta çıkışın bulunması şekil 3.8 'de grafiksel olarak gösterilmiştir.

Yukarıda elde edilen çıkış grafiğine bölüm 4 'de anlatılan netleştirme yöntemlerinden biri uygulanarak kontrol edilecek sisteme verilecek olan keskin sonuç hesaplanmış olur.

3.5.2.Cebirsel anlatım

Şekil 3.7 'de gösterildiği gibi e, ce ve u;

$$\begin{aligned} e &= \{\text{BP, KP, SF, KN, BN}\} \\ ce &= \{\text{BP, KP, SF, KN, BN}\} \\ u &= \{\text{BP, KP, SF, KN, BN}\} \end{aligned} \tag{3.34}$$

fuzzy alt kümeleri kapsarlar. Daha önce anlatılan kurallara ait fuzzy bağıntılar aşağıdaki gibi yazılabilir.

$$
\begin{aligned}
R_1 &= [\mu_{SF}(e) \wedge \mu_{KP}(ce)] \circ \mu_{KN}(u) \\
R_2 &= [\mu_{SF}(e) \wedge \mu_{SF}(ce)] \circ \mu_{SF}(u) \\
R_3 &= [\mu_{KN}(e) \wedge \mu_{KN}(ce)] \circ \mu_{KP}(u) \\
R_4 &= [\mu_{KN}(e) \wedge \mu_{SF}(ce)] \circ \mu_{BP}(u)
\end{aligned}
\qquad (3.35)
$$

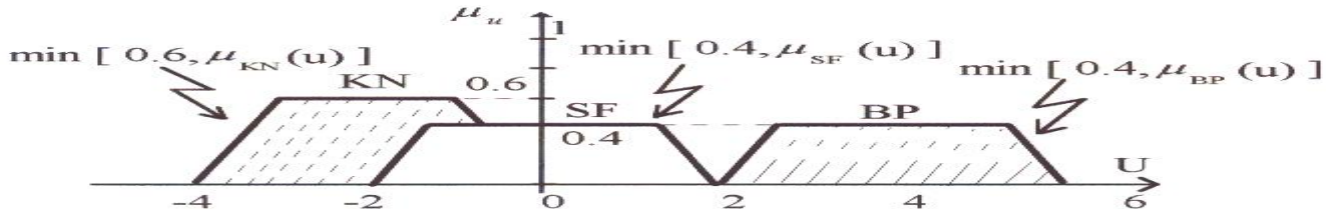
Fuzzy bağıntı tanımındaki max-min kompozisyonu kullanılarak R_1, R_2, R_3 ve R_4 bağıntıları aşağıdaki duruma getirilir.

$$
\begin{aligned}
R_1(e, ce, u) &= \max\{ \min[\min[\mu_{SF}(e), \mu_{KP}(ce)], \mu_{KN}(u)]\} \\
R_2(e, ce, u) &= \max\{ \min[\min[\mu_{SF}(e), \mu_{SF}(ce)], \mu_{SF}(u)]\} \\
R_3(e, ce, u) &= \max\{ \min[\min[\mu_{KN}(e), \mu_{KN}(ce)], \mu_{KP}(u)]\} \\
R_4(e, ce, u) &= \max\{ \min[\min[\mu_{KN}(e), \mu_{SF}(ce)], \mu_{BP}(u)]\}
\end{aligned}
\qquad (3.36)
$$

$e = -1$ ve $ce = 1.5$ için şekil 3.7 'den faydalanarak her fuzzy alt kümede bu değerler ilişkin aşağıdaki ağırlıklar bulunur.

$$
\begin{aligned}
\mu_{BN}(e)|_{e=-1} &= 0 & \mu_{BN}(ce)|_{ce=1.5} &= 0 \\
\mu_{KN}(e)|_{e=-1} &= 0.6 & \mu_{KN}(ce)|_{ce=1.5} &= 0 \\
\mu_{SF}(e)|_{e=-1} &= 0.6 & \mu_{SF}(ce)|_{ce=1.5} &= 0.4 \\
\mu_{KP}(e)|_{e=-1} &= 0 & \mu_{KP}(ce)|_{ce=1.5} &= 0.8 \\
\mu_{BP}(e)|_{e=-1} &= 0 & \mu_{BP}(ce)|_{ce=1.5} &= 0
\end{aligned}
\qquad (3.37)
$$

Bu ağırlıklara göre R_1, R_2, R_3 ve R_4 fuzzy bağıntıları aşağıdaki duruma gelirler.



Şekil 3.9 Çıkış ifadesi

4. NETLEŞTİRME YÖNTEMLERİ

DEFUZZIFACITION METHODS

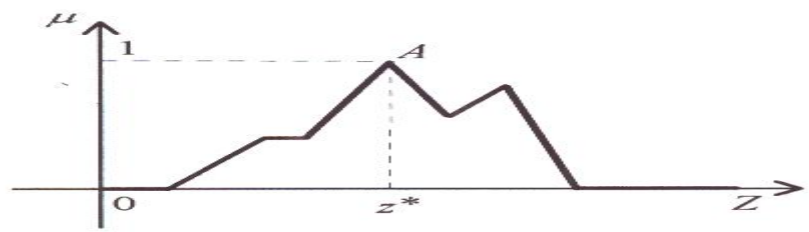
Yedi ayrı netleştirme yöntemi vardır. Bunlar aşağıda grafiksel ve cebirsel olarak anlatılmıştır, (Ross, 1995).

4.1. Maksimum-üyelik yöntemi

Maximum – Membership Methods

Yükseklik metodu olarak da bilinir. Bu yöntemde üyelik fonksiyonunda en büyük ağırlık keskin sonuç olarak alınır (Şekil 4.1). Aşağıdaki gibi ifade edilir.

$$\mu_c(z^*) \geq \mu_c(z) \quad \forall z \in Z \tag{4.1}$$



Şekil 4.1. Maks-Üyelik yöntemi

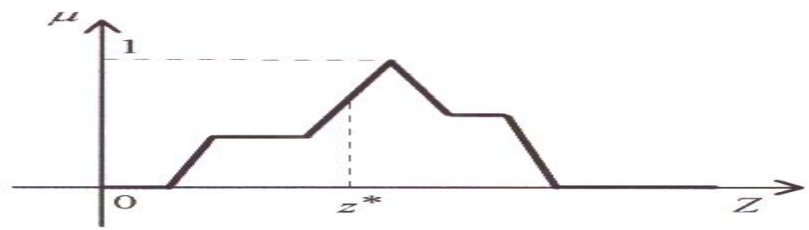
4.2. Merkezi yöntem

Central Methods

Bu yöntem ağırlık merkezini bulma veya alan merkezini bulma olarak adlandırılır. En çok kullanılan netleştirme yöntemlerin birisidir. Bu yöntem aşağıdaki gibi ifade edilir.

$$z^* = \frac{\int \mu_c(z).z.dz}{\int \mu_c(z).dz} \qquad (4.2)$$

Burada $\int$ cebirsel integrali gösterir. Bu yöntem şekil 4.2. 'de gösterilmiştir.



Şekil 4.2. Merkezi yöntem

Finding the centre of Gravity Methods

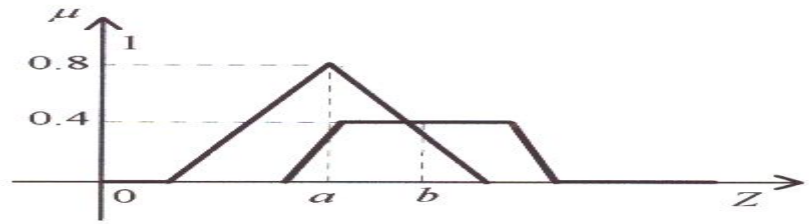
4.3. Ağırlık ortalamasını bulma yöntemi

Bu yöntem sadece çıkış grafiğinin simetrik olması durumunda uygulanır. Aşağıdaki gibi gösterilir.

$$z^* = \frac{\sum \mu_c(\bar{z}).\bar{z}}{\sum \mu_c(\bar{z})} \qquad (4.3)$$

$\sum$ cebirsel toplam sembolünü gösterir. Şekil 4.3 'de gösterildiği gibi çıkıştaki üyelik fonksiyonlarından en büyük ağırlıklar alınır ve aşağıdaki işlem yapılarak netleşmiş sonuç bulunur.

$$z^* = \frac{a(0.8) + b(0.4)}{0.8 + 0.4} \qquad (4.4)$$

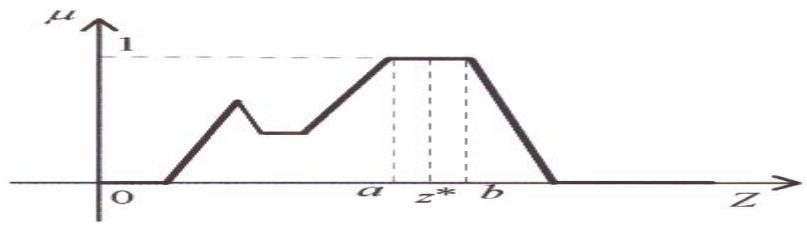


Şekil 4.3. Ağırlık ortalamasını bulma yöntemi

4.4. Maksimum üyelikleri ortalama yöntemi Average of Maximum Membership Method

Bu yöntem maksimum üyelik noktalarını ortalama yöntemi olarakda bilinir. Maksimum-üyelik yöntemine benzerdir. Farklı olarak maksimum noktaların evrensel kümedeki değerlerinin ortalaması alınır. Şekil 4.4 için netleşmiş sonuç aşağıdaki gibi bulunur.

$$z^* = \frac{a+b}{2} \qquad (4.5)$$



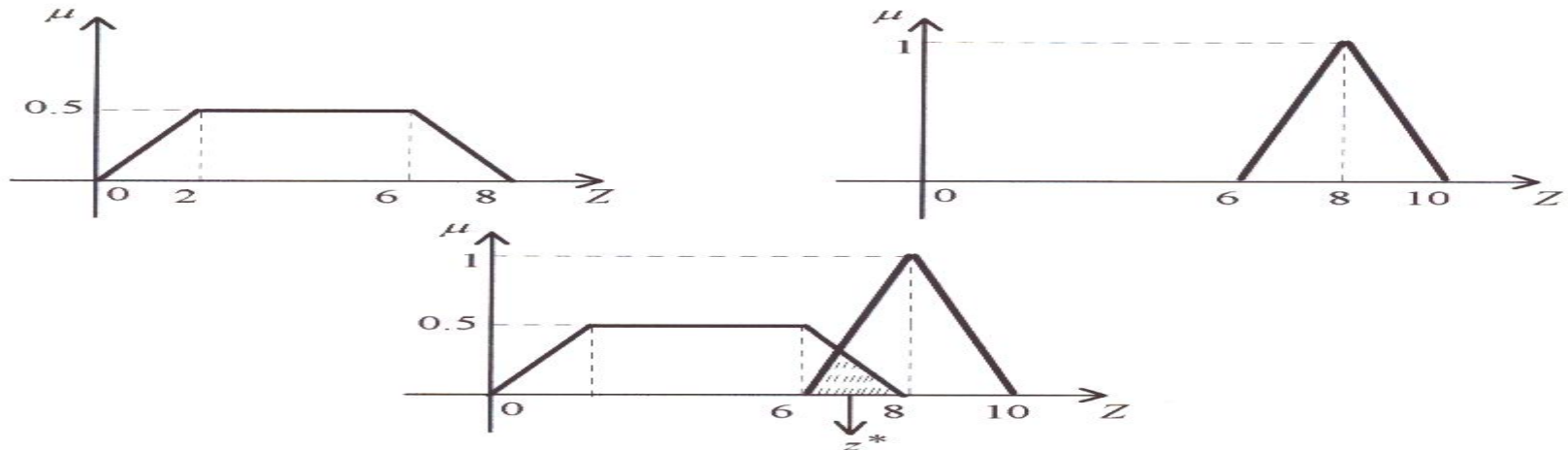
Şekil 4.4.Maksimum üyelikleri ortalama yöntemi

4.5. Toplamların merkezini bulma yöntemi Finding the centroid of totals Method

Diğer hesabı uzun olan netleştirme yöntemlerinden daha çabuk sonucun bulunmasını sağlar. Şekil 4.5. 'de gösterildiği gibi çıkıştaki fuzzy küme C_1 ve C_2 üyelik fonksiyonlarının bileşimi şeklindedir. Netleşmiş sonuç aşağıdaki gibi hesaplanır.

$$z^* = \frac{\int_z z \sum_{k=1}^{n} \mu_{c_k}(z).z.dz}{\int_z \sum_{k=1}^{n} \mu_{c_k}(z).dz} \qquad (4.6)$$

Bu yöntem ağırlıkları ortalama yöntemine benzer. Yanlız bu yöntemde sonuçtaki üyelik fonsiyonlarının merkezdeki ağırlıkları yerine bu üyelik fonksiyonlarının alanları alınarak hesap yapılır.



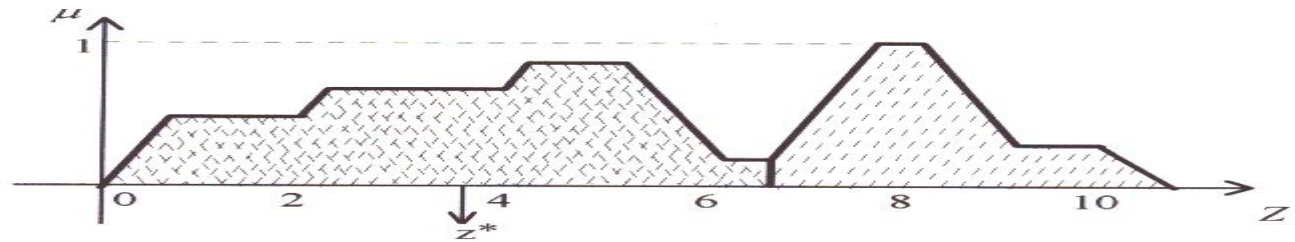
Şekil 4.5.Toplamların merkezini bulma yöntemi

4.6. En büyük alan merkezini bulma yöntemi Finding the Biggest Central of area Methode

Eğer çıkışta birden fazla içbükey bölge var ise bu içbükey bölgelerden en büyük alana sahip olan alınır. Netleşmiş z^* çıkış değeri hesaplamak için bu bölgeye daha önce anlatılan merkezi bulma yöntemi uygulanır. Bu yöntem grafik olarak şekil 4.6 'da gösterilir ve cebirsel olarak aşağıdaki gibi hesaplanır.

$$z^* = \frac{\int \mu_{c_m}(z).z.dz}{\int \mu_{c_m}(z).dz} \qquad (4.7)$$

Burada C_m, C_k 'daki en büyük içbükey alanı sembolize eder.



Şekil 4.6.En büyük alan merkezini bulma yöntemi

Finding the First and last maximum values method

4.7. İlk veya son maksimum değeri bulma yöntemi

Bu yöntemde çeşitli üyelik fonksiyonlarının bileşimi olan C_k 'nın çıkış evrensel kümesinde ağırlığı büyük olan en yakın değer veya ağırlığı büyük olan en son değer alınır.Aşağıdaki gibi ifade edilir.

İlk maksimum değerini alma yöntemi;

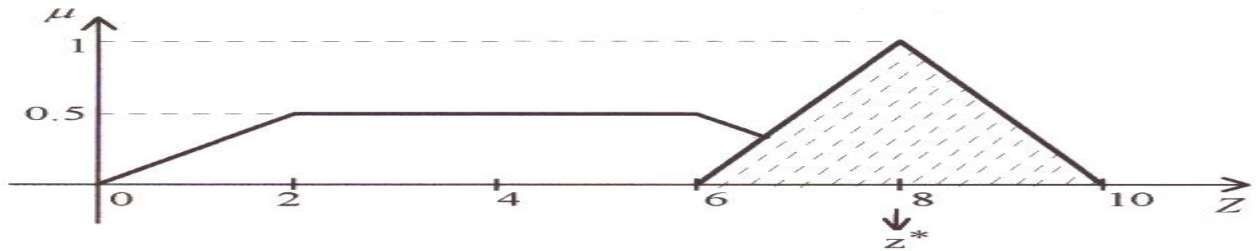
$$z^* = \inf_{z \in Z}\left\{z \in Z \mid \mu_{C_k} = hgt(C_k)\right\} \qquad (4.8)$$

şeklinde ve son maksimum değerini alma yöntemi;

$$z^* = \sup_{z \in Z}\left\{z \in Z \mid \mu_{C_k} = hgt(C_k)\right\} \qquad (4.9)$$

şeklinde gösterilir.

Burada infumum (inf) evrensel kümede ağırlığı en büyük olan ilk değeri ve subremum (sub) ağırlığı en büyük olan en son değeri gösterir.



Şekil 4.7.İlk veya son maksimum değeri bulma yöntemi

5. FUZZY LOJİK KONTROL

5.1. Temel Fuzzy Lojik Kontrol

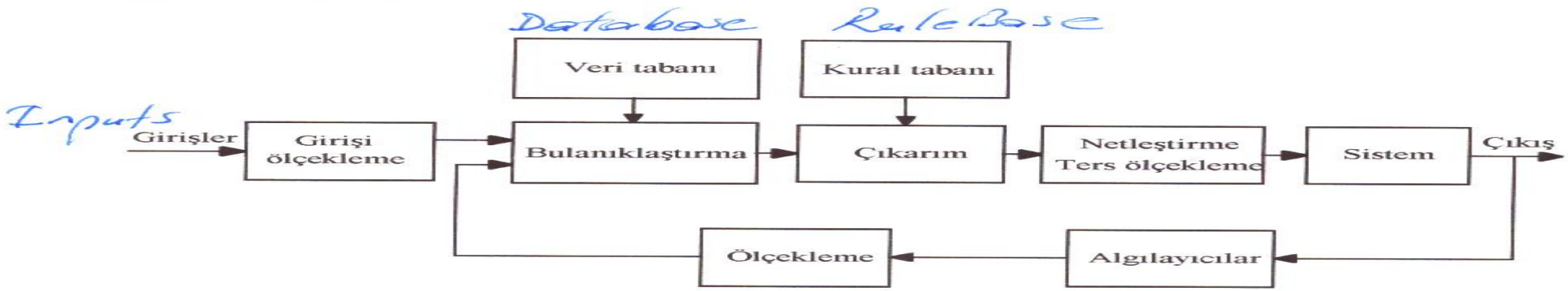
Temel Fuzzy Lojik Kontrol sisteminin blok diyagramı şekil 5.1 'de gösterilmiştir.

Şekil 5.1 'deki veri tabanı modülü giriş ve çıkışın fuzzy kısımlarının tümü hakkındaki verileri, kural tabanlı sistemde tanımlanan giriş değişkenlerine karşılık gelen üyelik fonksiyonlarını ve sistemin kontrolu için kontrol olaylarını veya çıkış değişkenlerini içerir.

Temel fuzzy lojik kontrol sisteminin tasarım basamakları aşağıdaki gibidir.

1. Giriş ve çıkış değişkenleri belirlenir.
2. Her değişkene karşılık gelen evrensel kümelerde fuzzy alt kümelerin bölümleri ve bu fuzzy alt kümelere karşılık gelen sonradan kurallarda kullanılacak sözel etiketler belirlenir.
3. Her fuzzy alt kümenin üyelik fonksiyonu belirlenir.
4. Sistemin kontrol edecek kurallar belirlenir.
5. Giriş ve çıkış normalize etmek için için ölçekleme faktörü belirlenir.

6. FLK girişleri bulanıklaştırılır.
7. Fuzzy çıkarımı kullanarak girişlerin değerleri için her kuralın çıkış değeri bulunur ve elde edilen bu çıkışlar birleştirilir.
9. Sonuçtaki çıkışa netleştirme yöntemlerinden biri uygulanarak keskin çıkış elde edilir.



Şekil 5.1 Temel Fuzzy Lojik Kontrol sisteminin blok diyagramı.

5.2. Genel Fuzzy Lojik Kontrol

Temel tasarım maddeleri aşağıdaki gibidir,(Lee,1990a).

1. Bulanıklaştırma metodları ve bulanıklaştırma işlemcisinin veya bulanıklaştırıcının ifadesi.
2. Veri tabanı;

a. Evrensel kümenin ayrıştırılması / normalize edilmesi.
b. Giriş ve çıkış uzayının bulanık bölümlere ayrılması .
c. Bölümlerin tamamlanması.
d. Temel bir bulanık kümenin üyelik fonksiyonunun seçimi.

3. Kural tabanı

a. İşlem giriş ve çıkış değişkenlerinin seçimi.
b. Fuzzy lojik kontrol kurallarının elde ediliş kaynağı.

3.6 EXAMPLE: FUZZY SYSTEM FOR WIND CHILL

Consider a fuzzy system to determine what the temperature feels like to a person under certain weather conditions. This is known as "wind chill." Wind chill has to do with the rate at which heat is removed from the person's body—the lower the

wind chill, the faster heat is removed. Wind chill cannot make a body any colder than the outside temperature; it only feels colder because heat is being removed at a faster rate.

Assume that an "expert" (i.e., someone who is outside a lot) says the wind chill is determined by the actual temperature and the wind speed (it actually depends on humidity also, but we omit that for this example). Suppose he/she has identified four fuzzy sets for TEMPERATURE:COLD, COOL, WARM, and HOT (characterized in Fig. 3.2), and three fuzzy sets for WIND SPEED:LOW, GENTLE, and HIGH (characterized in Fig. 3.3).

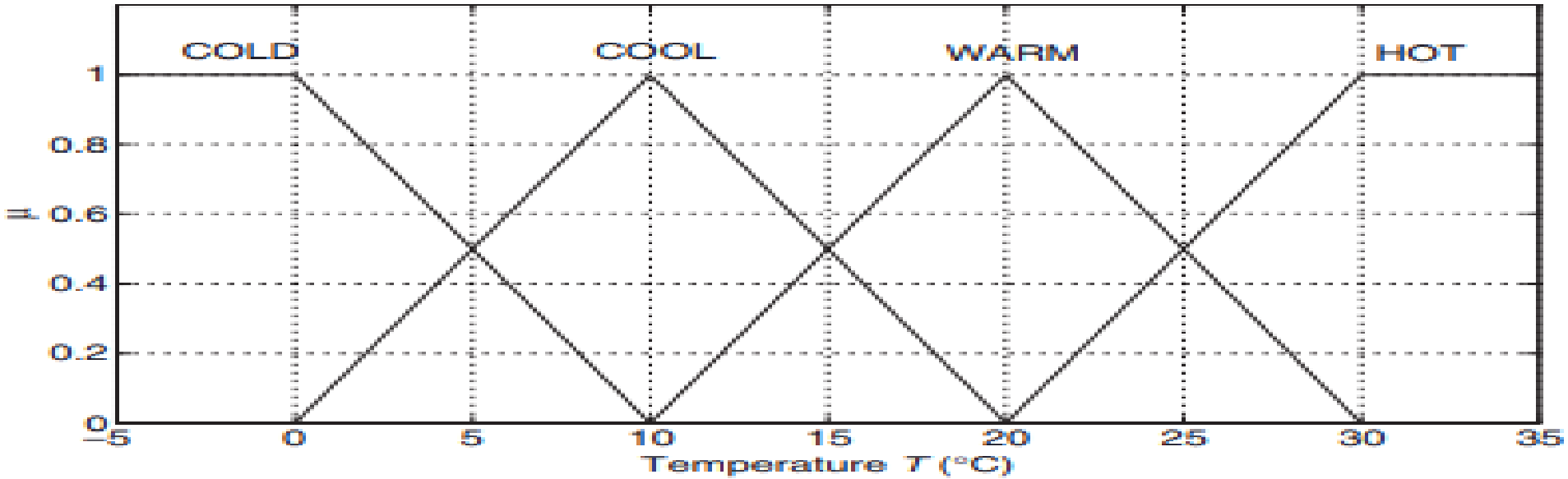


Figure 3.2. Fuzzy sets on the TEMPERATURE universe.

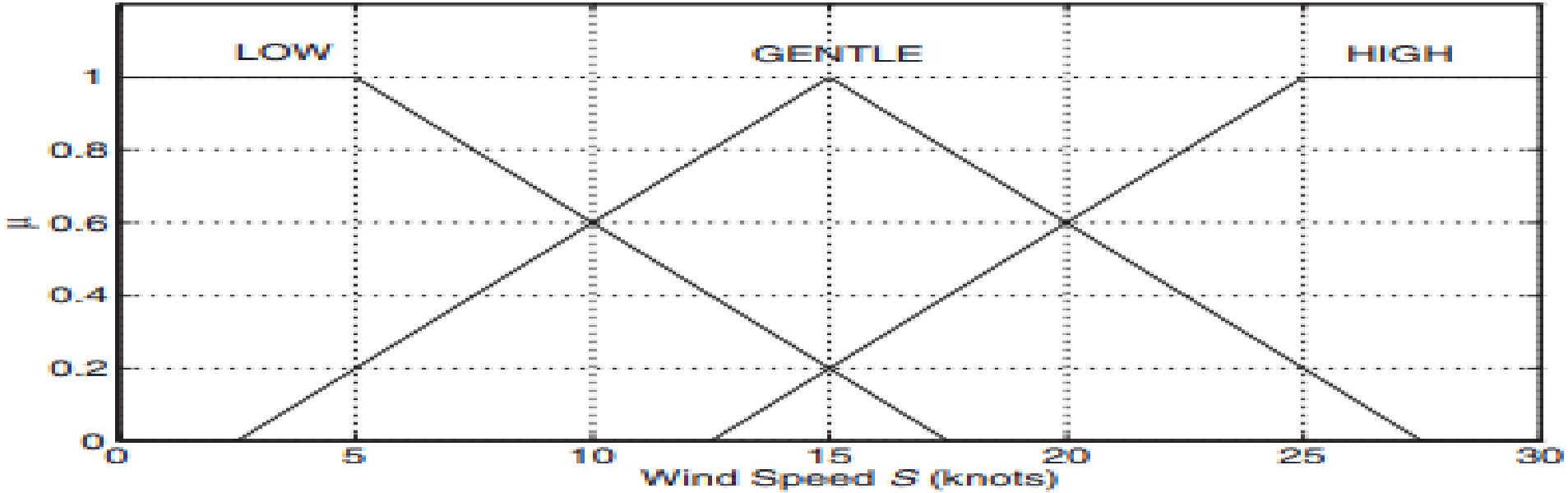


Figure 3.3. Fuzzy sets on the WIND SPEED universe.

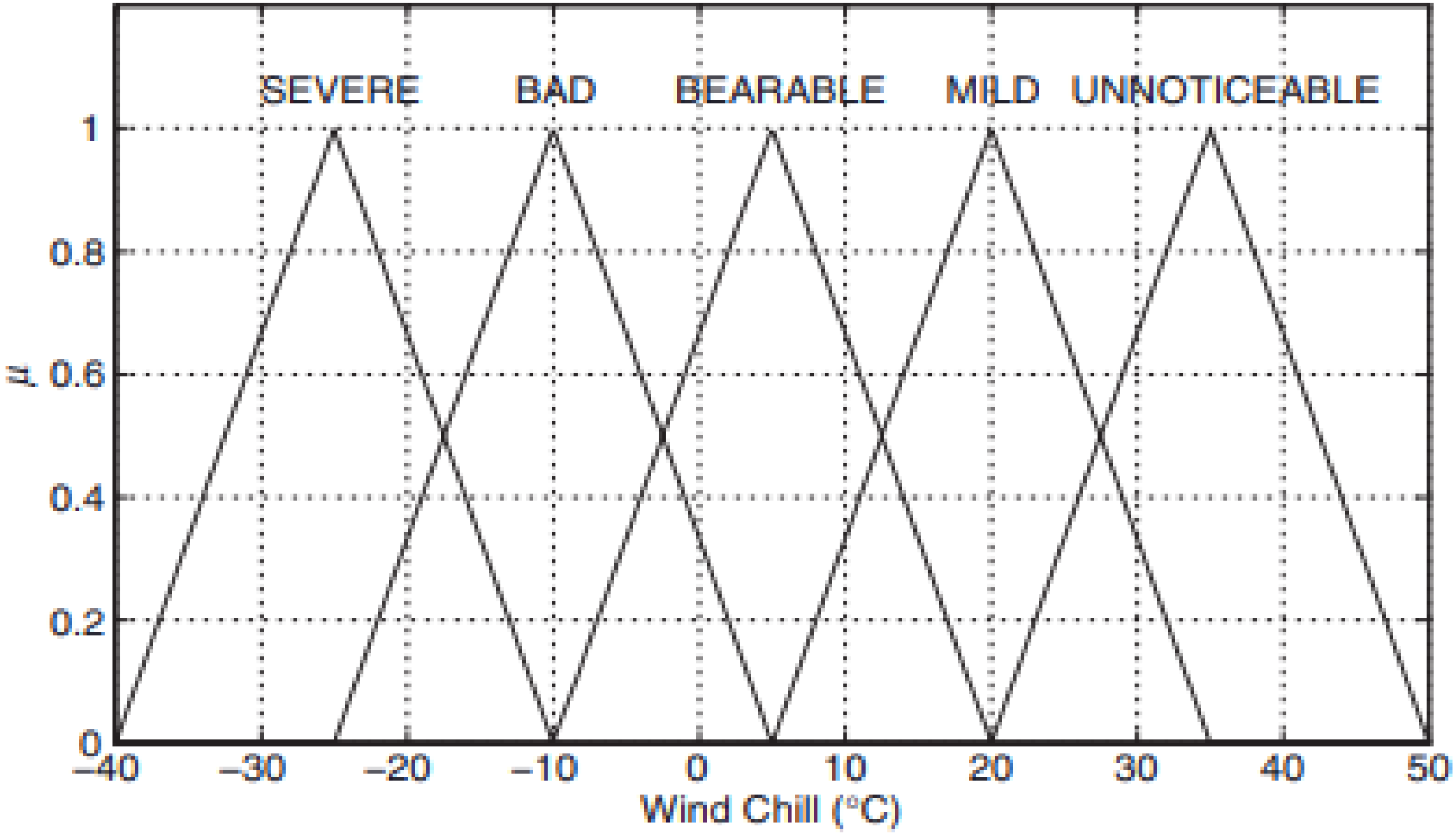


Figure 3.4. Fuzzy sets on the WIND CHILL universe.

Suppose also that he/she has identified five fuzzy sets for WIND CHILL:SEVERE, BAD, BEARABLE, MILD, and UNNOTICEABLE, characterized on the output universe of discourse as in Figure 3.4.

Notice from Figure 3.4 that the end memberships on the output universe (i.e., SEVERE and UNNOTICEABLE) do not saturate at 1 as the end memberships on the input universes do. To see why, look at the operations involved in the defuzzification step. The above defuzzification methods require the calculation of the areas of the implied fuzzy sets $\int \mu^{\hat{Q}'}$ (for COG defuzzification) and the centers of area q_i of the output fuzzy sets (for both COG and CA). Both of these calculations are ill-defined if the output fuzzy sets saturate at 1.

Also, note that the output of the fuzzy system is a *dependent* variable, while the inputs to the fuzzy system are *independent* variables. Therefore, we must allow for the inputs to be any values, but the outputs are prescribed by the fuzzy system, hence their range is restricted. For example, while it may be unlikely, we nevertheless must allow for the *possibility* that the outdoor temperature could go significantly less than 0°C or significantly greater than 30°C. In fact, we have no control whatever over the outdoor temperature (it is an *independent* variable).

Conversely, the lowest wind chill our fuzzy system will be capable of outputting is −25°C because this is the center of area of the SEVERE output fuzzy set. The output of the fuzzy system will never get below this value no matter what the outdoor temperature and wind speed are because of the way the system has been designed. A similar statement could be made about the UNNOTICEABLE fuzzy set. Therefore our Wind Chill fuzzy system is only accurate for wind chills between −25 and 30°C. A larger range could be designed if desired, but it too will be limited to some practical range of wind chills.

Suppose finally that the "expert" has given us the following common-sense rules for determining the wind chill:

1. If TEMPERATURE is COLD and WIND SPEED is LOW, then WIND CHILL is BEARABLE.
2. If TEMPERATURE is COLD and WIND SPEED is GENTLE, then WIND CHILL is BAD.
3. If TEMPERATURE is COLD and WIND SPEED is HIGH, then WIND CHILL is SEVERE.
4. If TEMPERATURE is COOL and WIND SPEED is LOW, then WIND CHILL is MILD.
5. If TEMPERATURE is COOL and WIND SPEED is GENTLE, then WIND CHILL is BEARABLE.
6. If TEMPERATURE is COOL and WIND SPEED is HIGH, then WIND CHILL is BAD.
7. If TEMPERATURE is WARM and WIND SPEED is LOW, then WIND CHILL is UNNOTICEABLE.
8. If TEMPERATURE is WARM and WIND SPEED is GENTLE, then WIND CHILL is MILD.
9. If TEMPERATURE is WARM and WIND SPEED is HIGH, then WIND CHILL is BEARABLE.
10. If TEMPERATURE is HOT and WIND SPEED is LOW, then WIND CHILL is UNNOTICEABLE.
11. If TEMPERATURE is HOT and WIND SPEED is GENTLE, then WIND CHILL is UNNOTICEABLE.
12. If TEMPERATURE is HOT and WIND SPEED is HIGH, then WIND CHILL is MILD.

Now the fuzzy system is completely specified (i.e., we have specified all premise and consequent fuzzy sets, together with the rule base). Note that there are 12 rules, one for each combination of TEMPERATURE and WIND SPEED. This kind of rule base is said to be *complete*. For conciseness, the rule base can be tabulated as in Table 3.1:

TABLE 3.1 WIND CHILL Corresponding to Various Combinations of TEMPERATURE and WIND SPEED[a]

Wind Chill		WIND SPEED		
		LOW	GENTLE	HIGH
Temperature	COLD	BEARABLE	BAD	SEVERE
	COOL	MILD	BEARABLE	BAD
	WARM	UNNOTICEABLE	MILD	BEARABLE
	HOT	UNNOTICEABLE	UNNOTICEABLE	MILD

[a]Tabulation of above rule base.

3.6.1 Wind Chill Calculation, *Minimum* T-Norm, COG Defuzzification

Let us calculate the wind chill corresponding to a temperature of 7°C and a wind speed of 22 knots, using *min* T-norm and COG defuzzification.

Inference

To evaluate the degree to which a temperature of 7°C qualifies as COLD, COOL, WARM, and HOT, evaluate these fuzzy sets' membership functions for $T = 7$. Referring to Figure 3.2, we have

$$\mu^{\text{COLD}}(7) = 0.3$$
$$\mu^{\text{COOL}}(7) = 0.7$$
$$\mu^{\text{WARM}}(7) = 0$$
$$\mu^{\text{HOT}}(7) = 0$$

Similarly, the degree to which a wind speed of 22 knots qualifies as LOW, GENTLE, and HIGH is calculated from Figure 3.3 as

$$\mu^{\text{LOW}}(22) = 0$$
$$\mu^{\text{GENTLE}}(22) = 0.44$$
$$\mu^{\text{HIGH}}(22) = 0.76$$

Using *minimum* T-norm, we obtain the following degrees of firing for the 12 rules in the rule base:

$$R_1 : \mu_1(7, 22) = \mu^{\text{COLD} \cap \text{LOW}} = \min(0.3, 0) = 0$$
$$R_2 : \mu_2(7, 22) = \mu^{\text{COLD} \cap \text{GENTLE}} = \min(0.3, 0.44) = 0.3$$
$$R_3 : \mu_3(7, 22) = \mu^{\text{COLD} \cap \text{HIGH}} = \min(0.3, 0.76) = 0.3$$
$$R_4 : \mu_4(7, 22) = \mu^{\text{COOL} \cap \text{LOW}} = \min(0.7, 0) = 0$$
$$R_5 : \mu_5(7, 22) = \mu^{\text{COOL} \cap \text{GENTLE}} = \min(0.7, 0.44) = 0.44$$
$$R_6 : \mu_6(7, 22) = \mu^{\text{COOL} \cap \text{HIGH}} = \min(0.7, 0.76) = 0.7$$
$$R_7 : \mu_7(7, 22) = \mu^{\text{WARM} \cap \text{LOW}} = \min(0, 0) = 0$$
$$R_8 : \mu_8(7, 22) = \mu^{\text{WARM} \cap \text{GENTLE}} = \min(0, 0.44) = 0$$
$$R_9 : \mu_9(7, 22) = \mu^{\text{WARM} \cap \text{HIGH}} = \min(0, 0.76) = 0$$
$$R_{10} : \mu_{10}(7, 22) = \mu^{\text{HOT} \cap \text{LOW}} = \min(0, 0) = 0$$
$$R_{11} : \mu_{11}(7, 22) = \mu^{\text{HOT} \cap \text{GENTLE}} = \min(0, 0.44) = 0$$
$$R_{12} : \mu_{12}(7, 22) = \mu^{\text{HOT} \cap \text{HIGH}} = \min(0, 0.76) = 0$$

Thus we see that for this input $(T, S) = (7, 22)$, Rule 6 is fired with the greatest certainty, Rule 5 is fired with less certainty, and so on. Many rules are not fired at all.

The membership arrangement in Figure 3.2 is called a *partition of unity* because the sum of all memberships equals 1 at every T. The membership arrangement in Figure 3.3 is not a partition of unity. Note that a group of Gaussian membership functions can never form a partition of unity.

Partitions of unity simplify defuzzification (see Chapter 4), but it is not absolutely necessary to use them. As always, fuzzy sets should be defined so that they most accurately reflect the quantities they describe, not merely to simplify calculations.

Since only four rules are fired for this input, we will create four nonzero implied fuzzy sets. To create the implied fuzzy sets for Rules 2, 3, 5, and 6, we attenuate each rule's consequent membership function by the degree of firing of the rule. Using *minimum* T-norm, we obtain the membership function characterizing Rule 2's implied fuzzy set as

$$\mu_2^{\text{implied}}(y) = \min_y \left\{ \mu_2(7, 22), \mu^{\text{BAD}}(y) \right\}$$

This minimum is taken pointwise in $\mathcal{Y}$. Similarly, the membership function characterizing Rules 3, 5, and 6's implied fuzzy sets are

$$\mu_3^{\text{implied}}(y) = \min_y \left\{ \mu_3(7, 22), \mu^{\text{SEVERE}}(y) \right\}$$

$$\mu_5^{\text{implied}}(y) = \min_y \left\{ \mu_4(7, 22), \mu^{\text{BEARABLE}}(y) \right\}$$

$$\mu_6^{\text{implied}}(y) = \min_y \left\{ \mu_6(7, 22), \mu^{\text{BAD}}(y) \right\}$$

The resulting implied fuzzy sets' membership functions are shown in Figure 3.5. Some of the memberships in Figure 3.5 have been slightly displaced so they can be seen more clearly.

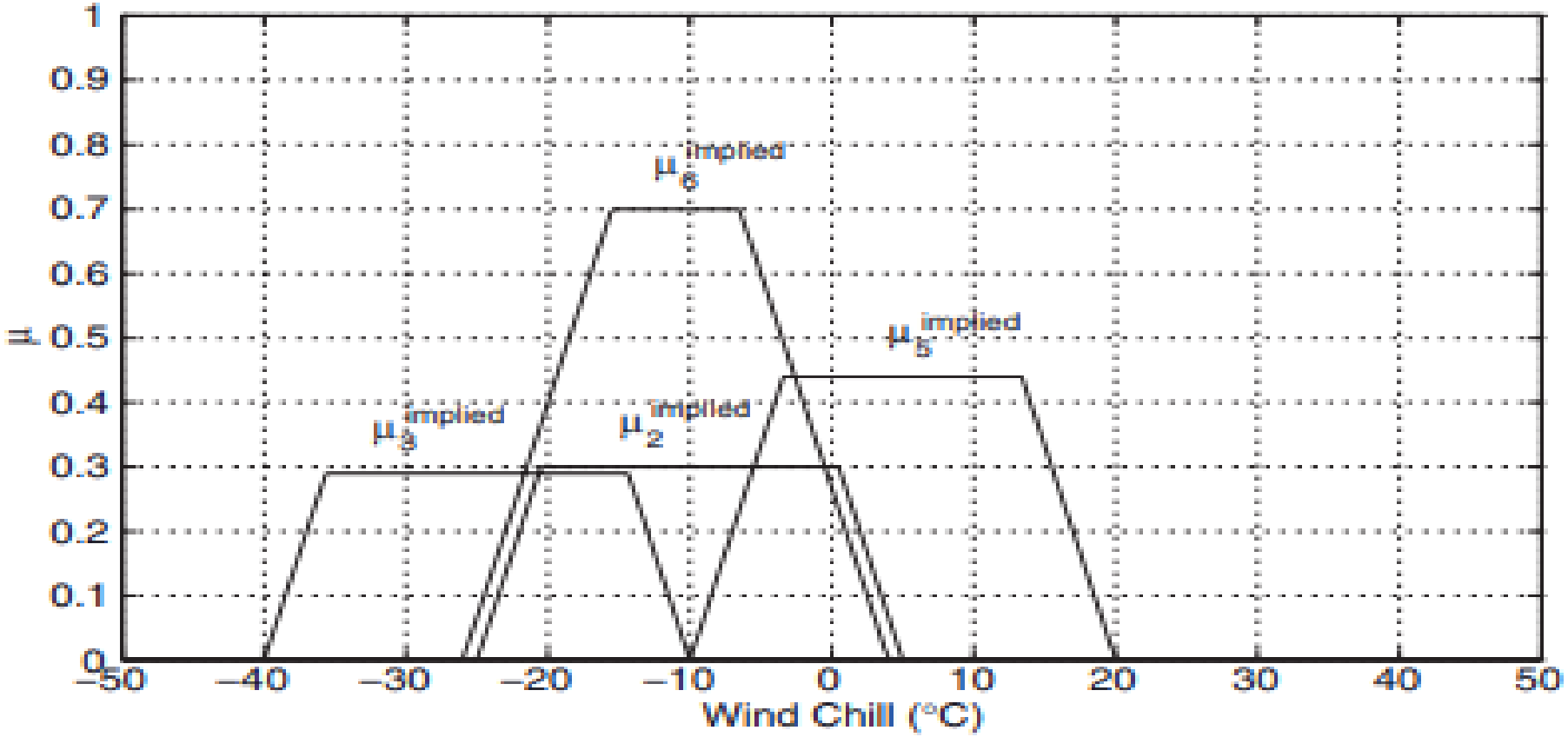


Figure 3.5. Implied fuzzy sets, *minimum* T-norm (slightly displaced to improve clarity).

Defuzzification

To calculate a crisp output using COG defuzzification, it is necessary to find the centers of area of the memberships characterizing the output fuzzy sets [q_i, $i = 2, 3, 5$, and 6 in (3.5)] and the areas of the four trapezoidal member-

ship functions characterizing the implied fuzzy sets in Figure 3.5 [$\int \mu_i^{\text{implied}}$, $i = 2, 3, 5, 6$ in (3.5)]. These are

$$q_2 = -10$$
$$q_3 = -25$$
$$q_5 = 5$$
$$q_6 = -10$$
$$\int \mu_2^{\text{implied}} = 7.65$$
$$\int \mu_3^{\text{implied}} = 7.65$$
$$\int \mu_5^{\text{implied}} = 10.296$$
$$\int \mu_6^{\text{implied}} = 13.65$$

Note: The area of a trapezoid with base w and height h is $w(h - h^2/2)$.

Therefore, the crisp output of the fuzzy system corresponding to the crisp input $(T, S) = (7, 22)$, is calculated using COG defuzzification to be

$$y^{\text{crisp}} = \frac{-10(7.65) - 25(7.65) + 5(10.296) - 10(13.65)}{7.65 + 7.65 + 10.296 + 13.65} = -8.9887°\text{C} \qquad (3.8)$$

Thus, the wind chill corresponding to a temperature of 7°C and a wind speed of 22 knots, calculated using the rule base and fuzzy sets specified above, using the *minimum* T-norm for conjunction in the premise and inference, and COG defuzzification, is −8.9887°C.

Since there are only two inputs, the input–output characteristic of the fuzzy system can be plotted. The characteristic is shown in Figure 3.6.

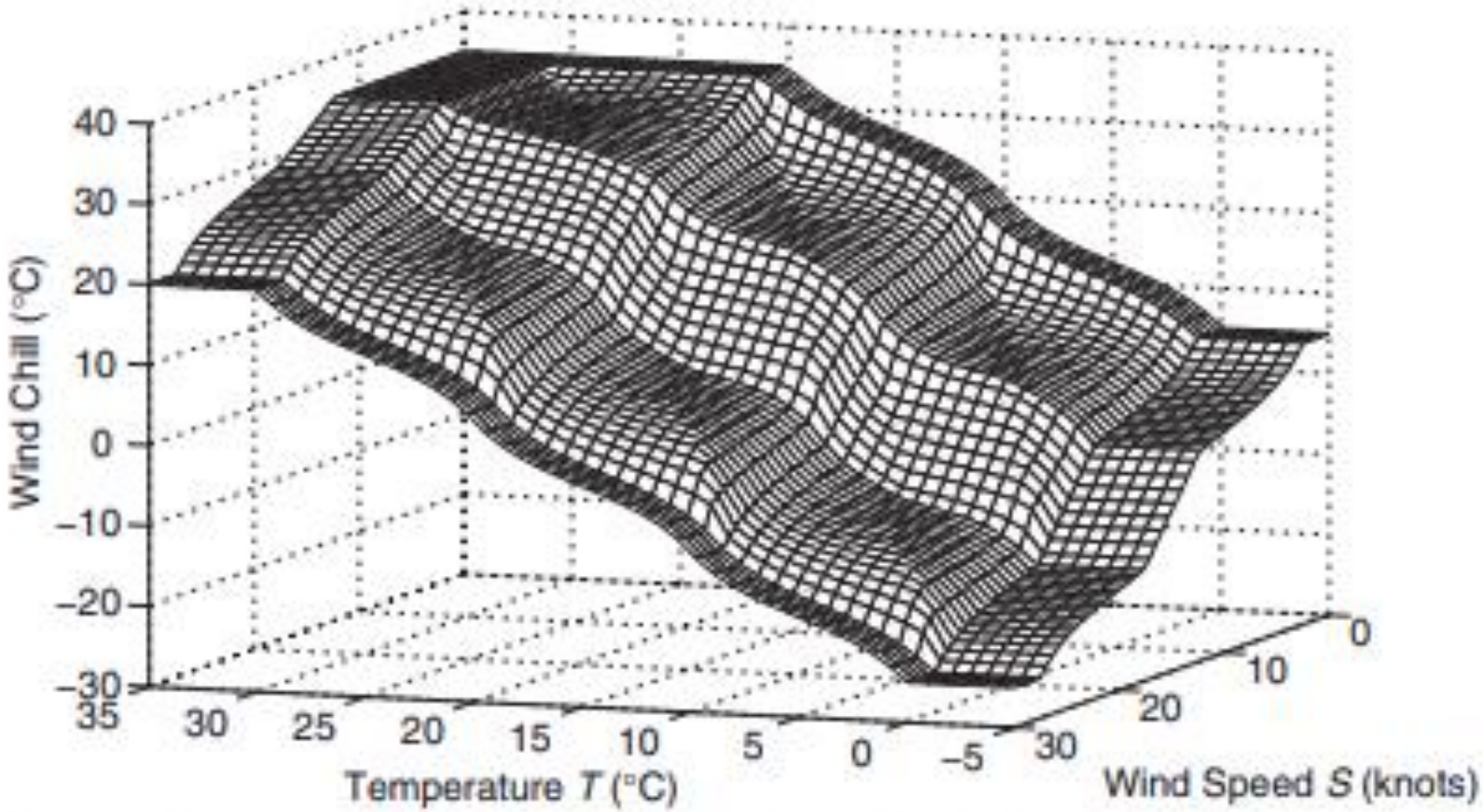


Figure 3.6. Input–output characteristic of *Wind Chill* fuzzy system, *min* T-norm, COG defuzzification.

The *general* shape of the characteristic reflects our common sense about wind chill: that is that higher wind chill temperatures result from higher temperatures combined with lower wind speeds, and lower wind chill temperatures result from lower temperatures combined with higher wind speeds. The *local* shape of the characteristic, that is, the waves and undulations, is a function of the membership shapes, T-norm, and defuzzification method used. Therefore, the local shape is not very important; the global shape is.

The flat areas at the extremes of the characteristic reflect the fact that the leftmost and rightmost TEMPERATURE memberships saturate at 0 and 30°C, respectively, the leftmost and rightmost WIND SPEED memberships saturate at 2.5 and 27.5 kn, respectively, and the leftmost and rightmost WIND CHILL membership centers are at −25 and 35°C, respectively. Therefore, the characteristic is only accurate for (T, S) such that $0°C \leq T \leq 30°C$ and $2.5 \leq S \leq 27.5$ kn. If the characteristic is examined closely, it can be seen that the point $(T, S) = (7, 22)$ corresponds to a wind chill of −8.9887°C.

3.6.2 Wind Chill Calculation, *Minimum* T-Norm, CA Defuzzification

If CA defuzzification is used instead of COG, the crisp output of the fuzzy system corresponding to the crisp input $(T, S) = (7, 22)$ is calculated to be

$$y^{\text{crisp}} = \frac{-10(0.3) - 25(0.3) + 5(0.44) - 10(0.7)}{0.3 + 0.3 + 0.44 + 0.7} = -8.7931°\text{C} \tag{3.9}$$

Thus, the wind chill corresponding to a temperature of 7°C and a wind speed of 22 knots, calculated using the rule base and fuzzy sets specified above, using the *minimum* T-norm for conjunction in the premise and inference, and CA defuzzification, is −8.7931°C.

The wind chill −8.7931°C calculated in (3.9) is close to the wind chill −8.9887°C calculated in (3.8), but not exactly equal to it. The difference is due to the difference in defuzzification methods used. Neither is correct or incorrect. The difference is analogous to the difference in percieved temperature different individuals might feel under the same conditions, or even the same person on different days.

The input–output characteristic of the fuzzy system is shown in Figure 3.7. Its shape is slightly different from that of Figure 3.6 due to the use of CA defuzzification rather than COG.

3.6.3 Wind Chill Calculation, *Product* T-Norm, COG Defuzzification

Let us calculate the crisp output corresponding to the same crisp input, but this time using *product* T-norm throughout.

Inference

Since we are using the same input as before $(T, S) = (7, 22)$, we have the same degrees of belongingness for all input fuzzy sets as before. Using *product* T-norm, we obtain the following degrees of firing for the rules:

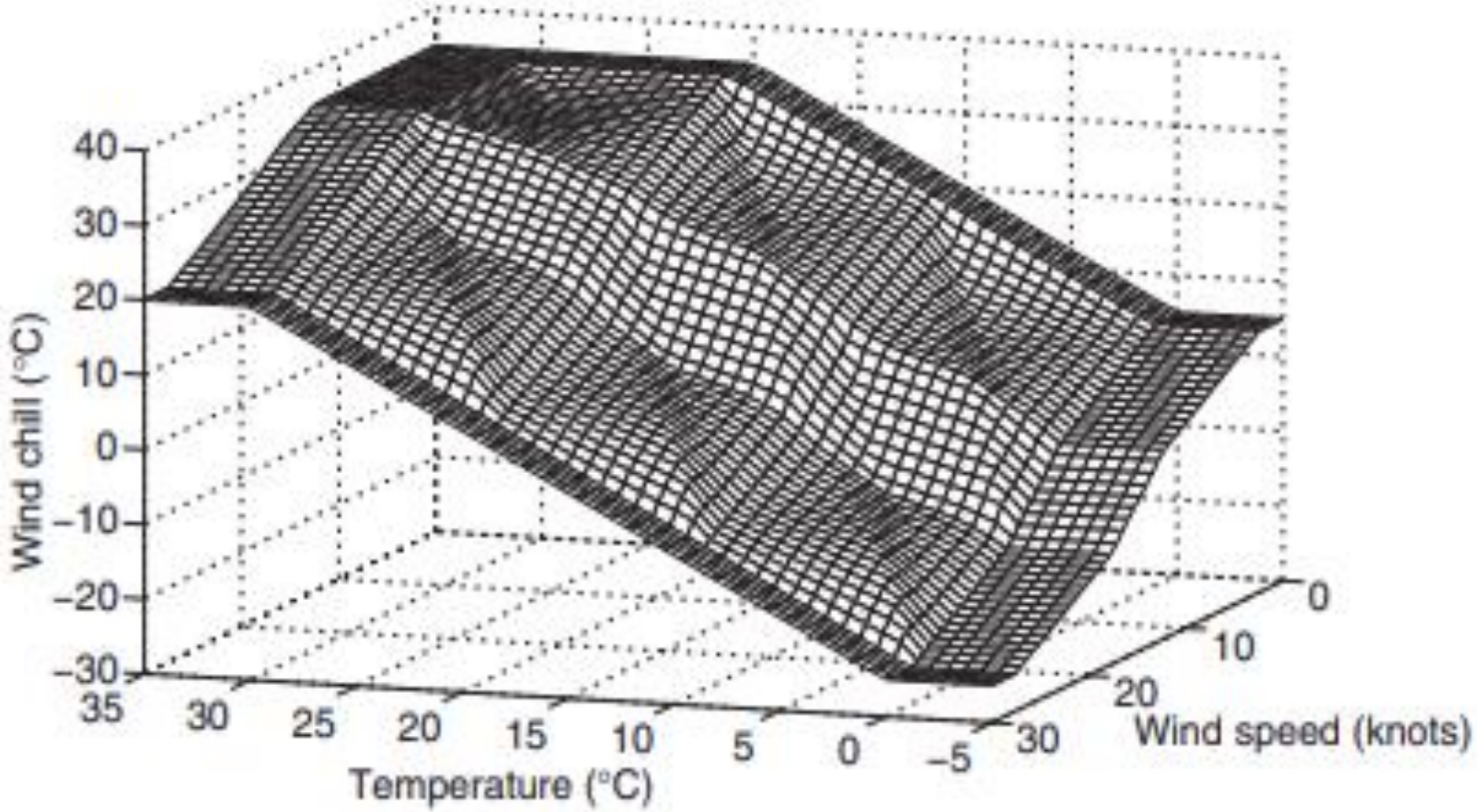


Figure 3.7. Input–output characteristic of *Wind Chill* fuzzy system, *min* T-norm, CA defuzzification.

$$R_1 : \mu_1(7, 22) = \mu^{\text{COLD} \cap \text{LOW}} = (0.3)(0) = 0$$
$$R_2 : \mu_2(7, 22) = \mu^{\text{COLD} \cap \text{GENTLE}} = (0.3)(0.44) = 0.132$$
$$R_3 : \mu_3(7, 22) = \mu^{\text{COLD} \cap \text{HIGH}} = (0.3)(0.76) = 0.228$$
$$R_4 : \mu_4(7, 22) = \mu^{\text{COOL} \cap \text{LOW}} = (0.7)(0) = 0$$
$$R_5 : \mu_5(7, 22) = \mu^{\text{COOL} \cap \text{GENTLE}} = (0.7)(0.44) = 0.308$$
$$R_6 : \mu_6(7, 22) = \mu^{\text{COOL} \cap \text{HIGH}} = (0.7)(0.76) = 0.532$$
$$R_7 : \mu_7(7, 22) = \mu^{\text{WARM} \cap \text{LOW}} = (0)(0) = 0$$
$$R_8 : \mu_8(7, 22) = \mu^{\text{WARM} \cap \text{GENTLE}} = (0)(0.44) = 0$$
$$R_9 : \mu_9(7, 22) = \mu^{\text{WARM} \cap \text{HIGH}} = (0)(0.76) = 0$$
$$R_{10} : \mu_{10}(7, 22) = \mu^{\text{HOT} \cap \text{LOW}} = (0)(0) = 0$$
$$R_{11} : \mu_{11}(7, 22) = \mu^{\text{HOT} \cap \text{GENTLE}} = (0)(0.44) = 0$$
$$R_{12} : \mu_{12}(7, 22) = \mu^{\text{HOT} \cap \text{HIGH}} = (0)(0.76) = 0$$

Again, only Rules 2, 3, 5, and 6 are fired. Using *product* T-norm, we obtain the membership functions characterizing Rules 2, 3, 5, and 6's implied fuzzy sets as

$$\mu_2^{\text{implied}}(y) = (\mu_2(7, 22))(\mu^{\text{BAD}}(y))$$
$$\mu_3^{\text{implied}}(y) = (\mu_3(7, 22))(\mu^{\text{SEVERE}}(y))$$
$$\mu_5^{\text{implied}}(y) = (\mu_5(7, 22))(\mu^{\text{BEARABLE}}(y))$$
$$\mu_6^{\text{implied}}(y) = (\mu_6(7, 22))(\mu^{\text{BAD}}(y))$$

The resulting membership functions are shown in Figure 3.8.

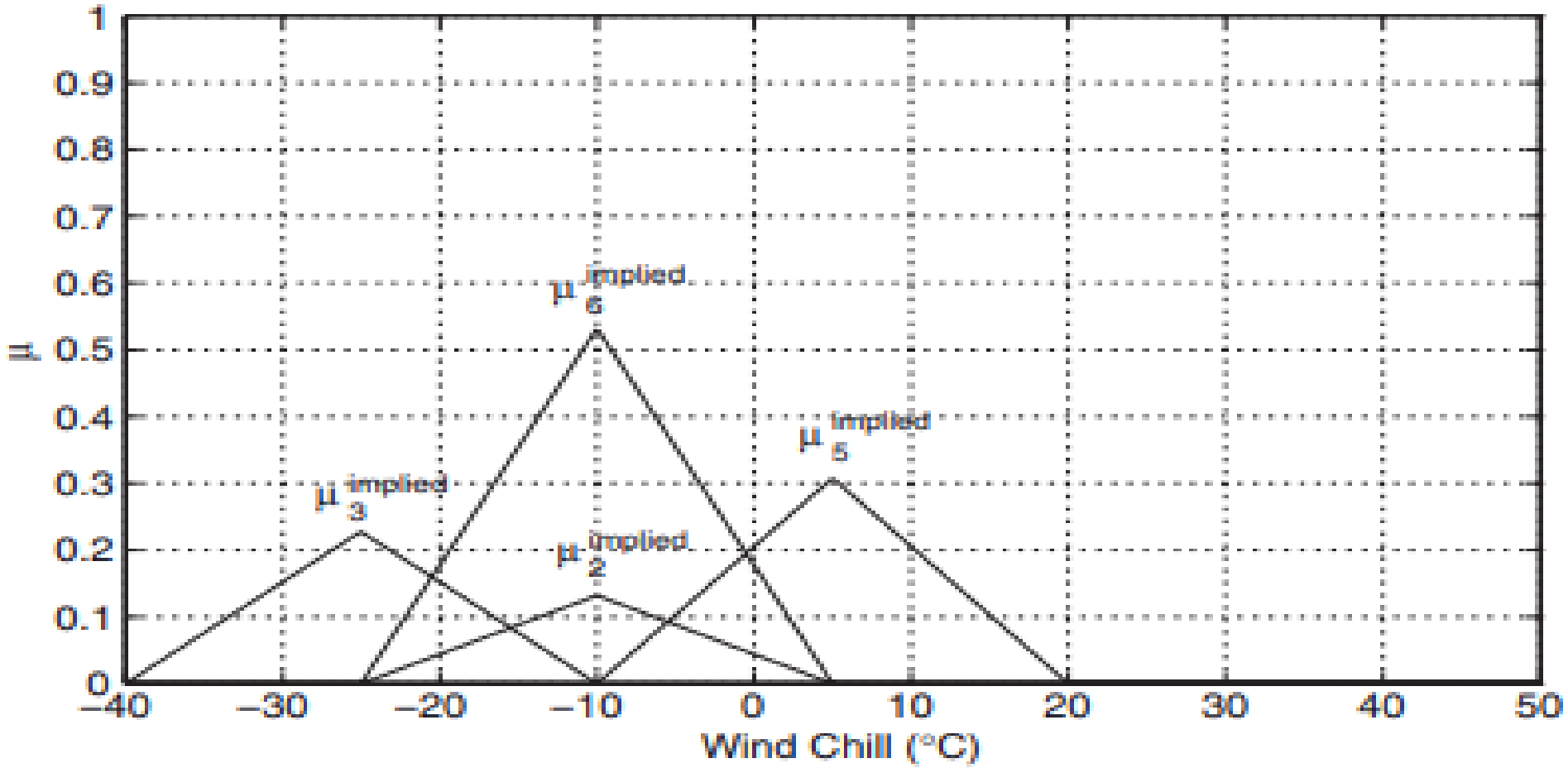


Figure 3.8. Implied fuzzy sets, *product* T-norm.

Defuzzification

The centers of area of the output fuzzy sets [q_i, $i = 2, 3\ 5, 6$ in (3.5)] are the same as before, and the areas of the four triangular membership functions characterizing the implied fuzzy sets in Figure 3.8 [$\int \mu_i^{\text{implied}}$, $i = 2, 3\ 5$, and 6 in (3.5)] are

$$\int \mu_2^{\text{implied}} = 1.98$$
$$\int \mu_3^{\text{implied}} = 3.42$$
$$\int \mu_5^{\text{implied}} = 4.62$$
$$\int \mu_6^{\text{implied}} = 7.98$$

Therefore, the crisp output of the fuzzy system corresponding to the crisp input $(T, S) = (7, 22)$ is calculated using COG defuzzification to be

$$y^{\text{crisp}} = \frac{-10(1.98) - 25(3.42) + 5(4.62) - 10(7.98)}{1.98 + 3.42 + 4.62 + 7.98} = -9.0°\text{C} \qquad (3.10)$$

Thus, the wind chill corresponding to a temperature of 7°C and a wind speed of 22 knots, calculated using the rule base and fuzzy sets specified above, using the *product* T-norm for conjunction in the premise and inference, and COG defuzzification, is –9.0°C. The input–output characteristic of the fuzzy system is shown in Figure 3.9. This is slightly different from the previous characteristics (Figs. 3.6 and 3.7) due to the different T-norm and defuzzification methods. The characteristic still has the same general shape as the others.

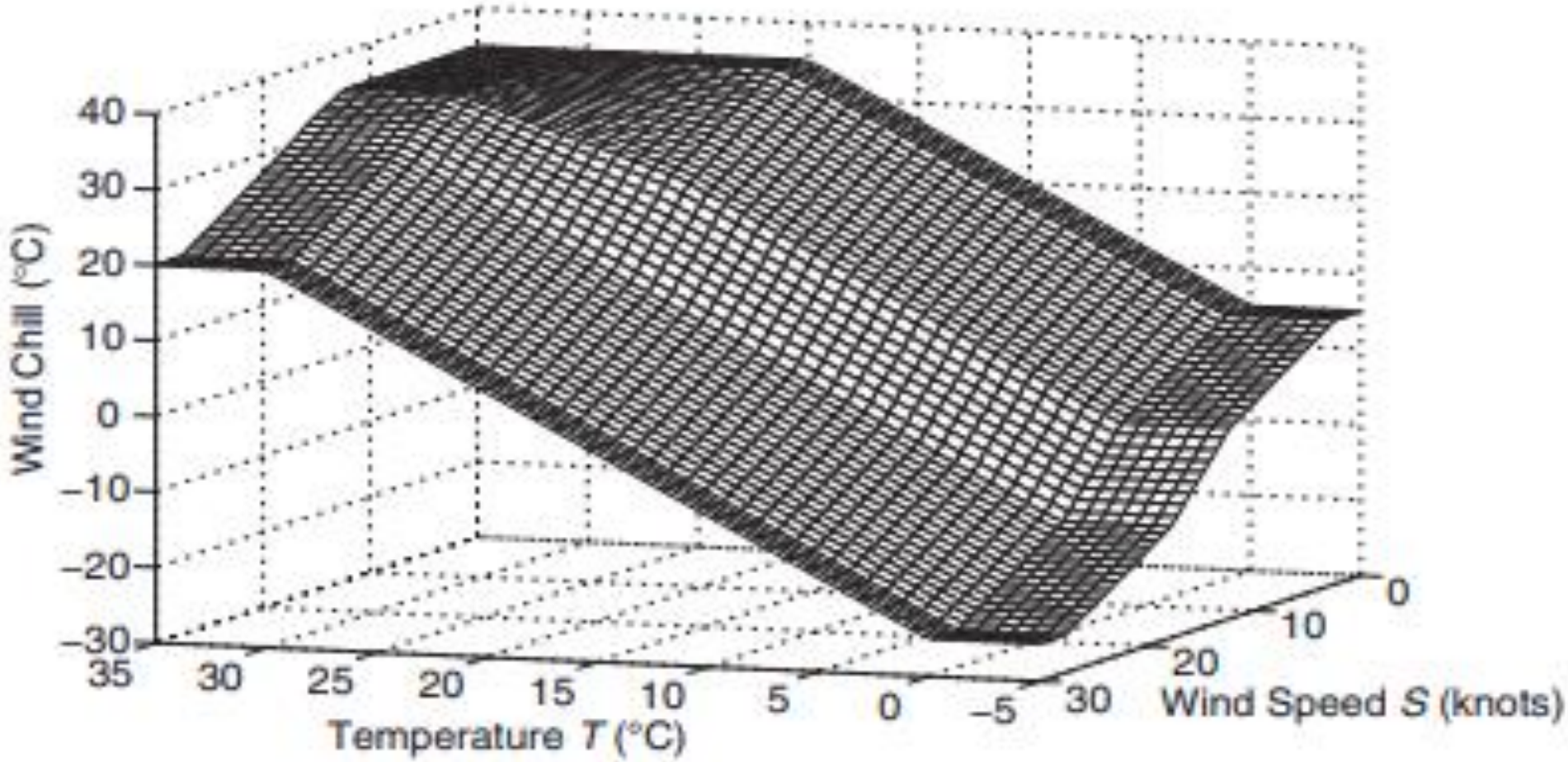


Figure 3.9. Input–output characteristic of *Wind Chill* fuzzy system, *product* T-norm, COG defuzzification.

3.6.4 Wind Chill Calculation, *Product* T-Norm, CA Defuzzification

If CA defuzzification is used instead of COG, the crisp output of the fuzzy system corresponding to the crisp input $(T, S) = (7, 22)$ is calculated to be

$$y^{\text{crisp}} = \frac{-10(0.132) - 25(0.228) + 5(0.308) - 10(0.532)}{0.132 + 0.228 + 0.308 + 0.532} = -9.0°\text{C} \qquad (3.11)$$

Thus, the wind chill corresponding to a temperature of 7°C and a wind speed of 22 knots, calculated using the rule base and fuzzy sets specified above, using the *product* T-norm for conjunction in the premise and inference, and CA defuzzification, is –9.0°C. The input–output characteristic is identical to that of Figure 3.9. This is due to the fact that the areas of the triangles in Figure 3.8 are proportional to the premise membership values of the corresponding rules. This is not true for the trapezoidal memberships of Figure 3.5.

3.6.5 Wind Chill Calculation, Singleton Output Fuzzy Sets, *Product* T-Norm, CA Defuzzification

Another possibility for output fuzzy sets is *singleton* fuzzy sets. As mentioned in Section 2.5, these are characterized by membership functions that are zero except for one point in the output universe, where they are 1. For example, the output fuzzy sets SEVERE, BAD, BEARABLE, MILD, and UNNOTICEABLE could be characterized by the membership functions shown in Figure 3.10.

Now, defuzzification must be CA since there is no area to compute. Also, since the singletons are normal, Eq. (3.7) is always used for defuzzification. Therefore, the above results for triangular output memberships, *product* T-norm, and CA defuzzification would also be obtained if the output fuzzy sets were singletons. In general, if *product* T-norm is used and the output memberships are symmetrical and

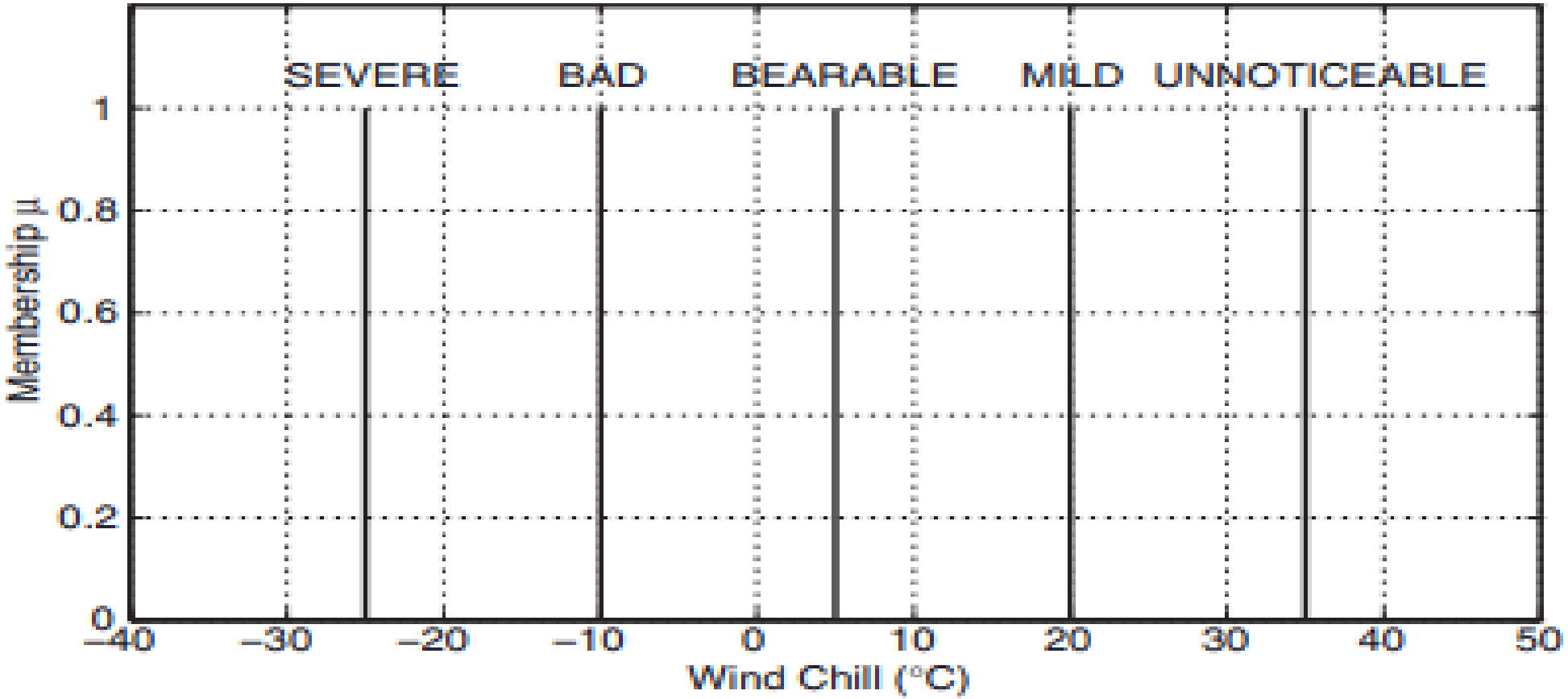


Figure 3.10. Membership functions characterizing singleton output fuzzy sets.

normal, the output fuzzy sets may be replaced by singletons. In the remaining chapters of this book, only singleton output fuzzy sets will be used.

FUZZY CONTROL WITH MAMDANI SYSTEMS

The basic idea of control is to command a system to perform as desired by monitoring the system ' s performance and adjusting its input in such a way as to force the performance to be as desired. The output or states of the system are measured and fed back to the controller. On the basis of this information, the controller decides how to change the system input in order to improve the system performance. Much of conventional control is " model based, " which means the controller design is based on a mathematical model of the system. Examples of model - based controllers are linear state feedback controllers, optimal controllers, H∞ controllers, and proportional - integral - derivative (PID) controllers (although a skilled expert can tune a PID controller to improve system performance even when there is no mathematical model of the system). In some cases, however, these methods fail because a suffi ciently accurate mathematical model of the system is not known. In such cases, if suffi cient knowledge about how to control the system is available from a human " expert, " a fuzzy system can be designed to effectively control the system even if the mathematical model is completely unknown [13,16 – 18] . In fact, one of the main uses for fuzzy systems is in closed - loop control of nonlinear systems whose mathematical models are unknown or poorly known.

4.1 TRACKING CONTROL WITH A MAMDANI FUZZY CASCADE COMPENSATOR

Mamdani fuzzy systems can be used to formulate compensators that are based on the user's common sense about how to control a system. This kind of controller needs no mathematical model of the system, hence it is not model based. Its design is rather based on expert knowledge.

Most plants to be controlled are continuous-time, therefore their inputs and outputs are piecewise-continuous functions of time. If the control objective is tracking, the controller configuration usually involves unity feedback with a cascade compensator, as in Figure 1.6. When the compensator is a fuzzy system, the configuration of Figure 4.1 results.

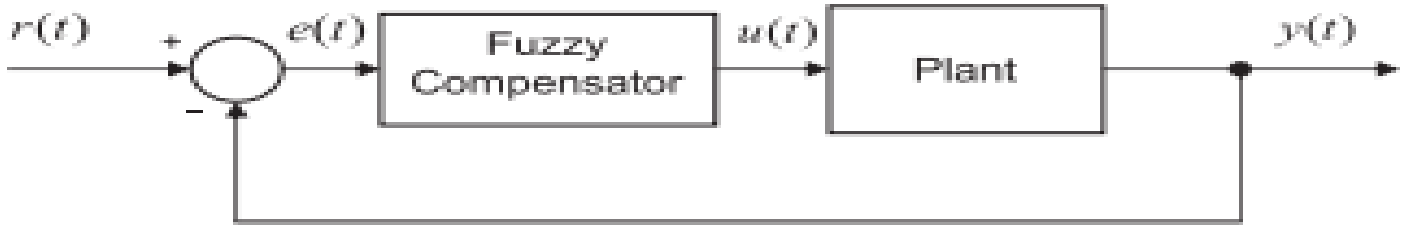


Figure 4.1. Closed-loop system with cascade fuzzy controller.

In Figure 4.1, the output of the summer is the continuous-time function $e(t) = r(t) - y(t)$, which is the tracking error. The fuzzy compensator takes $e(t)$ as its input and formulates a plant input to force the plant output to follow the reference input $r(t)$. In this chapter, the compensator will be designed from only expert knowledge about how to control the system. Therefore, no plant models will be needed.

Because the compensator is a fuzzy system, it is virtually always implemented with a digital computer. Therefore, time must be discretized with an appropriate sampling time depending on the speed of the analog signals. The input to the fuzzy compensator is a sampled version of $e(t)$, resulting in an output of the fuzzy compensator occurring at every sample time. The compensator's output must then be converted to a continuous-time signal to be fed to the plant. Because the compensator's input changes at every time step, the compensator's output also changes at every time step according to the fuzzy compensator's input–output characteristic (see Section 3.6). This characteristic does not change with time, therefore the fuzzy compensator implements a time-invariant mapping from $e(t)$ to $u(t)$ with $e(t)$ [hence $u(t)$] changing at every time step.

4.1.1 Initial Fuzzy Compensator Design: Ball and Beam Plant

Consider the ball and beam plant of Section 1.4. The system is depicted in Figure 4.2 and modeled (for simulation purposes only) by [19]:

$$\ddot{x} = 9.81 \sin kv \tag{4.1}$$

Note: The model (4.1) is actually a simplified model of the ball and beam [12].

In Figure 4.2, $x(t)$ is the position of the ball along the beam (with $x = 0$ defined as the center of the beam), and $\psi(t)$ is the beam angle commanded by the motor (with $\psi = 0$ defined as horizontal). The input to the ball and beam system is the voltage v supplied to the motor. The beam angle ψ is proportional to v (i.e., $\psi = kv$).

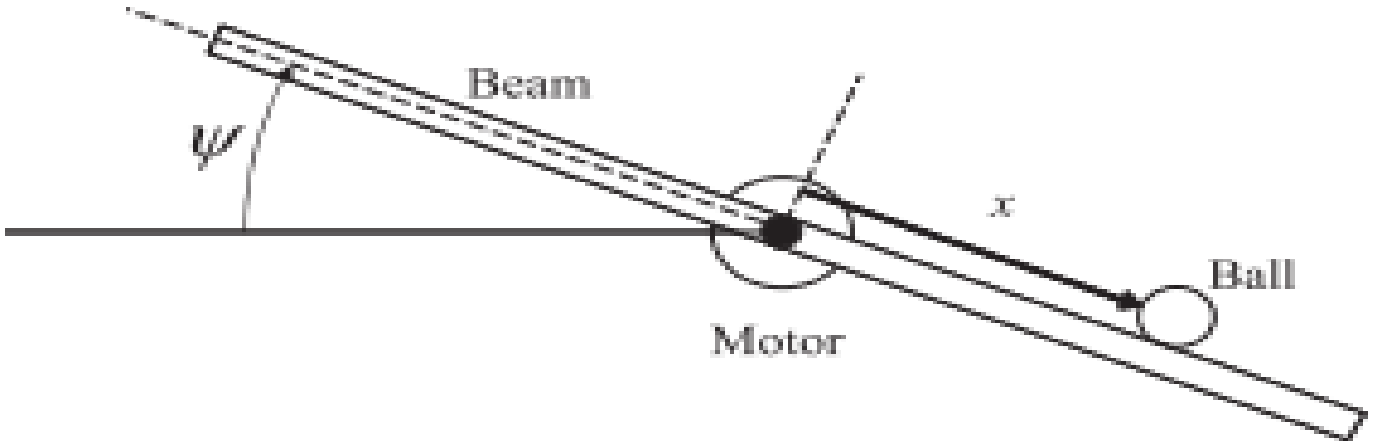


Figure 4.2. Ball and beam system.

The control problem is to adjust $v(t)$ so that the ball position $x(t)$ tracks an arbitrary reference signal $r(t)$. If the control task is to stop the ball motionless at the center of the beam, we have $r(t) = 0$. In that case, $e(t) = -x(t)$.

In general, determination of the information necessary to accomplish a given control task is not trivial, but for many problems the determination can be made using common sense. Suppose an "expert" has decided that the contol objective can be accomplished with knowledge of the ball's position and velocity. Accordingly, let the inputs to the fuzzy controller be e and $\dot{e}$, and the output be v.

Define five fuzzy sets on the e and $\dot{e}$ universes, with linguistic values *Negative Large* (NL), *Negative Small* (NS), *Zero* (Z), *Positive Small* (PS), and *Positive Large* (PL). These are characterized by the triangular memberships shown in Figures 4.3 and 4.4.

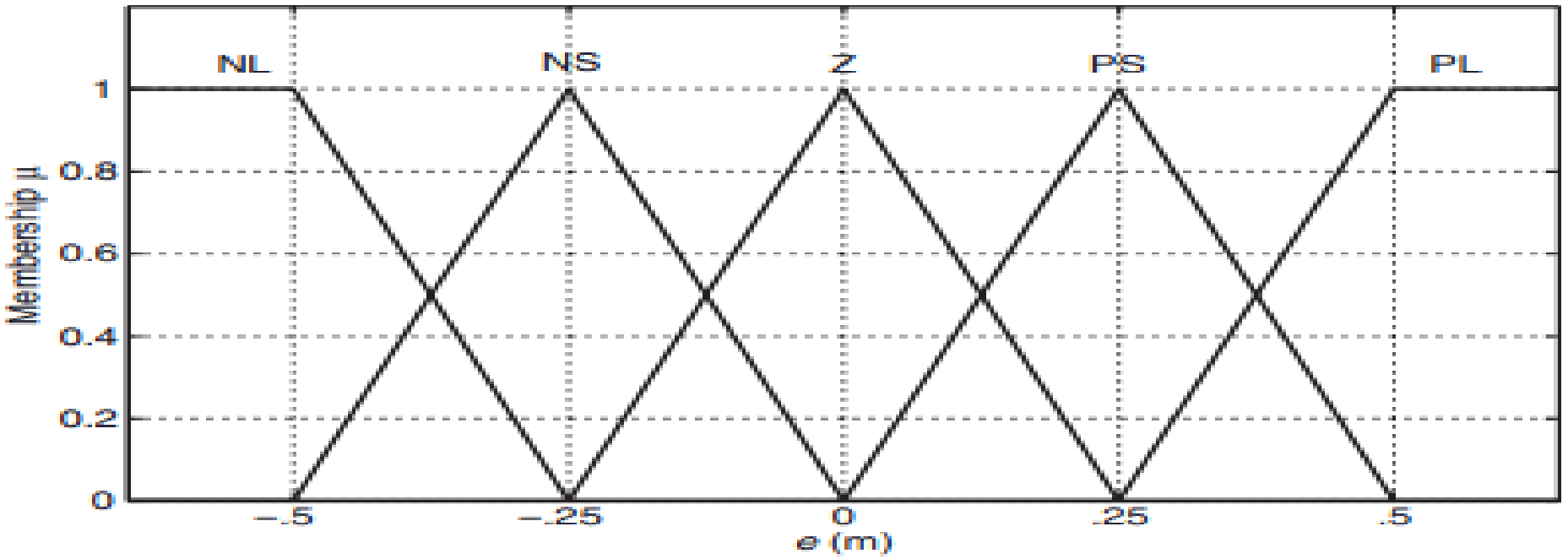


Figure 4.3. Triangular fuzzy sets on e universe.

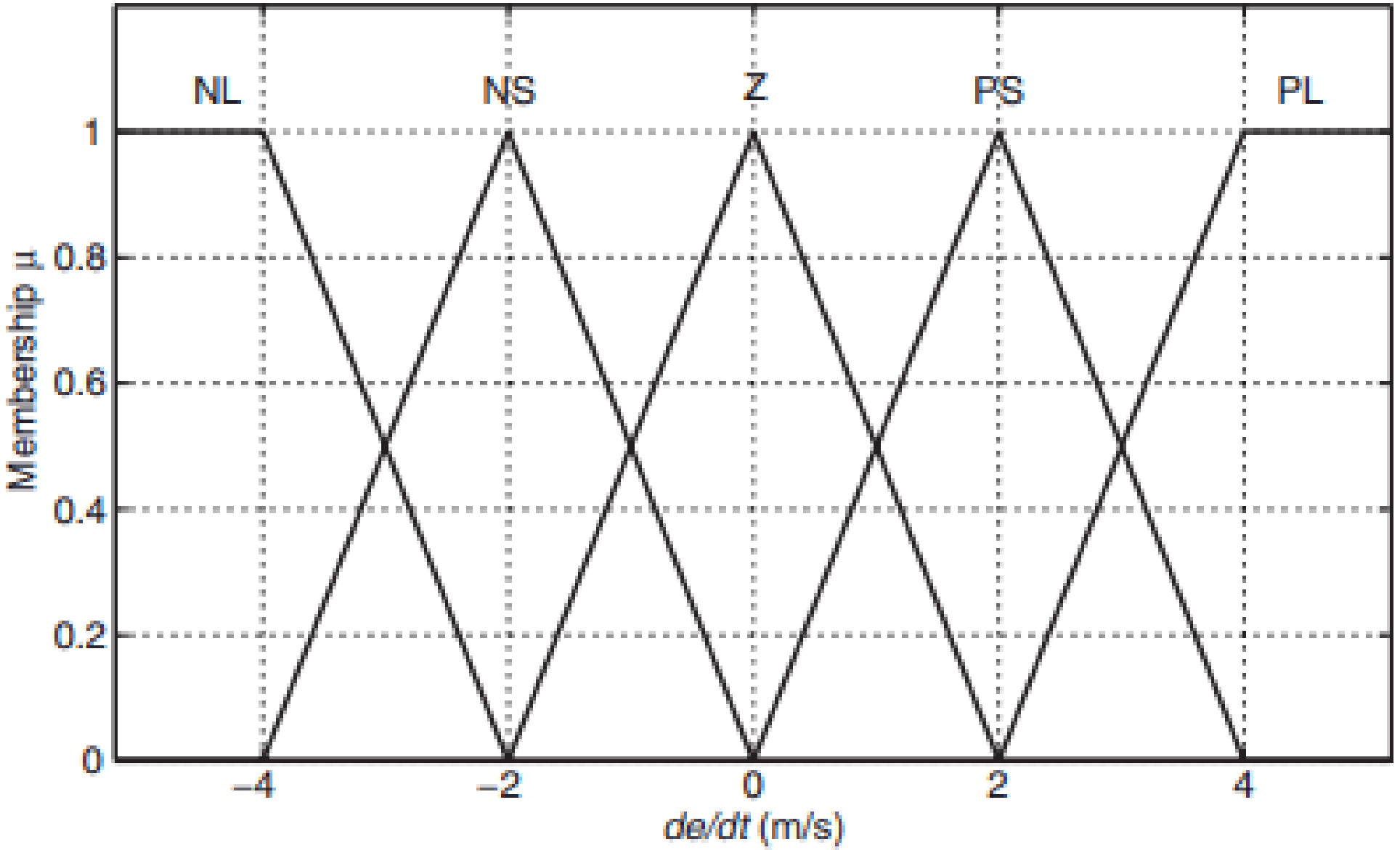


Figure 4.4. Triangular fuzzy sets on $\dot{e}$ universe.

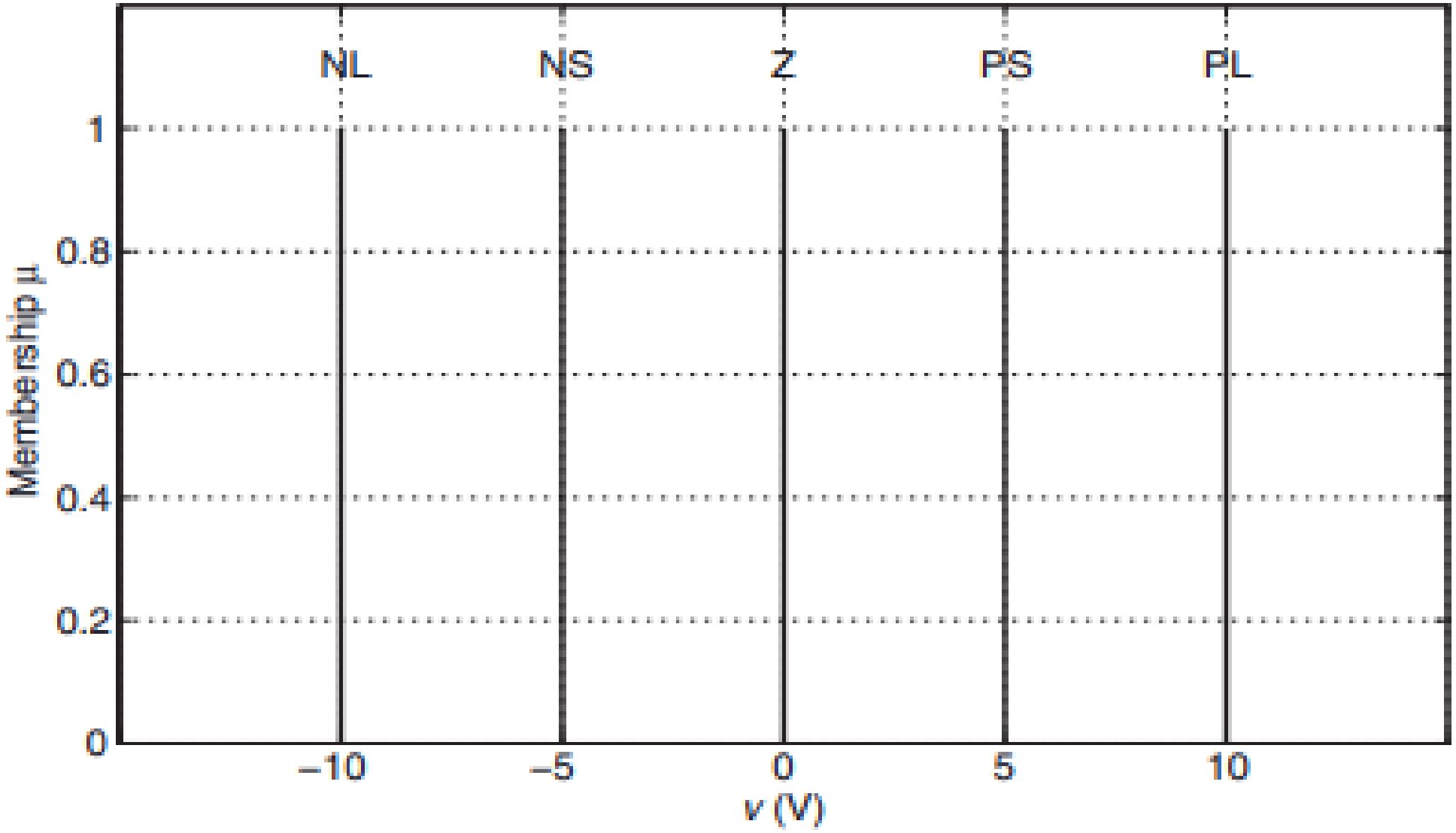


Figure 4.5. Fuzzy sets on v universe.

Similarly define five singleton fuzzy sets on the v universe, with linguistic values NL, NS, Z, PS, and PL. These are characterized by the memberships shown in Figure 4.5.

A good choice of input and output fuzzy sets is crucial to the success of any fuzzy controller. The locations of the fuzzy sets in Figure 4.3 were chosen because the beam is 1 m long with $x = 0$ at the center, therefore x (hence e) is always such that $-0.5 \leq e \leq 0.5$ m. The fuzzy sets in Figure 4.4 were chosen based on our estimation of the maximum ball velocity expected in normal operation of the beam. The effective universe of discourse $-4 \leq \dot{e} \leq 4$ m/s was arrived at by realizing that if the ball is dropped from a stationary position, its velocity will reach -4.4 m/s when it has fallen 1 m. This is an approximation of the maximum ball velocity if it is located motionless at one end of the beam with a beam angle of $\psi = \pi/2$ (i.e., it falls vertically 1 m). This is admittedly a crude estimation, but it is important to try to find a meaningful range for $\dot{e}$, as it is for all inputs and outputs. The singleton fuzzy sets in Figure 4.5 were chosen based on our estimate of the voltage range needed to actuate the beam in order to accomplish the control task (i.e., $-10 \leq v \leq 10$V).

Recall that in Chapter 3 we saw that using singleton output fuzzy sets gave comparable results to triangular, Gaussian, or other types of output fuzzy sets under certain conditions. For some applications, especially classification [20], it may be necessary to be very meticulous about shaping output membership functions. However, for control applications, it is usually sufficient to use singleton output fuzzy sets because the controller can be otherwise adjusted by adjusting scaling gains or other parameters.

Using singleton output memberships simplifies defuzzification as well, since no areas under memberships of implied fuzzy sets need to be calculated. Efficiency of calculation is crucial for control with a digital computer due to short sampling times available for the fuzzy controller to perform its calculations. For these reasons, we will use only *product* T-norm and singleton output fuzzy sets in the remainder of this book.

4.1.2 Rule Base Determination: Ball and Beam Plant

The rule base consists of 25 rules, one rule for each combination of the e and $\dot{e}$ fuzzy sets. The rules should reflect our common sense about how to balance the ball motionless at the center of the beam. Thus, for instance, if the ball is in the position depicted in Figure 4.2, x is positive, therefore e is negative. If the ball is also moving further to the right, e is growing more negative, therefore $\dot{e}$ is negative. In this situation, it is obvious that v should be negative in order to rotate the beam in a counterclockwise direction to slow the ball's rightward progress and move it to the left toward the center of the beam.

This is an example of heuristic expert knowledge; it is based on our past experience and common sense, not on any mathematical model of the ball and beam. A complete rule base for the compensator can be constructed by using this "expert knowledge" in each of the 25 possible combinations of the e and $\dot{e}$ fuzzy sets. Below we address several scenarios to indicate how the rule base is constructed.

Case 1: e is NL and ė is NL

This situation is depicted in Figure 4.6 (the large arrow indicates fast movement to the right).

In this case, because the ball is moving quickly to the right away from the center of the beam, v should be large and negative (i.e., NL) in order to quickly rotate the beam counterclockwise to move the ball to the left.

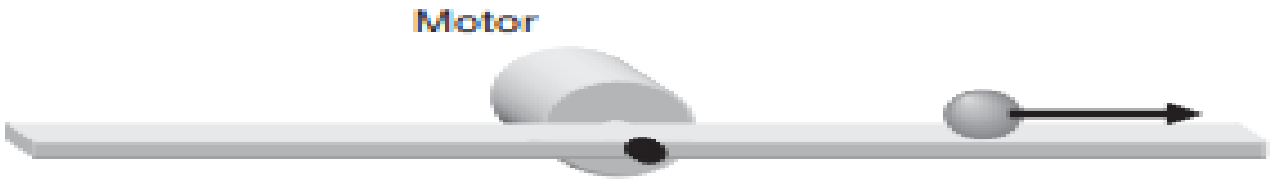


Figure 4.6. The parameter e is NL and $\dot{e}$ is NL.

Case 2: e is NL and ė is PL

This situation is depicted in Figure 4.7.

In this case, because the ball is moving quickly to the left toward the center of the beam, no rotation of the beam is necessary. Therefore, v should be zero (i.e., Z).

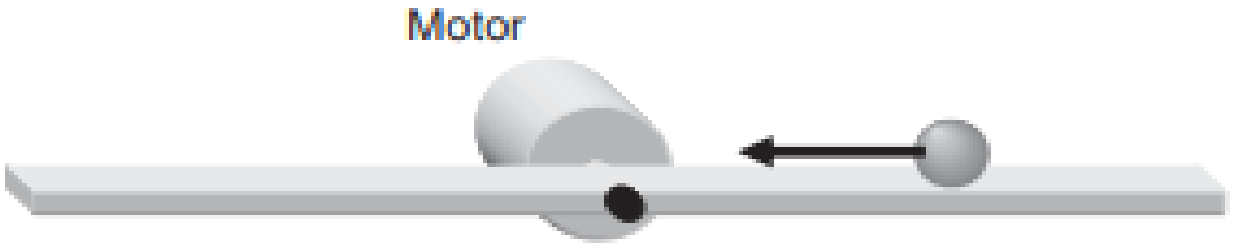


Figure 4.7. The parameter e is NL and c is PL.

Case 3: e is Z and ė is NS

This situation is depicted in Figure 4.8 (the small arrow indicates slow movement to the right).

In this case, because the ball is moving slowly to the right away from the center of the beam, a gentle counterclockwise rotation of the beam is necessary to move it leftward. Therefore, v should be a small negative amount (i.e., NS).

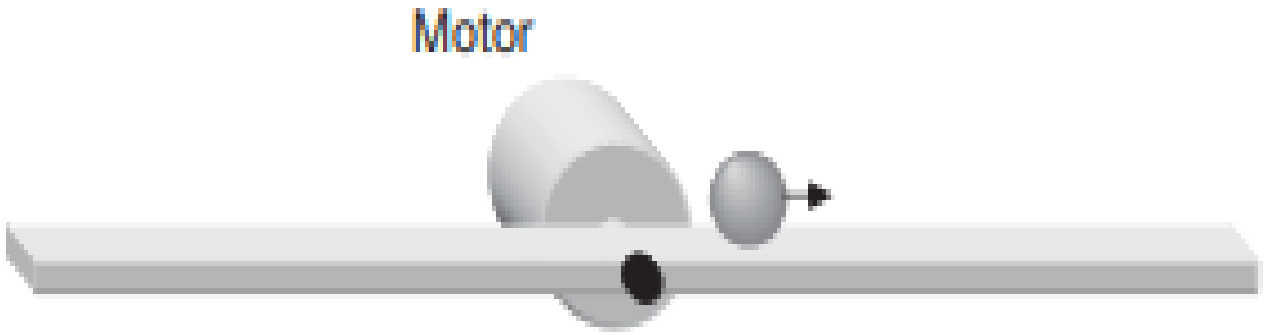


Figure 4.8. The parameter e is Z and $\dot{e}$ is NS.

The above three scenarios reflect our common sense (i.e., our "expert knowledge") about how to balance the ball motionless at the center of the beam. In this way, we can write all 25 rules in the rule base:

1. If e is NL and $\dot{e}$ is NL, then v is NL.
2. If e is NL and $\dot{e}$ is NS, then v is NL.
3. If e is NL and $\dot{e}$ is Z, then v is NL.
4. If e is NL and $\dot{e}$ is PS, then v is NS.
5. If e is NL and $\dot{e}$ is PL, then v is Z.
6. If e is NS and $\dot{e}$ is NL, then v is NL.
7. If e is NS and $\dot{e}$ is NS, then v is NL.
8. If e is NS and $\dot{e}$ is Z, then v is NS.
9. If e is NS and $\dot{e}$ is PS, then v is Z.
10. If e is NS and $\dot{e}$ is PL, then v is PS.
11. If e is Z and $\dot{e}$ is NL, then v is NL.
12. If e is Z and $\dot{e}$ is NS, then v is NS.
13. If e is Z and $\dot{e}$ is Z, then v is Z.
14. If e is Z and $\dot{e}$ is PS, then v is PS.
15. If e is Z and $\dot{e}$ is PL, then v is PL.
16. If e is PS and $\dot{e}$ is NL, then v is NS.
17. If e is PS and $\dot{e}$ is NS, then v is Z.
18. If e is PS and $\dot{e}$ is Z, then v is PS.
19. If e is PS and $\dot{e}$ is PS, then v is PL.
20. If e is PS and $\dot{e}$ is PL, then v is PL.
21. If e is PL and $\dot{e}$ is NL, then v is Z.
22. If e is PL and $\dot{e}$ is NS, then v is PS.
23. If e is PL and $\dot{e}$ is Z, then v is PL.
24. If e is PL and $\dot{e}$ is PS, then v is PL.
25. If e is PL and $\dot{e}$ is PL, then v is PL.

These are given in tabular form (Table 4.1) as follows:

TABLE 4.1 Tabulated Rule Base for Ball and Beam Controller

v \ $\dot{e}$	NL	NS	Z	PS	PL
NL	NL	NL	NL	NS	Z
NS	NL	NL	NS	Z	PS
e Z	NL	NS	Z	PS	PL
PS	NS	Z	PS	PL	PL
PL	Z	PS	PL	PL	PL

For later reference, let us define the *rule matrix* of this controller as

$$V_{e\dot{e}} = \begin{bmatrix} \text{NL} & \text{NL} & \text{NL} & \text{NS} & \text{Z} \\ \text{NL} & \text{NL} & \text{NS} & \text{Z} & \text{PS} \\ \text{NL} & \text{NS} & \text{Z} & \text{PS} & \text{PL} \\ \text{NS} & \text{Z} & \text{PS} & \text{PL} & \text{PL} \\ \text{Z} & \text{PS} & \text{PL} & \text{PL} & \text{PL} \end{bmatrix} \tag{4.2}$$

4.1.3 Inference: Ball and Beam Plant

Since *product* T-norm is used (see Section 3.6.5) the premise value of each rule at each time t is the product of the degrees of membership of $e(t)$ in the fuzzy set for e specified by the rule and $\dot{e}(t)$ in the fuzzy set for $\dot{e}$ specified by the rule. For instance, the premise value for Rule 1 at time t is

$$\mu_1(e(t), \dot{e}(t)) = \mu_e^{NL}(e(t))\,\mu_{\dot{e}}^{NL}(\dot{e}(t)) \tag{4.3}$$

Consider a particular time t at which $[e(t), \dot{e}(t)] = (-0.0625\,\text{m}, 3\,\text{m/s})$. This situation corresponds to the ball being slightly to the right of center and traveling rapidly to the left. Referring to Figure 4.3, a tracking error of $e = -0.0625$ is considered Z to an extent 0.75 and NS to an extent 0.25 (this e is not in any of the other three fuzzy sets on the e universe). Similarly, referring to Figure 4.4, $\dot{e} = 3$ is considered PL to an extent 0.5 and PS to an extent 0.5. Therefore, we have

$$\mu_e^{Z}(-0.0625) = 0.75 \tag{4.4a}$$
$$\mu_e^{NS}(-0.0625) = 0.25 \tag{4.4b}$$
$$\mu_{\dot{e}}^{PS}(3) = 0.5 \tag{4.4c}$$
$$\mu_{\dot{e}}^{PL}(3) = 0.5 \tag{4.4d}$$

Rules 9, 10, 14, and 15 are *on*, and the rest are not fired.

The premise values of the fired rules are

$$\mu_9(-0.0625, 3) = \mu_e^{NS}(-0.0625)\,\mu_{\dot{e}}^{PS}(3) = 0.25(0.5) = 0.125 \tag{4.5a}$$
$$\mu_{10}(-0.0625, 3) = \mu_e^{NS}(-0.0625)\,\mu_{\dot{e}}^{PL}(3) = 0.25(0.5) = 0.125 \tag{4.5b}$$
$$\mu_{14}(-0.0625, 3) = \mu_e^{Z}(-0.0625)\,\mu_{\dot{e}}^{PS}(3) = 0.75(0.5) = 0.375 \tag{4.5c}$$
$$\mu_{15}(-0.0625, 3) = \mu_e^{Z}(-0.0625)\,\mu_{\dot{e}}^{PL}(3) = 0.75(0.5) = 0.375 \tag{4.5d}$$

4.1.4 Defuzzification: Ball and Beam Plant

Since the output memberships are singletons (see Section 3.6.5), the crisp output of the fuzzy controller at time t, which is the voltage input to the motor at time t, is calculated using center average defuzzification as

$$v(t) = \frac{\sum_{i=1}^{25} q_i \mu_i(e(t), \dot{e}(t))}{\sum_{i=1}^{25} \mu_i(e(t), \dot{e}(t))} = \frac{q_9\mu_9 + q_{10}\mu_{10} + q_{14}\mu_{14} + q_{15}\mu_{15}}{\mu_9 + \mu_{10} + \mu_{14} + \mu_{15}} \tag{4.6}$$

where q_i is the location of the membership function characterizing the singleton fuzzy set specified in the consequent of Rule i. The crisp output of the fuzzy controller for the inputs $[e(t), \dot{e}(t)] = (-0.0625, 3)$ is

$$v(t) = \frac{0(0.125) + 5(0.125) + 5(0.375) + 10(0.375)}{0.125 + 0.125 + 0.375 + 0.375} = 6.25\ V \tag{4.7}$$

Thus the controller commands a clockwise rotation in order to stop the ball's leftward motion. Note that the denominator in (4.7) is unity due to the partitions of unity on the e and $\dot{e}$ universes (Figs. 4.3 and 4.4).

4.2 TUNING FOR IMPROVED PERFORMANCE BY ADJUSTING SCALING GAINS

As is done in many control systems both fuzzy and nonfuzzy, let us add adjustable scaling gains g_0 and g_1 for inputs e and $\dot{e}$, respectively, and adjustable scaling gain h for output v. These gains are used to tune the compensator to achieve better performance. The closed-loop system is shown in Figure 4.9, where the contents of the fuzzy compensator block are shown in Figure 4.10.

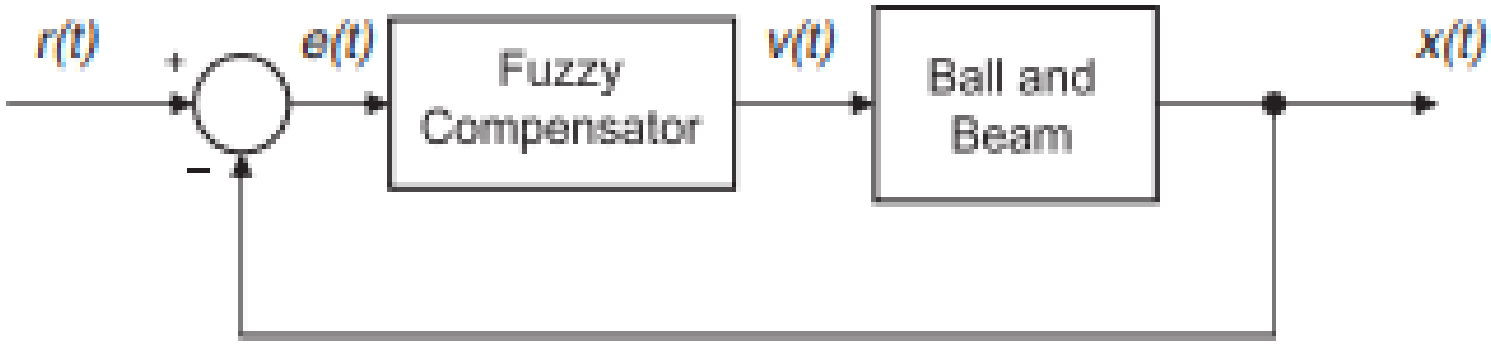


Figure 4.9. Closed-loop controller for ball and beam.

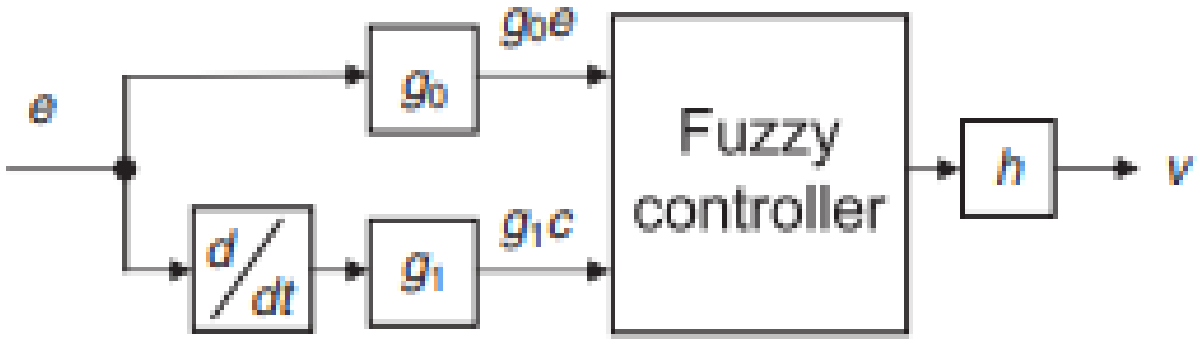


Figure 4.10. Fuzzy compensator block of Figure 4.9.

When the closed-loop system is simulated using a fourth-order Runge–Kutta integration routine with a step size of $\Delta t = 0.001$ s and an initial ball position of

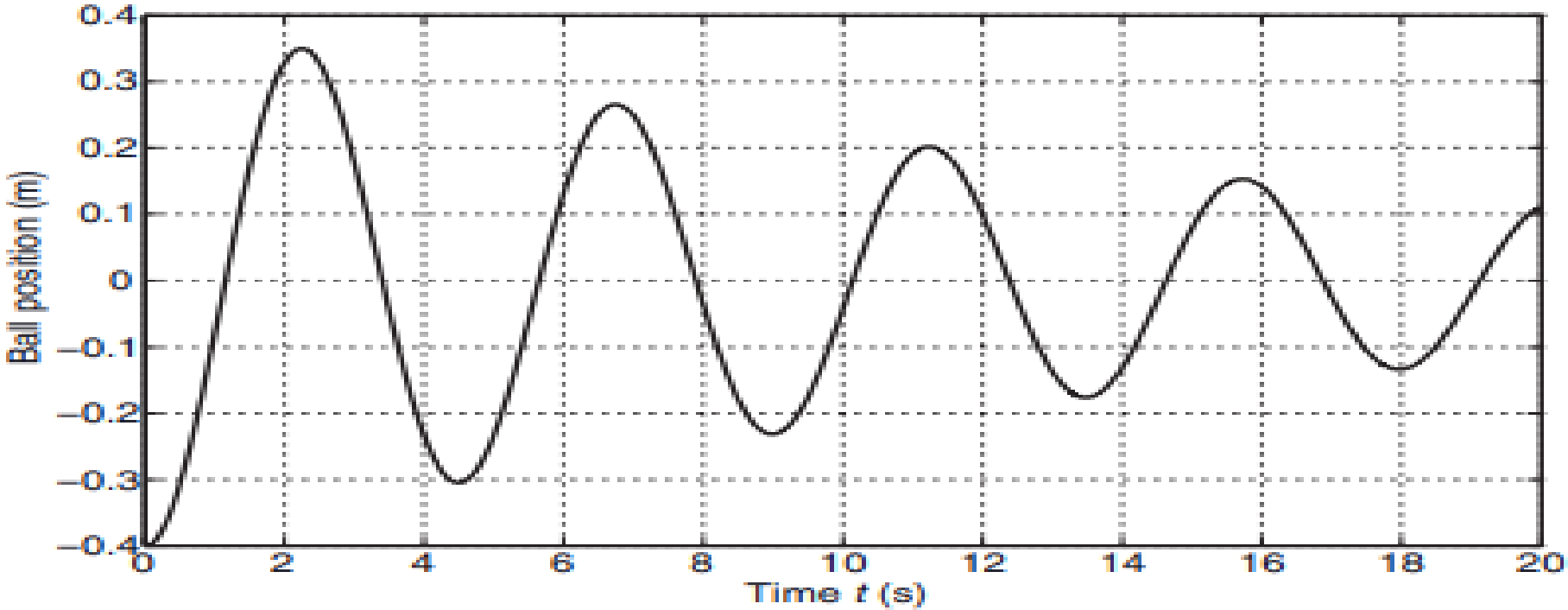


Figure 4.11. Ball travel, scaling gains $g_0 = 1$, $g_1 = 1$, $h = 1$.

$x(0) = -0.4$ m, the resulting ball movement is shown in Figure 4.11. The ball moves toward zero, however, the response is very oscillatory. The fuzzy controller is doing its job, but must be tuned to quicken the response and get rid of the oscillations. This can be done by adjusting the gains g_0, g_1, and h.

In some cases, the fuzzy controller originally designed with all scaling gains equaling unity may not produce a stable closed-loop system, even though the controller is correctly designed according to our best judgment. In such cases, the problem is likely that the universes of discourse are incorrectly sized for the problem. Then it may be necessary to adjust the scaling gains initially to produce a stable closed-loop system before the gains are further adjusted to improve the response.

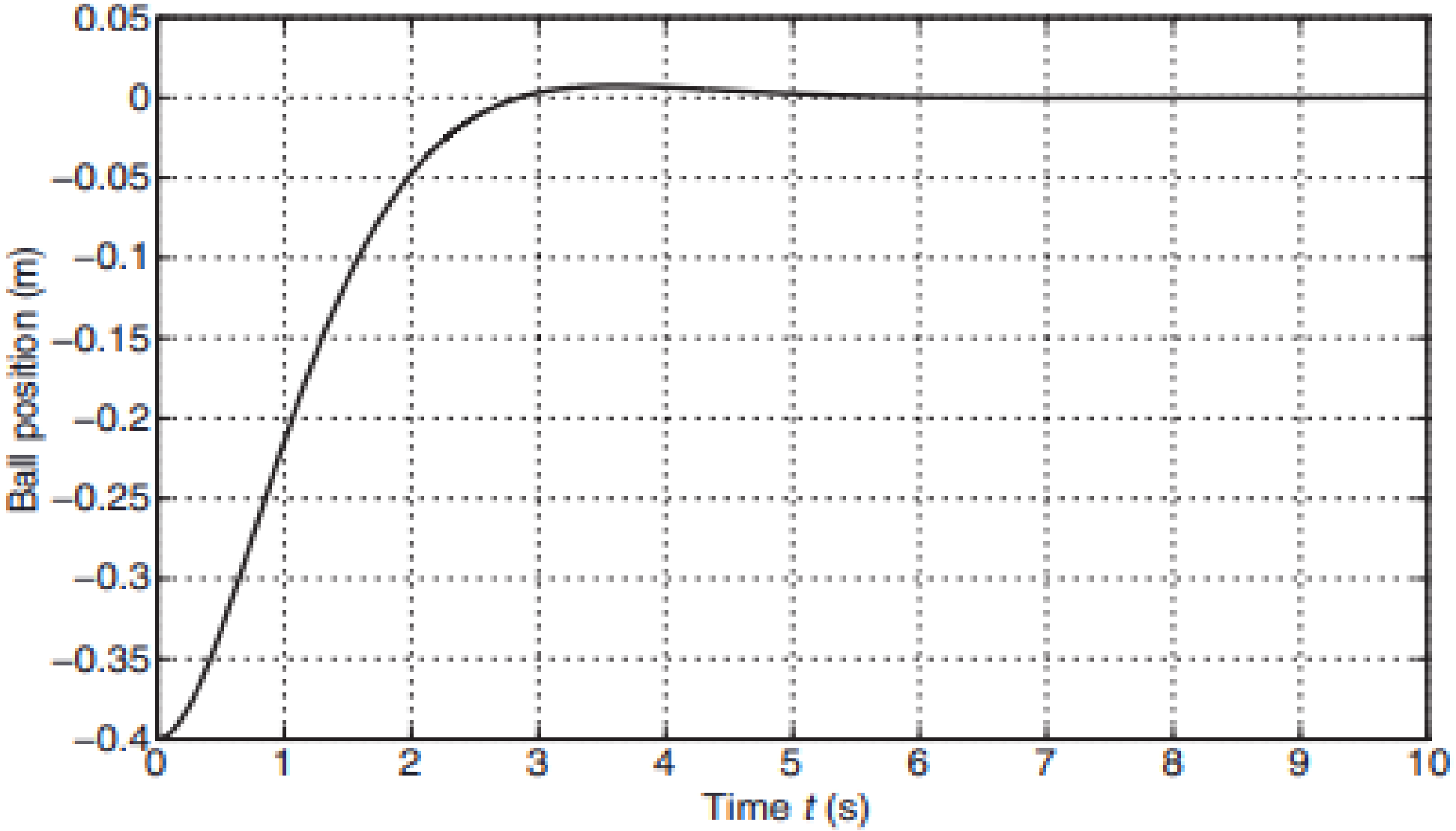


Figure 4.12. Ball travel, scaling gains $g_0 = 1$, $g_1 = 18$, $h = 1$.

When g_1 is increased to 18 to get rid of the oscillations, the behavior shown in Figure 4.12 results. This reduces the oscillations significantly, but the ball takes 6 s to settle motionless at the center of the beam, which is too slow.

In order to quicken the response, g_0 is increased to 3 while keeping $g_1 = 18$. The resulting response is shown in Figure 4.13. The time taken by the ball to reach the center of the beam is significantly decreased, however, the ball overshoots its target.

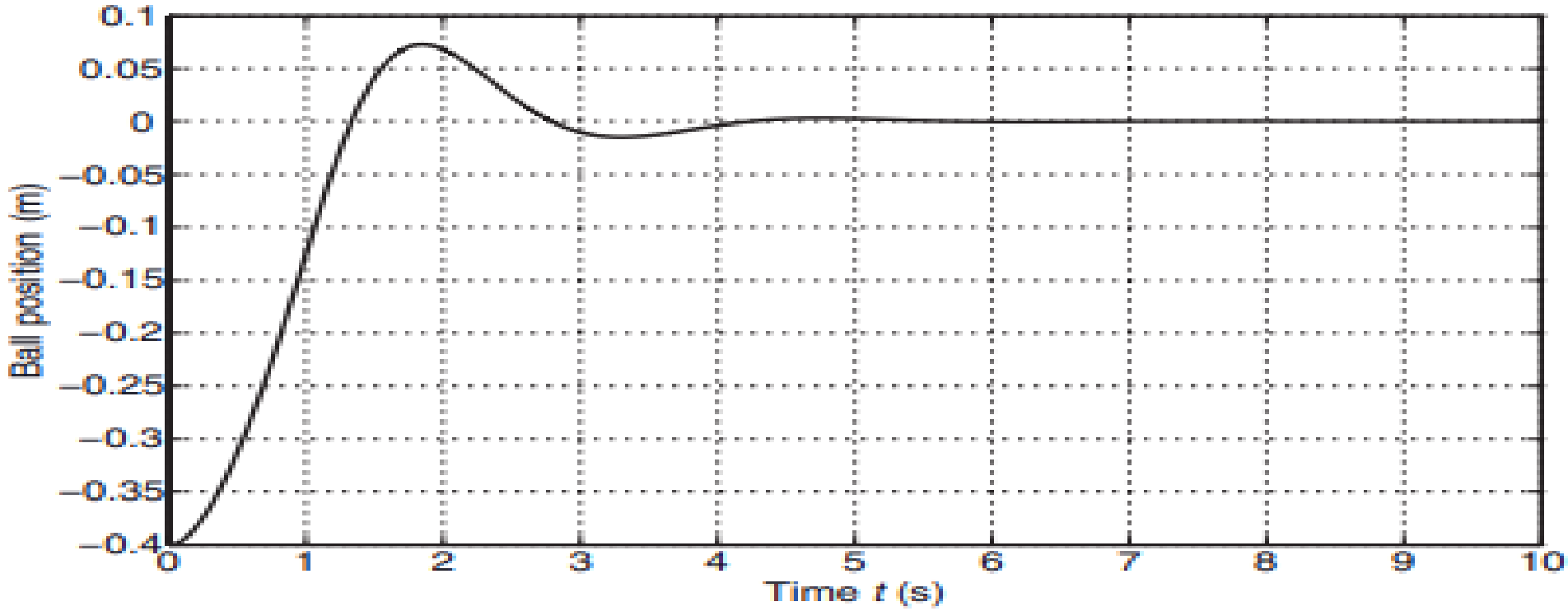


Figure 4.13. Ball travel, scaling gains $g_0 = 3$, $g_1 = 18$, $h = 1$.

When the output scaling gain h is increased to 7 while keeping g_0 and g_1 unchanged, the response of Figure 4.14 results.

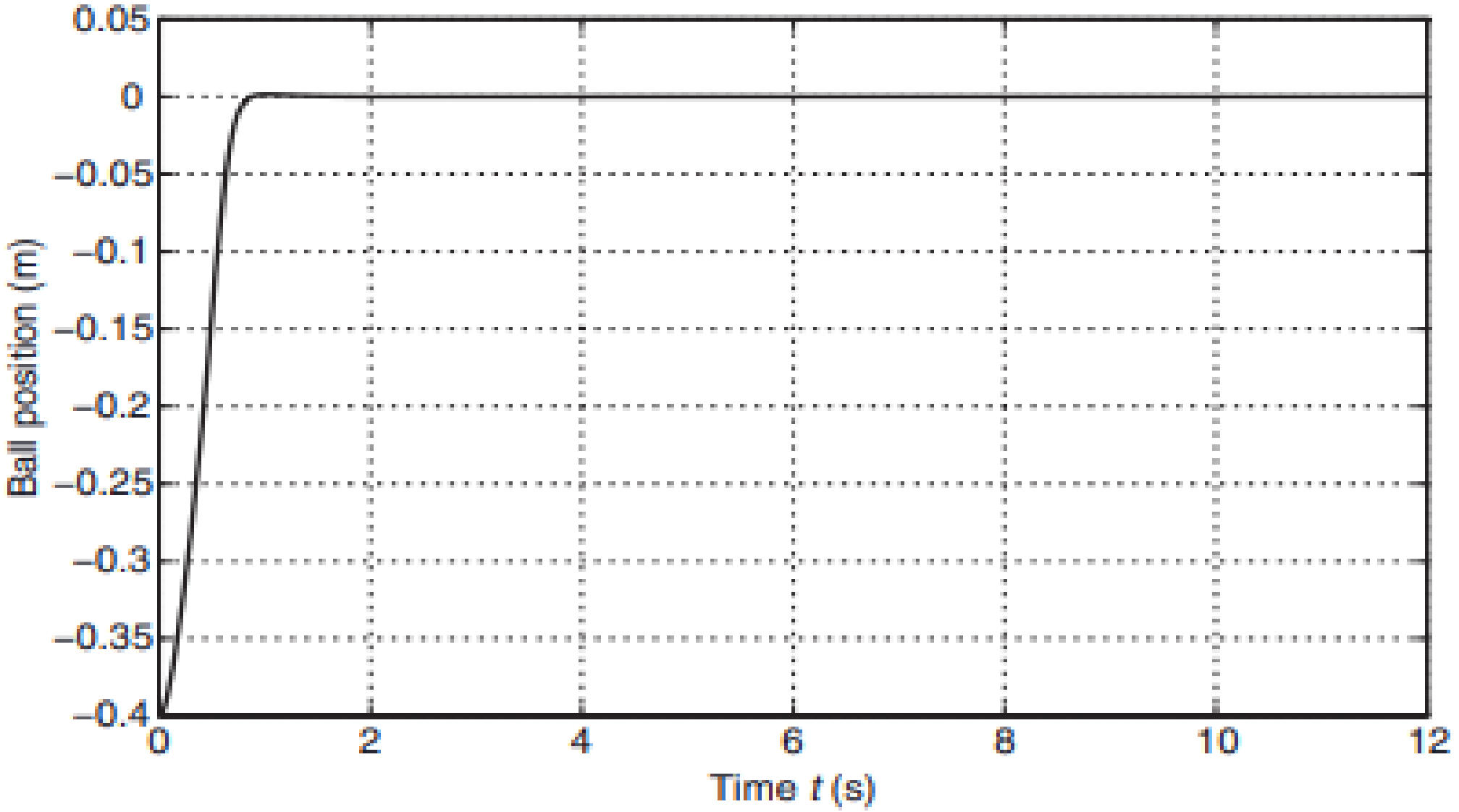


Figure 4.14. Ball travel, scaling gains $g_0 = 3$, $g_1 = 18$, $h = 7$.

This is a satisfactory response; the ball settles at the center of the beam within 1 s with no overshoot. However, note that, with $h = 7$, the controller now requires that the voltage source be able to supply voltages in the range $-70 \leq v \leq 70\,V$ at currents demanded by the motor. If a sufficiently powerful supply is not available, it may be necessary to accept a slower response. The control effort necessary to accomplish a control task is always a concern in the implementation of every practical control system.

The beam angle producing the response of Figure 4.14 is shown in Figure 4.15. It is seen that the beam angle jumps instantaneously to an angle of 40° to achieve the rapid ball response of Figure 4.14. This may be a problem: Such a quick movement may throw the ball off the beam! The output gain $h = 7$ was chosen because values of h less then 7 result in a slower response with overshoot, while values greater than 7 produce no significant improvement in the 1 s settling time seen in Figure 4.14.

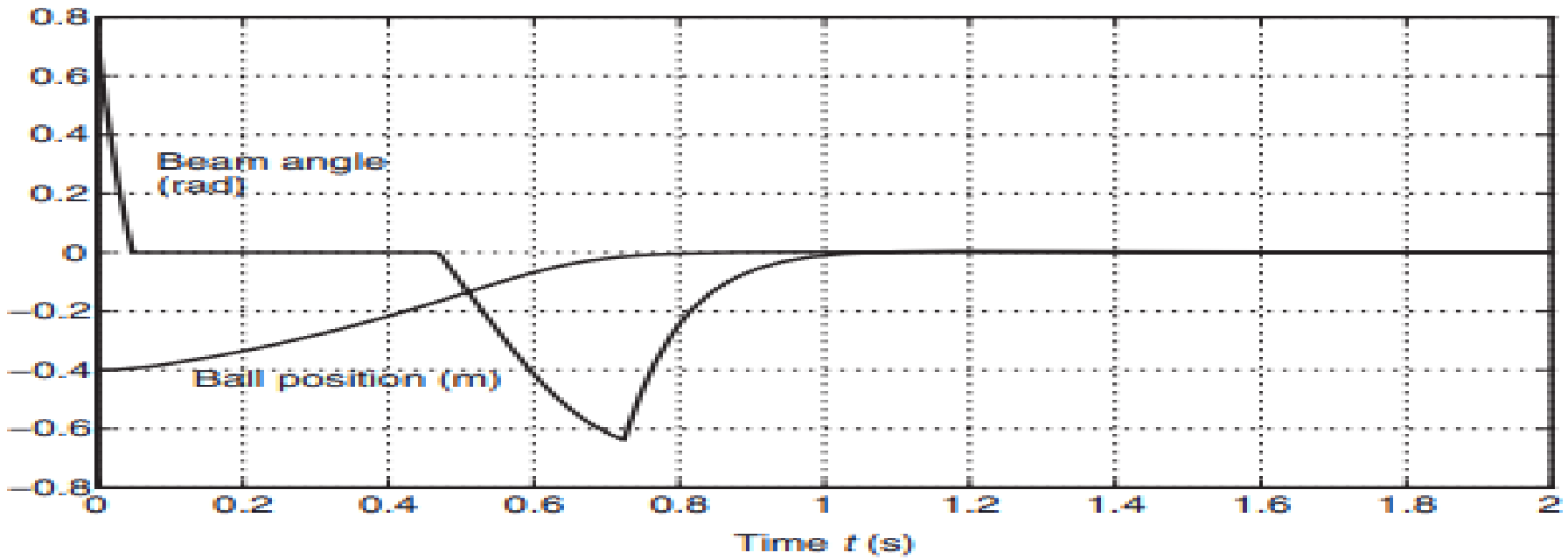


Figure 4.15. Ball travel (m) and corresponding beam angle (rad). $g_0 = 3$, $g_1 = 18$, $h = 7$.

4.3 EFFECT OF INPUT MEMBERSHIP FUNCTION SHAPES

Let us repeat the above controller design using Gaussian input membership functions rather than triangular. Therefore, the e and $\dot{e}$ fuzzy sets are characterized by the memberships in Figures 4.16 and 4.17. Note that, similar to triangular membership functions, adjacent Gaussians may cross wherever desired. The spreads of the Gaussians of Figures 4.16 and 4.17 have been chosen so that adjacent memberships cross each other at 0.5.

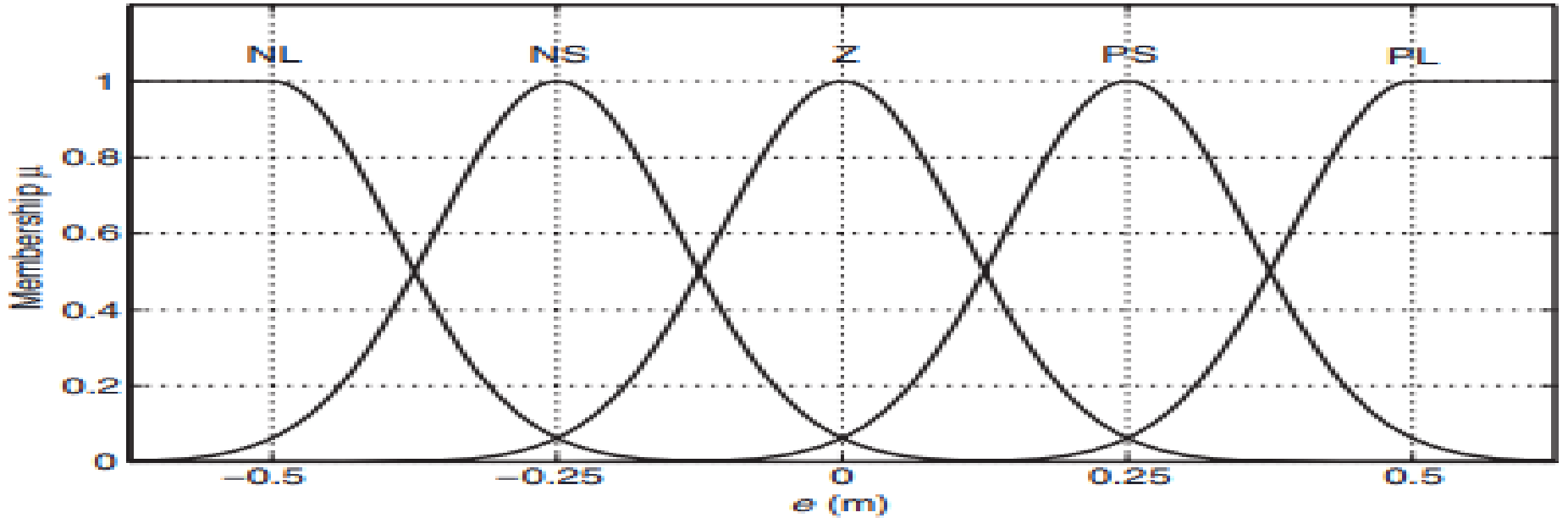


Figure 4.16. Gaussian fuzzy sets on e universe.

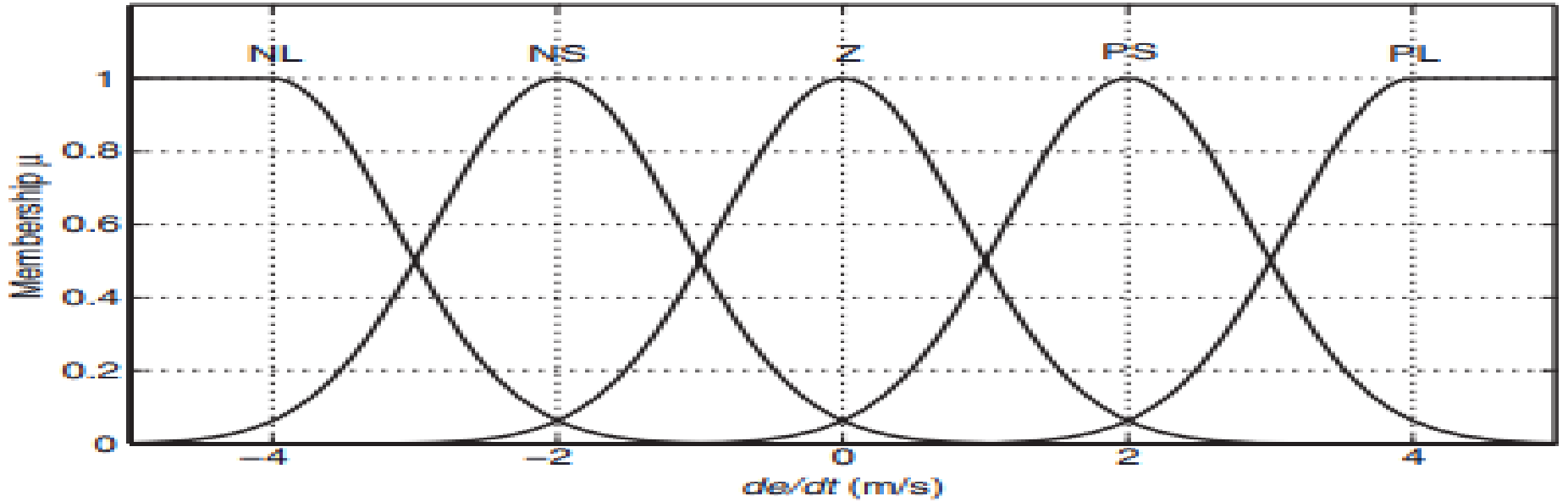


Figure 4.17. Gaussian fuzzy sets on $\dot{e}$ universe.

The rule base of Section 4.1.2 has not changed. However, the inference calculation is different due to the different membership function shapes. As above, consider a particular time t at which $[e(t), \dot{e}(t)] = (-0.0625\,\text{m}, 3\,\text{m/s})$. Referring to Figure 4.16, a tracking error of $e = -0.0625\,\text{m}$ is considered NL, NS, Z, PS, and PL all to nonzero extents. Specifically, we have the following:

$$\mu_e^{\text{NL}}(-0.0625) = 2.0530 \times 10^{-4} \tag{4.8a}$$

$$\mu_e^{\text{NS}}(-0.0625) = 0.2102 \tag{4.8b}$$

$$\mu_e^{\text{Z}}(-0.0625) = 0.8409 \tag{4.8c}$$

$$\mu_e^{\mathrm{PS}}(-0.0625) = 0.0131 \tag{4.8d}$$
$$\mu_e^{\mathrm{PL}}(-0.0625) = 8.0194 \times 10^{-7} \tag{4.8e}$$

Similarly, referring to Figure 4.17, $\dot{e} = 3$ m/s is considered NL, NS, Z, PS, and PL all to nonzero extents. Specifically, we have the following:

$$\mu_{\dot{e}}^{\mathrm{NL}}(3) = 1.7764 \times 10^{-15} \tag{4.9a}$$
$$\mu_{\dot{e}}^{\mathrm{NS}}(3) = 2.9802 \times 10^{-8} \tag{4.9b}$$
$$\mu_{\dot{e}}^{\mathrm{Z}}(3) = 0.0020 \tag{4.9c}$$
$$\mu_{\dot{e}}^{\mathrm{PS}}(3) = 0.5 \tag{4.9d}$$
$$\mu_{\dot{e}}^{\mathrm{PL}}(3) = 0.5 \tag{4.9e}$$

Because all input memberships are nonzero, all 25 rules are fired to some extent. However, the degrees of firing of most rules are very small. To avoid calculating the degrees of firing for rules that are minimally fired, let us consider a quantity to be in a fuzzy set only if its degree of membership in that set is at least 0.1. That is, we consider a quantity to be in a fuzzy set only if it is in the 0.1-cut of that set. Using this criterion, the ball position $e(t) = -0.0625$ m qualifies as Z to an extent 0.8409, as NS to an extent 0.2102, and not as any other linguistic value on the e universe. Similarly, the ball velocity $\dot{e}(t) = 3$ m/s qualifies as PS to an extent 0.5, as PL to an extent 0.5, and not as any other linguistic value on the $\dot{e}$ universe.

Once again, rules 9, 10, 14, and 15 are *on*, and the rest are not fired. The premise values of the fired rules are

$$\mu_9(-0.0625, 3) = \mu_e^{\mathrm{NS}}(-0.0625)\,\mu_{\dot{e}}^{\mathrm{PS}}(3) = 0.2102\,(0.5) = 0.1051 \tag{4.10a}$$
$$\mu_{10}(-0.0625, 3) = \mu_e^{\mathrm{NS}}(-0.0625)\,\mu_{\dot{e}}^{\mathrm{PL}}(3) = 0.2102\,(0.5) = 0.1051 \tag{4.10b}$$
$$\mu_{14}(-0.0625, 3) = \mu_e^{\mathrm{Z}}(-0.0625)\,\mu_{\dot{e}}^{\mathrm{PS}}(3) = 0.8409\,(0.5) = 0.4204 \tag{4.10c}$$
$$\mu_{15}(-0.0625, 3) = \mu_e^{\mathrm{Z}}(-0.0625)\,\mu_{\dot{e}}^{\mathrm{PL}}(3) = 0.8409\,(0.5) = 0.4204 \tag{4.10d}$$

The crisp output is calculated as in (4.6):

$$v = \frac{0(0.1051)+5(0.1051)+5(0.4204)+10(0.4204)}{0.1051+0.1051+0.4204+0.4204} = 6.5\,V \qquad (4.11)$$

This is similar to the crisp output calculated in (4.7) for triangular memberships. In fact, the closed-loop behavior of the plant with Gaussian controller is very close to that of the plant with triangular controller. This is not surprising, because the shape of Gaussian memberships is similar to that of triangular memberships. Neither controller is "better." The difference is analogous to the difference that occurs when two different people try to balance the ball on the beam by hand.

The input–output characteristic of the fuzzy controller with triangular memberships is shown in Figure 4.18, and that of the controller with Gaussian memberships is shown in Figure 4.19. The two characteristics are essentially the same, except for the local waves and undulations in the characteristic of Figure 4.19. These were discussed in Section 3.6.1. These local waves are due to the curvature of the Gaussians; they have nothing to do with the controller itself.

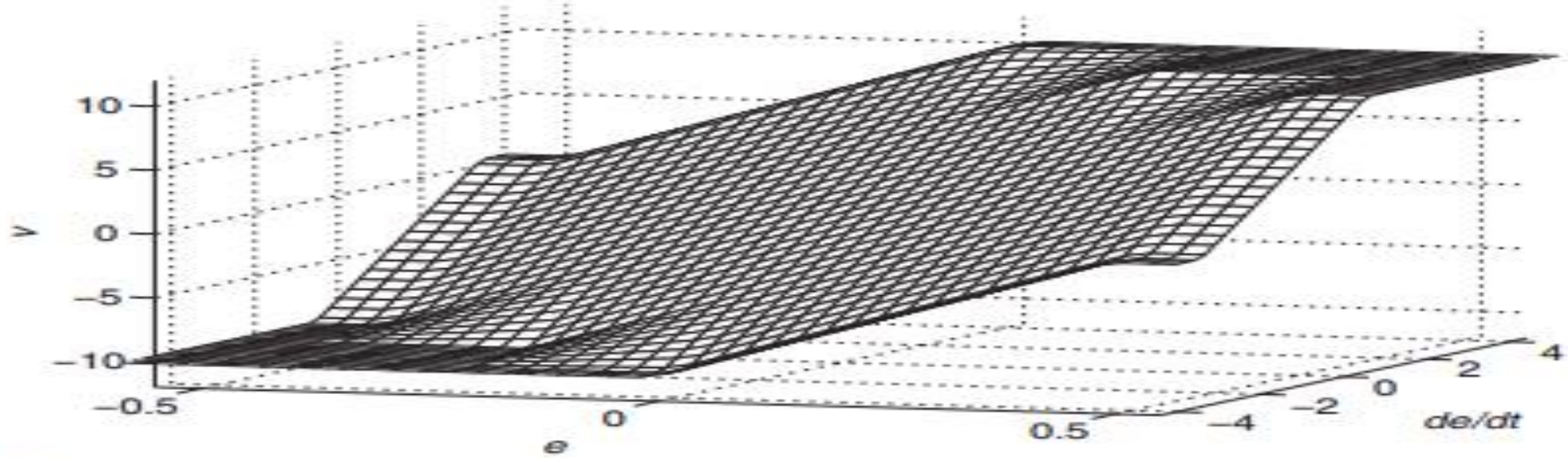


Figure 4.18. Input–output characteristic of fuzzy controller with triangular input memberships, *product* T-norm, and singleton output fuzzy sets.

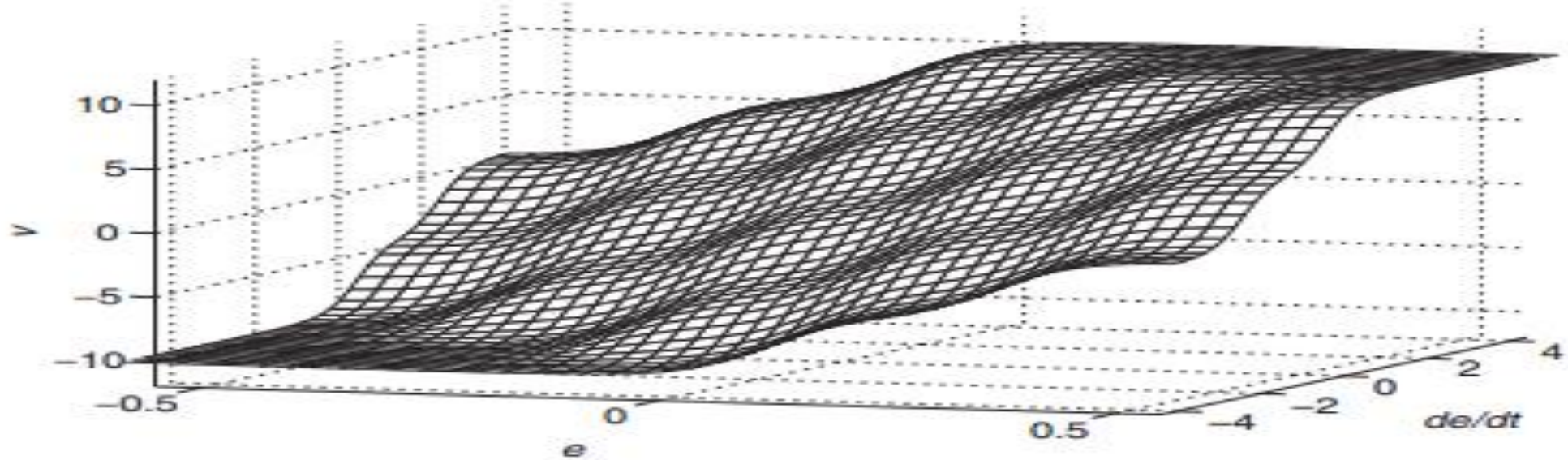


Figure 4.19. Input–output characteristic of fuzzy controller with Gaussian input memberships, *product* T-norm, and singleton output fuzzy sets.

4.4 CONVERSION OF PID CONTROLLERS INTO FUZZY CONTROLLERS

There is an exact correspondence between PID controllers and certain fuzzy controllers. This makes it possible to derive a fuzzy controller from a PID controller. It is very useful to be able to do this because designing a fuzzy controller can be initially difficult (as was seen in Sections 4.1 and 4.2). However, there exist well-known methods of designing PID controllers (e.g., Ziegler–Nichols tuning [3]).

Even when these methods are not used, an expert can often manually adjust a PID controller to achieve stable performance. For instance, in many applications if the response is too sluggish, it can be quickened by increasing the proportional

gain. Similarly, an effective strategy if the response is too oscillatory is often to increase the derivative gain. If the steady-state error is too large, it can be reduced by adjusting the integral gain. Using these and similar rules of thumb, an effective PID controller can be designed for a plant relatively easily (see Section 4.2, where these rules were used to tune the scaling gains g_0 and g_1). If this PID controller can then be converted into a fuzzy controller, it may be easier to adjust the fuzzy controller in a nonlinear manner to enhance robustness.

Consider a fuzzy PD controller with inputs $e(t)$ and $\dot{e}(t)$ and output $u(t)$ (the development for PI controllers and PID controllers is similar). Let the e universe of discourse contain n symmetrical triangular fuzzy sets forming a partition of unity, where n is odd, such that the middle triangle is centered at $e = 0$. Similarly, let the $\dot{e}$ universe contain n symmetrical triangular fuzzy sets forming a partition of unity such that the middle triangle is centered at $\dot{e} = 0$. Let the u universe contain $2n - 1$ equally spaced singleton fuzzy sets $Q^1, \ldots, Q^{2n-1}$, such that the middle singleton Q^n is located at $u = 0$. Finally, let the rule base of the controller be specified by the $n \times n$ rule matrix [see (4.2)]

$$U_{ut} = \begin{bmatrix} Q^1 & Q^2 & Q^3 & \cdots & Q^n \\ Q^2 & Q^3 & Q^4 & \cdots & Q^{n+1} \\ Q^3 & Q^4 & Q^5 & \cdots & Q^{n+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ Q^n & Q^{n+1} & Q^{n+2} & & Q^{2n-1} \end{bmatrix} \tag{4.12}$$

When *product* T-norm and center-average defuzzification are used, a fuzzy system constructed as above exhibits a linear input–output characteristic within its effective universe of discourse (see, e.g., Fig. 4.18). Let the effective universes of discourse for e, $\dot{e}$, and u be $[-e_{\max}\ e_{\max}]$, $[-\dot{e}_{\max}\ \dot{e}_{\max}]$, and $[-u_{\max}\ u_{\max}]$, respectively. Then, the fuzzy PD controller is equivalent to the following nonfuzzy PD controller within the fuzzy controller's effective universe (i.e., within the linear portion of the controller's input–output characteristic):

$$u(t) = \frac{u_{\max}}{e_{\max}} e(t) + \frac{u_{\max}}{\dot{e}_{\max}} \dot{e}(t) \tag{4.13}$$

Therefore, the fuzzy controller whose characteristic is shown in Figure 4.18 is equivalent to the nonfuzzy PD controller

$$u(t) = \frac{10}{0.5} e(t) + \frac{10}{4} \dot{e}(t) = 20e(t) + 2.5\dot{e}(t) \tag{4.14}$$

This suggests a method for deriving a fuzzy controller from a nonfuzzy PD controller. Assume that we have a nonfuzzy PD controller that gives satisfactory closed-loop behavior. In order for the equivalent fuzzy controller to be exactly equal to the nonfuzzy PD, it is necessary that the system trajectory always remain within the linear part of the characteristic. Therefore, with the nonfuzzy PD in operation, measure the maximum control effort necessary to accomplish the control task. This is the maximum absolute value of $u(t)$ that is output by the PD in controlling the plant. Call this $u_{\max}$. Then if the nonfuzzy PD controller is given by

$$u(t) = K_p e(t) + K_d \dot{e}(t) \tag{4.15}$$

the equivalent fuzzy controller has inputs $e(t)$, $\dot{e}(t)$, and output $u(t)$ with effective universes $\left[-\frac{\gamma u_{\max}}{K_p}, \frac{\gamma u_{\max}}{K_p}\right]$, $\left[-\frac{\gamma u_{\max}}{K_d}, \frac{\gamma u_{\max}}{K_d}\right]$, and $[-2\gamma u_{\max} \quad 2\gamma u_{\max}]$, respectively. The constant $\gamma > 1$ is to guarantee the system trajectories remain within the linear portion of the fuzzy system's input–output characteristic. Usually $\gamma = 2$ will suffice, but any sufficiently large γ will produce identical closed-loop behavior to the non-fuzzy PD compensator. After the equivalent fuzzy controller has been constructed, it can be easily altered to improve performance.

EXAMPLE 4.1

Consider the inverted pendulum plant of Figure 1.2. For simulation purposes, its mathematical model is given by

$$\ddot{\psi} = \frac{9.81 \sin \psi - \frac{2}{3} \cos \psi \left(0.25 \dot{\psi}^2 \sin \psi + F\right)}{0.5\left(\frac{4}{3} - \frac{1}{3} \cos^2 \psi\right)} \tag{4.16}$$

The error e is defined as $e = r(t) - \psi(t)$, where $\psi(t)$ is the angle of the rod from vertical and $r(t)$ is a reference trajectory. Suppose it has been determined that the PD controller

$$F(t) = -(30e(t) + 5\dot{e}(t)) \tag{4.17}$$

quickly balances the pendulum in the vertical-up position with no overshoot. Figures 4.20 and 4.21 show the rod angle response and corresponding PD controller output, respectively for an initial rod angle of −0.1 rad.

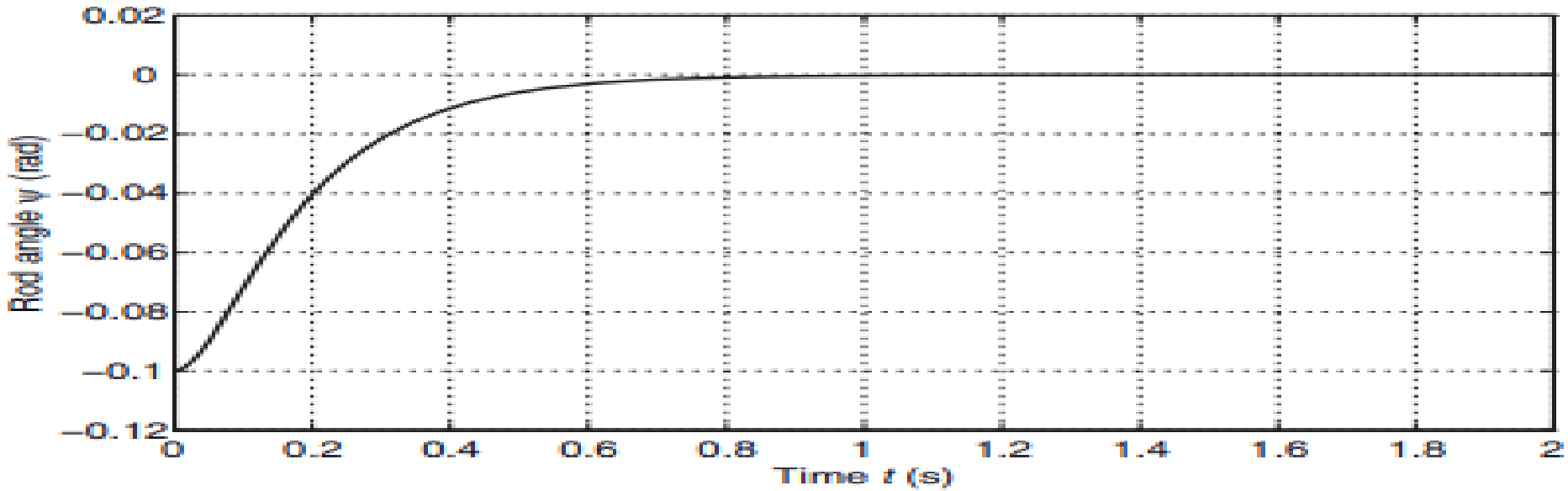


Figure 4.20. Rod angle response under nonfuzzy PD control (4.17).

From Figure 4.21, the maximum absolute value of the control effort is 3N. If we use $n = 5$ fuzzy sets for $e(t)$ and $\dot{e}(t)$, the input and output universes given in Figures 4.22–4.24 result.

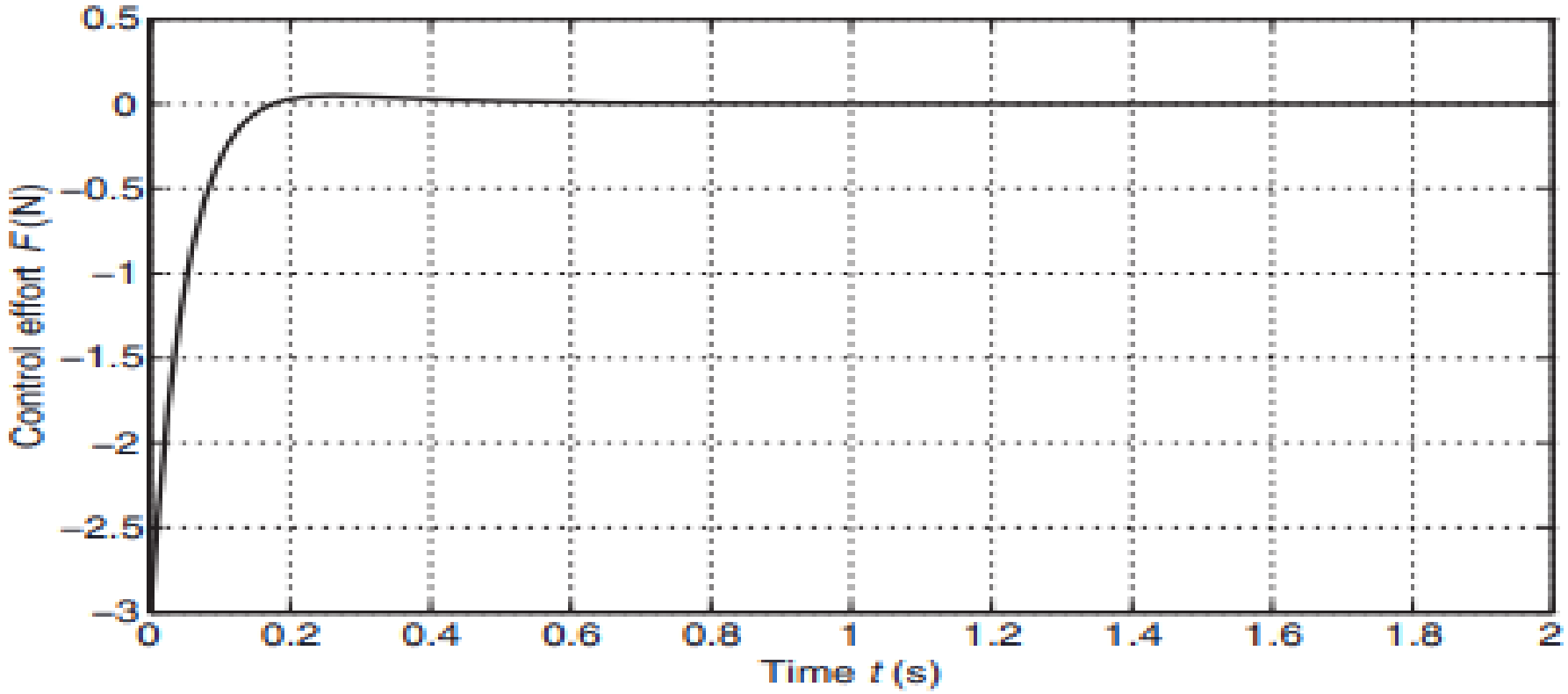


Figure 4.21. Control effort [output of nonfuzzy PD controller (4.17)] producing response of Figure 4.20.

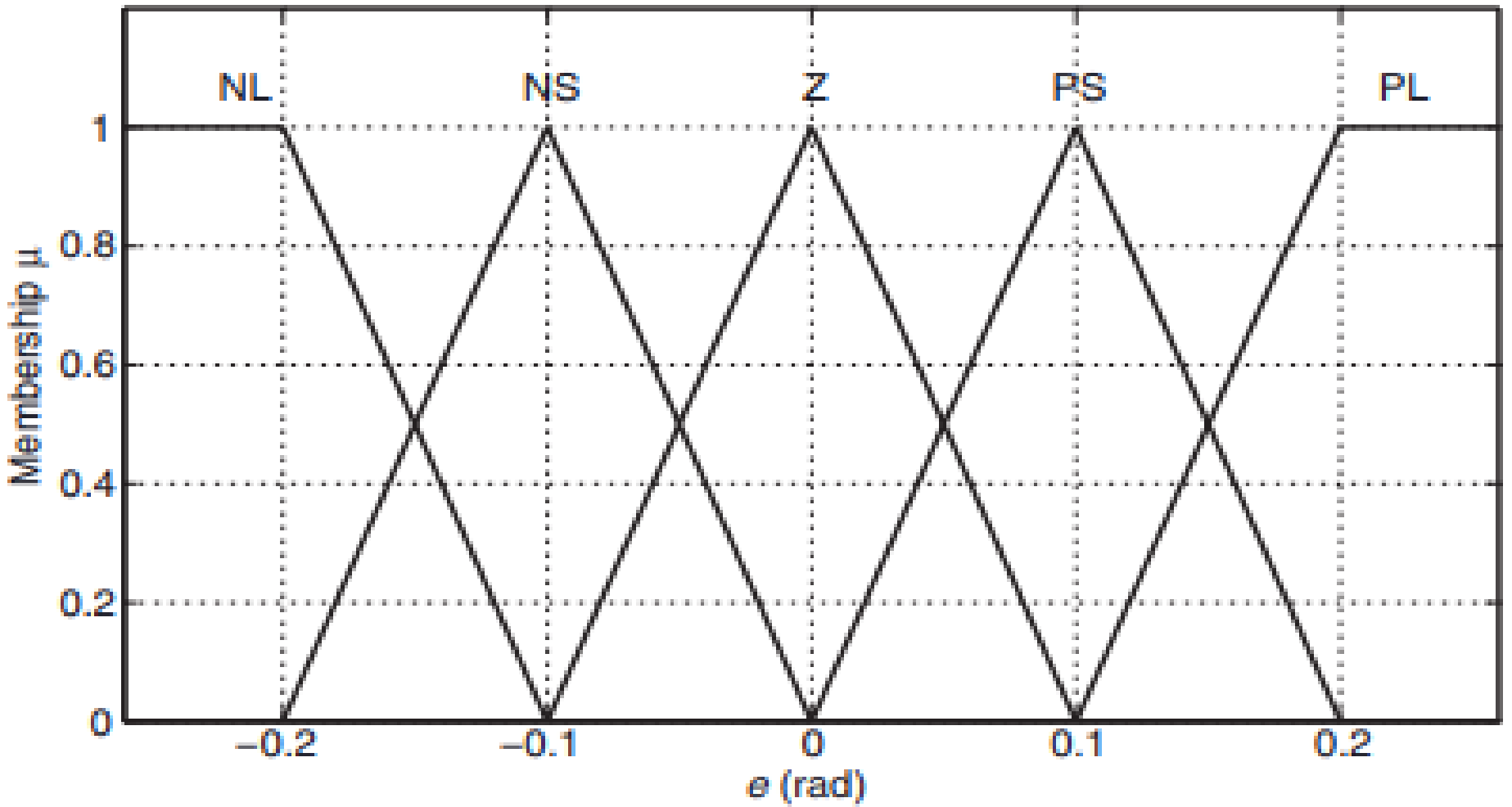


Figure 4.22. Fuzzy sets on e universe for equivalent fuzzy PD controller for inverted pendulum.

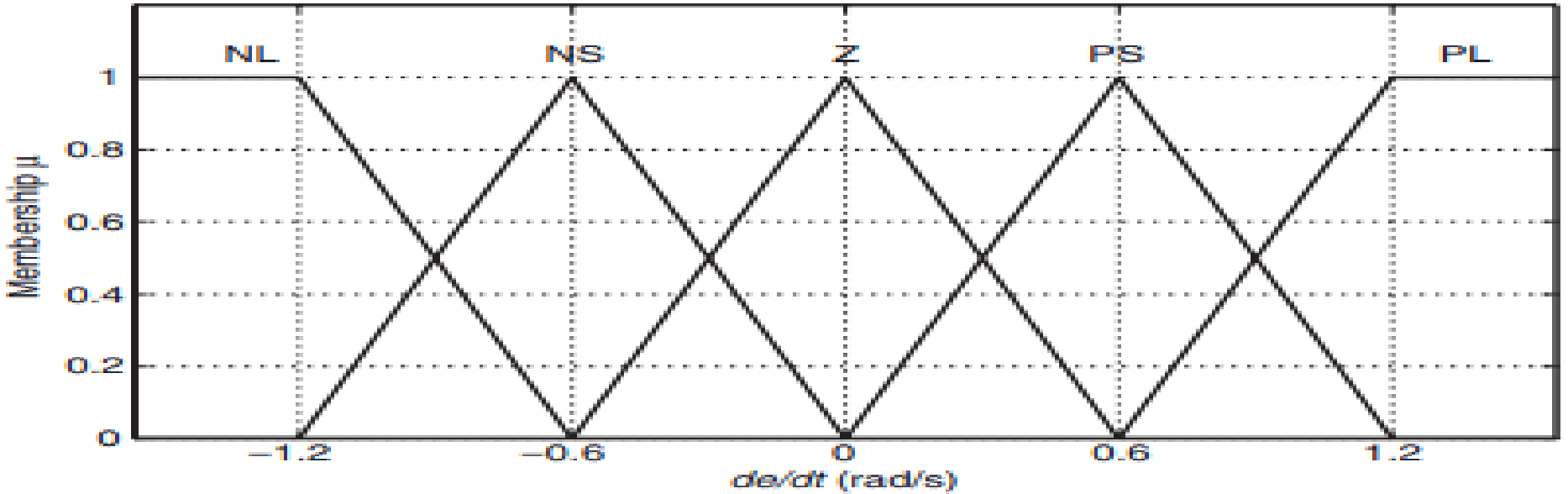


Figure 4.23. Fuzzy sets on $\dot{e}$ universe for equivalent fuzzy PD controller for inverted pendulum.

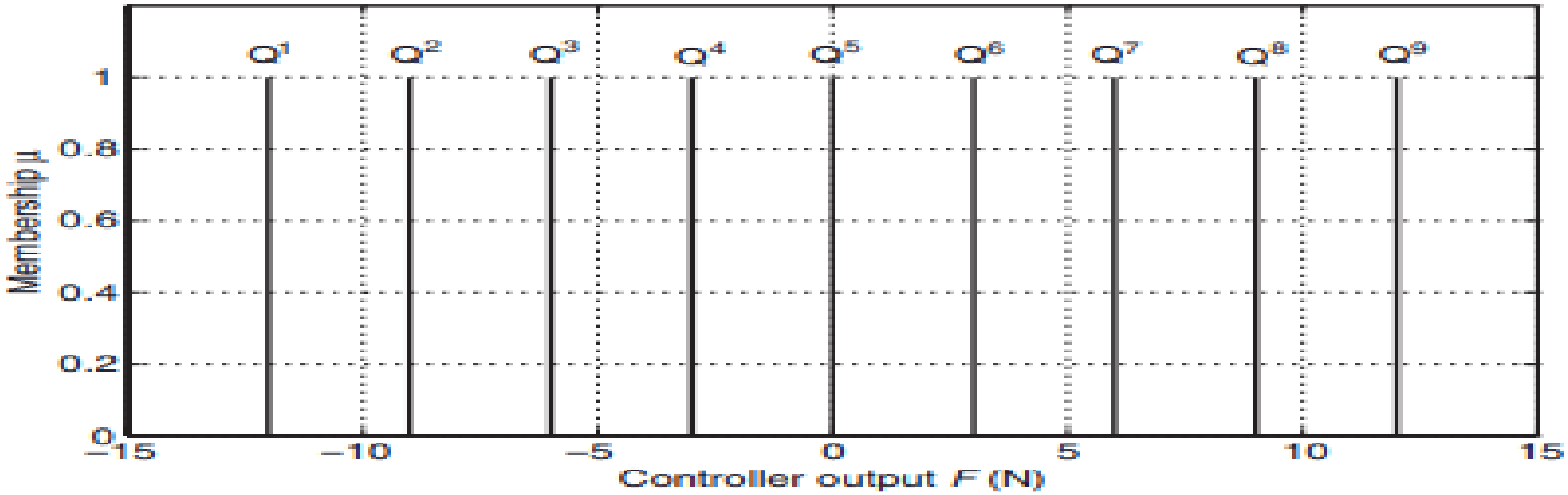


Figure 4.24. Fuzzy sets on F universe for equivalent fuzzy PD controller for inverted pendulum.

The tabulated rule base of the controller is, from (4.12),

TABLE 4.2 Tabulated Rule Base for Inverted Pendulum Controller

F		$\dot{e}$				
		NL	NS	Z	PS	PL
e	NL	Q^1	Q^2	Q^3	Q^4	Q^5
	NS	Q^2	Q^3	Q^4	Q^5	Q^6
	Z	Q^3	Q^4	Q^5	Q^6	Q^7
	PS	Q^4	Q^5	Q^6	Q^7	Q^8
	PL	Q^5	Q^6	Q^7	Q^8	Q^9

Since the nonfuzzy PD control law (4.17) is negated, the crisp output of the fuzzy compensator should be negated as well. The performance of the inverted pendulum in closed loop with this fuzzy PD compensator is identical to Figures 4.20 and 4.21 because the fuzzy compensator is identical to the nonfuzzy PD (4.17) initially designed for this plant.

□

4.4.1 Redesign for Increased Robustness

Let the rod be balanced in the vertical-up position with the above fuzzy PD controller. If the cart is bumped with an impulsive force of 50 N, this disturbance is enough to cause the rod to fall with this controller. Figure 4.25 shows this situation, where a 50 N force is applied between 1.5 and 1.55 s.

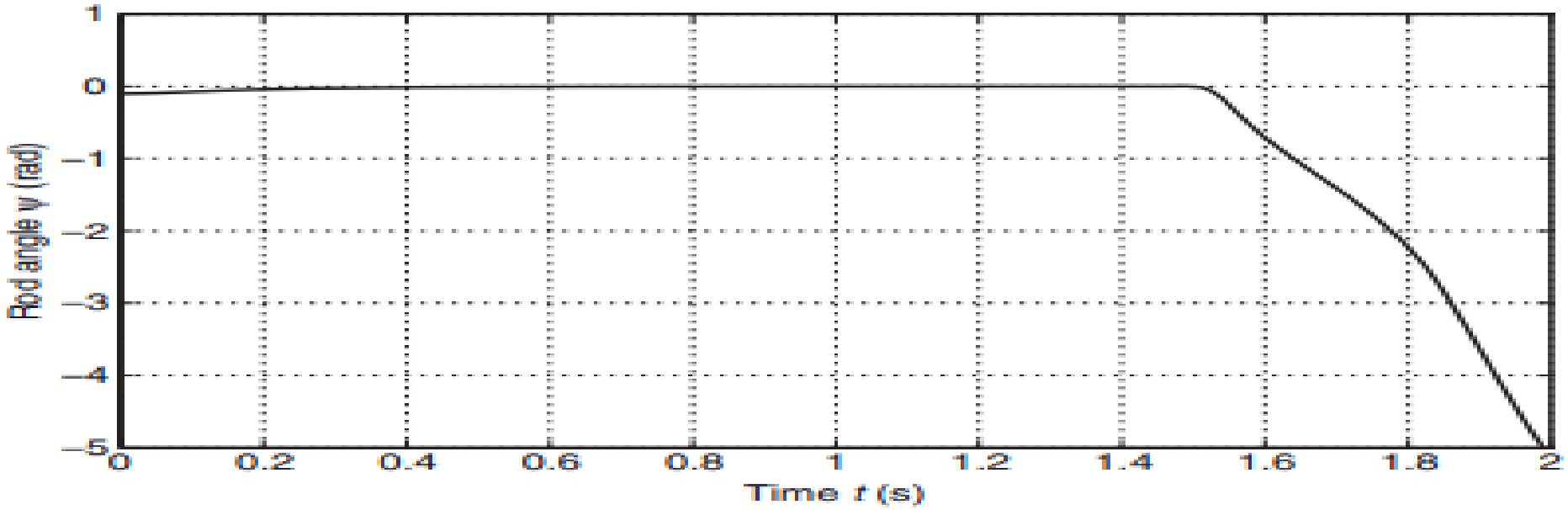


Figure 4.25. Rod angle, 50 N impulsive disturbance at $t = 1.5$ s.

The fuzzy PD can easily be redesigned on the basis of expert knowledge so that the rod's balance will not be destroyed by such impulsive disturbances. The redesign consists of repositioning the output singletons so that the controller delivers greater force when e and $\dot{e}$ are large, but when e and $\dot{e}$ are small the control is unchanged. Therefore, let the new output singletons be as in Figure 4.26.

The input–output characteristic of the redesigned nonlinear fuzzy PD controller is shown in Figure 4.27. In this figure, the characteristic of the redesigned nonlinear fuzzy PD is the same as that of the linear controller when small control

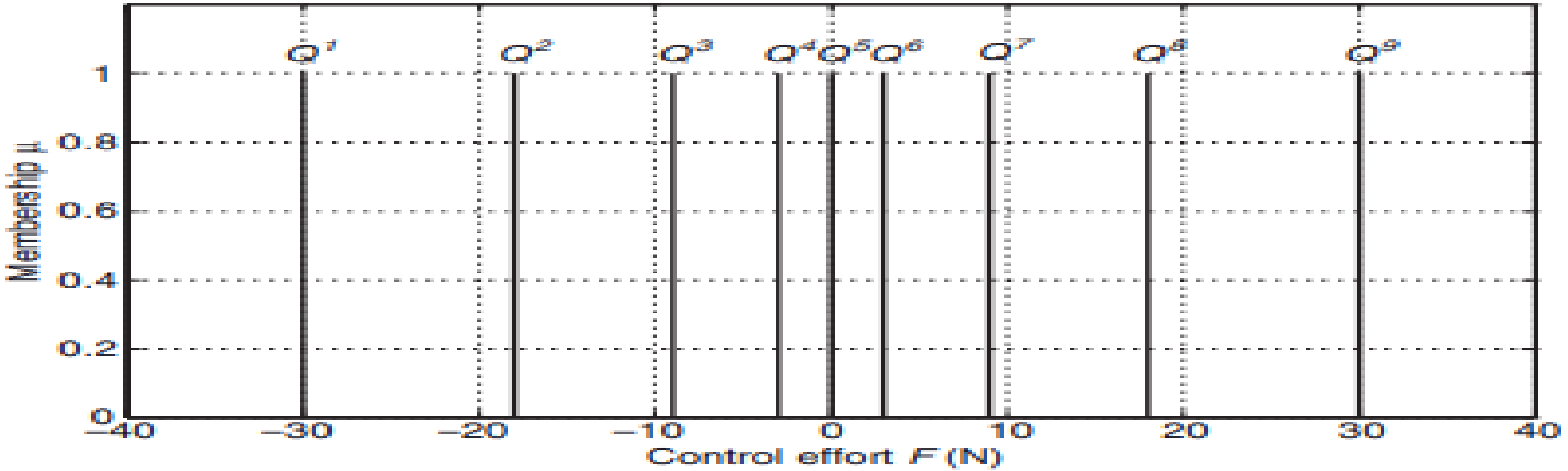


Figure 4.26. Redesigned output singletons for fuzzy PD controller for inverted pendulum.

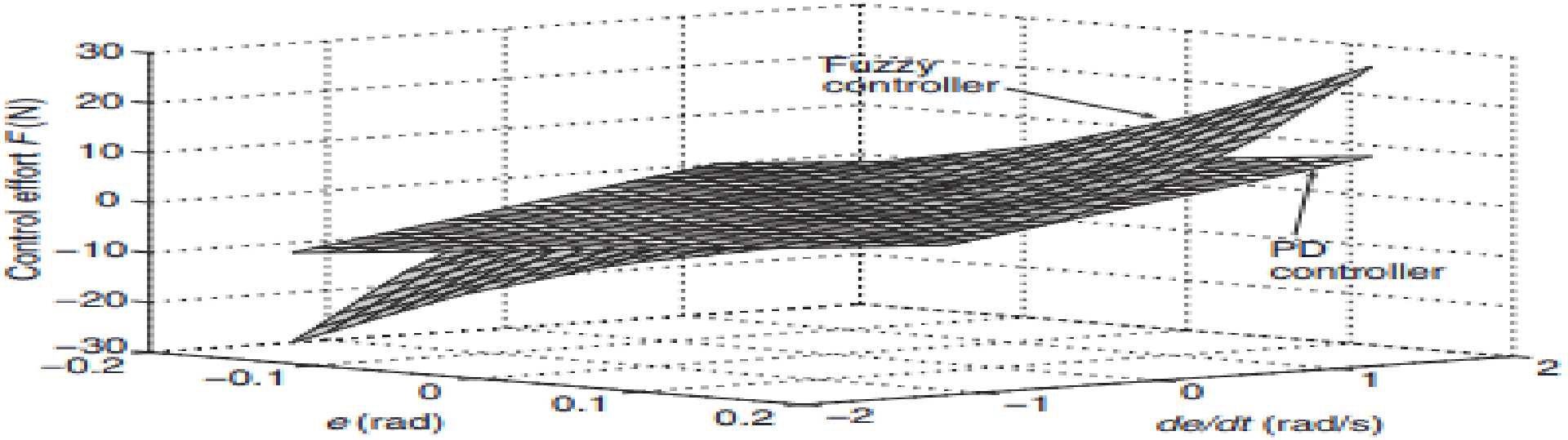


Figure 4.27. Input–output characteristics of linear PD and redesigned nonlinear fuzzy controllers.

effort is required, but for cases when large control effort is required the redesigned fuzzy controller differs significantly from the linear controller.

The response of the inverted pendulum under closed-loop control with the redesigned nonlinear fuzzy controller is shown in Figure 4.28. In this figure, we see that now the cart catches the rod before it falls, unlike Figure 4.25. Note that the fuzzy PD was easily redesigned using expert knowledge to handle the impulsive disturbance. The original nonfuzzy PD (4.17) may not be so easily redesigned to increase robustness.

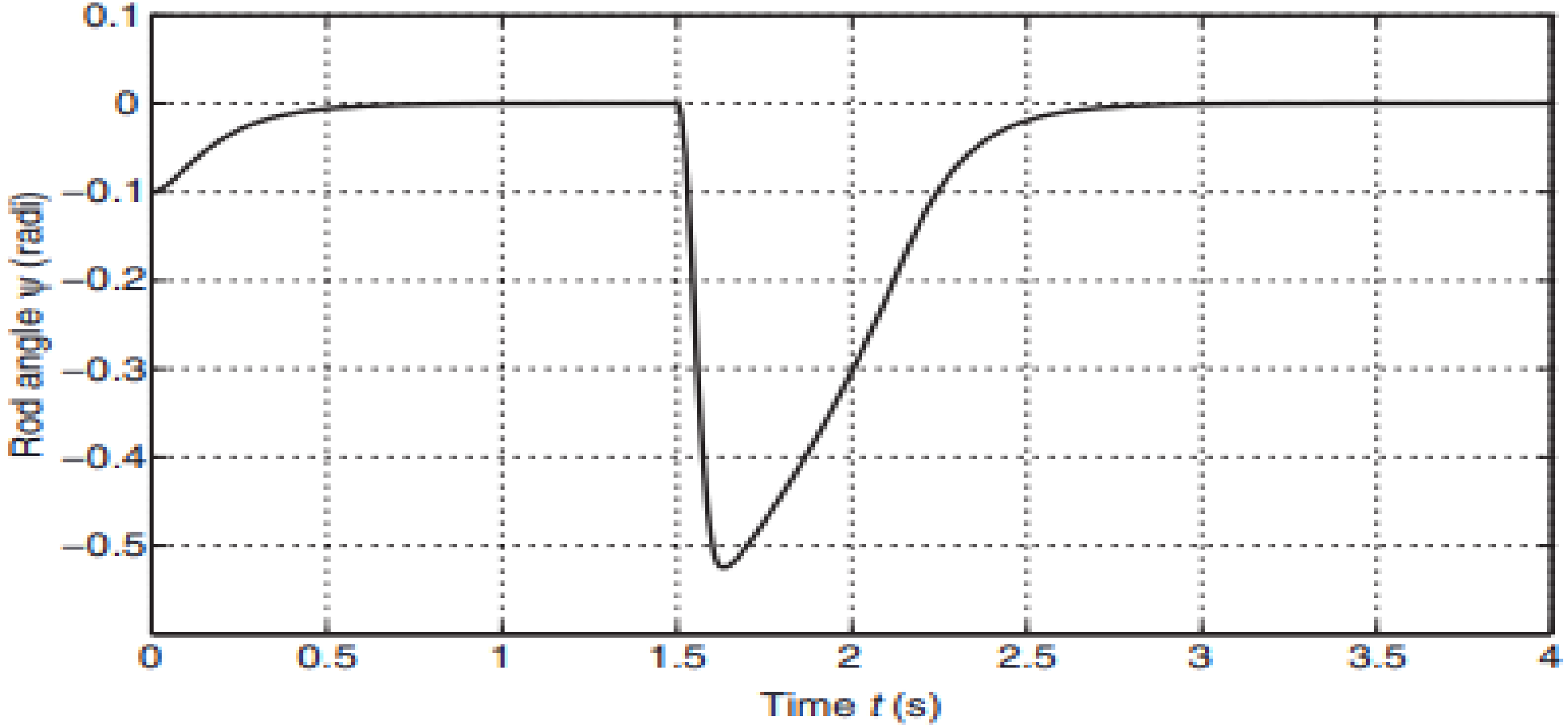


Figure 4.28. Rod angle with 50 N impulsive disturbance at $t = 1.5$ s, controlled with redesigned (nonlinear) fuzzy PD controller.

The reader is reminded that all designs of fuzzy controllers done in this chapter were done using only expert knowledge, that is, our common sense and experience with the processes, not any control theory or knowledge of mathematical models of the plants, which were needed only for purposes of simulation.

4.5 INCREMENTAL FUZZY CONTROL [13]

Consider the plant in Figure 4.1 with input $u(t)$ and output $y(t)$. So far, we have considered only *position form* control laws. Position form control laws prescribe what the control $u(t)$ should be (i.e., $u(t) = \ldots$).

An example of a position form control law is the well-known continuous-time PID control law given by

$$u(t) = K\left[e(t) + \frac{1}{T_i}\int_0^t e(\tau)\,d\tau + T_d\,\frac{de(t)}{dt}\right] \tag{4.18}$$

where $e(t)$ is the summer output and K, T_i, and T_d are constants chosen by the designer. Let $r(t) = r$, a constant. If we sample $e(t)$ every Δt seconds, we obtain the discrete-time position form PID control law given by

$$\begin{aligned} u(k) &= K\left[e(k) + \frac{\Delta t}{T_i}\sum_{i=0}^{k} e(i) + \frac{T_d}{\Delta t}c(k)\right] = Ke(k) + \frac{K\Delta t}{T_i}\sum_{i=0}^{k} e(i) + \frac{KT_d}{\Delta t}c(k) \\ &= K_p e(k) + K_i\sum_{i=0}^{k} e(i) + K_d c(k) \end{aligned} \tag{4.19}$$

where $c(k) = e(k) - e(k-1)$.

In *incremental form* control laws, the *change* in plant input, rather than the input itself, is prescribed by the controller. The change in input is given by $\Delta u(k) = u(k) - u(k-1)$. To formulate an incremental PID control law, we need [from (4.19)]

$$u(k-1) = K_p e(k-1) + K_i \sum_{i=0}^{k-1} e(i) + K_d c(k-1) \tag{4.20}$$

Therefore, the resulting PID control law in incremental form is given by

$$\Delta u(k) = K_p c(k) + K_i e(k) + K_d d(k) \tag{4.21}$$

where $d(k) = c(k) - c(k-1)$. Then, the control delivered to the plant is

$$u(k) = u(k-1) + \Delta u(k) \tag{4.22}$$

EXAMPLE 4.2

An incremental fuzzy controller is designed for the ball and beam of Section 4.1 using the method of Section 4.4 with $\Delta t = 0.01$ s. First, a nonfuzzy PD incremental controller is designed in an ad hoc manner using basic knowledge of PID controllers (increasing proportional gain quickens the response, increasing derivative gain decreases oscillations, etc.):

$$\Delta v(k) = P_{\text{incr}} c(k) + D_{\text{incr}} d(k) \tag{4.23}$$

This trial and error method results in gains of $P_{\text{incr}} = 500$, $D_{\text{incr}} = 58000$ to approximately duplicate the closed-loop performance of the ball and beam with position-form PD controller (4.17).

When the plant is run in closed loop with this controller, the maximum controller output is measured as $\Delta v_{\max} = 211.65$ V. Therefore, letting $\gamma = 1.89$ (for round numbers for locations of output singletons) and assuming there are three fuzzy sets on each universe, the effective universes of discourse for c, d, and Δv are, respectively, $-0.8 \le c \le 0.8$, $-0.0069 \le d \le 0.0069$, and $-800 \le \Delta v \le 800$, resulting in the following fuzzy sets on these universes (Figs. 4.29–4.31):

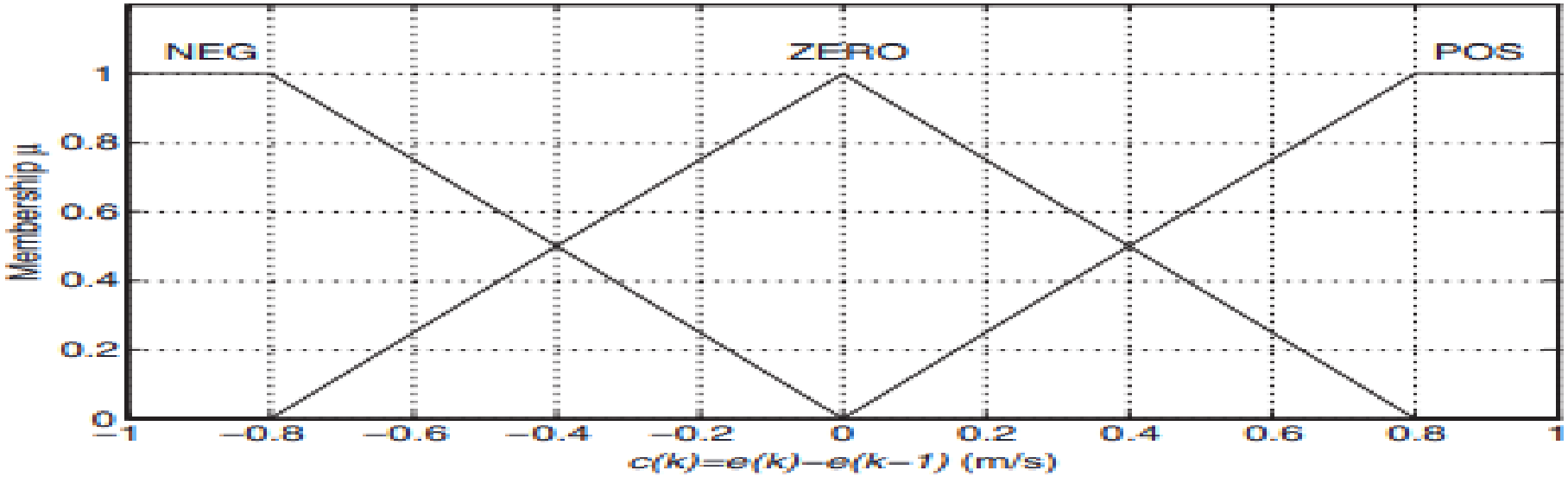


Figure 4.29. Fuzzy sets on $c(k)$ universe for fuzzy incremental PD controller for ball and beam plant.

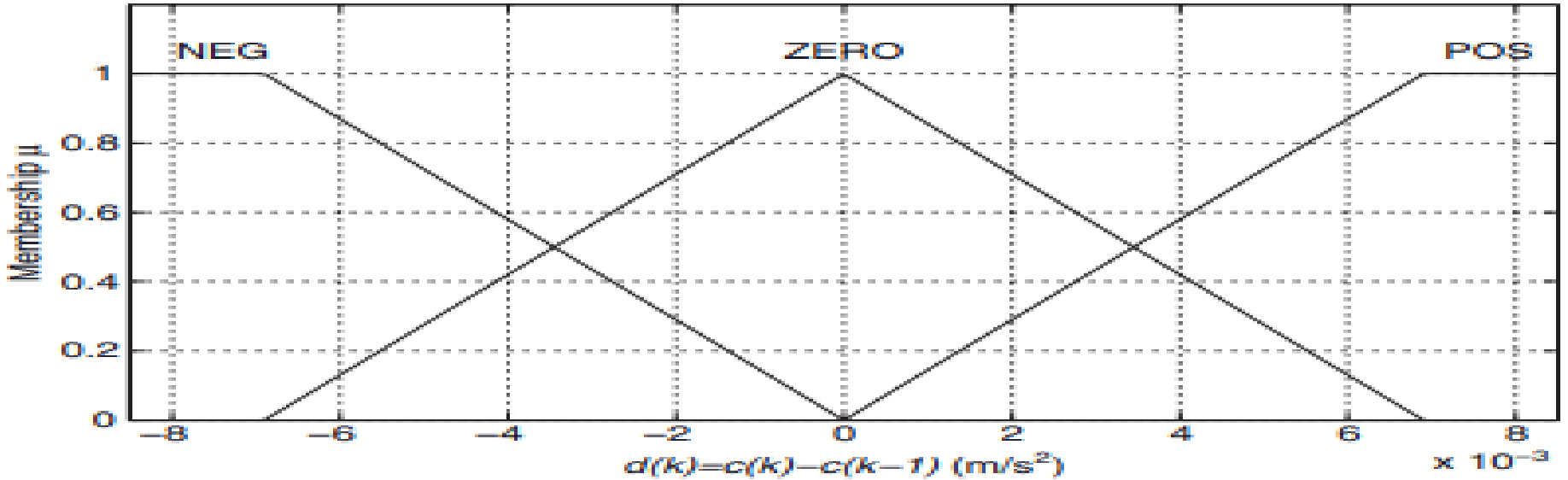


Figure 4.30. Fuzzy sets on $d(k)$ universe for fuzzy incremental PD controller for ball and beam plant.

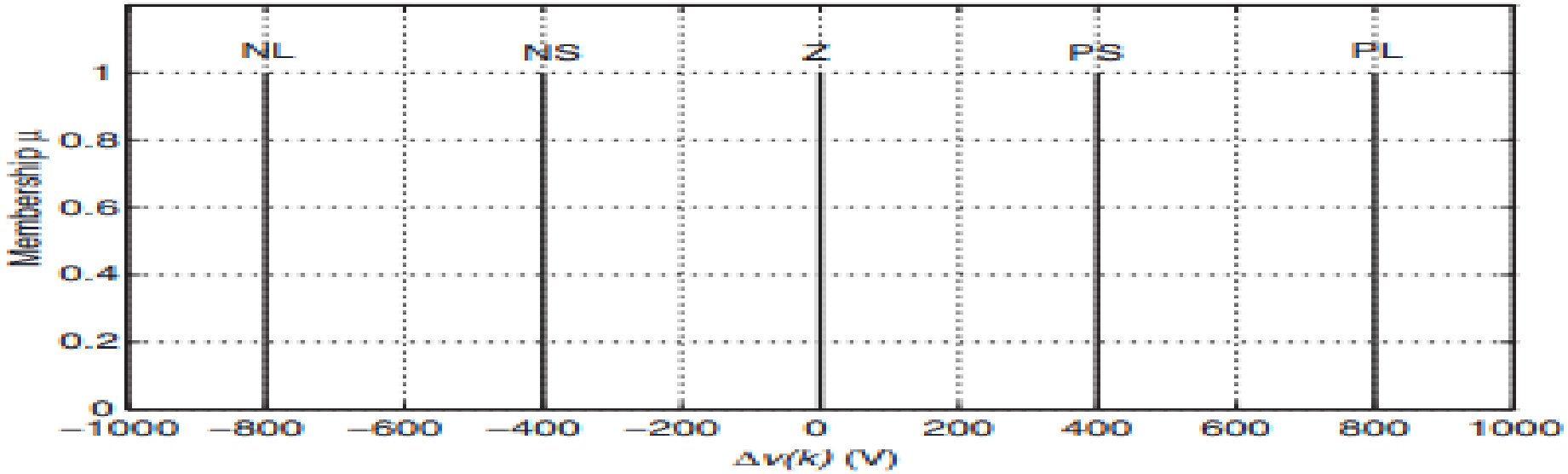


Figure 4.31. Fuzzy sets on $\Delta v(k)$ universe for fuzzy incremental PD controller for ball and beam plant.

The rule base of the fuzzy incremental PD controller is given in tabulated form in Table 4.3.

TABLE 4.3 Tabulated Rule Base for Incremental PD Ball and Beam Controller

Δv		d		
		N	Z	P
c	N	NL	NS	Z
	Z	NS	Z	PS
	P	Z	PS	PL

4.6 SUMMARY

Mamdani fuzzy systems have fuzzy sets in their rule consequents, as opposed to Takagi–Sugeno fuzzy systems (see Chapter 6), which have mathematical expressions in their rule consequents. We consider the most common interconnection for tracking: unity feedback with a fuzzy controller in casacade with the plant. Fuzzy control is not **model based**, that is, its design does not depend on a mathematical model of the plant, but on the designer's common sense and past experience with the process being controlled.

For the **ball and beam** controller (Sections 4.1–4.3), we first decide on the inputs that would be needed (e and $\dot{e}$), then determine proper universes of discourse for all inputs and outputs. These are based on our best common sense about the system, but scaling gains are included in the design to improve performance.

Section 4.4 introduces a method for **conversion of PID controllers into fuzzy controllers**. The equivalent fuzzy controller is exactly equal to the PID within its effective universe. This technique is useful because nonfuzzy PID controllers can be readily designed using well-known techniques. Once the equivalent fuzzy controller has been obtained, it can be easily adjusted in a nonlinear manner using heuristic knowledge to improve performance. Example 4.1 shows how this can be done. In this example, a nonfuzzy PD controller for an inverted pendulum plant is converted to a fuzzy controller. This fuzzy controller is then adjusted in a nonlinear manner to increase robustness of the system to impulsive disturbances.

Section 4.5 introduces **incremental fuzzy control**, which prescribes the *change* in input rather than the input itself. A fuzzy incremental PID is derived, requiring inputs $c(k)$, $e(k)$, and $d(k)$ where $e(k) = r - y(k)$, $c(k) = e(k) - e(k-1)$, and $d(k) = c(k) - c(k-1)$. In Example 4.2, the method of Section 4.4 is used to derive an incremental fuzzy PD controller for the ball and beam plant. This controller approximately duplicates the closed-loop performance of the position-form controller derived in Section 4.1.

MODELING AND CONTROL METHODS USEFUL FOR FUZZY CONTROL

5.1 CONTINUOUS-TIME MODEL FORMS

The four most common methods for describing continuous-time dynamic systems are the transfer function (for time-invariant linear systems), the impulse response (for time-varying or time-invariant linear systems), the input–output ordinary differential equation (for any type of continuous-time system), and the state-space description (for any type of system). Of these, only the linear or nonlinear time-invariant state-space descriptions are useful for fuzzy identification and control. Now we give brief summaries of these model structures.

5.1.1 Nonlinear Time-Invariant Continuous-Time State-Space Models

Let $x(t) = [x_1, x_2, \ldots, x_n]$ be the vector of states of a time-invariant continuous-time nth order single-input, single-output nonlinear system with input $u(t)$ and output $y(t)$. A very general form for the system model is

$$\dot{x} = F(x, u) \tag{5.1a}$$
$$y = H(x, u) \tag{5.1b}$$

where $F(x, u)$ and $H(x, u)$ are continuously differentiable functions of their arguments.

If (5.1) can be put in the form

$$\dot{x} = f(x) + g(x)u \tag{5.2a}$$
$$y = h(x) \tag{5.2b}$$

it is said to be *feedback linearizable* (a mathematically rigorous definition of feedback linearizability is more involved [1]. This means that it can be linearized by an appropriately designed feedback law. Under certain assumptions, by differentiating the output it is possible to transform (5.2) into the so-called *companion form* [21]:

$$y^{(m)} = \delta(x) + \eta(x)u \tag{5.3}$$

where $y^{(m)} = d^m y/dt^m$. The integer m is known as the *relative degree* of the system. If $m < n$, there can be *zero dynamics* ([1,21,22]), which are assumed stable in this book.

Models of the form (5.3) are used in indirect adaptive fuzzy control algorithms. In such algorithms, the system model is not known in advance; rather it is *identified* in this form using one of several fuzzy identification schemes (gradient, least squares, etc.) (see Chapter 9).

EXAMPLE 5.1

Consider the forced rigid pendulum shown in Figure 5.1. It has a massless shaft of length L with a mass M concentrated at the end, and a coefficient of friction B at the attach point. The external torque applied to the shaft at the attach point is τ. This is a version of the motor-driven robotic link of Section 1.4.

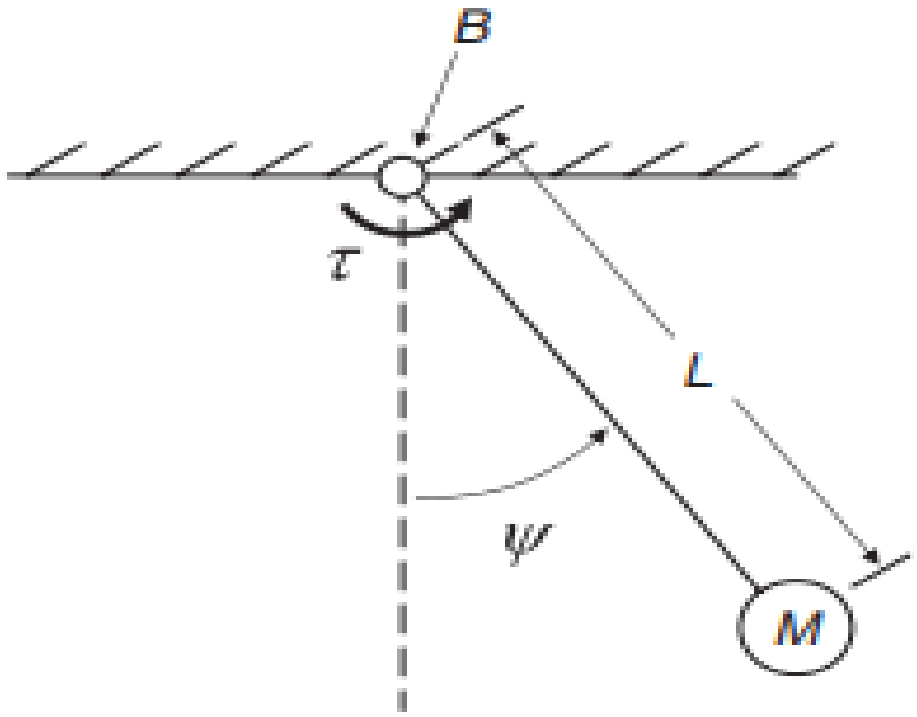


Figure 5.1. Forced rigid pendulum.

The rotational version of Newton's second law gives $\Upsilon = I\ddot{\psi}$, where ψ is the angle of the shaft from vertical, I is the moment of inertia of the pendulum about the attach point, and Υ is the sum of all torques acting on the shaft. Thus, the mathematical model of this system is

$$\tau - B\dot{\psi} - MgL\sin\psi = I\ddot{\psi}$$

where $I = ML^2$ and g is the acceleration of gravity. Let $M = 1\,\text{kg}$, $L = 1\,\text{m}$, $B = 1\,\text{kg-m/s}^2$ and $g = 9.81\,\text{m/s}^2$. If we define the states $x_1 = \psi$ and $x_2 = \dot{\psi}$, the output $y = \psi$, and the input $u = \tau$, the following state and output equations result:

$$\dot{x}_1 = x_2 \tag{5.4a}$$
$$\dot{x}_2 = -9.8\sin x_1 - x_2 + u \tag{5.4b}$$
$$y = x_1 \tag{5.4c}$$

This is in the form of (5.1) with $x = [x_1 \quad x_2]^T$,

$$F(x,u) = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -9.8\sin x_1 - x_2 + u \end{bmatrix} \tag{5.5}$$

and

$$H(x,u) = x_1$$

The right-hand sides of (5.4a) and (5.4b) can be rewritten to give

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -9.8\sin x_1 - x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{5.6a}$$
$$y = x_1 \tag{5.6b}$$

Since this is the form of (5.2) with $f(x)=\begin{bmatrix} x_2 \\ -9.8\sin x_1 - x_2 \end{bmatrix}$, $g(x)=\begin{bmatrix} 0 \\ 1 \end{bmatrix}$, and $h(x) = x_1$, the system is feedback linearizable.

Differentiating (5.6b) once, we have $\dot{y} = \dot{x}_1 (= x_2)$. This is not in the form of (5.3) because there is no u term present. Differentiating the output once again, we have $\ddot{y} = \dot{x}_2$. Then, the model can be expressed in the form of (5.3) with relative degree $m = 2$, $\delta(x) = -9.8\sin x_1 - x_2$ and $\eta(x) = 1$:

$$\ddot{y} = -9.8\sin x_1 - x_2 + u \tag{5.7}$$

The Matlab code to simulate the pendulum model (5.4) forced with an input torque of

$$u = \begin{cases} 2\sin(2\pi t), & t \le 2\,\text{s} \\ 0, & t > 2\,\text{s} \end{cases} \tag{5.8}$$

is in the Appendix . The resulting pendulum angle is shown in Figure 5.2.

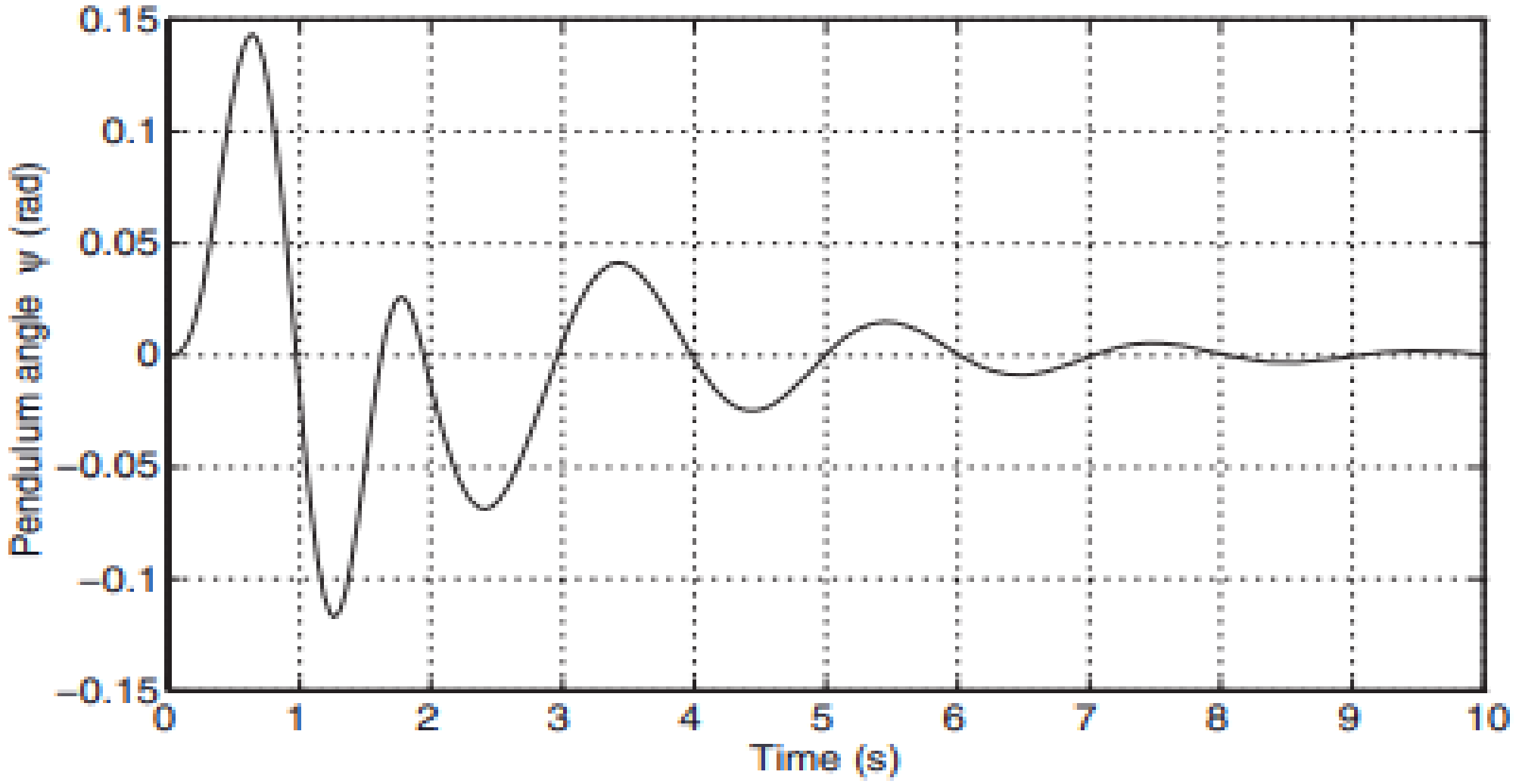


Figure 5.2. Pendulum angle, full nonlinear model (5.1).

5.1.2 Linear Time-Invariant Continuous-Time State-Space Models

Let the nonlinear system be modeled as in (5.1). Assume $x = 0$, $u = 0$ is an equilibrium point of the system [i.e., $F(0, 0) = 0$]. Then expanding $F(x, u)$ in a Taylor series about $x = 0$, $u = 0$, we have

$$F(x,u) = F(0,0) + \left.\frac{\partial F}{\partial x}\right|_{\substack{x=0\\u=0}} x + \left.\frac{\partial F}{\partial u}\right|_{\substack{x=0\\u=0}} u + \dots$$

In this series, the higher-order terms (i.e., those involving higher than first powers of x or u or cross-products between these) can be ignored in the vicinity of the equilibrium point because these terms are vanishingly small in this region. Therefore, in the vicinity of $x = 0$, $u = 0$, the system is approximately described by the linear model

$$\dot{x}(t) = Ax(t) + bu(t) \tag{5.9a}$$
$$y(t) = cx(t) \tag{5.9b}$$

where $A = \left.\frac{\partial F}{\partial x}\right|_{\substack{x=0\\u=0}}$, $b = \left.\frac{\partial F}{\partial u}\right|_{\substack{x=0\\u=0}}$, and $c = \left.\frac{\partial H}{\partial x}\right|_{\substack{x=0\\u=0}}$

EXAMPLE 5.2

The model of the forced pendulum of Example 5.1 can be expressed as (5.1) with $F(x, u)$ as in (5.5). Then,

$$\left.\frac{\partial F}{\partial x}\right|_{\substack{x=0\\u=0}} = \left.\begin{bmatrix} \dfrac{\partial F_1}{\partial x_1} & \dfrac{\partial F_1}{\partial x_2} \\ \dfrac{\partial F_2}{\partial x_1} & \dfrac{\partial F_2}{\partial x_2} \end{bmatrix}\right|_{\substack{x=0\\u=0}} = \left.\begin{bmatrix} 0 & 1 \\ -9.8\cos x_1 & -1 \end{bmatrix}\right|_{\substack{x=0\\u=0}} = \begin{bmatrix} 0 & 1 \\ -9.8 & -1 \end{bmatrix}$$

$$\left.\frac{\partial F}{\partial u}\right|_{\substack{x=0\\u=0}} = \left.\begin{bmatrix} \dfrac{\partial F_1}{\partial u} \\ \dfrac{\partial F_2}{\partial u} \end{bmatrix}\right|_{\substack{x=0\\u=0}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

This gives the model (5.9) linearized about the equilibrium point $x_1 = 0$, $x_2 = 0$, $u = 0$:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -9.8 & -1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \tag{5.10a}$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \tag{5.10b}$$

When (5.10) is simulated with the input torque of (5.8), the behavior of the linearized system is almost identical to that of the original nonlinear system (5.4). This occurs because $\psi(t)$ and $\dot{\psi}(t)$ remain close to their equilibrium values of $\psi = 0$, $\dot{\psi} = 0$, hence higher-order terms in the Taylor series are negligibly small.

5.2 MODEL FORMS FOR DISCRETE-TIME SYSTEMS

Most systems to be controlled are continuous-time systems. If such a system is to be controlled by connecting it to a digital computer, which is virtually always the case with fuzzy control, the control input must eventually be represented in discrete-time form, necessitating some type of method that approximates continuous-time signals by discrete-time signals. This could be done by calculating the continuous-time control signal from a continuous-time model in the form of (5.1), (5.3), or (5.9) and then discretizing it, or by calculating a discrete-time control signal from a discrete-time model of the system. In the former case, the approximation error is the result of discretizing a continuous-time control signal. In the latter case, the error is the result of approximating a continuous-time system by a discrete-time system.

If the system is discrete time, either inherently or through sampling, all signals are considered as existing only at discrete time instants, $k\Delta t$, where $k = 0, 1, 2, \ldots$ and Δt is the sampling interval for sampled continuous-time systems. If the system is inherently discrete, $\Delta t = 1$. For sampled continuous-time systems, the sampling interval Δt is usually supressed, that is, $x(k\Delta t)$ is written $x(k)$.

In this section, we give two forms of discrete-time models for linear systems.

5.2.1 Input–Output Difference Equation Model for Linear Discrete-Time Systems

One common method of describing the input–output behavior of a discrete-time linear system is by an input–output difference equation involving present and past inputs and outputs. This is known as an *autoregressive moving average* (ARMA) model. The general form of ARMA model used in this book is

$$y(k+1) = \alpha(q^{-1})y(k) + \beta(q^{-1})u(k) \tag{5.11}$$

with

$$\alpha(q^{-1}) = a_1 + a_2 q^{-1} + \cdots + a_n q^{-(n-1)} \tag{5.12a}$$
$$\beta(q^{-1}) = b_0 + b_1 q^{-1} + \cdots + b_m q^{-m} \tag{5.12b}$$

where k is an integer representing time, $u(k)$ is the system input at time k, $y(k)$ is the system output at time k, a_i, $i = 1, 2, \ldots, n$ and b_j, $j = 0, 1, \ldots, m$ are constants, and q^{-1} is the backward-shift operator defined by

$$q^{-1}y(k) = y(k-1) \tag{5.13}$$

By incrementing the time index in (5.11) by $n - 1$, the following model results:

$$\begin{aligned} &y(k+n) - a_1 y(k+n-1) - a_2 y(k+n-2) - \cdots - a_n y(k) \\ &\quad = b_0 u(k+n-1) + b_1 u(k+n-2) + \cdots + b_m u(k+n-m-1) \end{aligned} \tag{5.14}$$

Implementing the forward-shift operator q in (5.13) results in

$$\begin{aligned} &q^n y(k) - a_1 q^{n-1} y(k) - a_2 q^{n-2} y(k) - \cdots - a_n y(k) \\ &\quad = b_0 q^{n-1} u(k) + b_1 q^{n-2} u(k) + \cdots + b_m q^{n-m-1} u(k) \end{aligned}$$

from which we can write the system pulse transfer function:

$$\frac{y(k)}{u(k)} = \frac{b_0 q^{n-1} + b_1 q^{n-2} + \cdots + b_m q^{n-m-1}}{q^n - a_1 q^{n-1} - a_2 q^{n-2} - \cdots - a_n} \tag{5.15}$$

5.2.2 Linear Time-Invariant Discrete-Time State-Space Models

If we define the states $x_1(k) = y(k)$, $x_2(k) = y(k+1)$, $x_3(k) = y(k+2)$, … , $x_n(k) = y(k+n-1)$ in (5.14), the following state equations result

$$x(k+1) = Ax(k) + bu(k) \tag{5.16a}$$

$$y(k) = cx(k) \tag{5.16b}$$

where $x(k) = [x_1(k)\ x_2(k)\ \ldots\ x_n(k)]^T$ and

$$A=\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \cdots & 1 \\ a_n & a_{n-1} & a_{n-2} & \cdots & a_1 \end{bmatrix}, b=\begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, c^T=\begin{bmatrix} 0 \\ \vdots \\ b_m \\ \vdots \\ b_0 \end{bmatrix}$$

The above gives a method for converting from input–output difference equation form to linear discrete-time state-space form. It is also possible to convert to linear discrete-time state-space form directly from linear continuous-time state space form (5.9). To do this, discretize time and define $x(k) = x(k\Delta t)$, where Δt is the sample time and k is an integer. Approximate $\dot{x}$ as

$$\dot{x} = \frac{x(k+1)-x(k)}{\Delta t}$$

Then (5.9) becomes

$$\frac{x(k+1)-x(k)}{\Delta t} = Ax(k)+bu(k)$$
$$y(k)=cx(k)$$

which yields

$$x(k+1)=(I+A\Delta t)x(t)+b\Delta tu(k)$$
$$y(k)=cx(k)$$

EXAMPLE 5.3

Consider the linearized continuous-time model of the forced pendulum (5.10). If the sampling time Δt is 0.01 s, the discrete-time linearized model of the forced pendulum is

$$x(k+1)=\begin{bmatrix} 1 & 0.01 \\ -0.098 & 0.99 \end{bmatrix}x(k)+\begin{bmatrix} 0 \\ 0.01 \end{bmatrix}u(k) \tag{5.17a}$$

$$y(k)=\begin{bmatrix} 1 & 0 \end{bmatrix}x(k) \tag{5.17b}$$

□

The Matlab code to simulate the pendulum model (5.17) forced with the input torque of (5.8) is given in the Appendix. The behavior of the linearized discrete-time system is nearly identical to that of the original nonlinear system. This is because the sampling time $\Delta t = 0.01$ s is small enough that all signals are essentially constant over one sampling interval.

EXAMPLE 5.4

Applying the forward shift operator (5.13) to (5.17) yields

$$qx(k) = \begin{bmatrix} 1 & 0.01 \\ -0.098 & 0.99 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} u(k)$$

$$y(k) = [1 \quad 0]x(k)$$

Solving for $x(k)$ in the first equation and substituting this into the second equation yields

$$y(k) = [1 \quad 0]\left(qI - \begin{bmatrix} 1 & 0.01 \\ -0.098 & 0.99 \end{bmatrix} \right)^{-1} \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} u(k)$$

Therefore, the pulse transfer function of the forced pendulum is

$$\frac{y(k)}{u(k)} = \frac{0.0001}{q^2 - 1.99q + 0.991}$$

which gives rise to the linear input–output difference equation in the form of (5.11) describing the forced pendulum:

$$y(k+1) = 1.99y(k) - 0.991y(k-1) + 0.0001u(k-1) \tag{5.18}$$

The Matlab code to simulate the pendulum model (5.18)) forced with the input torque of (5.8) is given in the Appendix . The behavior of the system (5.18) is essentially identical to that of the original nonlinear system.

EXAMPLE 5.5

Consider the linear continuous-time time-invariant system given by (5.9) with

$$A = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & 2 & 1 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

Design a linear state feedback control law $u = -kx$ such that the poles of the closed-loop system are $-1, -1 + j, -1 - j, -2$.

Solution: The characteristic polynomial of A is

$$|\lambda I - A| = \lambda^4 - \lambda^3 - 4\lambda^2 + \lambda + 3$$

This can also be found using the Matlab command *poly*. Then, we have $d_0 = 3$, $d_1 = 1$, $d_2 = -4$, $d_3 = -1$ and

$$W = \begin{bmatrix} 1 & -4 & -1 & 1 \\ -4 & -1 & 1 & 0 \\ -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

The controllability matrix of the system is

$$C = [b \quad Ab \quad A^2b \quad A^3b] = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 1 & 4 \\ 0 & -1 & 0 & -2 \\ -1 & 0 & -2 & -1 \end{bmatrix}$$

Therefore,

$$T = CW = \begin{bmatrix} 5 & -3 & -2 & 1 \\ -1 & 0 & 1 & 0 \\ 2 & 1 & -1 & 0 \\ 0 & 2 & 1 & -1 \end{bmatrix}$$

The desired closed-loop characteristic polynomial is

$$(\lambda+1)(\lambda+1-j)(\lambda+1+j)(\lambda+2) = \lambda^4 + 5\lambda^3 + 10\lambda^2 + 10\lambda + 4$$

This can also be found using the Matlab command *poly*. Then we have $\delta_0 = 4$, $\delta_1 = 10$, $\delta_2 = 10$, $\delta_3 = 5$. Then, from (5.25), the feedback matrix k is

$$k = [1 \quad 9 \quad 14 \quad 6]T^{-1} = [0 \quad 41 \quad 21 \quad -6]$$

and the feedback control law is

$$u = -kx = -41x_2 - 21x_3 + 6x_4 \tag{5.27}$$

To check, verify that the eigenvalues of $A - bk$ are $-1, -1+j, -1-j, -2$ as desired. □

If the number of states is small (say $n \leq 3$), it may be easier to calculate k by simply calculating the characteristic polynomial of $A - bk$ with variable k and equating this to the desired closed-loop characteristic polynomial.

EXAMPLE 5.10

Consider the forced pendulum (5.4). Find a feedback control law so that the closed-loop system is linear with unity zero-frequency gain and poles at $s = -1 \pm j$.

Solution: The desired transfer function (unity zero-frequency gain, poles at $s = -1 \pm j$) is

$$\frac{\psi(s)}{\tau(s)} = \frac{2}{s^2 + 2s + 2}$$

Considering the model (5.7), which is equivalent to (5.4), we have $\delta(\psi, \dot{\psi}) = -9.8 \sin \psi - \ddot{\psi}$ and $\eta(\psi, \dot{\psi}) = 1$.

Then by (5.39) and (5.41) the necessary control law is

$$\tau = 9.8 \sin \psi + \ddot{\psi} - 2\dot{\psi} - 2\psi + 2r = 9.8 \sin \psi - 2\psi - \dot{\psi} + 2r \qquad (5.42)$$

where $r(t)$ is the external input to the system. The block diagram of the closed-loop system is shown in Figure 5.3.

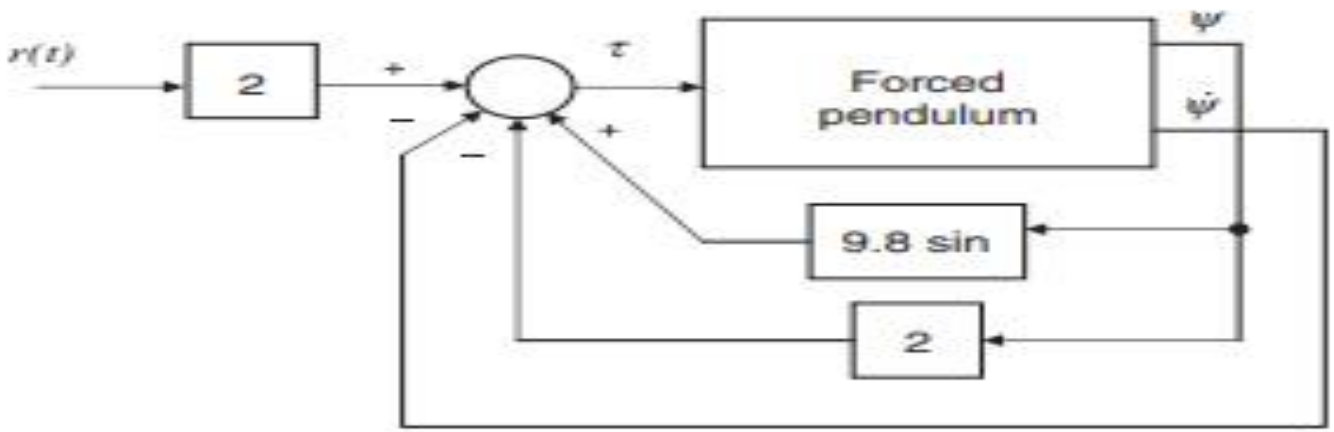


Figure 5.3. Block diagram of forced pendulum (5.4) with feedback linearizing control (5.42).

TAKAGI – SUGENO FUZZY SYSTEMS

6.1 TAKAGI–SUGENO FUZZY SYSTEMS AS INTERPOLATORS BETWEEN MEMORYLESS FUNCTIONS

Consider a T–S fuzzy system with R rules of the form:

$$R_i \qquad \text{If } x_1 \text{ is } P_1^K \text{ and } x_2 \text{ is } P_2^L \text{ and } \cdots \text{ and } x_n \text{ is } P_n^M, \text{ then } q^i = f_i(\bullet) \qquad (6.1)$$

The input universes $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n$ are as before, with various fuzzy sets defined on them, but now the consequents are memoryless functions $f_i(\bullet)$. The crisp output of this system is

$$y^{\text{crisp}} = \frac{\sum_{i=1}^{R} q^i \mu_i(\underline{x})}{\sum_{i=1}^{R} \mu_i(\underline{x})} \qquad (6.2)$$

where $\mu_i(\underline{x})$ is the premise membership value of Rule i. The crisp output can be expressed as

$$y^{\text{crisp}} = q^1\xi_1(t) + q^2\xi_2(t) + \cdots + q^R\xi_R(t) \tag{6.3}$$

where

$$\xi_i(t) = \frac{\mu_i(x(t))}{\sum_{j=1}^{R}\mu_j(x(t))} \quad i = 1, \ldots, R \tag{6.4}$$

are called *fuzzy basis functions* [28].

In this section, we assume the functions $f_i(\bullet)$ are affine functions of the inputs:

$$f_i(\bullet) = a_0^i + a_1^i x_1 + a_2^i x_2 + \cdots + a_n^i x_n \tag{6.5}$$

where the coefficients a_j^i are constants. The output y^{crisp} is a nonlinear function of the inputs $x_1, \ldots, x_n$.

EXAMPLE 6.1

Consider a two-input T–S fuzzy system with rules

1. If x_1 is P_1^1 and x_2 is P_2^1, then $q^1 = 1 + x_1 + x_2$.
2. If x_1 is P_1^1 and x_2 is P_2^2, then $q^2 = 2x_1 + x_2$.
3. If x_1 is P_1^2 and x_2 is P_2^1, then $q^3 = -1 + x_1 - 2x_2$.
4. If x_1 is P_1^2 and x_2 is P_2^2, then $q^4 = -2 - x_1 + 0.5x_2$.

Inside the effective universe of discourse the fuzzy sets P_1^1, P_1^2, P_2^1, and P_2^2 are characterized by the following membership functions:

$$\mu_1^1(x_1) = \exp\left[-\frac{1}{2}\left(\frac{x_1 + 1}{0.8493}\right)^2\right]$$

$$\mu_1^2(x_1) = \exp\left[-\frac{1}{2}\left(\frac{x_1 - 1}{0.8493}\right)^2\right]$$

$$\mu_2^1(x_2) = \exp\left[-\frac{1}{2}\left(\frac{x_2 + 1}{0.8493}\right)^2\right]$$

$$\mu_2^2(x_2) = \exp\left[-\frac{1}{2}\left(\frac{x_2 - 1}{0.8493}\right)^2\right]$$

These are shown in Figures 6.1 and 6.2.

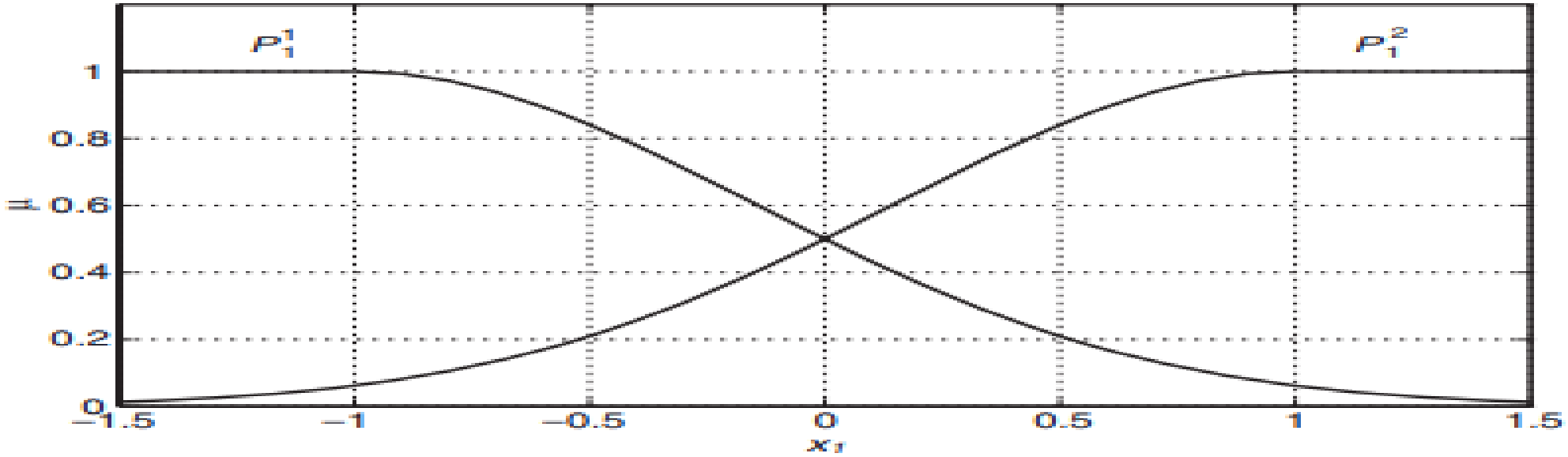


Figure 6.1. Memberships for x_1 input.

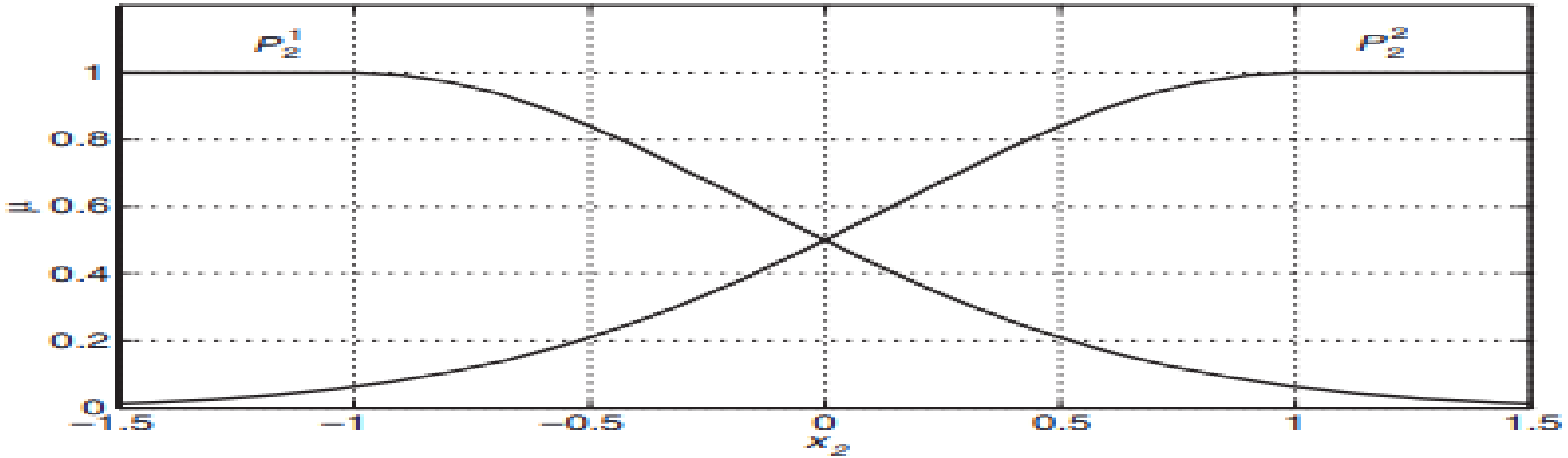


Figure 6.2. Memberships for x_2 input.

Given the crisp input (x_1, x_2), the crisp output of this system is [see (6.3), (6.4)]:

$$y^{crisp}(x_1, x_2) = \sum_{i=1}^{4} q^i(x_1, x_2)\xi_i(x_1, x_2)$$

where $\xi_i(x_1, x_2) = \mu_i(x_1, x_2) \Big/ \sum_{j=1}^{4} \mu_j(x_1, x_2)$, $i = 1, 2, 3, 4$ and, assuming *product* T-norm,

$$\mu_1(x_1, x_2) = \exp\left[-\frac{1}{2}\left(\frac{x_1 + 1}{0.8493}\right)^2\right]\exp\left[-\frac{1}{2}\left(\frac{x_2 + 1}{0.8493}\right)^2\right] \qquad (6.6a)$$

$$\mu_2(x_1, x_2) = \exp\left[-\frac{1}{2}\left(\frac{x_1+1}{0.8493}\right)^2\right]\exp\left[-\frac{1}{2}\left(\frac{x_2-1}{0.8493}\right)^2\right] \quad (6.6b)$$

$$\mu_3(x_1, x_2) = \exp\left[-\frac{1}{2}\left(\frac{x_1-1}{0.8493}\right)^2\right]\exp\left[-\frac{1}{2}\left(\frac{x_2+1}{0.8493}\right)^2\right] \quad (6.6c)$$

$$\mu_4(x_1, x_2) = \exp\left[-\frac{1}{2}\left(\frac{x_1-1}{0.8493}\right)^2\right]\exp\left[-\frac{1}{2}\left(\frac{x_2-1}{0.8493}\right)^2\right] \quad (6.6d)$$

For the particular crisp input $(x_1, x_2) = (0.6, -0.3)$, we calculate $\mu_1(0.6, -0.3) = 0.1207$, $\mu_2(0.6, -0.3) = 0.0525$, $\mu_3(0.6, -0.3) = 0.6373$, and $\mu_4(0.6, -0.3) = 0.2774$. These values yield the following fuzzy basis function values:

$$\xi_1(0.6, -0.3) = 0.1207/(0.1207 + 0.0525 + 0.6373 + 0.2774) = 0.1110$$
$$\xi_2(0.6, -0.3) = 0.0525/(0.1207 + 0.0525 + 0.6373 + 0.2774) = 0.0483$$
$$\xi_3(0.6, -0.3) = 0.6373/(0.1207 + 0.0525 + 0.6373 + 0.2774) = 0.5858$$
$$\xi_4(0.6, -0.3) = 0.2774/(0.1207 + 0.0525 + 0.6373 + 0.2774) = 0.2550$$

The consequent functions are evaluated for $(x_1, x_2) = (0.6, -0.3)$ as

$$q^1(0.6, -0.3) = 1 + 0.6 - 0.3 = 1.3$$
$$q^2(0.6, -0.3) = 2(0.6) - 0.3 = 0.9$$
$$q^3(0.6, -0.3) = -1 + 0.6 - 2(-0.3) = 0.2$$
$$q^4(0.6, -0.3) = -2 - 0.6 + .5(-0.3) = -2.75$$

Therefore, the crisp output of the T–S fuzzy system corresponding to the input $(x_1, x_2) = (0.6, -0.3)$ is

$$y^{\text{crisp}}(0.6, -0.3) = 1.3(0.1110) + 0.9(0.0483) + 0.2(0.5858) - 2.75(0.2550)$$
$$= -0.3962$$

The characteristic surface of the fuzzy system is shown in Figure 6.3. This was found by evaluating the crisp output corresponding to numerous points in the domain $-2 \leq x_1 < 2$, $-2 \leq x_2 < 2$. The surface is an interpolation between the four affine planar functions $q^1(x_1, x_2)$, $q^2(x_1, x_2)$, $q^3(x_1, x_2)$, and $q^4(x_1, x_2)$. Small portions of these planes are shown on the corresponding edges of the characteristic surface of Figure 6.3. It is seen that on the appropriate edges of the surface, the planes match well with the surface. For instance, if x_1 and x_2 are both large negative numbers (i.e., $x_1 < -1$, $x_2 < -1$), Rule 1 would be fired with certainty, while Rules 2, 3, and 4 would be fired very little. Therefore, the crisp output of the fuzzy system would be essentially q^1.

Referring to Figure 6.3, in the area of the surface where both x_1 and x_2 are between -1 and -2, the surface is indeed essentially equal to q^1. Inside the effective universe of discourse (i.e., $-1 \leq x_1 < 1$, $-1 \leq x_2 < 1$), the fuzzy system interpolates between the planes.

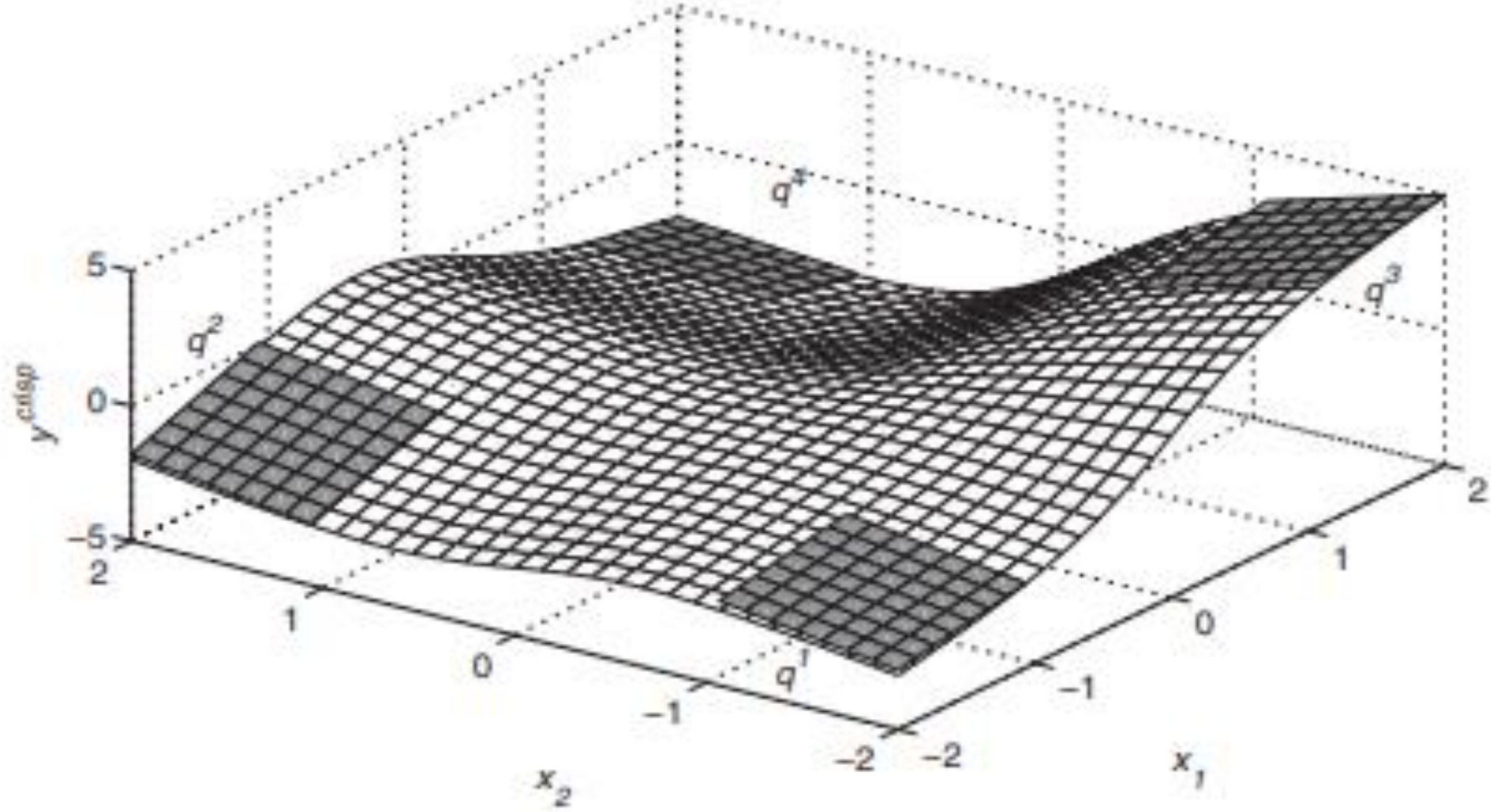


Figure 6.3. Characteristic surface of the T–S fuzzy system with individual consequents plotted at appropriate corners.

6.2 TAKAGI–SUGENO FUZZY SYSTEMS AS INTERPOLATORS BETWEEN CONTINUOUS-TIME LINEAR STATE-SPACE DYNAMIC SYSTEMS

Consider a T–S fuzzy system with R rules of the form:

R_i: If $x_1(t)$ is P_1^K and $x_2(t)$ is P_2^L and ... and $x_n(t)$ is P_n^M, then

$$\dot{x}^i(t) = A_i x(t) + b_i u(t) \tag{6.7}$$

The contents of the $n \times 1$ vector $x(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^T$ are the system states that vary continuously with time t. The input universes $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n$ are as before, with various fuzzy sets defined on them, but now the consequents are continuous-time linear constant-coefficient dynamic systems involving the states. The quantities A_i and b_i are $n \times n$ and $n \times 1$ constant-coefficient matrices, respectively, for $i = 1, 2, \ldots, n$. This fuzzy system does not have a single crisp output, rather it produces a time-varying system described by the differential equation:

$$\dot{x}(t) = \frac{\sum_{i=1}^{R} \mu_i(x(t))(A_i x(t) + b_i u(t))}{\sum_{i=1}^{R} \mu_i(x(t))} \tag{6.8}$$

or

$$\dot{x}(t) = A(t)x(t) + b(t)u(t) \tag{6.9}$$

where $A(t) = \xi_1(t)A_1 + \xi_2(t)A_2 + \ldots + \xi_R(t)A_R$ and $b(t) = \xi_1(t)b_1 + \xi_2(t)b_2 + \ldots + \xi_R(t)b_R$ and $\xi_i(t)$, $i = 1, \ldots, R$ are the fuzzy basis functions described in (6.4). Matrices $A(t)$ and $b(t)$ contain functions of the system states, making the system (6.9) a nonlinear time-varying system.

Consider a two-input T–S fuzzy system with rules

1. If $x_1(t)$ is P_1^1 and $x_2(t)$ is P_2^1, then $\dot{x}^1(t) = A_1x(t) + b_1u(t)$.
2. If $x_1(t)$ is P_1^1 and $x_2(t)$ is P_2^2, then $\dot{x}^2(t) = A_2x(t) + b_2u(t)$.
3. If $x_1(t)$ is P_1^2 and $x_2(t)$ is P_2^1, then $\dot{x}^3(t) = A_3x(t) + b_3u(t)$.
4. If $x_1(t)$ is P_1^2 and $x_2(t)$ is P_2^2, then $\dot{x}^4(t) = A_4x(t) + b_4u(t)$.

where $A_1 = \begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix}$, $b_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$, $A_2 = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}$, $b_2 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

$$A_3 = \begin{bmatrix} 0 & 1 \\ -3 & -2 \end{bmatrix}, b_3 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}, A_4 = \begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix}, b_4 = \begin{bmatrix} 0 \\ -2 \end{bmatrix}$$

The fuzzy sets P_1^1, P_1^2, P_2^1, and P_2^2 are characterized by the membership functions shown in Figures 6.4 and 6.5.

For an arbitrary input (x_1, x_2) inside the effective universe of discourse (i.e., $-1 \le x_1 < 1$, $-1 \le x_2 < 1$), the premise values for the above four rules are calculated as:

$$\mu_1(t) = (-0.5x_1(t) + 0.5)(-0.25x_2(t) + 0.5) \quad (6.10a)$$
$$\mu_2(t) = (-0.5x_1(t) + 0.5)(0.25x_2(t) + 0.5) \quad (6.10b)$$
$$\mu_3(t) = (0.5x_1(t) + 0.5)(-0.25x_2(t) + 0.5) \quad (6.10c)$$
$$\mu_4(t) = (0.5x_1(t) + 0.5)(0.25x_2(t) + 0.5) \quad (6.10d)$$

where we have used the equations of the membership functions in Figures 6.4 and 6.5 inside the effective universes and *product* T-norm.

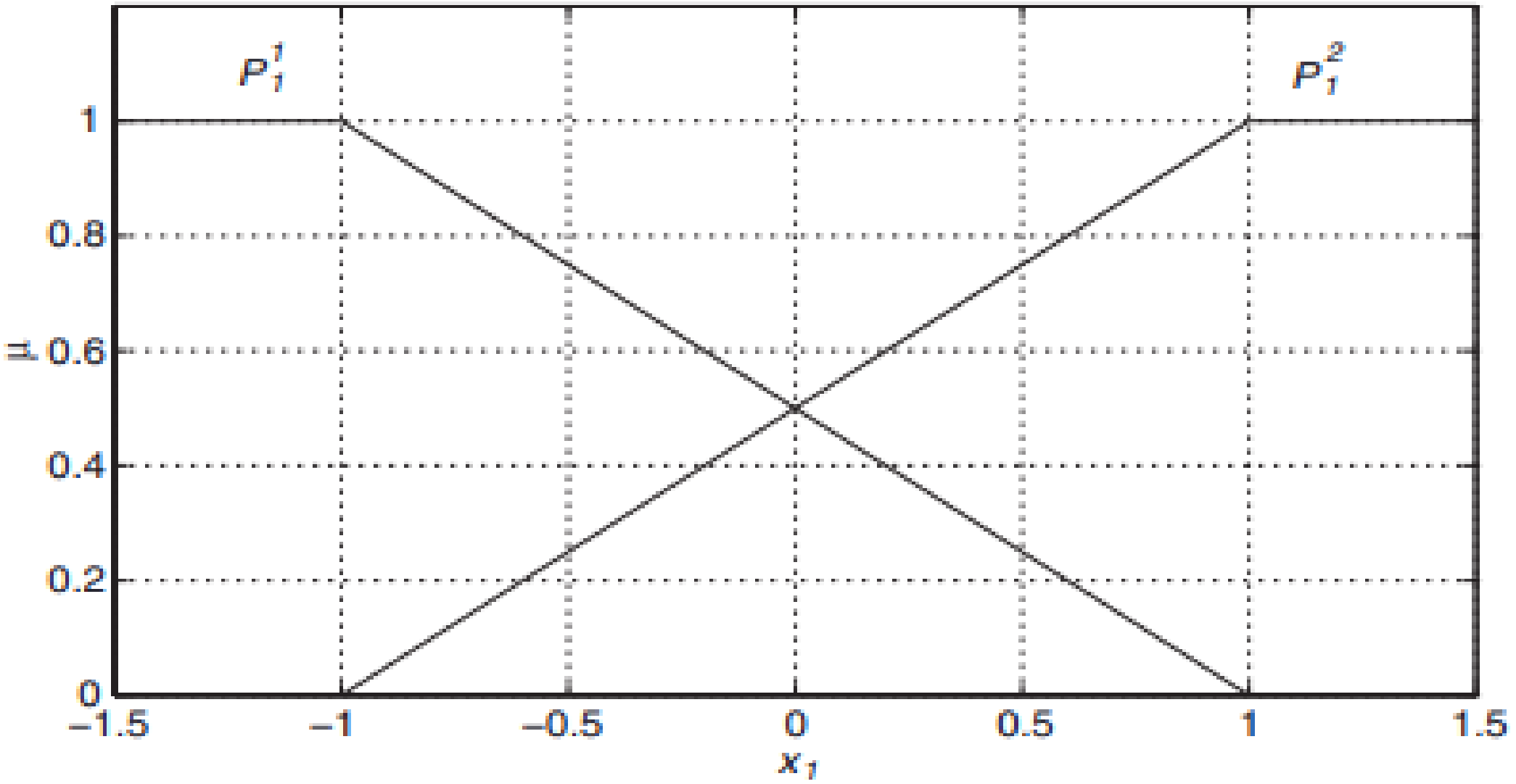


Figure 6.4. Memberships for fuzzy sets on x_1 universe.

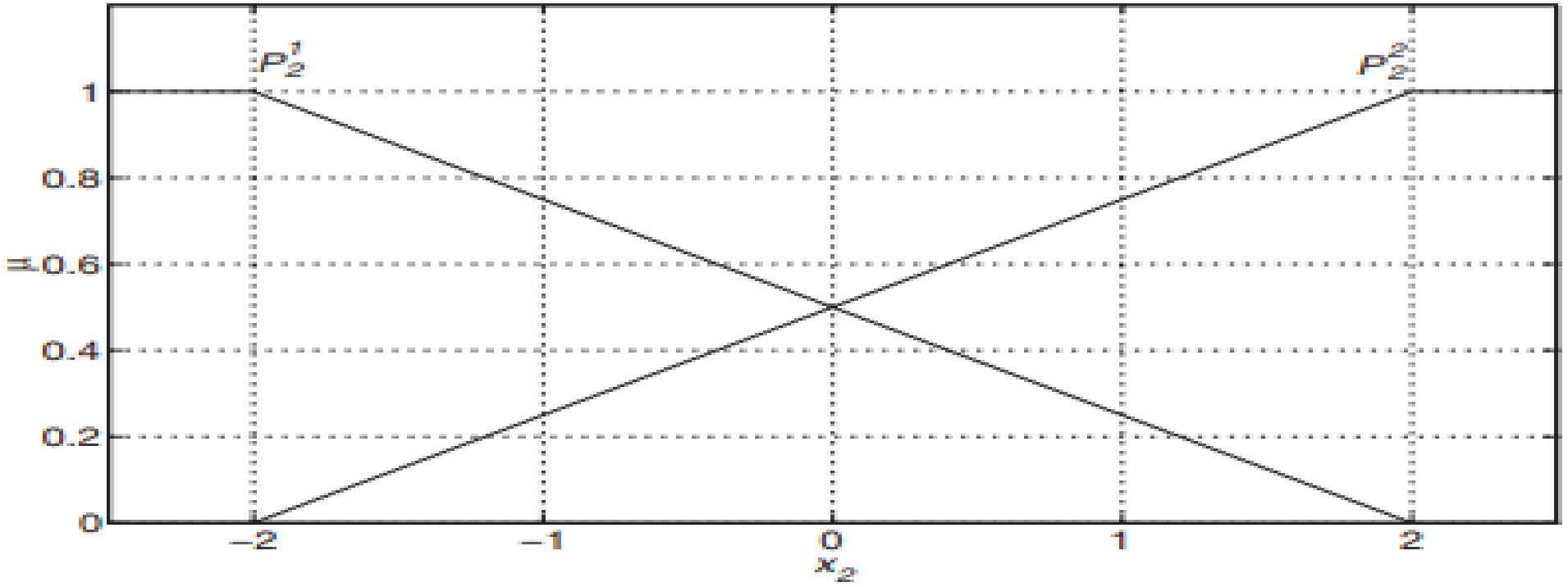


Figure 6.5. Memberships for fuzzy sets on x_2 universe.

If we consider a particular instant of time t_1 at which $x_1(t_1) = 0.4$ and $x_2(t_1) = -0.8$, we have the following premise values for the rules: $\mu_1(t_1) = 0.21$, $\mu_2(t_1) = 0.09$, $\mu_3(t_1) = 0.49$, and $\mu_4(t_1) = 0.21$. These yield the following basis function values:

$$\xi_1(t_1) = 0.21/(0.21 + 0.09 + 0.49 + 0.21) = 0.21$$
$$\xi_2(t_1) = 0.09/(0.21 + 0.09 + 0.49 + 0.21) = 0.09$$
$$\xi_3(t_1) = 0.49/(0.21 + 0.09 + 0.49 + 0.21) = 0.49$$
$$\xi_4(t_1) = 0.21/(0.21 + 0.09 + 0.49 + 0.21) = 0.21$$

Here we see the advantage of partitions of unity on the input universes combined with product T-norms: in the calculation of y^{crisp}, the denominator, which is the sum of the premise values of all rules, always equals unity. As a result, the basis functions ξ_i are merely the premise values μ_i, saving computer resources to calculate them.

The system at time t_1 is described by the state-space differential equation

$$\dot{x}(t) = A(t_1)x(t) + b(t_1)u(t)$$

where

$$\begin{aligned} A(t) &= \xi_1(t_1)A_1 + \xi_2(t_1)A_2 + \xi_3(t_1)A_3 + \xi_4(t_1)A_4 \\ &= 0.21\begin{bmatrix} 0 & 1 \\ -2 & -2 \end{bmatrix} + 0.09\begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix} + 0.49\begin{bmatrix} 0 & 1 \\ -3 & -2 \end{bmatrix} + 0.21\begin{bmatrix} 1 & 1 \\ -2 & -2 \end{bmatrix} \\ &= \begin{bmatrix} 0.21 & 1 \\ -2.4 & -2 \end{bmatrix} \end{aligned}$$

and

$$b(t_1) = \xi_1(t_1)b_1 + \xi_2(t_1)b_2 + \xi_3(t_1)b_3 + \xi_4(t_1)b_4$$
$$= 0.21\begin{bmatrix}0\\2\end{bmatrix} + 0.09\begin{bmatrix}2\\2\end{bmatrix} + 0.49\begin{bmatrix}-2\\2\end{bmatrix} + 0.21\begin{bmatrix}0\\-2\end{bmatrix} = \begin{bmatrix}-0.8\\1.16\end{bmatrix}$$

To summarize, the system at time t_1 is described by

$$\begin{bmatrix}\dot{x}_1\\\dot{x}_2\end{bmatrix} = \begin{bmatrix}0.21 & 1\\-2.4 & -2\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix} + \begin{bmatrix}-0.8\\1.16\end{bmatrix}u$$

Note that this describes the system at only one time t_1. As t changes, $A(t)$ and $b(t)$ change.

A simulation of this system was carried out using fourth-order Runge–Kutta integration with a step size of $\Delta t = 0.01$ s. An input of $u(t) = 3\sin(\pi t)$ with initial condition $x_1(0) = x_2(0) = 0$ yields the state trajectories shown in Figure 6.6. The nonlinear nature of the system can be seen in this response. The Matlab simulation program producing Figure 6.6 is given in the Appendix.

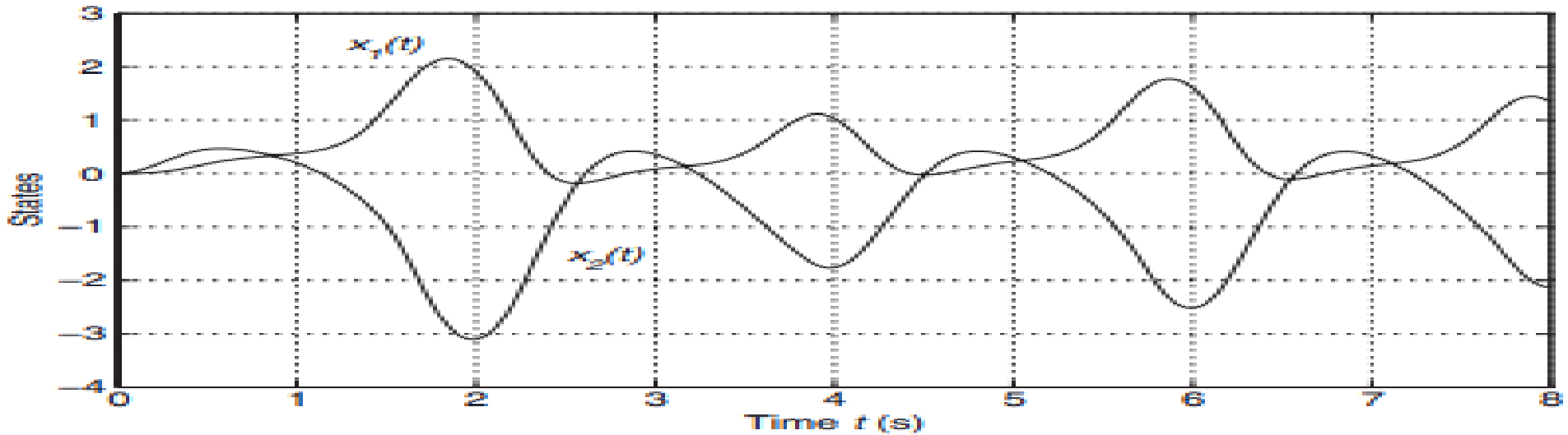


Figure 6.6 State trajectories of T–S fuzzy system that interpolates four linear continuous-time constant-coefficient state-space systems, Example 6.2.

6.3 TAKAGI–SUGENO FUZZY SYSTEMS AS INTERPOLATORS BETWEEN DISCRETE-TIME LINEAR STATE-SPACE DYNAMIC SYSTEMS

T–S fuzzy systems can be used to create time-varying discrete-time state-space dynamic systems. Consider a T–S fuzzy system with R rules of the form:

R_i: If $x_1(k)$ is P_1^K and $x_2(k)$ is P_2^L and ... and $x_n(k)$ is P_n^M, then

$$x^i(k+1) = A_i x(k) + b_i x(k). \tag{6.11}$$

The contents of the $n \times 1$ vector $x(k) = [x_1(k), x_2(k), \ldots, x_n(k)]^T$ are the system states that vary with the discrete time $k \in \{0, 1, 2, 3, \ldots\}$. The input universes $\mathcal{X}_1$, $\mathcal{X}_2$, ... , $\mathcal{X}_n$ are as before, with various fuzzy sets defined on them, but now the consequents are discrete-time linear constant-coefficient state-space systems. This fuzzy system does not have a single crisp output, rather it produces a time-varying system described by the state-space difference equation:

$$x(k+1) = \frac{\sum_{i=1}^{R} \mu_i(k)(A_i x(k) + b_i x(k))}{\sum_{i=1}^{R} \mu_i(k)} \tag{6.12}$$

or

$$x(k+1) = A(k)x(k) + b(k)u(k) \tag{6.13}$$

where $A(k) = \xi_1(k)A_1 + \xi_2(k)A_2 + \ldots + \xi_R(k)A_R$ and $b(k) = \xi_1(k)b_1 + \xi_2(k)b_2 + \ldots + \xi_R(k)b_R$ with fuzzy basis functions

$$\xi_i(k) = \frac{\mu_i(k)}{\sum_{j=1}^{R} \mu_j(k)} \tag{6.14}$$

As k varies, the states $x(k)$ also vary. Therefore, the fuzzy basis functions and the matrices $A(k)$ and $b(k)$ vary with time. Matrices $A(k)$ and $b(k)$ contain functions of the system states, making the system (6.13) a nonlinear time-varying system.

EXAMPLE 6.3

Consider a two-input T–S fuzzy system with rules

1. If $x_1(k)$ is P_1^1 and $x_2(k)$ is P_2^1, then $x^1(k+1) = A_1x(k) + b_1u(k)$.
2. If $x_1(k)$ is P_1^1 and $x_2(k)$ is P_2^2, then $x^2(k+1) = A_2x(k) + b_2u(k)$.
3. If $x_1(k)$ is P_1^2 and $x_2(k)$ is P_2^1, then $x^3(k+1) = A_3x(k) + b_3u(k)$.
4. If $x_1(k)$ is P_1^2 and $x_2(k)$ is P_2^2, then $x^4(k+1) = A_4x(k) + b_4u(k)$.

where $A_1 = \begin{bmatrix} 0.1 & 0 \\ -0.2 & 0.2 \end{bmatrix}$, $b_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $A_2 = \begin{bmatrix} 0 & 0.1 \\ 1 & -0.2 \end{bmatrix}$, $b_2 = \begin{bmatrix} 0.1 \\ 1 \end{bmatrix}$

$$A_3 = \begin{bmatrix} -0.3 & 1 \\ 0 & 0.2 \end{bmatrix}, b_3 = \begin{bmatrix} -0.1 \\ 1 \end{bmatrix}, A_4 = \begin{bmatrix} 0.1 & 1 \\ 0.2 & 0.5 \end{bmatrix}, b_4 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

Let the fuzzy sets P_1^1, P_1^2, P_2^1, and P_2^2 be characterized by the Gaussian membership functions shown in Figures 6.1 and 6.2. Then for an arbitrary input (x_1, x_2) and

product T-norm, the premise values for the above four rules are calculated as in (6.6).

If we consider a particular instant of time k_1 at which $x_1(k_1) = 0.4$ and $x_2(k_1) = -0.8$, the instantaneous fuzzy basis function values are

$$\xi_1(k_1) = \frac{\mu_1(k_1)}{\sum_{i=1}^{4}\mu_i(k_1)} = \frac{0.25}{0.25+0.0272+0.7578+0.0825} = 0.2237$$

$$\xi_2(k_1) = \frac{\mu_2(k_1)}{\sum_{i=1}^{4}\mu_i(k_1)} = \frac{0.0272}{0.25+0.0272+0.7578+0.0825} = 0.0243$$

$$\xi_3(k_1) = \frac{\mu_3(k_1)}{\sum_{i=1}^{4}\mu_i(k_1)} = \frac{0.7578}{0.25+0.0272+0.7578+0.0825} = 0.6782$$

$$\xi_4(k_1) = \frac{\mu_4(k_1)}{\sum_{i=1}^{4}\mu_i(k_1)} = \frac{0.0825}{0.25+0.0272+0.7578+0.0825} = 0.0738$$

Then the system at time k_1 is described by the difference equation

$$x(k+1) = A(k_1)x(k) + b(k_1)u(k)$$

where

$$\begin{aligned} A(k_1) &= \xi_1(k_1)A_1 + \xi_2(k_1)A_2 + \xi_3(k_1)A_3 + \xi_4(k_1)A_4 \\ &= 0.2237\begin{bmatrix} 0.1 & 0 \\ -0.2 & 0.2 \end{bmatrix} + 0.0243\begin{bmatrix} 0 & 0.1 \\ 1 & -0.2 \end{bmatrix} + 0.6782\begin{bmatrix} -0.3 & 1 \\ 0 & 0.2 \end{bmatrix} \\ &\quad + 0.0738\begin{bmatrix} 0.1 & 1 \\ 0.2 & 0.5 \end{bmatrix} = \begin{bmatrix} -0.1737 & 0.7544 \\ -0.0056 & 0.2124 \end{bmatrix} \end{aligned}$$

and

$$b(k_1) = \xi_1(k_1)b_1 + \xi_2(k_1)b_2 + \xi_3(k_1)b_3 + \xi_4(k_1)b_4$$
$$= 0.2237\begin{bmatrix}0\\1\end{bmatrix} + 0.0243\begin{bmatrix}1\\1\end{bmatrix} + 0.6782\begin{bmatrix}-1\\1\end{bmatrix} + 0.0738\begin{bmatrix}0\\-1\end{bmatrix} = \begin{bmatrix}-0.06538\\0.8524\end{bmatrix}$$

To summarize, the system at time k_1 is described by

$$\begin{bmatrix}x_1(k+1)\\x_2(k+1)\end{bmatrix} = \begin{bmatrix}-0.1737 & 0.7544\\-0.0056 & 0.2124\end{bmatrix}\begin{bmatrix}x_1(k)\\x_2(k)\end{bmatrix} + \begin{bmatrix}-0.06538\\0.8524\end{bmatrix}u(k)$$

Note that this describes the system at only one time k_1. As k changes, $A(k)$ and $b(k)$ change.

This system was simulated with $x_1(0) = x_2(0) = 0$, a sampling time of $\Delta t = 0.2$ s, and $u(t) = 3\sin(\pi t)$, where $t = k\Delta t$ and k is an integer. The resulting state trajectories

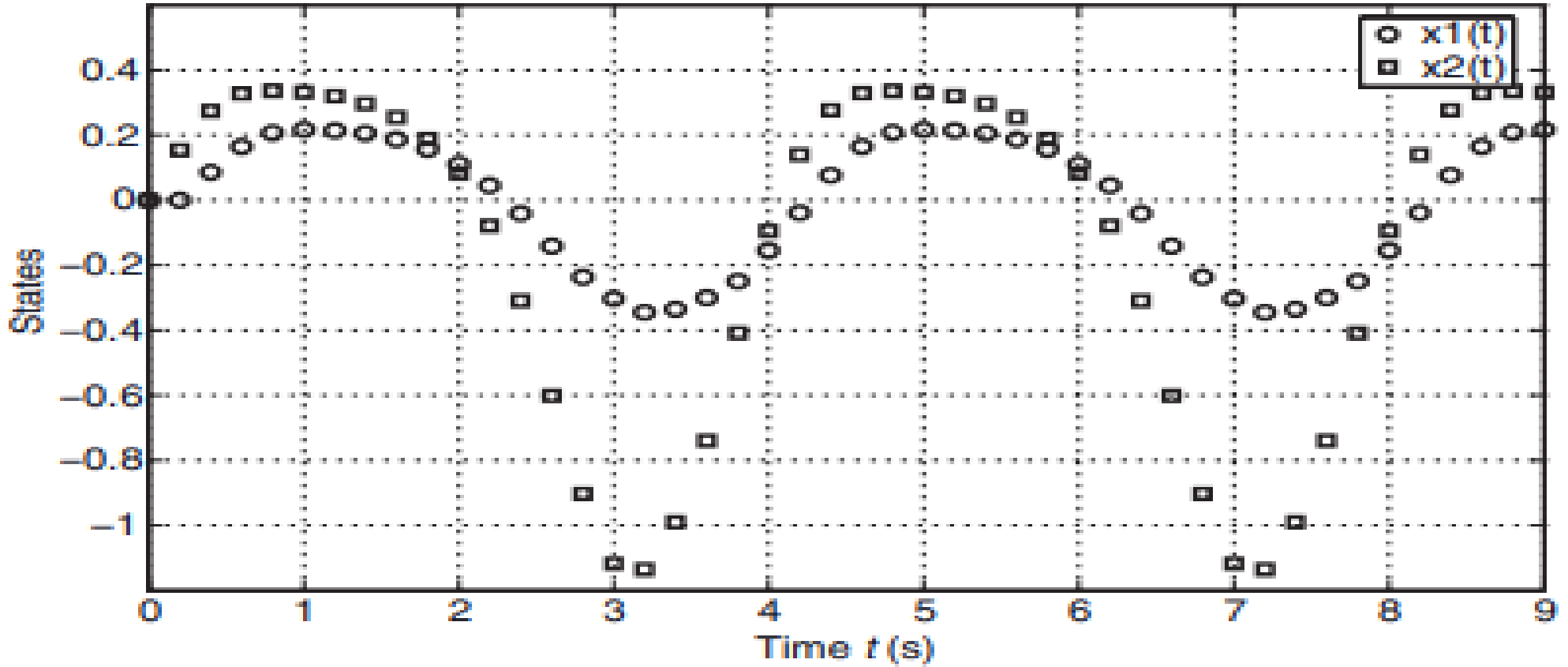


Figure 6.7. Input and corresponding states for discrete-time system created by a four-rule T–S fuzzy system.

are shown in Figure 6.7. The nonlinear nature of the system can be seen in this response. The Matlab simulation program producing Figure 6.7 is given in the Appendix.

PARALLEL DISTRIBUTED CONTROL WITH TAKAGI – SUGENO FUZZY SYSTEMS

If a plant has a model in the form of a Takagi–Sugeno (T–S) fuzzy system that interpolates linear dynamic systems, which we call the *plant fuzzy system*, one approach to stabilizing it is called *parallel distributed control* (PDC) [9,17,29]. The basic philosophy of PDC is to create a *controller fuzzy system* with rule premises identical to those of the plant fuzzy system. In the controller fuzzy system, each rule's consequent is a control law designed to control the linear system in the corresponding consequent of the plant fuzzy system. The overall control law is thus a weighted average of the individual control laws, just as the overall nonlinear system is a weighted average of the linear systems in each consequent of the plant fuzzy system.

Parallel distributed control is important because it constitutes a method of controlling nonlinear systems. Furthermore, as will be seen in Chapter 10, a nonlinear system can sometimes be identified online as a T–S fuzzy system. If a parallel distributed controller can then be designed based on this identification, a type of real-time control known as *adaptive control* can be effected for nonlinear systems.

The control methods utilized in this book in parallel distributed control schemes are the ones contained in Chapter 5, that is pole placement, tracking, and model reference. Pole placement via parallel distributed control has the distinct advantage that there exist mathematical tools to prove closed-loop stability, as discussed in Theorems 7.1 and 7.2. Such tools are lacking for control via Mamdani fuzzy systems.

7.1 CONTINUOUS-TIME SYSTEMS

Let the plant fuzzy system rule base consist of R rules of the form:

$$R_i: \quad \text{If } x_1(t) \text{ is } P_1^K \text{ and } x_2(t) \text{ is } P_2^L \text{ and } \cdots \text{ and } x_n(t) \text{ is } P_n^M, \text{ then} \qquad \dot{x}^i(t) = A_i x(t) + b_i u(t) \tag{7.1}$$

Then the system is given by the nonlinear/time-varying model given by (6.8) and (6.9).

The parallel distributed pole placement controller for this system is another fuzzy system with R rules of the form:

$$R_i \qquad \text{If } x_1(t) \text{ is } P_1^K \text{ and } x_2(t) \text{ is } P_2^L \text{ and } \cdots \text{ and } x_n(t) \text{ is } P_n^M, \text{ then } u^i(t) = -k_i x(t) \tag{7.2}$$

where the eigenvalues of $A_i - b_i k_i$ are as desired in the left half-plane (see Section 5.3.1). The resulting control law is

$$u(t) = -\left[\sum_{i=1}^{R} k_i \xi_i(t)\right] x(t) \tag{7.3}$$

where $\xi_i(t)$ are the fuzzy basis functions defined in (6.4).

Not every PDC controller produces a stable closed-loop system, despite the fact that each consequent system in the plant fuzzy system is stabilized by the control law in the corresponding consequent of the controller fuzzy system. Nevertheless, we have the following stability result for pole placement via parallel distributed control.

THEOREM 7.1 (CONTINUOUS-TIME PARALLEL DISTRIBUTED POLE PLACEMENT)

In the parallel distributed pole placement scheme described by (7.1)–(7.3), assume each consequent system of the plant fuzzy system is controllable. Then, $\underline{x} = \underline{0}$ is globally asymptotically stable if there exists a positive definite symmetric matrix G such that for all $i = 1, \ldots, R$ and $j = 1, \ldots, R$,

$$G(A_i - b_i k_j) + (A_i - b_i k_j)^{\mathrm{T}} G < 0 \tag{7.4}$$

where $M < 0$ means that the matrix M is negative definite.

The proof follows directly from Lyapunov stability theory. Note that the inequality (7.4) must be satisfied by *one* matrix G for all combinations of i and j. The inequality (7.4) is called a *continuous-time linear matrix inequality* (LMI) [29–31]. LMIs, such as (7.4), are difficult to solve by hand. Therefore, there are various computational tools to solve them.

EXAMPLE 7.1

Consider a nonlinear two-state dynamic system described by the T–S fuzzy system

1. If x_1 is P_1^1 and x_2 is P_2^1, then $\dot{x}^1(t) = A_1 x(t) + b_1 u(t)$.
2. If x_1 is P_1^1 and x_2 is P_2^2, then $\dot{x}^2(t) = A_2 x(t) + b_2 u(t)$.
3. If x_1 is P_1^2 and x_2 is P_2^1, then $\dot{x}^3(t) = A_3 x(t) + b_3 u(t)$.
4. If x_1 is P_1^2 and x_2 is P_2^2, then $\dot{x}^4(t) = A_4 x(t) + b_4 u(t)$.

where

$$A_1 = \begin{bmatrix} 0 & 1 \\ -2 & 2 \end{bmatrix}, b_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 1 \\ -3 & 2 \end{bmatrix}, b_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0 & 1 \\ -2 & 1 \end{bmatrix}, b_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, A_4 = \begin{bmatrix} 0 & 1 \\ -3 & 1 \end{bmatrix}, b_4 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Notice that each consequent system is controllable and unstable. The eigenvalues of A_1 are $1 \pm j$, those of A_2 are $1 \pm j\sqrt{2}$, those of A_3 are $0.5 \pm j1.32$, and those of A_4 are $0.5 \pm j1.66$. Therefore, this T–S fuzzy system is nonlinear and unstable. To stabilize it, we design a parallel distributed pole placement controller with state feedback control laws to place the closed-loop eigenvalues of each system at $-1 \pm j$. The parallel distributed controller is given by

1. If $x_1(t)$ is P_1^1 and $x_2(t)$ is P_2^1, then $u^1(t) = -k_1 x(t)$.
2. If $x_1(t)$ is P_1^1 and $x_2(t)$ is P_2^2, then $u^2(t) = -k_2 x(t)$.
3. If $x_1(t)$ is P_1^2 and $x_2(t)$ is P_2^1, then $u^3(t) = -k_3 x(t)$.
4. If $x_1(t)$ is P_1^2 and $x_2(t)$ is P_2^2, then $u^4(t) = -k_4 x(t)$.

where $k_1 = [0 \quad 4]$, $k_2 = [-1 \quad 4]$, $k_3 = [0 \quad 3]$, and $k_4 = [-1 \quad 3]$. Then the control law applied to the plant is

$$u(t) = -\left[k_1 \xi_1(t) + k_2 \xi_2(t) + k_3 \xi_3(t) + k_4 \xi_4(t)\right] x(t) \tag{7.5}$$

The requirement (7.4) for stability amounts to 16 inequalities for this particular problem. These are

$$
\begin{aligned}
&G(A_1 - b_1k_1) + (A_1 - b_1k_1)^{\mathrm{T}}G < 0\\
&G(A_1 - b_1k_2) + (A_1 - b_1k_2)^{\mathrm{T}}G < 0\\
&G(A_1 - b_1k_3) + (A_1 - b_1k_3)^{\mathrm{T}}G < 0\\
&G(A_1 - b_1k_4) + (A_1 - b_1k_4)^{\mathrm{T}}G < 0\\
&G(A_2 - b_2k_1) + (A_2 - b_2k_1)^{\mathrm{T}}G < 0\\
&G(A_2 - b_2k_2) + (A_2 - b_2k_2)^{\mathrm{T}}G < 0\\
&G(A_2 - b_2k_3) + (A_2 - b_2k_3)^{\mathrm{T}}G < 0\\
&G(A_2 - b_2k_4) + (A_2 - b_2k_4)^{\mathrm{T}}G < 0\\
&G(A_3 - b_3k_1) + (A_3 - b_3k_1)^{\mathrm{T}}G < 0\\
&G(A_3 - b_3k_2) + (A_3 - b_3k_2)^{\mathrm{T}}G < 0\\
&G(A_3 - b_3k_3) + (A_3 - b_3k_3)^{\mathrm{T}}G < 0\\
&G(A_3 - b_3k_4) + (A_3 - b_3k_4)^{\mathrm{T}}G < 0\\
&G(A_4 - b_4k_1) + (A_4 - b_4k_1)^{\mathrm{T}}G < 0\\
&G(A_4 - b_4k_2) + (A_4 - b_4k_2)^{\mathrm{T}}G < 0\\
&G(A_4 - b_4k_3) + (A_4 - b_4k_3)^{\mathrm{T}}G < 0\\
&G(A_4 - b_4k_4) + (A_4 - b_4k_4)^{\mathrm{T}}G < 0
\end{aligned}
$$

As stated above, all 16 of these inequalities must be satisfied by one positive definite symmetric matrix G.

It may be verified that the matrix

$$G = \begin{bmatrix} 1509 & 347 \\ 347 & 653 \end{bmatrix}$$

satisfies these 16 LMIs. This G was found with the aid of the Matlab *LMI Control Toolbox*. Therefore, we are assured that the above parallel distributed controller renders the point $\underline{x} = \underline{0}$ globally asymptotically stable in the above closed-loop system.

Let the fuzzy sets P_1^1, P_1^2, P_2^1, and P_2^2 be characterized by the membership functions shown in Figures 6.1 and 6.2. With initial conditions $x_1(0) = 1$, $x_2(0) = -1$, and with no feedback, the state trajectories of Figure 7.1 result. The trajectories increase unboundedly because the open-loop system is unstable. With the same initial conditions and parallel distributed pole placement feedback control law (7.5), the trajectories of Figure 7.2 result, indicating a stable closed-loop system. The Matlab program producing Figures 7.1 and 7.2 is given in the Appendix.

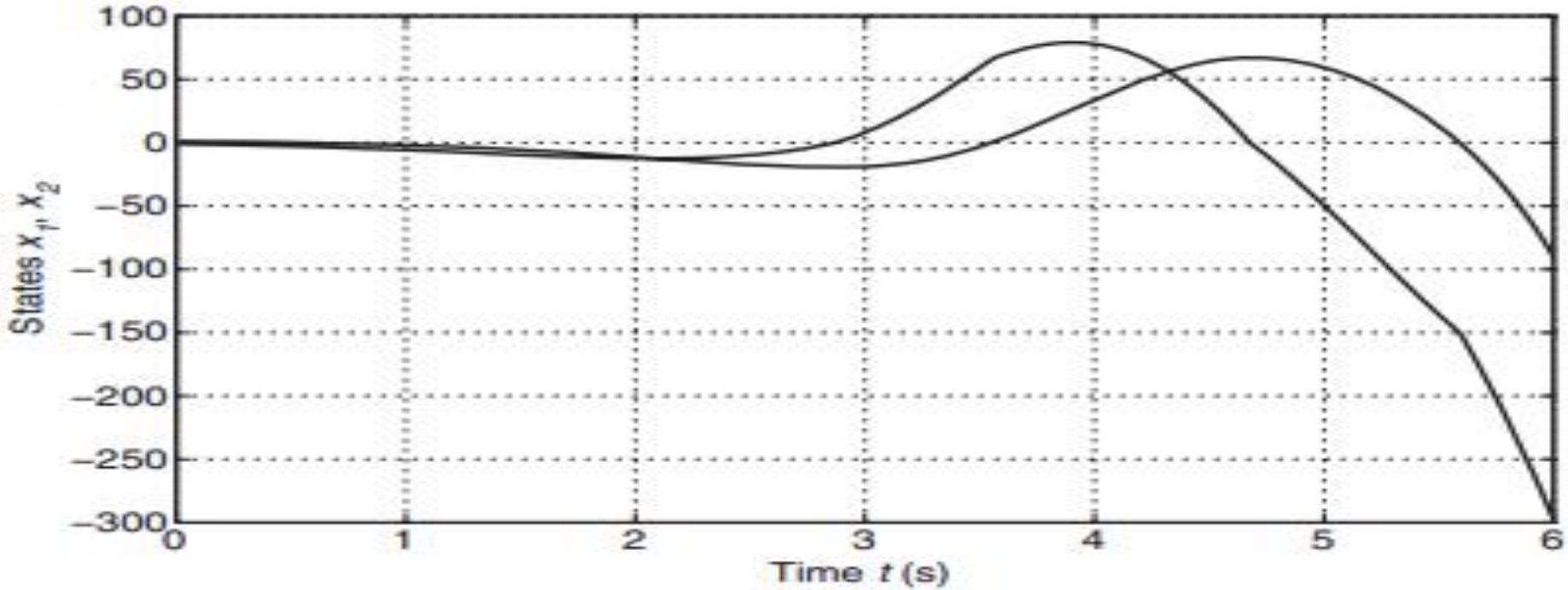


Figure 7.1. State trajectories of open-loop continuous-time T–S plant.

7.2 DISCRETE-TIME SYSTEMS

Let the plant fuzzy system rule base consist of R rules of the form:

$$R_i: \quad \text{If } x_1(k) \text{ is } P_1^K \text{ and } x_2(k) \text{ is } P_2^L \text{ and } \cdots \text{ and } x_n(k) \text{ is } P_n^M, \text{ then}$$
$$x^i(k+1) = A_i x(k) + b_i u(k) \tag{7.6}$$

Then the system obeys the nonlinear/time-varying model given by (6.12) and (6.13).

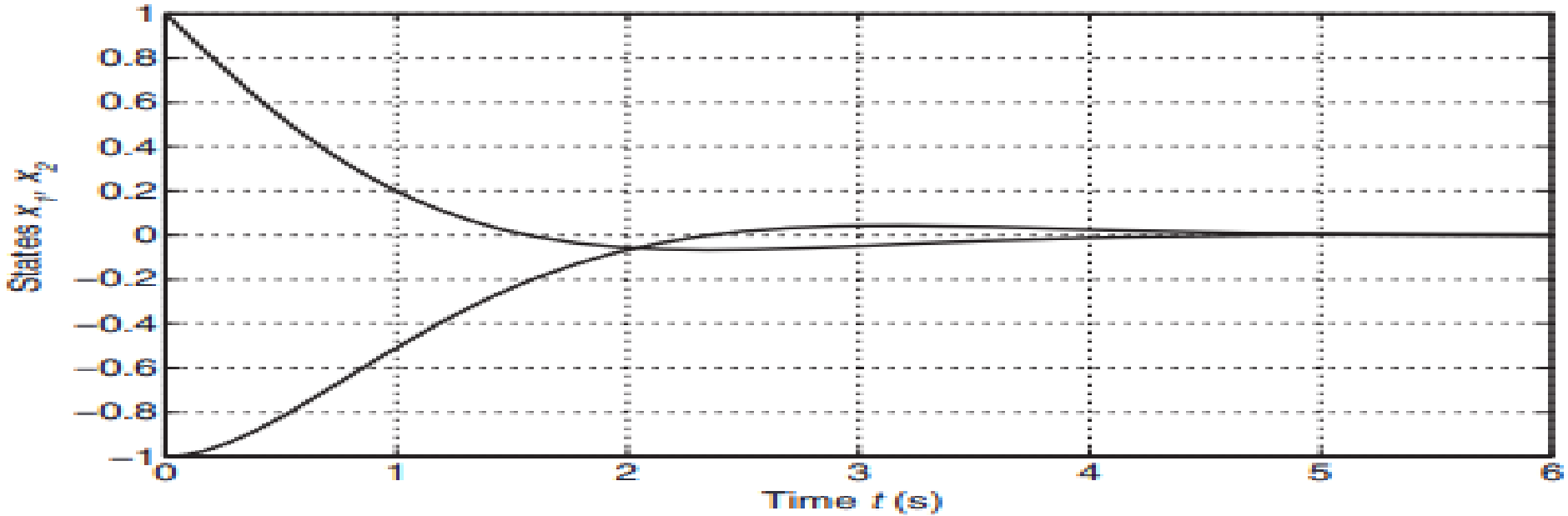


Figure 7.2. State trajectories of continuous-time T–S plant with parallel distributed pole placement.

The parallel distributed pole-placement controller for this system is another fuzzy system with R rules of the form:

$$R_i: \quad \text{If } x_1(k) \text{ is } P_1^K \text{ and } x_2(k) \text{ is } P_2^L \text{ and } \cdots \text{ and } x_n(k) \text{ is } P_n^M, \text{ then } u^i(k) = -k_i x(k) \tag{7.7}$$

where the eigenvalues of $A_i - b_i k_i$ are as desired inside the unit circle (see Section 5.3.1). The resulting control law is

$$u(k) = -\left[\sum_{i=1}^{R} k_i \xi_i(k)\right] x(k) \tag{7.8}$$

where ξ_i are the fuzzy basis functions defined in (6.14). The stability result for discrete-time parallel distributed control is as follows.

THEOREM 7.2 (DISCRETE-TIME PARALLEL DISTRIBUTED POLE PLACEMENT)

In the parallel distributed pole placement scheme described by (7.6)–(7.8), assume each consequent system of the plant fuzzy system is controllable. Then $\underline{x} = \underline{0}$ is globally asymptotically stable if there exists a positive definite symmetric matrix G such that for all $i = 1, \ldots, R$ and $j = 1, \ldots, R$,

$$(A_i - b_i k_j)^T G (A_i - b_i k_j) - G < 0 \tag{7.9}$$

The inequality (7.9) is called a *discrete-time LMI*.

EXAMPLE 7.2

Consider a nonlinear dynamic system described by the T–S fuzzy system

1. If $x_1(k)$ is P_1^1 and $x_2(k)$ is P_2^1, then $x^1(k+1) = A_1x(k) + b_1u(k)$.
2. If $x_1(k)$ is P_1^1 and $x_2(k)$ is P_2^2, then $x^2(k+1) = A_2x(k) + b_2u(k)$.
3. If $x_1(k)$ is P_1^2 and $x_2(k)$ is P_2^1, then $x^3(k+1) = A_3x(k) + b_3u(k)$.
4. If $x_1(k)$ is P_1^2 and $x_2(k)$ is P_2^2, then $x^4(k+1) = A_4x(k) + b_4u(k)$.

where

$$A_1 = \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix}, b_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, A_2 = \begin{bmatrix} 1 & 1 \\ -1 & 2 \end{bmatrix}, b_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
$$A_3 = \begin{bmatrix} 0 & 1 \\ -0.2 & 1 \end{bmatrix}, b_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, A_4 = \begin{bmatrix} 1 & 1 \\ -0.3 & 1 \end{bmatrix}, b_4 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The eigenvalues of A_1 are $0.5 \pm j0.866$, those of A_2 are $1.5 \pm j0.866$, those of A_3 are 0.2764, 0.7236, and those of A_4 are $1 \pm j0.5477$. Therefore, this T–S fuzzy system is nonlinear and unstable, because the eigenvalues of A_1, A_2, and A_4 have magnitudes that are not less than unity. To stabilize this system, we design a parallel distributed controller with state feedback control laws to place the closed-loop eigenvalues of each system at −0.5, 0.5. The parallel distributed controller is given by

1. If $x_1(k)$ is P_1^1 and $x_2(k)$ is P_2^1, then $u^1(k) = -k_1x(k)$.
2. If $x_1(k)$ is P_1^1 and $x_2(k)$ is P_2^2, then $u^2(k) = -k_2x(k)$.
3. If $x_1(k)$ is P_1^2 and $x_2(k)$ is P_2^1, then $u^3(k) = -k_3x(k)$.
4. If $x_1(k)$ is P_1^2 and $x_2(k)$ is P_2^2, then $u^4(k) = -k_4x(k)$.

where

$$k_1 = [-1.25 \quad 1], k_2 = [-0.25 \quad 3], k_3 = [-0.45 \quad 1], k_4 = [0.45 \quad 2]$$

Then the control law applied to the system is

$$u(k) = -\left[k_1\xi_1(k) + k_2\xi_2(k) + k_3\xi_3(k) + k_4\xi_4(k)\right]x(k) \tag{7.10}$$

Let the fuzzy sets P_1^1, P_1^2, P_2^1, and P_2^2 be characterized by the membership functions shown in Figures 6.1 and 6.2. With $\Delta t = 0.1$ s, initial conditions $x_1(0) = 1$, $x_2(0) = -1$, and with no feedback, the state trajectories of Figure 7.3 result. The trajectories increase unboundedly because the open-loop system is unstable. With the same initial conditions and parallel distributed pole placement feedback control law (7.10), the trajectories of Figure 7.4 result, indicating a stable closed-loop system. The Matlab program producing Figures 7.3 and 7.4 is given in the Appendix.

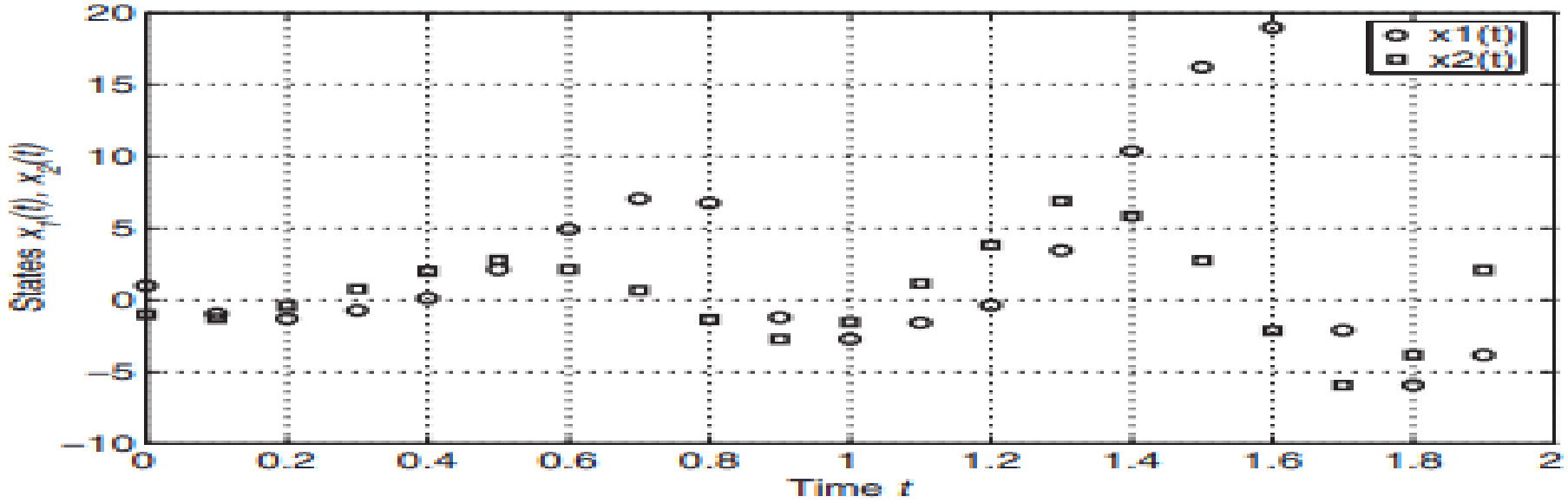


Figure 7.3. State trajectories of open-loop discrete-time T–S plant.

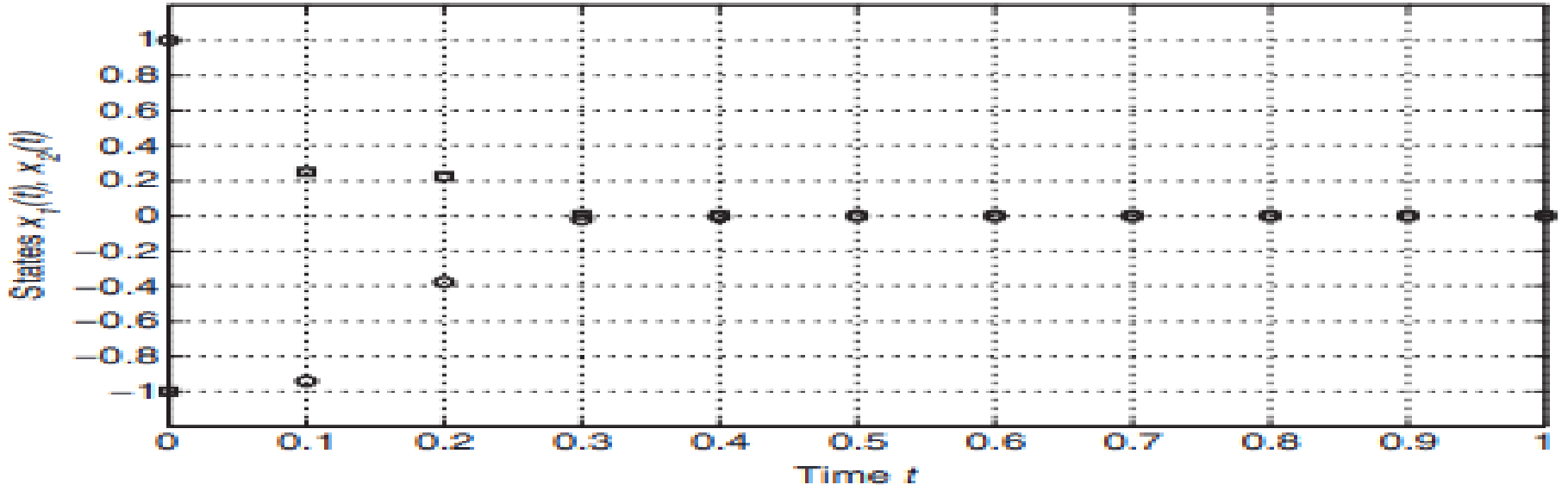


Figure 7.4. State trajectories of discrete-time T–S plant with parallel distributed pole placement.

7.3 PARALLEL DISTRIBUTED TRACKING CONTROL

The tracking control problem consists of determining a plant input such that the plant output tracks a desired reference signal. A parallel distributed-type controller can be designed to force the output of a nonlinear system modeled as a T–S fuzzy system to track a desired reference signal. The controller design is explained in Section 5.3.2 for a single linear plant. The parallel distributed tracking strategy is to apply this design procedure to every consequent plant in the plant fuzzy system. For tracking, the consequents in the plant fuzzy system are assumed to be in the form of (5.11), that is, input–output difference equations.

Let the plant fuzzy system rule base consist of R rules of the form:

$$R_i: \quad \text{If } y(k) \text{ is } P_1^K \text{ and } y(k-1) \text{ is } P_2^L \text{ and } \cdots \text{ and } y(k-n+1) \text{ is } P_n^M, \text{ then } y^i(k+1) = \alpha_i(q^{-1})\, y(k) + \beta_i(q^{-1})\, u(k) \tag{7.11}$$

where

$$\alpha_i(q^{-1}) = a_{i,1} + a_{i,2}q^{-1} + \cdots + a_{i,n}q^{-(n-1)} \tag{7.12a}$$

$$\beta_i(q^{-1}) = b_{i,0} + b_{i,1}q^{-1} + \cdots + b_{i,m}q^{-m} = b_{i,0} + \beta_i'(q^{-1}) \tag{7.12b}$$

and q^{-1} is the backward shift operator defined by $q^{-1}y(k) = y(k-1)$.

The parallel distributed one-step-ahead tracking controller for this system is another fuzzy system with R rules of the form:

$$R_i: \quad \text{If } y(k) \text{ is } P_1^K \text{ and } y(k-1) \text{ is } P_2^L \text{ and } \cdots \text{ and } y(k-n+1) \text{ is } P_n^M, \text{ then } u^i(k) = \frac{1}{b_{i,0}}\left[-\beta_i'(q^{-1})\, u(k) + r(k+1) - \alpha_i(q^{-1})\, y(k)\right] \tag{7.13}$$

The resulting control law is

$$u(k) = \sum_{i=1}^{R} u^i(k)\, \xi(k) \tag{7.14}$$

where $\xi_i(k)$, $i = 1, \ldots, R$ are the fuzzy basis functions defined in (6.14). This design procedure applies to n-step-ahead tracking problems with obvious modifications.

EXAMPLE 7.3

Consider the T–S fuzzy system with rule base:

1. If $y(k)$ is P_1^1 and $y(k-1)$ is P_2^1, then

$$y^1(k+1)=1.5y(k)-0.4y(k-1)+u(k)+0.6u(k-1)$$

2. If $y(k)$ is P_1^1 and $y(k-1)$ is P_2^2, then

$$y^2(k+1)=0.4y(k)-1.8y(k-1)+1.2u(k)+u(k-1)$$

3. If $y(k)$ is P_1^2 and $y(k-1)$ is P_2^1, then

$$y^3(k+1)=1.2y(k)+0.5y(k-1)+1.5u(k)+0.7u(k-1)$$

4. If $y(k)$ is P_1^2 and $y(k-1)$ is P_2^2, then

$$y^4(k+1)=0.8y(k)-1.6y(k-1)+1.5u(k)+u(k-1)$$

The consequent systems are in the form of the consequent systems of (7.11) with

$$\alpha_1(q^{-1})=1.5-0.4q^{-1},\ \alpha_2(q^{-1})=0.4-1.8q^{-1},\ \alpha_3(q^{-1})=1.2+0.5q^{-1}$$
$$\alpha_4(q^{-1})=0.8-1.6q^{-1},\ \beta_1(q^{-1})=1+0.6q^{-1},\ \beta_2(q^{-1})=1.2+q^{-1}$$

$$\beta_3(q^{-1}) = 1.5 + 0.7q^{-1},\ \beta_4(q^{-1}) = 1.5 + q^{-1},\ b_{1,0} = 1,\ \beta_1'(q^{-1}) = 0.6q^{-1},\ b_{2,0} = 1.2$$

$$\beta_2'(q^{-1}) = q^{-1},\ b_{3,0} = 1.5,\ \beta_3'(q^{-1}) = 0.7q^{-1},\ b_{4,0} = 1.5, \text{ and } \beta_4'(q^{-1}) = q^{-1}$$

Note that each consequent system is unstable because the polynomials

$$q^2[1 - q^{-1}\alpha_1(q^{-1})] = q^2 - 1.5q + 0.4,\ q^2[1 - q^{-1}\alpha_2(q^{-1})] = q^2 - 0.4q + 1.8$$

$$q^2[1 - q^{-1}\alpha_3(q^{-1})] = q^2 - 1.2q - 0.5, \text{ and } q^2[1 - q^{-1}\alpha_4(q^{-1})] = q^2 - 0.8q + 1.6$$

each have roots outside the unit circle.

As an example, consider the third consequent plant above, that is,

$$y^3(k+1) = 1.2y(k) + 0.5y(k-1) + 1.5u(k) + 0.7u(k-1)$$

For this plant, we have $\alpha_3(q^{-1}) = 1.2 + 0.5q^{-1}$, $\beta_3(q^{-1}) = 1.5 + 0.7q^{-1}$, $b_{3,0} = 1.5$, and $\beta_3'(q^{-1}) = 0.7q^{-1}$. Therefore, (5.31) yields

$$(1.5 + 0.7q^{-1})u^3(k) = r(k+1) - (1.2 + 0.5q^{-1})y(k)$$

to give the tracking control law

$$u^3(k) = \frac{1}{1.5}[-0.7u(k-1) - 1.2y(k) - 0.5y(k-1) + r(k+1)]$$

Using this design technique on all four consequent plants yields the parallel distributed one-step-ahead tracking controller:

1. If $y(k)$ is P_1^1 and $y(k-1)$ is P_2^1, then

$$u^1(k) = -0.6u(k-1) - 1.5y(k) + 0.4y(k-1) + r(k+1)$$

2. If $y(k)$ is P_1^1 and $y(k-1)$ is P_2^2, then

$$u^2(k) = \frac{1}{1.2}[-u(k-1) - 0.4y(k) + 1.8y(k-1) + r(k+1)]$$

3. If $y(k)$ is P_1^2 and $y(k-1)$ is P_2^1, then

$$u^3(k) = \frac{1}{1.5}[-0.7u(k-1) - 1.2y(k) - 0.5y(k-1) + r(k+1)]$$

4. If $y(k)$ is P_1^2 and $y(k-1)$ is P_2^2, then

$$u^4(k) = \frac{1}{1.5}[-u(k-1) - 0.8y(k) + 1.6y(k-1) + r(k+1)]$$

Let the input fuzzy sets P_1^1, P_1^2, P_2^1, and P_2^2 be defined as in Figure 6.4. Let the desired reference signal be $r(k) = 0.5\sin(0.2\pi k)$. With $\Delta t = 1$ second (hence $t = k\Delta t = k$), initial conditions $y(0) = 1$, $y(-1) = -1$, a plot of $y(t)$ together with $r(t)$ is shown in Figure 7.5 and the corresponding tracking error $y(t) - r(t)$ is shown in Figure 7.6.

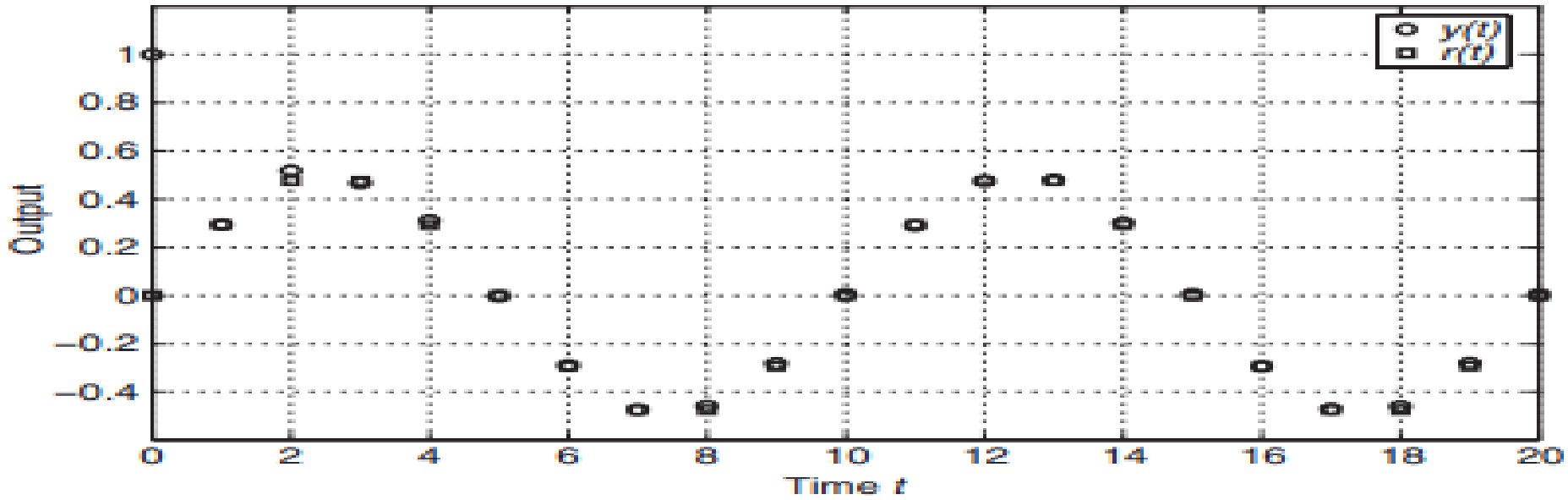


Figure 7.5. Tracking performance $y(t)$ and $r(t)$ of parallel distributed one-step-ahead tracking controller.

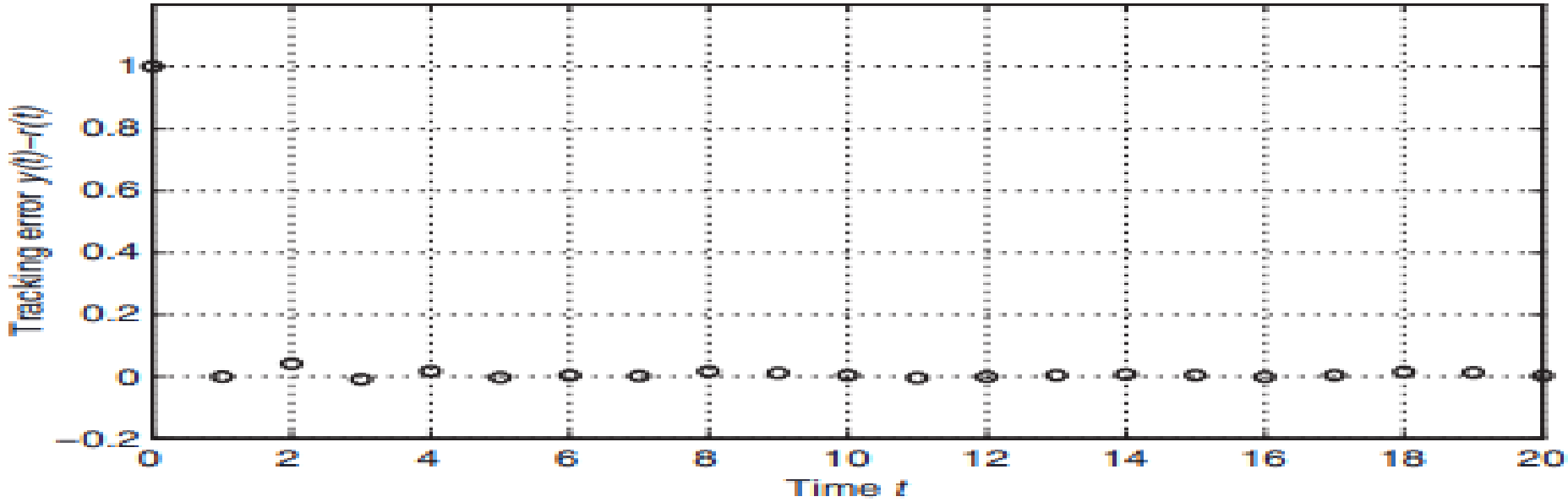


Figure 7.6. Tracking error $y(t) - r(t)$ of parallel distributed one-step-ahead tracking controller.

The tracking is not perfect; for this example the tracking error always remains in the range of $\pm 10^{-3}$ because of the interpolation properties of the T–S fuzzy plant and controller. The tracking would be perfect if each individual control law were applied to each individual plant. This is not the case, however, with parallel distributed-type controllers. Instead, we only have a weighted average of controllers controlling a weighted average of plants. Therefore, the overall control is only an approximation of the controller needed to effect perfect tracking for the overall plant, which is in fact nonlinear.

The Matlab code that produced Figures 7.5 and 7.6 is given in the Appendix.

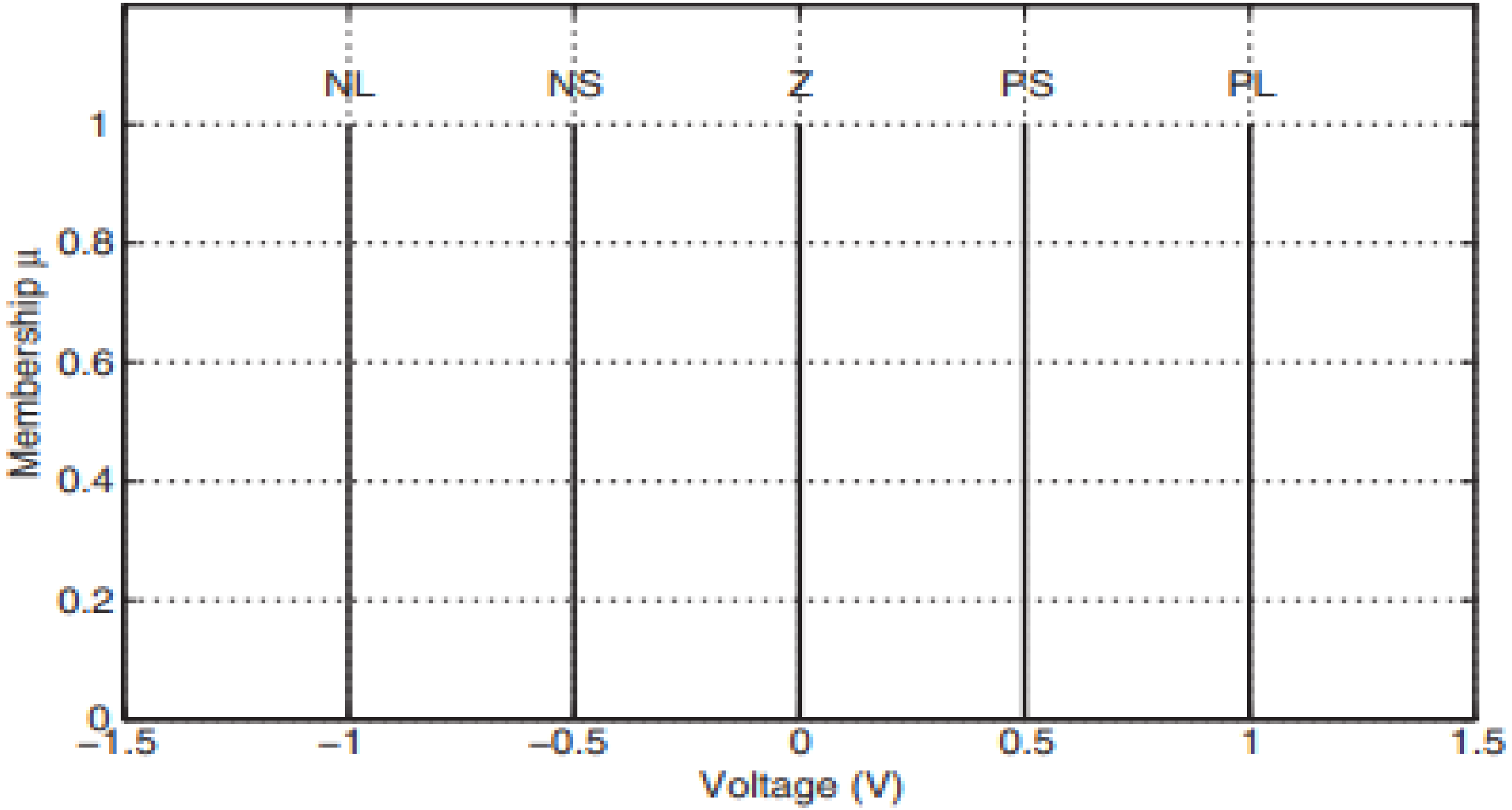


Figure 2.13. Membership functions characterizing five singleton fuzzy sets.

MODELING OF DYNAMIC PLANTS AS FUZZY SYSTEMS

9.1 MODELING KNOWN PLANTS AS T–S FUZZY SYSTEMS [29]

Many nonlinear systems with known mathematical models may be exactly modeled on a bounded domain in the state space with Takagi–Sugeno (T–S) fuzzy systems. Consider a nonlinear time-invariant system for which $x = 0$ is the equilibrium point. The basic idea of modeling it with a T–S fuzzy system is to express it as a series of linear dynamic systems, each one the consequent of one of the rules. Functions z_i are determined such that nonlinear terms in the plant model can be expressed as $z_i x$, where x is a plant state. Each of these functions z_i becomes an input to the fuzzy system. Two fuzzy sets forming a partition of unity are defined on each z_i universe, centered at the maximum and minimum of z_i.

Assume a scalar nonlinear function $z(x)$. If $z(x)$ is to be modeled in the domain $\mathcal{X}$, let $b_m = \min_{\mathcal{X}}(z)$ and $b_M = \max_{\mathcal{X}}(z)$. Then, create two fuzzy sets P^1 and P^2 on $\mathcal{X}$ characterized by triangular membership functions $\mu^1(z)$ and $\mu^2(z)$ as

$$\mu^1(z) = \frac{b_M - z}{b_M - b_m} \tag{9.1a}$$

$$\mu^2(z) = \frac{z - b_m}{b_M - b_m} \tag{9.1b}$$

Then, z can be exactly represented on $\mathcal{X}$ as

$$z = \mu^1(z) b_m + \mu^2(z) b_M \tag{9.2}$$

This technique can be used to exactly model a nonlinear dynamic system as a weighted average of linear systems.

EXAMPLE 9.1

Consider the nonlinear system $\dot{x} = f(x)$ and assume that $f(x)$ can be factored as $z(x)x$ [e.g., x^3 can be factored as $(x^2)x$ so that $z(x) = x^2$]. Now the nonlinear system can be expressed as $\dot{x} = z(x)x$. In order to exactly model the system on the bounded domain $\mathcal{X}$ as a T–S fuzzy system, let $b_m = \min_{\mathcal{X}}(z)$ and $b_M = \max_{\mathcal{X}}(z)$.

Create two fuzzy sets MIN and MAX on $\mathcal{X}$ characterized by triangular membership functions $\mu^{\text{MIN}}(z)$ and $\mu^{\text{MAX}}(z)$, as in (9.1). These are shown in Figure 9.1. Then the nonlinear system is exactly modeled on $\mathcal{X}$ by the T–S fuzzy system:

1. If z is MIN, then $\dot{x}^1 = b_m x$.
2. If z is MAX, then $\dot{x}^2 = b_M x$.

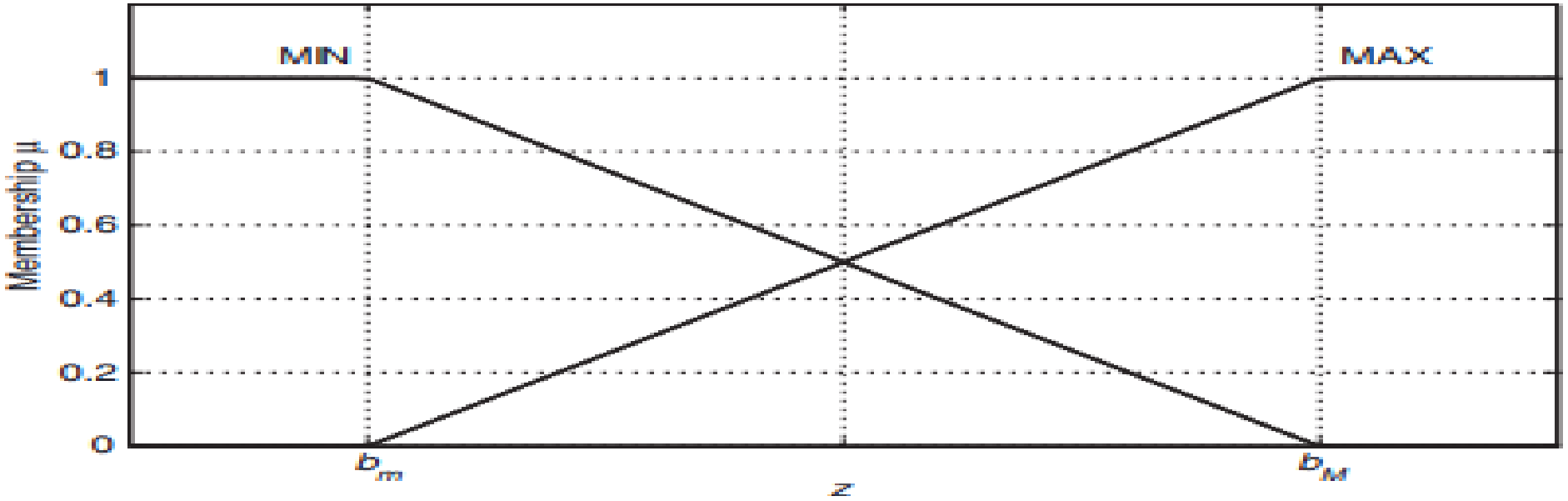


Figure 9.1. Fuzzy sets on z universe.

EXAMPLE 9.2

A nonlinear system has the mathematical model:

$$\dot{x}_1 = x_1 x_2 \tag{9.3a}$$

$$\dot{x}_2 = x_1 - x_2^3 + (1 + \cos^2 x_1)u \tag{9.3b}$$

It can be verified that this system is open-loop unstable [21,22]. Define $z_1 = x_1$, $z_2 = x_2^2$, and $z_3 = 1 + \cos^2 x_1$. Then, (9.3) can be rewritten as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & z_1 \\ 1 & -z_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ z_3 \end{bmatrix} u$$

We wish to derive a T–S fuzzy system whose behavior exactly duplicates (9.3). Assume the bounded domain $\mathcal{X}$ is defined by $x_1 \in [-2, 2]$ and $x_2 \in [-2, 2]$ [i.e., the T–S fuzzy system will duplicate (9.3) in this domain]. Then, $\min_{\mathcal{X}} z_1 = b_{1m} = -2$, $\max_{\mathcal{X}} z_1 = b_{1M} = 2$, $\min_{\mathcal{X}} z_2 = b_{2m} = 0$, $\max_{\mathcal{X}} z_2 = b_{2M} = 4$, $\min_{\mathcal{X}} z_3 = b_{3m} = 1$, $\max_{\mathcal{X}} z_3 = b_{3M} = 2$.

For z_1, z_2, and z_3, (9.1) yields

$$\mu^{\text{NEG}}(z_1) = \frac{2 - z_1}{4}$$

$$\mu^{\text{POS}}(z_1) = \frac{z_1 + 2}{4}$$

$$\mu^{\text{SMALL}}(z_2) = \frac{4 - z_2}{4}$$

$$\mu^{\text{LARGE}}(z_2) = \frac{z_2}{4}$$

$$\mu^{\text{SMALL}}(z_3) = \frac{2 - z_3}{1}$$

$$\mu^{\text{LARGE}}(z_3) = \frac{z_3 - 1}{1}$$

These memberships characterize fuzzy sets on the z_1 (Fig. 9.2), z_2 (Fig. 9.3), and z_3 (Fig. 9.4), universes.

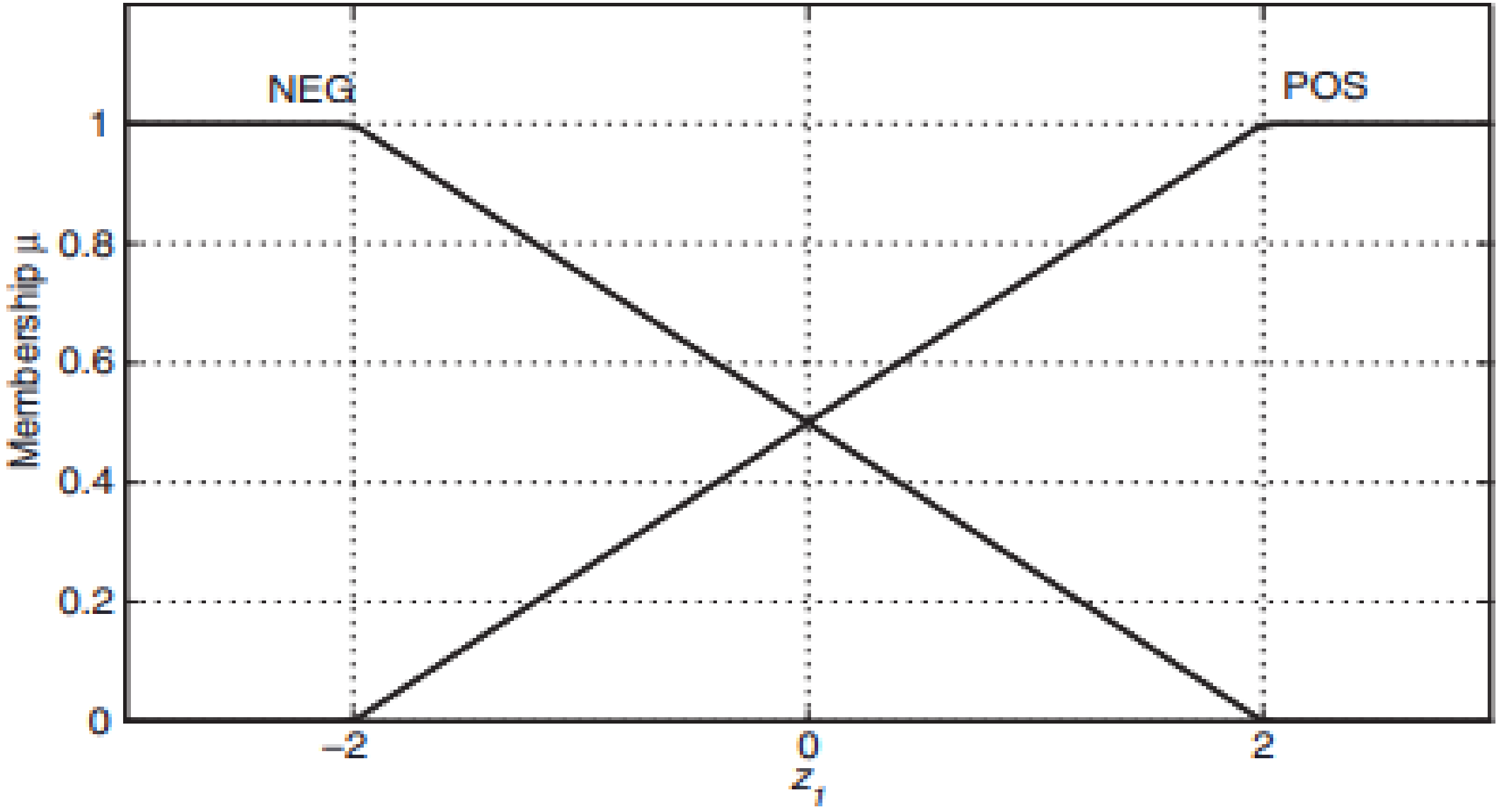


Figure 9.2. Fuzzy sets on z_1 universe.

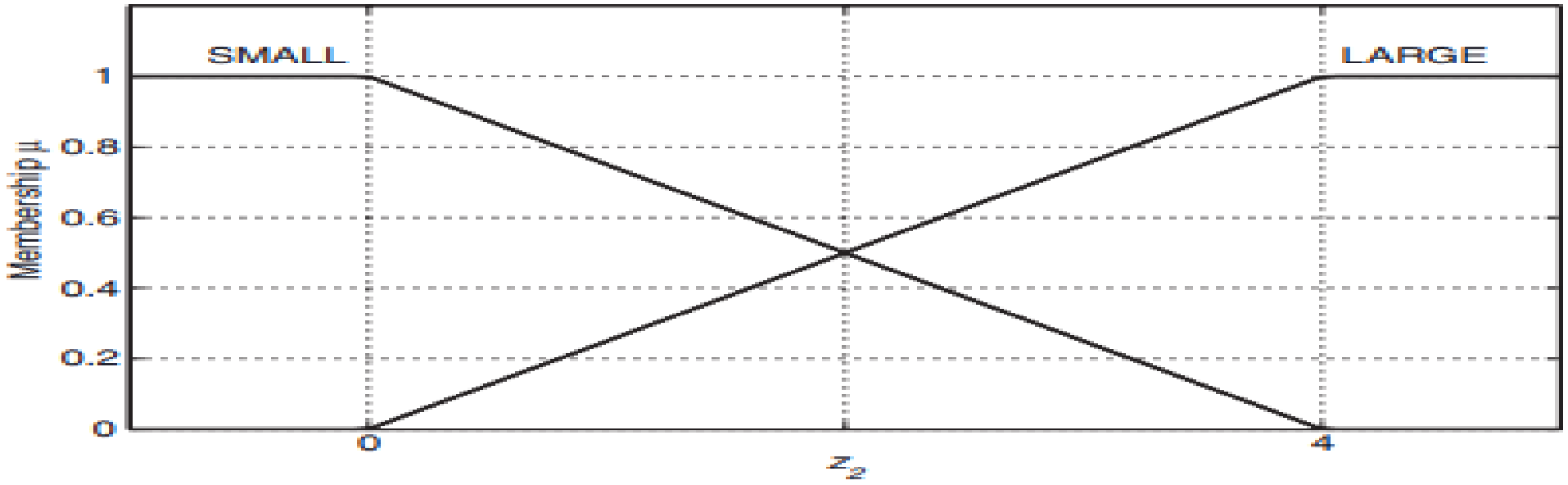


Figure 9.3. Fuzzy sets on z_2 universe.

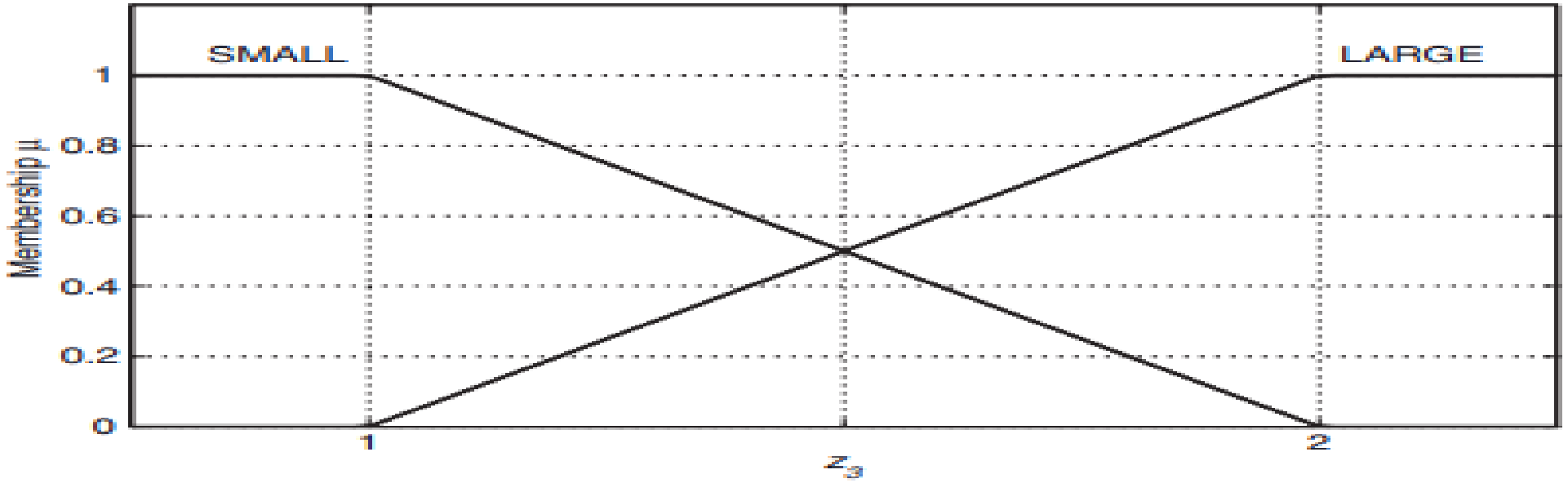


Figure 9.4. Fuzzy sets on z_3 universe.

The rule base of the equivalent fuzzy system is

1. If z_1 is NEG and z_2 is SMALL and z_3 is SMALL, then $\dot{x}^1 = \begin{bmatrix} 0 & -2 \\ 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$.
2. If z_1 is NEG and z_2 is SMALL and z_3 is LARGE, then $\dot{x}^2 = \begin{bmatrix} 0 & -2 \\ 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u$.
3. If z_1 is NEG and z_2 is LARGE and z_3 is SMALL, then $\dot{x}^3 = \begin{bmatrix} 0 & -2 \\ 1 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$.
4. If z_1 is NEG and z_2 is LARGE and z_3 is LARGE, then $\dot{x}^4 = \begin{bmatrix} 0 & -2 \\ 1 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u$.
5. If z_1 is POS and z_2 is SMALL and z_3 is SMALL, then $\dot{x}^5 = \begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$.
6. If z_1 is POS and z_2 is SMALL and z_3 is LARGE, then $\dot{x}^6 = \begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u$.
7. If z_1 is POS and z_2 is LARGE and z_3 is SMALL, then $\dot{x}^7 = \begin{bmatrix} 0 & 2 \\ 1 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$.
8. If z_1 is POS and z_2 is LARGE and z_3 is LARGE, then $\dot{x}^8 = \begin{bmatrix} 0 & 2 \\ 1 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u$.

This T–S fuzzy model exactly duplicates (9.3) in $x_1 \in [-2, 2]$, $x_2 \in [-2, 2]$. Note that the $1 + \cos^2 x_1$ term does not have to be expressed as linear in x because it multiplies the input u and not a state.

Now that we have a T–S fuzzy model of (9.3), we can design a parallel distributed controller as in Section 7.1. Let us design a PDC that places the closed-loop eigenvalues of each consequent of the plant fuzzy system at $-5 \pm j5$. Then the required controller fuzzy system has the rule base

1. If z_1 is NEG and z_2 is SMALL and z_3 is SMALL, then $u^1(t) = -[-24 \quad 10]x(t)$.
2. If z_1 is NEG and z_2 is SMALL and z_3 is LARGE, then $u^2(t) = -[-12 \quad 5]x(t)$.
3. If z_1 is NEG and z_2 is LARGE and z_3 is SMALL, then $u^3(t) = -[-24 \quad 6]x(t)$.
4. If z_1 is NEG and z_2 is LARGE and z_3 is LARGE, then $u^4(t) = -[-12 \quad 3]x(t)$.
5. If z_1 is POS and z_2 is SMALL and z_3 is SMALL, then $u^5(t) = -[26 \quad 10]x(t)$.
6. If z_1 is POS and z_2 is SMALL and z_3 is LARGE, then $u^6(t) = -[13 \quad 5]x(t)$.
7. If z_1 is POS and z_2 is LARGE and z_3 is SMALL, then $u^7(t) = -[26 \quad 6]x(t)$.
8. If z_1 is POS and z_2 is LARGE and z_3 is LARGE, then $u^8(t) = -[13 \quad 3]x(t)$.

To prove asymptotic stability, Theorem 7.1 requires that one positive definite symmetric G be bond to satisfy the 64 linear matrix inequalities (7.4) for all $i = 1, \ldots, 8$ and $j = 1, \ldots, 8$. However, it can be verified via simulation (though this is not a proof!) that the above fuzzy PDC controller does render $(x_1, x_2) = (0, 0)$ of (9.3) asymptotically stable.

9.2 IDENTIFICATION IN INPUT–OUTPUT DIFFERENCE EQUATION FORM

Assume an unknown plant with single input u and single output y. In order to find a fuzzy system to model it, it must be excited by an input while data $\{y(k), u(k)\}$, $k = 1, 2, 3, \ldots$ is taken from the plant. If the plant is continuous-time, which is usually the case, samples of the signals are taken every Δt s. In that case, $t = k\Delta t$ with $k = 1, 2, 3, \ldots$. Usually the Δt is suppressed and, for example, $y(k\Delta t)$ is written simply as $y(k)$.

The input sequence $u(k)$ should have sufficient frequency content to identify the plant. In general, the more complex the plant, the more different frequencies should be contained in u. It is generally not known what types of inputs provide sufficient excitation for nonlinear systems, but inputs consisting of many frequencies have the best chance of success.

If the plant is unknown, we can attempt to estimate it as an R-rule T–S fuzzy system with rules in the form:

$$\begin{aligned} R_i: \quad & \text{If } y(k) \text{ is } A_1^K \text{ and } y(k-1) \text{ is } A_2^L \text{ and } \cdots \text{ and } y(k-n+1) \text{ is } A_n^M, \text{ then} \\ & y^i(k+1) = a_1^i y(k) + a_2^i y(k-1) + \cdots + a_n^i y(k-n+1) + \\ & \qquad b_1^i u(k) + b_2^i u(k-1) + \cdots + b_n^i u(k-n+1) \end{aligned} \tag{9.4}$$

where the a and b coefficients and, in the case of gradient identification the input fuzzy sets, are to be determined. We note that (9.4) is not the only possibility of model structure in the consequent. The number of past inputs and outputs in the RHS regressions could be different from each other, and the delay of the model, which equals 1 in (9.4), could be other than 1.

9.2.1 Batch Least-Squares Identification in Input–Output Difference Equation Form

If least squares is to be used to identify a fuzzy model to estimate the plant, the input fuzzy sets must be fixed *a priori* so that the fuzzy system can be expressed in a form that is linear in the parameters (see Section 8.1). Therefore, the fuzzy basis functions are also known. If the known premise value of Rule i at time k is $\mu_i(y(k), \ldots, y(k-n+1))$ and defining the known fuzzy basis functions

$$\xi_i(k) = \frac{\mu_i(y(k), \cdots, y(k-n+1))}{\sum_{i=1}^{R} \mu_i(y(k), \cdots, y(k-n+1))} \tag{9.5}$$

for $i = 1, \ldots, R$, the output of the fuzzy system can be expressed as

$$\begin{aligned}\hat{y}(k+1) = &\left[a_1^1 y(k) + \cdots + a_n^1 y(k-n+1) + b_1^1 u(k) + \cdots + b_n^1 u(k-n+1)\right]\xi_1(k) + \\ &\left[a_1^2 y(k) + \cdots + a_n^2 y(k-n+1) + b_1^2 u(k) + \cdots + b_n^2 u(k-n+1)\right]\xi_2(k) + \cdots + \\ &\left[a_1^R y(k) + \cdots + a_n^R y(k-n+1) + b_1^R u(k) + \cdots + b_n^R u(k-n+1)\right]\xi_R(k)\end{aligned} \tag{9.6}$$

or $\hat{y}(k+1) = \phi^{\mathrm{T}}(k)\theta$, where

$$\phi(k) = \begin{bmatrix} y(k)\xi_1(k) \\ \vdots \\ y(k)\xi_R(k) \\ \vdots \\ y(k-n+1)\xi_1(k) \\ \vdots \\ y(k-n+1)\xi_R(k) \\ u(k)\xi_1(k) \\ \vdots \\ u(k)\xi_R(k) \\ \vdots \\ u(k-n+1)\xi_1(k) \\ \vdots \\ u(k-n+1)\xi_R(k) \end{bmatrix} \quad \text{and} \quad \theta = \begin{bmatrix} a_1^1 \\ \vdots \\ a_1^R \\ \vdots \\ a_n^1 \\ \vdots \\ a_n^R \\ b_1^1 \\ \vdots \\ b_1^R \\ \vdots \\ b_n^1 \\ \vdots \\ b_n^R \end{bmatrix} \tag{9.7}$$

From M input–output measurements, form matrices

$$\Phi = \begin{bmatrix} \phi^T(n) \\ \phi^T(n+1) \\ \vdots \\ \phi^T(M-1) \end{bmatrix} - (M-n) \times 2nR$$

and

$$Y = [y(n+1) \quad y(n+2) \quad \cdots \quad y(M)]^T - (M-n)\times 1$$

Then the batch least-squares estimate of θ based on the M measurements taken is calculated as in (8.6).

EXAMPLE 9.3

Consider the motor-driven robotic link described in Section 1.4.3 and modeled by the truth model [19]:

$$\ddot{\psi} = -64 \sin \psi - 5\dot{\psi} + 4 \times 10^4 u \tag{9.8}$$

where ψ is the angle of the link from the vertical-down position and u is the current input to the motor. The output is the measured link angle $y(t) = \psi(t)$. The system is simulated with a fourth-order Runge–Kutta integration algorithm with a step size of 0.001 s. For the input–output data, we save pairs $\{u(k\Delta t), y(k\Delta t)\}$ for $k = 1, \ldots, 998$ and $\Delta t = 0.01$ s, that is, we save input and output signals every 10 steps through the RK4 integration routine.

The input function chosen for the identification is a Gaussian pseudorandom sequence with zero mean and a standard deviation of 2×10^{-2}, giving the output shown in Figure 9.5.

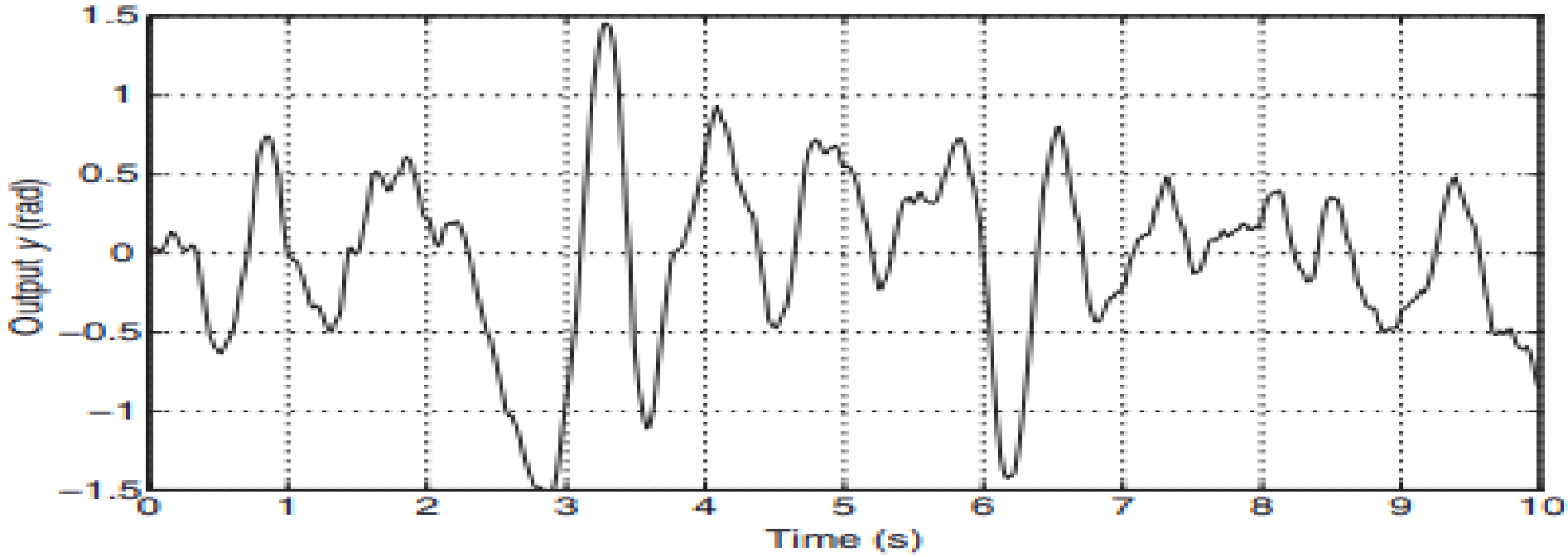


Figure 9.5. Output portion of training data.

Deline m input fuzzy sets characterized by triangular membership functions equally spaced on the universe $-\pi/2 \leq y \leq \pi/2$ rad and forming partitions of unity. By trying various numbers of inputs n and input fuzzy sets m, it is determined that a sufficient number of inputs to the fuzzy system is $n = 2$, with $m = 2$ fuzzy sets on each input universe. Therefore, the number of rules is $R = 4$, and there are 16 unknown parameters to be determined via batch least squares. This results in $\Phi - 996 \times 16$ and $Y - 996 \times 1$. The fuzzy sets on each input universe are shown in Figure 9.6.

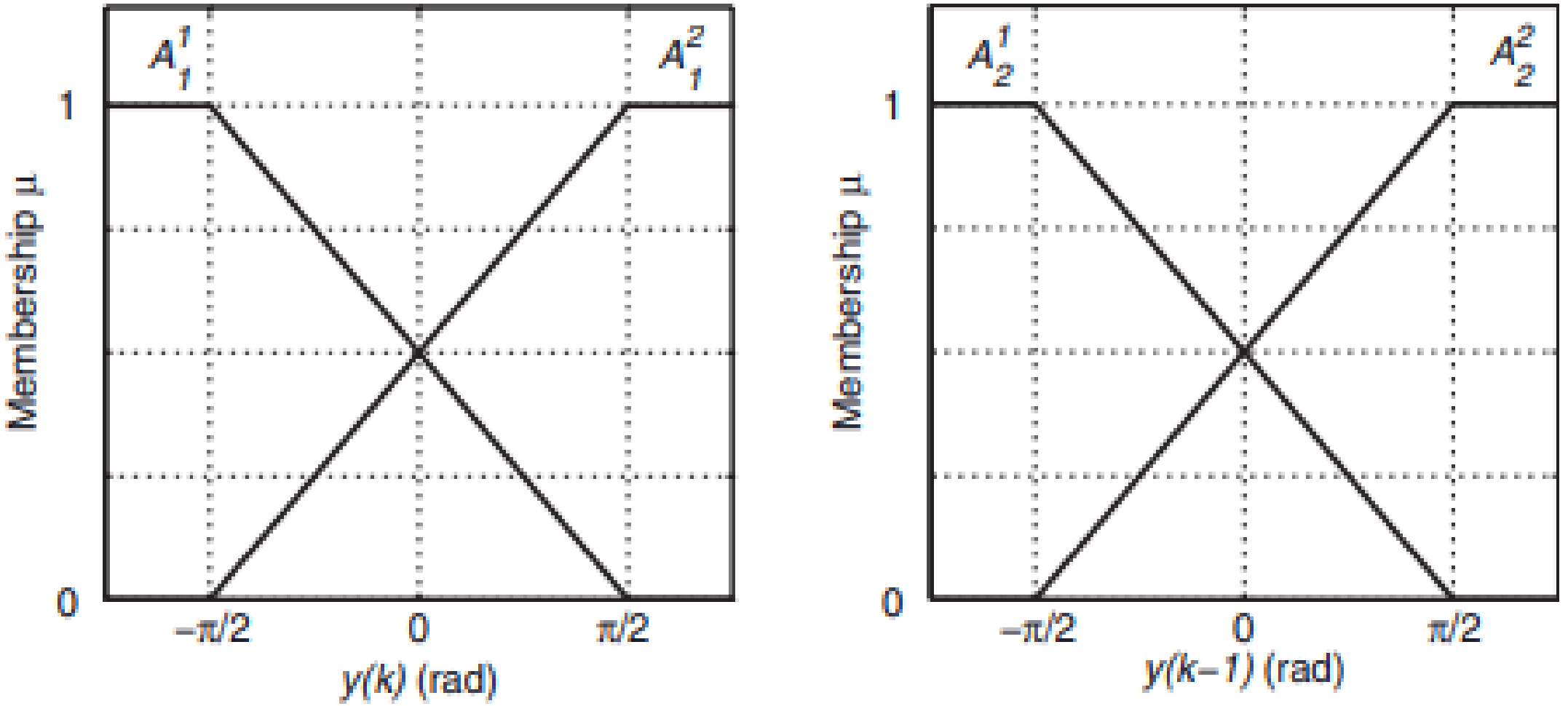


Figure 9.6. Input fuzzy sets for T–S fuzzy system approximating motor-driven robotic link.

The corresponding batch least-squares estimate of consequent parameters from (8.6) is

$$\theta^* = \begin{bmatrix} a_1^{1*} \\ a_1^{2*} \\ a_1^{3*} \\ a_1^{4*} \\ a_2^{1*} \\ a_2^{2*} \\ a_2^{3*} \\ a_2^{4*} \\ b_1^{1*} \\ b_1^{2*} \\ b_1^{3*} \\ b_1^{4*} \\ b_2^{1*} \\ b_2^{2*} \\ b_2^{3*} \\ b_2^{4*} \end{bmatrix} = \begin{bmatrix} 2.46 \\ 1.19 \\ 2.52 \\ 1.43 \\ -1.58 \\ -1.13 \\ -0.735 \\ -0.417 \\ 0.0494 \\ -0.601 \\ 0.929 \\ -0.0703 \\ -0.0314 \\ 1.41 \\ -1.35 \\ -0.0766 \end{bmatrix}$$

Thus the T–S fuzzy system approximating the motor-driven robotic link is given by the rule base:

1. If $y(k)$ is A_1^1 and $y(k-1)$ is A_2^1, then

$$\hat{y}^1(k+1) = 2.46y(k) - 1.58y(k-1) + 0.0494u(k) - 0.0314u(k-1)$$

2. If $y(k)$ is A_1^1 and $y(k-1)$ is A_2^2, then

$$\hat{y}^2(k+1) = 1.19y(k) - 1.13y(k-1) - 0.601u(k) + 1.41u(k-1)$$

3. If $y(k)$ is A_1^2 and $y(k-1)$ is A_2^1, then

$$\hat{y}^3(k+1) = 2.52y(k) - 0.735y(k-1) + 0.929u(k) - 1.35u(k-1)$$

4. If $y(k)$ is A_1^2 and $y(k-1)$ is A_2^2, then

$$\hat{y}^4(k+1) = 1.43y(k) - 0.417y(k-1) - 0.0703u(k) - 0.0766u(k-1)$$

where the fuzzy sets A_1^1, A_1^2, A_2^1, and A_2^2 are specified in Figure 9.6. The output of the fuzzy system is

$$\hat{y}(k+1)) = \frac{\sum_{j=1}^{4} \hat{y}^j \mu_j(y(k), y(k-1))}{\sum_{j=1}^{4} \mu_j(y(k), y(k-1))}$$

where the premise membership values μ_j, $j = 1, \ldots, 4$ are calculated as

$$\mu_1(y(k), y(k-1)) = \mu_1^1(y(k), y(k-1))\mu_2^1(y(k), y(k-1))$$
$$\mu_2(y(k), y(k-1)) = \mu_1^1(y(k), y(k-1))\mu_2^2(y(k), y(k-1))$$
$$\mu_3(y(k), y(k-1)) = \mu_1^2(y(k), y(k-1))\mu_2^1(y(k), y(k-1))$$
$$\mu_4(y(k), y(k-1)) = \mu_1^2(y(k), y(k-1))\mu_2^2(y(k), y(k-1))$$

and fuzzy set A_j^i is characterized by the membership function μ_j^i.

To test the approximating T–S fuzzy system (called *model validation*), its behavior is compared with that of the true robotic link for a variety of inputs. These inputs should be significantly different from the input used to obtain the training data. For instance, if the input current to both the link and the fuzzy system is $u(t) = 10^{-5}$ sign(sin(0.2πt), the output link angles of Figure 9.7 result. This shows good agreement between the actual link and the T–S fuzzy model. The agreement using a variety of other inputs is also good, therefore the T–S fuzzy system is deemed to approximate the motor-driven robotic link plant to sufficient accuracy.

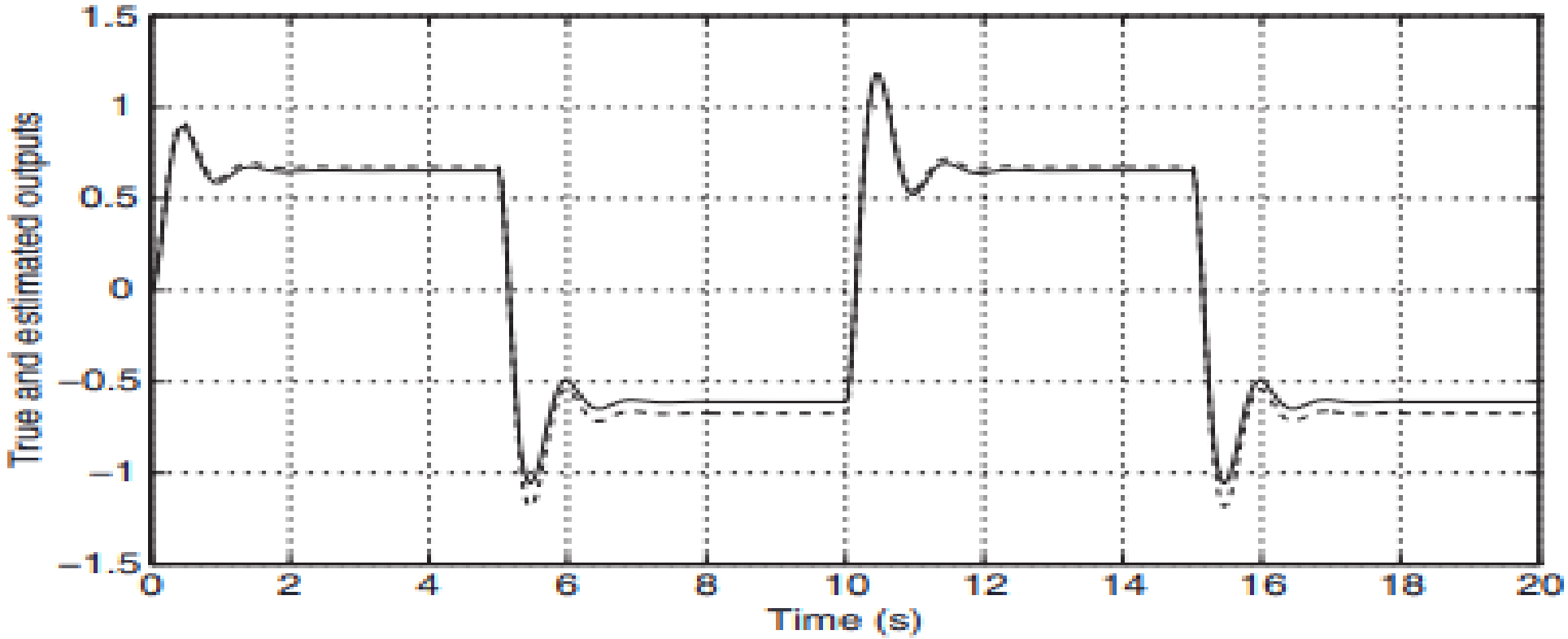


Figure 9.7. Comparison of outputs of true robotic link y (dashed) and approximating fuzzy system $\hat{y}$ (solid), square wave input.

Although it is not the subject of this discussion, we point out that now that we have an accurate T–S model of the link, it is possible to use the methods of 5.3.2 (tracking) and 5.3.3 (model reference) in parallel distributed controllers to control it, similarly to Examples 7.3 and 7.4.