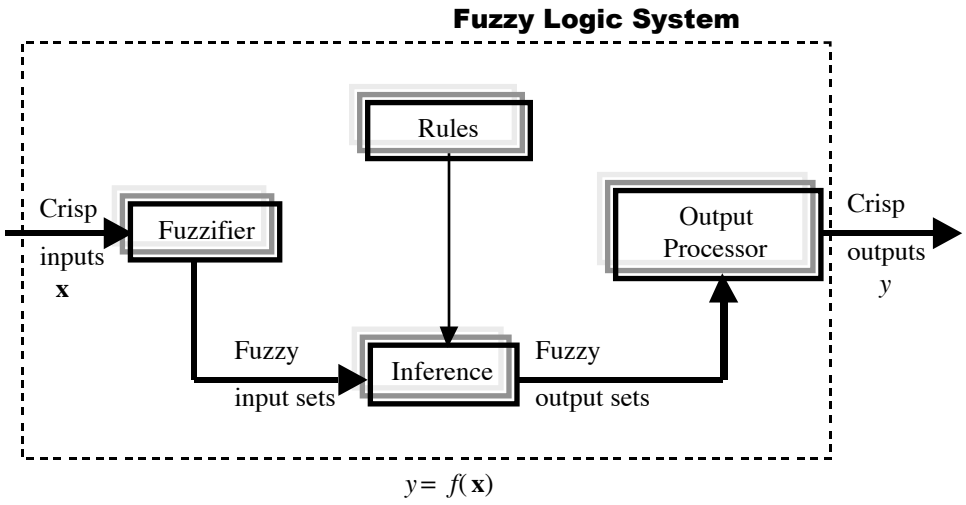
Introduction to Rule-Based Fuzzy Logic Systems

by Jerry M. Mendel

University of Southern California



CONTENTS

Introduction to Rule-Based Fuzzy Logic Systems

A Self-Study Course

This course was designed around Chapters 1, 2, 4–6, 13 and 14 of *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and new Directions* by Jerry M. Mendel, Prentice-Hall 2001. The goal of this self-study course is to provide training in the field of rule-based fuzzy logic systems.

In this course, which is the first of two self-study courses, the participant will focus on rule-based fuzzy logic systems when no uncertainties are present. This is analogous to first studying deterministic systems before studying random systems. In the follow-on self-study course *New Directions in Rule-Based Fuzzy Logic Systems: Handling Uncertainties*, the participant will learn about expanded and richer kinds of rule-based fuzzy logic systems, ones that can directly model uncertainties and minimize their effects. The present course (or equivalent knowledge) is a prerequisite to the follow-on course.

Prerequisites

This course is directed at participants who have had no formal training in fuzzy logic and want to learn about rule-based fuzzy logic systems. It assumes a college undergraduate degree, preferably in electrical engineering or computer science.

Course Objectives

After completing this course, you should be able to:

- Describe many differences between fuzzy sets and crisp sets, and fuzzy logic and crisp logic
- Describe numerous applications for rule-based fuzzy logic systems (FLSs)
- Demonstrate how a fuzzy set is described by a membership function
- Compute set theoretic operations for fuzzy sets using membership functions
- Demonstrate compositions of fuzzy relations and compute their membership functions
- Describe and use Zadeh's Extension Principle
- Explain the transition from crisp logic to fuzzy logic
- Demonstrate membership functions for rules
- Explain how rules are fired and implement the firing of rules
- Describe and demonstrate how a FLS can be used to forecast a time-series
- Describe and demonstrate how a FLS can be used as a fuzzy logic advisor for making social or engineering judgments
- Describe the architectures of three type-1 FLSs
- Compute the input-output relationships for these three FLSs
- Demonstrate and implement a variety of design methods for optimizing the design parameters of these three FLSs
- Describe the nature of and the order of all computations needed to design and implement these three FLSs
- Explain what software is available to implement and design these three FLSs

- Explain four kinds of uncertainties that can occur in a rule-based FLS
- Describe why a type-1 FLS cannot directly model and minimize the effects of the uncertainties

Course Components

This course includes:

- A study guide including learning objectives, reading assignments, and practice problems (with solutions)
- A final exam and its solution.
- The textbook is not included.

How to use this course

This course was developed assuming the reader would complete the lessons sequentially, i.e., Lesson 1 followed by Lesson 2, etc. Similarly, the tasks in each lesson should be completed sequentially in the following order:

1. Read the objectives of the lesson.
2. Read the assigned sections of the text and the *Study Guide* in the order indicated.
3. Review the key points of the chapter.
4. Solve the practice problems.
5. Review the practice problem solutions.
6. Review the objectives of the lesson and determine if they have been met. If so, proceed to the next lesson. If not, review 2 through 5 above until the objectives are met.

After finishing Lesson 15 take the final exam.

Acknowledgements

I would like to take this opportunity to thank Qilian Liang for his careful review of the Study Guide and Li-Xin Wang for contributing the write-up in Lesson 13 about fuzzy logic control.

Lesson 1: INTRODUCTION & OVERVIEW

Learning Objectives

The main purpose of this lesson is to provide some motivation for and a broad overview of the entire course. After completing this lesson you will be able to:

- Explain the difference between logic and fuzzy logic.
- Explain why FL is needed.
- Describe a brief history about the development of fuzzy logic (FL), including information about the founder of FL, Lotfi Zadeh.
- Describe the four components that make up a rule-based fuzzy logic system (FLS).
- List some applications for FLSs.
- Explain the difference between a FLS and a neural network.

Reading Assignment

I. What is Fuzzy Logic (FL)?

We answer this question by contrasting FL with logic.

According to the *Encyclopedia Britannica*, "Logic is the study of propositions and their use in argumentation." According to *Webster's Dictionary of the English Language*, "logic is the science of formal reasoning, using principles of valid inference," and " logic is the science whose chief end is to ascertain the principles on which all valid reasoning depends, and which may be applied to test the legitimacy of every conclusion that is drawn from premises." Although multi-valued logic exists, we are most familiar with two-valued (dual-valued) logic in which a proposition is either *true* or *false*. This kind of logic is also referred to as *crisp logic*.

Traditional (sometimes called *Western*) logic was first systematized by Aristotle thousands of years ago, in ancient Athens. There are two fundamental laws of classical logic:

> *Law of the Excluded Middle*: A set and its complement must comprise the universe of discourse.
>
> *Law of Contradiction*: An element can either be in its set or its complement; it cannot simultaneously be in both.

These two laws sound similar, but the Law of Contradiction forbids something being simultaneously true and not true, whereas the Law of the Excluded Middle forbids anything other than something being true or not true. Shakespeare's Hamlet exemplified the Law of Contradiction when he said "To be or not to be, that is the question."

Fuzzy logic (FL) is a type of logic that includes more than just true or false values. It is the logic that deals with situations where you can't give a clear yes/no (true/false) answer. In FL, propositions are represented with *degrees of truthfulness or falsehood*, i.e., FL uses a continuous range of truth values in the interval [0, 1] rather than just true or false values. In FL, both of the two fundamental laws of classical logic can be broken, i.e., it is possible for an element to simultaneously be in its set and its complement but to different degrees, the sums of which add up to unity. This will be made very clear in Lesson 3. So, Zadeh's Hamlet might have said "To be somewhat and not to be somewhat, that is the cunundrum." FL includes classical dual-valued logic as a special case.

II. Why is FL Needed?
The following quotes address this question:

"An argument, which is only convincing if it is precise loses all its force if the assumptions on which it is based are slightly changed, while an argument which is convincing but imprecise may well be stable under small perturbations of its underlying axioms." [J. Schwartz, "The pernicious influence of mathematics in science," in Nagel, Suppes, and Tarski, *Logic Methodology and Philosophy of Science*, Standford, 1962.]

"All traditional logic habitually assumes that precise symbols are being employed. It is therefore not applicable to this terrestrial life but only to an imagined celestial existence."
[B. Russell, "Vagueness," *Australasian J. Psychol. Philos.*, vol. 1, pp. 84–92, 1923.]

"As the complexity of a system increases, our ability to make precise and yet significant statements about its behavior diminishes until a threshold is reached beyond which precision and significance (or relevance) become almost mutually exclusive characteristics." [L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," Memorandum ERL-M 411, Berkeley, Oct. 1973.] This is called *The Principle of Incompatibility*.

"As we move into the information era, human knowledge becomes increasingly important. We need a theory to formulate human knowledge in a systematic manner and put it into engineering systems, together with other information like mathematical models and sensory measurements." [L.-X. Wang, *A Course in Fuzzy Systems and Control*," Prentice-Hall, Upper Saddle River, NJ, 1997]

Read the *IEEE Spectrum* (June 1995, pp. 32-35) profile of Zadeh that is a supplement to this lesson and appears at the end of the *Study Guide*.

III. An Impressionistic Brief History of FL (In literature, *impressionism* is a "mode of treatment in which scene, character, and emotion are depicted through the author's or character's impressions rather than by strict objective detail." [*New Webster's Dictionary of the English Language*, Delair Publ. Co., 1981])

Lotfi Zadeh is the founding father of FL. His first seminal paper on fuzzy sets appeared in 1965, although he began to formulate ideas about them at least four years earlier. Fuzzy sets met with great resistance in the West, perhaps because of the negative connotations associated with the word "fuzzy." Let's face it, "fuzzy" does not conjure up visions of scientific or mathematical rigor. For decades after 1965 some—albeit, a relatively small number of—people, along with Zadeh, developed the rigorous mathematical foundations of fuzzy sets and fuzzy logic. Interestingly enough, Chinese and Japanese researchers devoted a large effort to fuzzy sets and fuzzy logic. A popular hypothesis for this is that "fuzzy" fits in quite nicely with Eastern philosophies and religions (e.g., the complimentarity of Yin and Yang). But, until the early 1970's fuzzy logic was a theory looking for an application. Then, a major breakthrough occurred in 1975 when Mamdani and Assilian showed how to use rule-based FL to control a non-linear dynamical system. It was relatively easy to do this, and it was a fast way to design a control system. Although the design did not lend itself to the well-accepted, important, critical and rigorous examinations called for by control theory, it did demonstrate an important real application for FL. Other applications of rule-based FL began to appear, two very notable ones in Japan—control of the Sendai cities' subway system, and control of a water treatment system. Commercial products began to appear, e.g. fuzzy shower, fuzzy washing machine, fuzzy rice-cooker, and, in Japan, the word "fuzzy" took on the connotation of "intelligent," and in 1990 received an award. Western industries took notice—there was big money to be made—and the decade of the '90s rolled in, during which FL achieved a high degree of acceptability (there still is an on-going debate between subjective probabilists and fuzzy theorists about whether FL is the same as or is different from subjective probability). The IEEE established the *IEEE Transactions on Fuzzy Systems*, and established the IEEE Conference and Fuzzy Systems (FUZZ); there are many other journals devoted to fuzzy systems (e.g., *Fuzzy Sets and Systems*); and, there are many workshops and conferences devoted either exclusively to or that include sessions on fuzzy technologies. In 1995, the IEEE awarded Zadeh its highest honor, its Medal of Honor, which is comparable to the Nobel Prize. Fuzzy logic is now widely used in many industries and fields to solve practical problems, and is still a subject of intense research by academics all over the world. Although many applications have been found for FL, it is its application to rule-based systems that has most significantly demonstrated its importance as a powerful design methodology. Such rule-based fuzzy logic systems (FLSs) are what this course is all about.

If you are interested in a less impressionistic history of FL, then see, for example, the books by McNeill and Freilberger (1992), Wang (1997), or Kosko (1993a). One of the best histories of FL appears in the recent textbook by Yen and Langari (1999, pp. 3–18).

IV. Four Components That Make Up a Rule-Based Fuzzy Logic System (FLS)
Read pages 3–8 of the textbook.

FL has led to a new *architecture* for problem solving. This architecture processes its inputs non-linearly and is built upon a class of logical propositions—rules. Rules can be extracted from experts and can then be quantified using the mathematics of FL that you will learn in this course; doing this leads to the architecture of a FLS. Or, we can a priori assume the architecture of a FLS, using the mathematics of FL, and tune the parameters of the FLS to solve a problem. The latter approach is in the spirit of using a neural network (NN) to solve a problem, where the

architecture of the NN is assumed ahead of time and its parameters are tuned to solve a problem. The former approach is truly unique to FL. The two approaches can be combined, allowing an architecture to be developed that can be based on a combination of linguistic and numerical information. Both approaches have an important role to play in problem solving and are described in this course.

V. Applications for FLSs

FL and rule-based FLSs have been applied in many different fields and industries, far too numerous to catalog. Consumer applications include cameras, camcorders, washing machines, microwave ovens, vacuum cleaners and rice-cookers. There are many control system applications, including: the already-mentioned control of the Sendai subway system, control of a cement kiln, traffic junction control, gas cooling control, robot control, and autonomous orbital operations. The automobile industry has found a broad range of applications for FLSs, including automobile speed control, anti-skid braking system, transmission system, and fuel injector. FL is also used by financial investment companies in some security investment systems, and by businesses to help make complicated decisions. FLSs have also been proposed for digital communications (e.g., modulation classification, equalization). For more comprehensive discussions about the applications of FL and FLSs, see, e.g., McNeill and Freilberger's 1992 book, Kosko's 1997 book, Yen and Langari's 1999 book, Cox's 1995 book, or Lin and Lee's 1996 book. For a complete listing of these books, see the References at the end of this course's textbook.

VI. Difference Between a Rule-Based FLS and a Neural Network

A rule-based FLS is built upon IF-THEN propositions from logic, whereas a neural network is built upon simple biological models of a neuron. Just as today's NNs are a far cry from biological neurons, today's rule-based FLSs are a far cry from propositional logic.

Today, fuzzy and neural are being combined. A *fuzzy neural network* is a NN that uses FL in some way, e.g., the weights of the NN may be modeled as fuzzy sets. A *neural fuzzy system* is a FLS that uses NN concepts in some way, e.g., the parameters of the FLS may be tuned using a back-propagation method, or data may first be clustered using a NN after which the clusters play the roles of the antecedents in a FLS.

VII. Coverage

After this introductory first lesson, there are a series of four lessons that will provide you with the basic tools that are needed in order to mathematically describe a rule-based FLS. Three lessons—Lessons 2–4—are about fuzzy sets and relations and one lesson—Lesson 5—is about fuzzy logic. Lesson 6 then describes two applications that are treated in the rest of this course as *case studies*, namely *forecasting of time-series* and *knowledge mining using surveys*. We then turn to three specific architectures for FLSs. Four lessons—Lessons 7–10—cover many aspects of the very widely used *singleton type-1 FLS* (also known as a *Mamdani FLS*), ranging from analysis to design to applications. Lesson 11 then covers all aspects of a *non-singleton type-1 FLS*, also ranging from analysis to design to applications. The non-singleton FLS lets us model

the inputs to the FLS as fuzzy numbers, whereas the singleton FLS does not, and, because a non-singleton FLS is very similar to a singleton FLS, we spend only one lesson on it. Finally, Lesson 12 covers many aspects of a *type-1 TSK FLS*, again ranging from analysis to design to applications. The TSK FLS is very popular in control systems applications of FL and is also becoming popular in signal processing applications of a FLS. Lesson 13 lets you explore some applications of a type-1 FLS, namely: rule-based pattern classification, equalization of time-varying non-linear digital communication channels, and fuzzy logic control. Its main purpose is to let you see how one or more of the FLSs already studied can be used to solve some real-world problems. Lesson 14 focuses on *computation*, both for implementing a FLS during its *operation* and during the *design* of the FLS. It enumerates all computations for singleton and non-singleton type-1 Mamdani FLSs and a singleton type-1 TSK FLS, and, overviews on-line software that is available for these computations. Finally, Lesson 15 focuses on the shortcomings of type-1 FLSs and how they can be overcome.

Key Points

- Fuzzy logic is a type of logic that includes more than just true or false values; it uses a continuous range of truth values in the interval [0, 1].
- Fuzzy logic lets us combine linguistic knowledge and numerical data in a systematic way.
- Lotfi Zadeh is the founder of fuzzy logic.
- A rule-based fuzzy logic system is comprised of four elements: rules, fuzzifier, inference engine and output processor.
- A FLS is a new architecture for problem solving, one that processes its inputs nonlinearly and is built upon IF-THEN rules.
- FL and FLSs have been applied in many different fields and industries.
- Today, fuzzy and neural are being combined into fuzzy neural networks and neural fuzzy systems.

Questions

1. Consider an engineering project that you are working on or have recently worked on. What are some IF-THEN rules for that project?
2. What are the antecedents and consequent(s) for the just-stated rules?
3. Why do you think that fuzzy logic as a discipline has encountered so much resistance?

Note: No solutions are provided for these questions because each participant will have their own answers to them. Deeper answers to Question 3 than are given in Section III above, can be found in the references mentioned at the end of that section.

Lesson 2: FUZZY SETS–Part 1

Learning Objectives

This lesson is the first of a series of four that will provide you with the basic tools that are needed in order to mathematically describe a rule-based FLS. In all of these lessons we begin by describing concepts that should be familiar to you—crisp sets, crisp relations, and crisp logic—and show how they can be generalized to related fuzzy concepts—fuzzy sets, fuzzy relations, and fuzzy logic. We spend three lessons on fuzzy sets and relations and one lesson on fuzzy logic.

The main purpose of this lesson is to explain the transition from crisp to fuzzy sets, emphasizing the concepts of a membership function and linguistic variables. After completing this lesson you will be able to:

- Explain how a fuzzy set is a generalization of a crisp set.
- Demonstrate what a *membership function* is and how it differs for crisp and fuzzy sets.
- Explain and demonstrate what we mean by a *linguistic variable*.
- Explain some terminology about fuzzy sets.

Reading Assignment

Read pages 19–25 of the textbook.

Key Points

- A crisp set can be defined using a membership function (MF) that only has two values, 0 or 1.
- A fuzzy set is a generalization of a crisp set to MFs that have values in the closed interval [0, 1].
- A crisp set is a special case of a fuzzy set.
- Linguistic variables are variables whose values are not numbers but words or sentences in a natural or artificial language.
- Membership functions are associated with terms—linguistic variables—which appear in the antecedents of consequents of rules, or in phrases.
- Popular shapes for MFs are triangles, Gaussian, trapezoidal, piece-wise linear, and bell-shaped.
- There is no unique MF for a term; even when its shape is agreed upon, there are parameters for the shape that can be chosen in different ways. The freedom to make such choices provides fuzzy logic systems with design degrees of freedom.
- The terms *support of a fuzzy set*, *fuzzy singleton*, and *normal fuzzy set* let us communicate about fuzzy sets.

Practice Problems

Complete Exercise 1–2 (all six parts).

Lesson 3: FUZZY SETS–Part 2

Learning Objectives

This lesson is the second of a series of four that will provide you with the basic tools that are needed in order to mathematically describe a rule-based FLS. The main purposes of this lesson are to describe the transitions from set theoretic operations for crisp sets to those for fuzzy sets as well as the transitions from crisp relations and compositions on the *same* product space to those for fuzzy relations and compositions on the *same* product space, and to introduce the concept of a *hedge*. After completing this lesson you will be able to:

- Demonstrate how the basic crisp set theoretic operations of union, intersection and complement can be computed using membership functions.
- Explain basic set-theoretic properties for crisp sets (e.g., associativity, DeMorgan's Laws, Law of Excluded Middle, Law of Contradiction).
- Describe the generalizations of the set theoretic operations of union, intersection and complement to fuzzy sets, and how they can be computed using membership functions.
- Explain what *t-norms* and *t-conorms* are.
- Explain basic set-theoretic properties for fuzzy sets (e.g., associativity, DeMorgan's Laws, Law of Excluded Middle, Law of Contradiction).
- Demonstrate crisp relations and compositions on the same product space.
- Demonstrate fuzzy relations and compositions on the same product space and explain how these differ from their crisp counterparts.
- Explain the concept of a *hedge* and list some hedges and their MFs.

Reading Assignment

Read pages 26–36, 517–520, and 42–44 (in this order) of the textbook.

Key Points

- The basic crisp set theoretic operations of union, intersection and complement can be computed using crisp membership functions whose values are either 0 or 1. The maximum and minimum functions can be used for union and intersection, respectively.
- Operations on crisp sets satisfy many properties including associativity, DeMorgan's Laws, Law of Excluded Middle, Law of Contradiction, and these properties can be proved using Venn diagrams or membership functions.
- The basic fuzzy set theoretic operations of union, intersection and complement can be computed using fuzzy membership functions whose values are in the closed interval [0, 1]. The maximum and minimum functions can be used for fuzzy union and fuzzy intersection, respectively, but they are not the only operations that can be used.
- T-norms are operators that can be used for fuzzy intersection.

- T-conorms are operators that can be used for fuzzy union.
- When minimum t-norm and maximum t-conorm are used for fuzzy intersection and fuzzy union, respectively, then operations on fuzzy sets satisfy all set theoretic properties except for the Laws of Excluded Middle and Contradiction, and these properties can be proved using membership functions.
- When product t-norm and maximum t-conorm are used for fuzzy intersection and fuzzy union, respectively, then operations on fuzzy sets do not satisfy the Laws of Excluded Middle and Contradiction, and do not satisfy some other set theoretic properties; hence, this t-norm/t-conorm pair must be used with care.
- A crisp relation represents the presence or absence of association, interaction, or interconnectedness between the elements of two or more sets.
- A crisp relation can be described using either a relational matrix or a sagittal diagram.
- A crisp relation is a crisp set.
- The intersection and union of two crisp relations is called a *composition* of the crisp relations.
- Fuzzy relations represent a *degree* of presence or absence of association, interaction, or interconnectedness between the elements of two or more fuzzy sets.
- Binary fuzzy relations are fuzzy relations between two fuzzy sets.
- The intersection and union of two fuzzy relations are called *compositions* of the fuzzy relations. They can be computed using t-norm or t-conorm operators.
- A linguistic hedge is an operation that modifies the meaning of a term, i.e. of a fuzzy set.
- Hedges can be viewed as operators that act on a fuzzy set's MF to modify it.

Practice Problems

Complete Exercises 1-9, 1-11 and 1-19a.

Lesson 4: FUZZY SETS–Part 3

Learning Objectives

This lesson is the third of a series of four that will provide you with the basic tools that are needed in order to mathematically describe a rule-based FLS. The main purpose of this lesson is to describe the transition from *crisp* relations and compositions on *different* product spaces that share a common set to those for *fuzzy* relations and compositions on *differen*t product spaces that share a common set. Computing the MF for the composition of two fuzzy sets on different product spaces that share a common set is, as we will see in Lesson 7, the most important computation of a rule-based FLS. A second purpose of this lesson is to explain Zadeh's Extension Principle, which is widely used in many applications of FL. After completing this lesson you will be able to:

- Explain what is meant by "different product spaces."
- Demonstrate how to compute the MFs for compositions of *crisp* relations on different product spaces that share a common set using max-min and max-product composition formulas.
- Demonstrate how to compute the MFs for compositions of *fuzzy* relations on different product spaces that share a common set using the sup-star composition formula.
- Describe the Extension Principle and demonstrate how to use it in different situations.

Reading Assignment

Read pages 36–42 of the textbook.

The "sup" in the sup-star composition is short for *supremum*. If S is a set of real numbers bounded from above, then there is a smallest real number y such that $x \le y$ for all $x \in S$. The number y is called the *least upper bound* or *supremum* of S and is denoted $\sup_{x \in S}(x)$. We use the maximum for the supremum.

The *sup-star composition*, which is given in Equation (1-45), is the most important formula for a rule-based FLS; but, it is not proven in the text. Because of its importance, we provide a proof of it here. Since an understanding of the proof is not essential to the *use* of the sup-star composition, you may consider the proof as *optional* reading.

First, we define the composition of two fuzzy relations.

We have already learned that an element belongs to a fuzzy set if it has a non-zero membership in that set. In this respect, *the composition of two fuzzy relations* means:

> If $R(U,V)$ and $S(V,W)$ (R and S, for short) are two type-1 fuzzy relations on $U \times V$ and $V \times W$ respectively, then the composition of these two relations, denoted $R(U,V) \circ S(V,W) \equiv R \circ S(U,W)$, is defined as a subset $R \circ S(U,W)$ of $U \times W$ such that

(u,w) $R \circ S$ if and only if the membership for any pair (u,w), u U and w W, is non-zero [i.e., $\mu_{R\circ S}(u,w) \neq 0$] for at least one v V such that $\mu_R(u,v) \neq 0$ *and* $\mu_S(v,w) \neq 0$.

We shall show that this condition is equivalent to the sup-star composition

$$\mu_{R\circ S}(u,w) = \sup_{v\ V}\left[\mu_R(u,v) \bigstar \mu_S(v,w)\right]$$

A Side: In the proof given next, we use the following method. Let **A** be the statement "$\mu_{R\circ S}(u,w) \neq 0$," and **B** be the statement "there exists at least one v V such that $\mu_R(u,v) \neq 0$ *and* $\mu_S(v,w) \neq 0$." We prove that "**A** iff **B**" by first proving that $\overline{\mathbf{B}}$ $\overline{\mathbf{A}}$ (equivalent to proving that **A** **B**, i.e., necessity of **B**) and then proving that $\overline{\mathbf{A}}$ $\overline{\mathbf{B}}$ (equivalent to proving that **B** **A**, i.e., sufficiency of **B**).

Proof of (1-45): *Necessity*–If there exists no v V such that $\mu_R(u,v) \neq 0$ *and* $\mu_S(v,w) \neq 0$, then this means that for every v V, either $\mu_R(u,v)$ or $\mu_S(v,w)$ is equal to zero (or both are zero), which in turn implies that $\mu_R(u,v) \bigstar \mu_S(v,w) = 0$ for every v V, i.e. the supremum of $\mu_R(u,v) \bigstar \mu_S(v,w)$ over v V is zero. Hence, $\mu_{R\circ S}(u,w) = 0$, as it should be.

Sufficiency–If the sup-star composition is zero then it must be true that $\mu_R(u,v) \bigstar \mu_S(v,w) = 0$ for every v V, which means that for every v V, either $\mu_R(u,v)$ or $\mu_S(v,w)$ (or both) is zero. This means that there is no v V such that $\mu_R(u,v) \neq 0$ and $\mu_S(v,w) \neq 0$. ■

Read pages 44–47 of the textbook.

Using the Extension Principle in a Rule-Based FLS.
In engineering applications of rule-based FLSs, it can happen that functions of a measured variable are used as either the antecedents or consequent of a rule. Some examples are $x_1 = \ln x$ and $x_2 = \sin f$. If we are given the MFs for *x* and *f*, then we will need to determine the MFs for x_1 and x_2. This is done using the Extension Principle. An alternative, of course, is to think directly in terms of MFs for $\ln x$ and $\sin f$; but, doing this may be very unnatural, i.e. it is usually much more natural to converse in terms of measured quantities and not in terms of functions of those quantities.

Key Points

- The composition of two *crisp* relations on different product spaces that share a common set can be computed in different ways, including relational matrices and sagittal diagrams; but, using formulas to do this, such as the max-min or max-product compositions (or their shortcuts), are very efficient because they can be easily implemented on a digital computer.
- The composition of two *fuzzy* relations on different product spaces that share a common set is performed using the sup-star composition, where "star" denotes a t-norm operator.

- The most important application of the sup-star composition in a rule-based FLS is when one of the relations is a fuzzy set.
- The Extension Principle (EP) lets us extend mathematical relationships between non-fuzzy variables to fuzzy variables.
- When using the EP, we must be careful to distinguish between one-to-one and one-to-many mappings, and, single- and multiple-variable mappings, so as to use the proper version of it in each case.

Practice Problems

Complete Exercises 1–16 and 1–20 (b).

Lesson 5–FUZZY LOGIC

Learning Objectives

This lesson is the fourth of a series of four that will provide you with the basic tools that are needed in order to mathematically describe a rule-based FLS. The main purposes of this lesson are to review the elements of crisp logic, make the transition from crisp to fuzzy logic, obtain membership functions for rules, and to provide pictorial explanations of the firing of rules. After completing this lesson you will be able to:

- Explain that rules are a form of *propositions*, and describe what propositions are.
- Demonstrate the role of *truth tables* in crisp logic.
- Explain the major elements of crisp logic and demonstrate the truth table for five operations that are frequently applied to propositions.
- Explain the concept of a *tautology* and demonstrate how to use it to determine MFs for crisp implications (rules).
- Describe the firing of crisp rules using Modus Ponens and Modus Tollens.
- Explain the transition from crisp logic to fuzzy logic.
- Describe Generalized Modus Ponens and demonstrate how to implement it using a sup-star composition formula.
- Create insightful pictorial diagrams that show the steps of the Generalized Modus Ponen sup-star composition.
- Explain what “engineering implications” are and why they are needed.

Reading Assignment

Read pages 48–59 of the textbook.

Because of the importance of the sup-star composition (1-74), we now illustrate its computation when there is some uncertainty about the measurement of input variable x, in which case the measurement can be modeled as a fuzzy number. These results are used in Lesson 11.

Let the measured value of x be denoted x'. In our two examples below we create a fuzzy number centered about x' by using the following Gaussian membership function for A:

$$\mu_A(x) = \exp\left[-\tfrac{1}{2}\left(\frac{x - x'}{\sigma_A}\right)^2\right]$$

Here we consider a single-antecedent rule whose antecedent membership function is also assumed to be a Gaussian, namely

$$\mu_A(x) = \exp\left[-\tfrac{1}{2}\left(\frac{x - m_A}{\sigma_A}\right)^2\right]$$

Example 5–1: Calculation of the sup-star composition for Gaussian MFs and Product t-norm

In this example, we assume product implication and product t-norm.

(a) First, we show that the sup-star composition in (1-74) can be expressed as

$$\mu_{B^*}(y) = \sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\right] \times \mu_B(y)$$

Derivation: Using product implication, $\mu_{A \to B}(x, y) = \mu_A(x)\mu_B(y)$, and using product t-norm $\bigstar = \times$; hence,

$$\mu_{B^*}(y) = \sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\mu_B(y)\right] = \left(\sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\right]\right) \times \mu_B(y)$$

(b) Next, we show that $\sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\right]$ occurs at $x = x_{\max} = \left(\sigma_A^2 m_A + \sigma_A^2 x'\right) / \left(\sigma_A^2 + \sigma_A^2\right)$.

Derivation: Let $f(x) \equiv \mu_{A^*}(x)\mu_A(x)$, and substitute the exponential MFs stated above into it, to see that

$$f(x) = \exp\left[-\tfrac{1}{2}\left(\frac{x - x'}{\sigma_{A^*}}\right)^2 + \left(\frac{x - m_A}{\sigma_A}\right)^2\right] \equiv \exp\left\{-\tfrac{1}{2}\varphi(x)\right\}$$

To maximize $f(x)$ we must minimize $\varphi(x)$; hence, we proceed as follows:

$$\frac{\partial \varphi(x)}{\partial x} = 2\left(\frac{x - x'}{\sigma_{A^*}^2}\right) + 2\left(\frac{x - m_A}{\sigma_A^2}\right)$$

$$\frac{\partial \varphi(x)}{\partial x} = 0 \qquad x = x_{\max}$$

$$(x_{\max} - x')\sigma_A^2 + (x_{\max} - m_A)\sigma_{A^*}^2 = 0$$

$$x_{\max}(\sigma_A^2 + \sigma_{A^*}^2) = \sigma_{A^*}^2 m_A + \sigma_A^2 x'$$

$$x_{\max} = \frac{\sigma_{A^*}^2 m_A + \sigma_A^2 x'}{\sigma_A^2 + \sigma_{A^*}^2} \qquad \text{QED}$$

(c) Finally, we show that $\sup_{x\ X}\left[\mu_{A^*}(x)\mu_A(x)\right] = \exp\left\{-\frac{1}{2}\left(x' - m_A\right)^2 \Big/ \left(\sigma_{A^*}^2 + \sigma_A^2\right)\right\}$.

Derivation: Substitute $x = x_{\max}$ into $f(x) \equiv \mu_{A^*}(x)\mu_A(x)$ to obtain:

$$f(x_{\max}) = \sup_{x\ X}\left[\mu_{A^*}(x)\mu_A(x)\right] = \mu_{A^*}(x_{\max})\mu_A(x_{\max})$$

$$f(x_{\max}) = \exp\left\{-\tfrac{1}{2}\varphi(x_{\max})\right\} = \exp\left\{-\frac{1}{2}\left[\frac{x_{\max} - x'}{\sigma_{A^*}}\right]^2 + \left[\frac{x_{\max} - m_A}{\sigma_A}\right]^2\right\}$$

where

$$\frac{x_{\max} - x'}{\sigma_{A^*}} = \frac{\sigma_{A^*}^2 m_A + \sigma_A^2 x' - (\sigma_A^2 + \sigma_{A^*}^2)x'}{(\sigma_A^2 + \sigma_{A^*}^2)\sigma_{A^*}} = \frac{\sigma_{A^*}(m_A - x')}{(\sigma_A^2 + \sigma_{A^*}^2)}$$

and

$$\frac{x_{\max} - m_A}{\sigma_A} = \frac{\sigma_{A^*}^2 m_A + \sigma_A^2 x' - (\sigma_A^2 + \sigma_{A^*}^2)m_A}{(\sigma_A^2 + \sigma_{A^*}^2)\sigma_A} = \frac{\sigma_A(x' - m_A)}{(\sigma_A^2 + \sigma_{A^*}^2)}$$

So,

$$\varphi(x_{\max}) = \frac{\sigma_{A^*}^2(m_A - x')^2 + \sigma_A^2(x' - m_A)^2}{(\sigma_A^2 + \sigma_{A^*}^2)^2} = \frac{(x' - m_A)^2}{(\sigma_A^2 + \sigma_{A^*}^2)}$$

Hence,

$$f(x_{\max}) = \exp\left\{-\frac{1}{2}\frac{(x' - m_A)^2}{(\sigma_A^2 + \sigma_{A^*}^2)}\right\}$$

Example 5-2: Calculation of the sup-star composition for Gaussian MFs and Minimum t-norm
In this example, we assume minimum implication and minimum t-norm.

(a) First, we show that the sup-star composition in (1-74) can be expressed as

$$\mu_{B^*}(y) = \min\left\{\sup_{x\ X} \min\left[\mu_{A^*}(x), \mu_A(x)\right], \mu_B(y)\right\}$$

Derivation: Using minimum implication, $\mu_{A \to B}(x,y) = \min\left[\mu_A(x), \mu_B(y)\right]$, and using minimum t-norm ★ = minimum; hence,

$$\mu_{B^*}(y) = \sup_{x \in X}\left(\min\left[\mu_{A^*}(x), \min\left[\mu_A(x), \mu_B(y)\right]\right]\right)$$

$$\mu_{B^*}(y) = \sup_{x \in X} \min\left\{\min\left[\mu_{A^*}(x), \mu_A(x)\right], \min\left[\mu_{A^*}(x), \mu_B(y)\right]\right\}$$

$$\mu_{B^*}(y) = \min\left\{\sup_{x \in X} \min\left[\mu_{A^*}(x), \mu_A(x)\right], \sup_{x \in X} \min\left[\mu_{A^*}(x), \mu_B(y)\right]\right\}$$

$$\mu_{B^*}(y) = \min\left\{\sup_{x \in X} \min\left[\mu_{A^*}(x), \mu_A(x)\right], \min\left[\sup_{x \in X} \mu_{A^*}(x), \mu_B(y)\right]\right\}$$

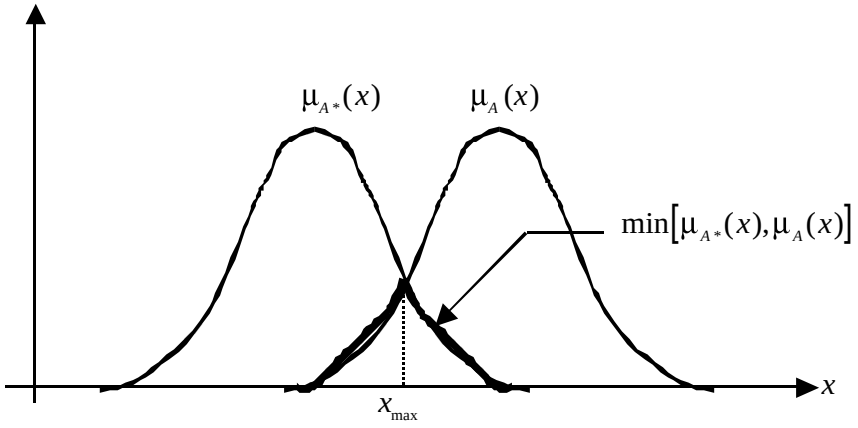
$$\mu_{B^*}(y) = \min\left\{\sup_{x \in X} \min\left[\mu_{A^*}(x), \mu_A(x)\right], \min\left[1, \mu_B(y)\right]\right\}$$

$$\mu_{B^*}(y) = \min\left\{\sup_{x \in X} \min\left[\mu_{A^*}(x), \mu_A(x)\right], \mu_B(y)\right\}$$

(b) Next, we show that $\sup_{x \in X} \min\left[\mu_A(x), \mu_A(x)\right]$ occurs at the intersection point of the two Gaussian membership functions, namely at

$$x = x_{\max} = \left(\sigma_A m_A + \sigma_A x'\right) / \left(\sigma_A + \sigma_A\right).$$

Derivation: We do this in the figure below, where it is clear that $x_{\max}$ occurs at the *intersection* of the two Gaussian membership functions.



In order to get a formula for $x_{\max}$, we set

$$\exp\left[-\tfrac{1}{2}\left(\frac{x - x'}{\sigma_{A^*}}\right)^2\right] = \exp\left[-\tfrac{1}{2}\left(\frac{x - m_A}{\sigma_A}\right)^2\right]$$

We must take into account the fact that, at the point where the two exponential functions cross each other, one is increasing and the other is decreasing; hence, $x_{\max}$ is the solution to

$$\frac{x_{\max} - x'}{\sigma_{A^*}} = -\left(\frac{x_{\max} - m_A}{\sigma_A}\right)$$

$$\sigma_A(x_{\max} - x') + \sigma_{A^*}(x_{\max} - m_A) = 0$$

$$x_{\max} = \frac{\sigma_A x' + \sigma_{A^*} m_A}{\sigma_{A^*} + \sigma_A}$$

(c) Finally, we show that $\sup_x \min\left[\mu_{A^*}(x), \mu_A(x)\right] = \exp\left[-\frac{1}{2}\left(\frac{x' - m_A}{\sigma_{A^*} + \sigma_A}\right)^2\right]$.

Derivation: From part (b), it is clear that

$$\sup_x \min\left[\mu_{A^*}(x), \mu_A(x)\right] = \mu_{A^*}(x_{\max}) = \mu_A(x_{\max})$$

So, for example,

$$\mu_A(x_{\max}) = \exp\left[-\frac{1}{2}\frac{1}{\sigma_A^2}\left\{\frac{\sigma_A x' + \sigma_{A^*} m_A}{\sigma_{A^*} + \sigma_A} - m_A\right\}^2\right] = \exp\left[-\frac{1}{2}\frac{1}{\sigma_A^2}\frac{\sigma_A x' + \sigma_{A^*} m_A - \sigma_{A^*} m_A - \sigma_A m_A}{(\sigma_{A^*} + \sigma_A)^2}\right]$$

$$\mu_A(x_{\max}) = \exp\left[-\frac{1}{2}\left(\frac{x' - m_A}{\sigma_{A^*} + \sigma_A}\right)^2\right]$$

Key Points

- A proposition is an ordinary statement involving terms that have been defined.
- Rules are a form of proposition.
- Propositions can be combined in many ways using conjunction, disjunction, implication, negation, and equivalence.
- The IF part of an implication is called the *antecedent*, whereas the THEN part is called the *consequent*.
- A *truth table* shows relationships between several propositions; the truth table for the five operations that are frequently applied to propositions is Table 1–2.
- A tautology is a proposition formed by combining other propositions; tautologies can be proven to be true or false using truth tables.

- For our work in rule-based FLSs, the following tautologies for an implication are most important because they let us establish MFs for the implication: $(p \rightarrow q) \leftrightarrow \sim[\, p \quad (\sim q)]$ and $(p \rightarrow q) \leftrightarrow (\sim p) \quad q$.
- Logic, set theory and Boolean Algebra are mathematically equivalent; any statement that is true is one system becomes a true statement in the other simply by making some changes in notation.
- Crisp rules are fired using inference mechanisms known as *Modus Ponens* and *Modus Tollens*; only Modus Pollens plays a role in a FLS.
- The transition from crisp logic to FL is done by replacing crisp logic's MFs by fuzzy MFs, and Modus Ponens by *Generalized Modus Ponens*.
- Generalized Modus Ponens is a fuzzy composition where the first fuzzy relation is a fuzzy set.
- The MF of a fired rule is given by the sup-star composition.
- Singleton fuzzification simplifies the computation of the sup-star composition by eliminating the need to perform the supremum operation.
- When all MFs are Gaussian then it is possible to compute the sup-star composition analytically for both product and minimum t-norms.
- Pictorial descriptions of the sup-star composition provide insight into its operations, and demonstrate a problem with using fuzzy versions of classical crisp implications, namely a bias in the MF of a fired rule.
- Mamdani implications—product and minimum—overcome the problem of a bias in the MF of a fired rule; but, their MFs are a departure from those of classical crisp implications.

Practice Problems

Complete Exercises 1–23 (c) and 1–26.

Lesson 6–CASE STUDIES

Learning Objectives

This lesson describes two applications that are treated in the rest of this course as case studies. These applications are *forecasting of time-series* and *knowledge mining using surveys*. After completing this lesson you will be able to:

- Describe how to formulate time-series forecasting problems.
- Explain the difference between *training* and *testing* sets of data.
- Demonstrate three ways to extract rules from numerical training data.
- Describe the Mackey-Glass chaotic time-series.
- Describe a six-step methodology for knowledge mining using surveys.
- Explain what a fuzzy logic advisor (FLA) is and demonstrate how it can be used for making social or engineering judgments.

Reading Assignment

Read pages 110–118 of the textbook.

Although we focus on the Mackey-Glass chaotic time-series in this course and in Chapters 5 and 6 of the textbook, it is by no means the only chaotic time series that has been used to demonstrate the forecasting capabilities of a FLS, e.g. the *Duffing equation* is considered by Mendel and Mouzouris in their 1997 paper.

Table 4–1 needs some additional explanation in relation to this course. Although it refers to six kinds of forecasters, in this course we will only cover three kinds: singleton type-1, non-singleton type-1, and TSK. The Mackey-Glass equation may be chaotic, but it is deterministic, i.e., even though it is very sensitive to its initial conditions (a property of a chaotic system), once they have been chosen, then each time we run a simulation of that equation we obtain exactly the same results. A singleton type-1 forecaster is useful when no uncertainties are present, i.e., there is no measurement noise so that the measurements that activate the forecaster are perfect, and, training and testing data are noise-free. A non-singleton type-1 forecaster tries to handle the situation when the data is corrupted by measurement noise, both during the design and operation of the forecaster. It does so by modeling the measurements as type-1 fuzzy numbers. Unfortunately, this leaves a lot to be desired; but, we can not do better within the framework of a type-1 FLS. To do better we must use a type-2 FLS, as described in the next course *New Directions in Rule-Based Fuzzy Logic Systems: Handling Uncertainties*. Finally, we use a totally different time series (a stream of compressed video) to illustrate the forecasting capabilities of a TSK forecast. That series is random but has no measurement noise associated with it either during its design or operation.

Read pages 119–126 of the textbook.

Sometimes a FLA is comprised of FL sub-advisors. Here we describe three architectures for such a FLA, assuming for illustrative purposes that there are three sub-advisors. The extension of these results to more than three sub-advisors is straightforward.

There are many different ways to combine/use three FL sub-advisors. First, however, we explain why one would construct sub-advisors. To ask people questions that use more than two antecedents is very difficult, because people usually can not correlate more than two things at a time. So, if more than two indicators are present for a social or engineering judgment, we can rank order them in importance (if this ordering is known ahead of time—or it may have to be established) and then use one or two of the indicators at a time to create the sub-advisors, after which results from the sub-advisors are combined to give the overall output of the FLA.

1. Parallel Architecture: Overall Decision Maker

In the figure for this architecture–Figure 1 below—we have partitioned the indicators in **x** into three subsets, each of which is the input to its own FLA. I assume that the dimensions of each of these subsets is one or two (if there are more than 6 indicators, then more sub-advisors will be needed). Simple one- or two-antecedent questions can be created in order to construct the sub-advisors. The output of each sub-advisor is for the same social judgment or engineering judgment, and the three outputs are aggregated in the *Combiner* block. Examples of a *Combiner* are: $y(\mathbf{x}) = \max[y_1(\mathbf{x}_1), y_2(\mathbf{x}_2), y_3(\mathbf{x}_3)]$ and $y(\mathbf{x}) = [y_1(\mathbf{x}_1) + y_2(\mathbf{x}_2) + y_3(\mathbf{x}_3)]/3$. Note that a final decision is only made at the output of the *Combiner* and not at the outputs of the sub-advisors.

This architecture would be used in Fig. 4-3 for both the Consensus and Individual's FLAs. It would be used in Fig. 4-6 for the Consensus FLA.

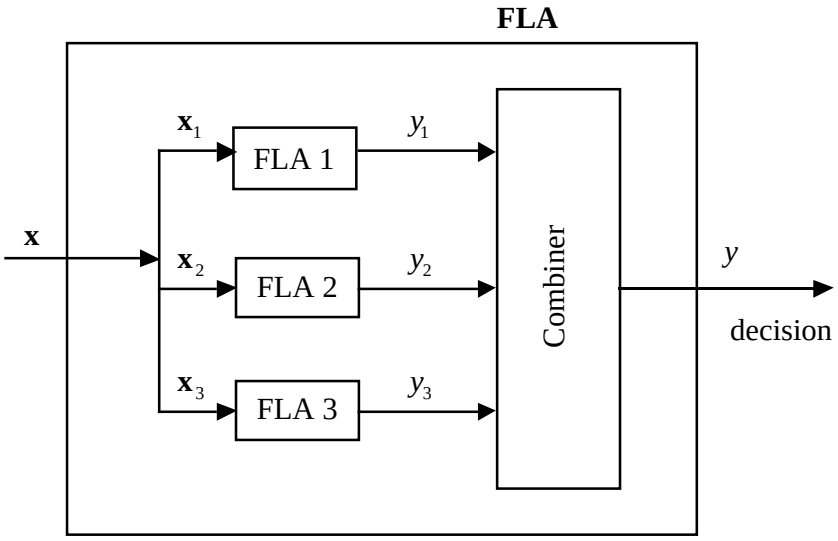


Figure 1: Parallel architecture: overall decision maker.

2. Parallel Architecture: Aggregate Decision Maker

In the figure for this architecture—Figure 2—we have again partitioned the indicators in **x** into three subsets, each of which is the input to its own FLA. Again, I assume that the dimensions of each of these subsets is one or two (if there are more than 6 indicators, then more sub-advisors will be needed). Simple one- or two-antecedent questions can be created in order to construct the sub-advisors.

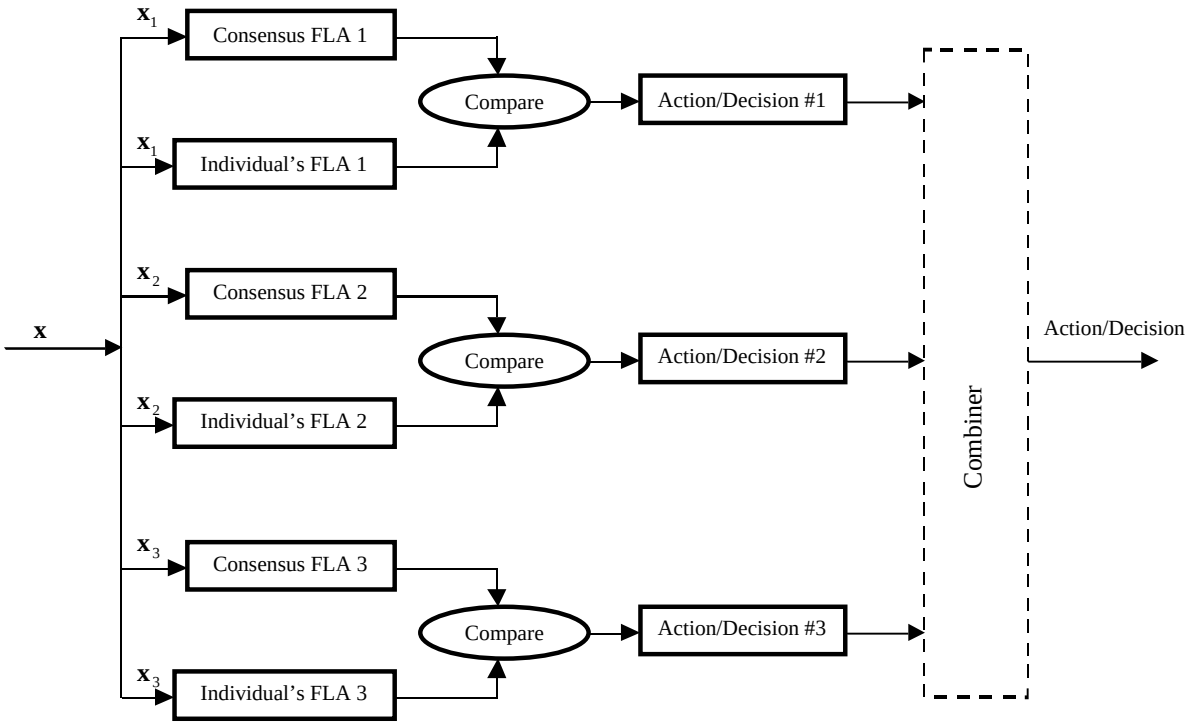


Figure 2: Parallel architecture: aggregate decision maker.

The architecture of the overall FLA is different than the architecture shown in Fig. 4-3. Now actions or decisions are made at the output of each sub-advisor and it is those actions or decisions that are passed on to the *Combiner*. The *Combiner* could use a majority-rules strategy, or some other strategy.

I have shown the block for the *Combiner* dashed because instead of combining actions and decisions it may be important to examine the actions/decisions at the output of each sub-advisor. For social judgments, an individual could be sensitized at the sub-advisor level with the hope that in so doing he or she would become sensitized at the aggregate level.

3. Hierarchical Architecture

In the figure for this architecture—Figure 3—we have again partitioned the indicators in **x** into three subsets. FLA 1 has antecedents that depend on the indicators in $\mathbf{x}_1$. The output of that sub-

advisor, $y_1(\mathbf{x}_1)$, acts as one of the indicators of FLA 2. The output of that sub-advisor, $y_2(\mathbf{x}_2, y_1(\mathbf{x}_1))$, then acts as one of the indicators of FLA 3. The output of FLA 3 is considered to be the overall output of the FLA, namely

$$y(\mathbf{x}) = y_3[\mathbf{x}_3, y_2] = y_3[\mathbf{x}_3, y_2(\mathbf{x}_2, y_1(\mathbf{x}_1))]$$

The output of each sub-advisor can be the same social judgment, but conditioned on different antecedents. The questions for FLA 1 are the standard ones. Those for FLAs 2 and 3 are not. For example a question for FLA 2 would have to be structured like:

IF judgement y made on the basis of indicators $\mathbf{x}_1$ is ____
and indicator x_{21} is _____ and indicator x_{22} is _____
THEN judgment y is _____

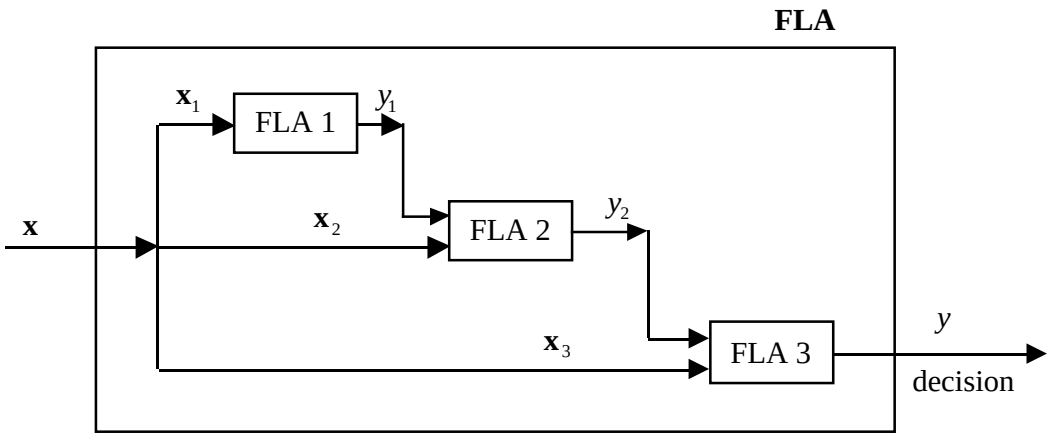


Figure 3: Hierarchical architecture.

We immediately see a potential problem for this architecture, namely if a sub-advisor indicator vector has two elements, then the questions associated with that sub-advisor will have three antecedents. Such three-antecedent questions are very difficult for people to answer. So, the overall indicator vector must be partitioned more finely so that each sub-advisor has at most two-antecedents. This can lead to an architecture that has a lot of sub-advisors.

For engineering judgments, when rules are extracted from data, it is possible to use the hierarchical architecture without having to worry about the dimension of the antecedents, since questions will not be asked of people.

Key Points

- To design a FLS forecaster data is partitioned into *training* and *testing* subsets. The number of elements in each subset depends on the size of the window of data points that is used to forecast the next data point.
- The training data are used in a FLS forecaster to establish its rules.
- One way to extract rules from numerical training data is: Let the data establish the fuzzy sets that appear in the antecedents and consequents of the rules.
- Another way to extract rules from numerical training data is: Pre-specify fuzzy sets for the antecedents and consequents and then associate the data with those fuzzy sets.
- A third way to extract rules from numerical training data is: Establish the architecture of a FLS and use the data to optimize its parameters.
- Chaotic behavior can be described as bounded fluctuations of the output of a non-linear system with high degree of sensitivity to initial conditions.
- The Mackey-Glass equation is a non-linear delay differential equation that is known to exhibit chaos when its delay parameter is greater than 17.
- Knowledge mining, as used in this course, means extracting information in the form of IF–THEN rules from people.
- *Judgment* means an assessment of the *level* of a variable of interest.
- A six step methodology for knowledge mining involves: identifying the behavior of interest, determining the indicators of the behavior of interest, establishing scales for each indicator and the behavior of interest, establishing names and interval information for each of the indicator's fuzzy sets and behavior of interest's fuzzy sets, establishing rules, and, surveying people (experts) to provide a consequent for each rule.
- Rules that are extracted from people about a judgment can be modeled using a FLS called a fuzzy logic advisor (FLA).
- FLAs can be used in different ways for social judgments or engineering judgments, e.g. they can be used to sensitize people about social judgments.
- FLAs can be comprised of sub-advisors that can be organized in a variety of architectures; this is useful so that people can be asked questions with at most one or two antecedents.

Practice Problem

Participate in the survey given in Table 4-2 (see, also, the discussion given on p. 77) by: (1) providing your start and end points for the five range labels, and (2) re-computing the mean and standard deviation values for the start and end points for each label using those shown in the table (obtained from 47 students) and your new values.

Lesson 7–SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–Part1

Learning Objectives

This lesson is the first in a series of four that cover many aspects of a very widely used FLS—a *singleton type-1 FLS* (also known as a *Mamdani FLS*)—ranging from analysis to design to applications. The main purpose of this lesson is to explain how to quantify the input–output operations of the inference mechanism in a FLS, and how this quantification is made simple when measurements are treated as perfect–the singleton case. After completing this lesson you will be able to:

- Describe the architecture of a type-1 singleton FLS.
- Demonstrate the broad range of IF-THEN rules that can be included within the framework of a rule-based FLS.
- Derive the MF that appears at the output of the inference engine for a single fired rule, using the sup-star composition.
- Demonstrate different ways for combining rule output MFs for multiple fired rules.
- Describe what is meant by *singleton fuzzification*.
- Demonstrate the tremendous simplification of the sup-star composition in the case of singleton fuzzification.
- Demonstrate pictorial descriptions of the firing of rules and the combining of multiple-fired rules that provide a lot of insight into the operation of a FLS.

Reading Assignment

Read pages 131–142 of the textbook.

Key Points

- A singleton type-1 FLS consists of rules, fuzzifier, inference mechanism and defuzzifier.
- A multiple-antecedent multiple-consequent rule can always be considered as a group of multi-input single-output rules
- Many *non-obvious* rules can be cast into the form of a *standard* IF-THEN rule, so that a rule-based FLS is quite broad in it applicability.
- The MF of a fired rule, $\mu_{B'}(y)$, is given by $\mu_{B'}(y) = \sup_{\mathbf{x}\ \mathbf{x}} \left[\mu_{A_{\mathbf{x}}}(\mathbf{x}) \star \mu_{A' \to G'}(\mathbf{x}, y)\right], \quad y \quad Y.$
- The fuzzy inference engine can be interpreted as a *system* whose output is $\mu_{B'}(y)$.
- Fired rules can be combined in different ways; there is no one best way to do this.
- A singleton fuzzifier has a MF that is non-zero at only one point, $x_i = x_i'$.
- For singleton fuzzification the supremum operation in the sup-star composition is very easy to evaluate because the MF of the input is non-zero only at one point, $x_i = x_i'$.

- For singleton fuzzification, the MF of a fired rule, $\mu_{B'}(y)$, is given by

$$\mu_{B'}(y) = \mu_{G'}(y) \star \left[\mu_{F_1'}(x_1') \star \cdots \star \mu_{F_p'}(x_p') \right], \quad y \quad Y$$

- Pictorial descriptions of input and antecedent operations, consequent operations, and combined output fuzzy sets provide lots of insight into the operations of the fuzzy inference mechanism.

Practice Problem

Example 5-1 is one of the most important ones given in Chapter 5, because it provides a geometric interpretation for the operations that occur within the inference engine. In this exercise, I want you to provide the figures that are comparable to the ones given in Figures 5-4–5-6, but using *triangular* MFs. Do this for both the minimum and product t-norms.

Lesson 8–SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–Part 2

Learning Objectives

This lesson is the second in a series of four that cover many aspects of a very widely used FLS—a *singleton type-1 FLS*—ranging from analysis to design to applications. The main purposes of this lesson are to complete the mathematical description of a singleton type-1 FLS, examine all of the design choices that have to be made, and explain why a FLS will work well. After completing this lesson you will be able to:

- Describe five popular methods for defuzzification: centroid, center-of-sums, height, modified height, and center-of-sets.
- Explain why there is no *one* singleton type-1 FLS, and demonstrate that the many choices that need to be made to specify or design such a FLS lead to a rich variety of FLSs.
- Demonstrate the input–output formula for a singleton type-1 FLS as a new kind of basis function expansion—a fuzzy basis function (FBF) expansion.
- Demonstrate that each rule, whether it derives from expert linguistic knowledge or is extracted from numerical data, can be associated with one FBF.
- Explain what a universal approximation theorem is, and describe a singleton type-1 FLS as a universal approximator.
- Demonstrate what is meant by *rule explosion*.

Reading Assignment

Read pages 142–148 of the textbook.

Section 5.5.2: Here we derive (5-16) by beginning with the additive combiner depicted in Figure 5-3, assuming product implication and product t-norm, and formally determining the center of gravity of its output $\mu_B(y) = \sum_{l=1}^{M} w_l \mu_{B^l}(y)$.

Derivation: From the last line of (5-10) we know that the MF for the additive combiner can be expressed as:

$$\mu_B(y) = \sum_{l=1}^{M} w_l \mu_{B^l}(y) = \sum_{l=1}^{M} w_l \mu_{G^l}(y) \prod_{i=1}^{p} \mu_{F_i^l}(x_i) = \sum_{l=1}^{M} w_l f^l \mu_{G^l}(y)$$

where $f^l = \prod_{i=1}^{p} \mu_{F_i^l}(x_i)$. The centroid of $\mu_B(y)$ is computed as follows:

$$\text{Centroid of } \mu_B(y) = \frac{\int_{y\ Y} y\mu_B(y)dy}{\int_{y\ Y} \mu_B(y)dy} = \frac{\int_{y\ Y} y\sum_{l=1}^{M} w_l f^l \mu_{G^l}(y)dy}{\int_{y\ Y} \sum_{l=1}^{M} w_l f^l \mu_{G^l}(y)dy}$$

$$\text{Centroid of } \mu_B(y) = \frac{\sum_{l=1}^{M} w_l f^l \int_{y\ Y} y\mu_{G^l}(y)dy}{\sum_{l=1}^{M} w_l f^l \int_{y\ Y} \mu_{G^l}(y)dy} \times \frac{\int_{y\ Y} \mu_{G^l}(y)dy}{\int_{y\ Y} \mu_{G^l}(y)dy}$$

$$\text{Centroid of } \mu_B(y) = \frac{\sum_{l=1}^{M} w_l f^l c_{G^l} a_{G^l}}{\sum_{l=1}^{M} w_l f^l a_{G^l}}$$

where c_{G^l} is the centroid of the *l*th consequent set G^l, i.e.

$$c_{G^l} = \int_{y\ Y} y\mu_{G^l}(y)dy \Big/ \int_{y\ Y} \mu_{G^l}(y)dy = \int_{y\ Y} y\mu_{G^l}(y)dy \Big/ a_{G^l}$$

and a_{G^l} is the area of that set. This completes the derivation. ■

It is interesting to see if this result also holds for product implication but minimum t-norm [between $\mu_{G^l}(y)$ and the firing level in (5-10)]. In this case, (5-10) becomes

$$\mu_{B^l}(y) = \min\left[\mu_{G^l}(y), f^l\right]$$

where f^l is defined above. Clearly, the previous derivation of the Centroid of $\mu_B(y)$ depends on the separability of $\mu_{G^l}(y)$ and f^l in the equation for $\mu_{B^l}(y)$, something that can not be guaranteed when $\mu_{B^l}(y) = \min\left[\mu_{G^l}(y), f^l\right]$; hence, Kosko's SAM is of very limited value.

Note that the center-of-sums defuzzifier is still applicable in this case, because (5-14) is in terms of the centroid and area of *output* fuzzy sets and not *consequent* fuzzy sets. These quantities can be computed numerically from knowledge of $\mu_{B^l}(y)$, as calculated from (5-10). ■

Read pages 149–157.

Key Points

- Defuzzification produces a crisp output from the fuzzy sets that appear at the output of a FLS's inference block.
- There are many kinds of defuzzifiers.
- The defuzzifiers that are based on some sort of *center of gravity computation* are: centroid, center-of-sums, height, modified height, and center-of-sets.

- Many choices need to be made in order to specify or design a type-1 FLS; they provide the designer with many design degrees of freedom.
- A FLS can be interpreted as a fuzzy basis function (FBF) expansion, which places a FLS into the more global perspective of function approximation.
- FBFs are not radial basis functions and they are not orthogonal basis functions.
- Every rule in a FLS, whether it comes from linguistic knowledge or is extracted from data, can be associated with a FBF.
- A FLS is a universal approximator, i.e., it can uniformly approximate any real continuous non-linear function to arbitrary degree of accuracy.
- Universal approximation is an existence theorem that helps to explain why a FLS is so successful in engineering applications, but is does not tell us how to specify a FLS.
- Rule explosion refers to rapid growth in the maximum number of rules that may be required in a FLS, e.g. if there are p input variables, each of which is divided into r overlapping regions, then a complete FLS must contain p^r rules.

Practice Problems

Complete Exercises 5–4 and 5–6.

Lesson 9–SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–Part 3

Learning Objectives

This lesson is the third in a series of four that cover many aspects of a very widely used FLS—a *singleton type-1 FLS*—ranging from analysis to design to applications. The main purpose of this lesson is to learn how to *design* singleton type-1 FLSs when a collection of training data is available. By "design" we mean specify or optimize the parameters that characterize the FLS. After completing this lesson you will be able to:

- Describe how training data can be interpreted as a collection of IF-THEN rules.
- Enumerate how many design parameters there can be in a specific FLS design, and the relation of that number to the number of possible rules in the FLS.
- Describe three high-level designs that can be associated with singleton type-1 FLSs.
- Explain a singleton type-1 FLS as a three-layered architecture.
- Demonstrate some design methods that can be used for the three kinds of designs, namely: one-pass methods, least-squares method, and a back-propagation (steepest descent) method,
- Demonstrate how to compute the derivatives that are needed for the back-propagation method.

Reading Assignment

Read pages 157–166 of the textbook. Omit Sections 5.9.4 and 5.9.5.

The following material supplements Section 5.9.3.

I. Interpretation of a Type-1 FLS as a Three-Layered Architecture

A singleton (or non-singleton) type-1 FLS can be viewed as a three-layered architecture. This was first discovered by my former Ph. D. student Li-Xin Wang, around 1990, as part of his Ph. D. research. This architecture suggests the possibility of back-propagating errors from the output of the FLS to earlier layers, in analogy with back-propagation in a feed-forward neural network (NN) (see discussions about this on the top of p. 166 in the textbook). It is important to note, though, that the three-layer architecture for the FLS is merely a re-interpretation of the FLS and is not a physical architecture—implementation. This is different from the layered architecture of a NN, where that architecture is usually viewed as a physical implementation of the network.

Starting with (5-24) and (5-25), we re-express $y(\mathbf{x})$ as follows:

$$y(\mathbf{x}) = f_s(\mathbf{x}) \equiv h/g$$

where

$$h = \sum_{l=1}^{M} \bar{y}^l w^l \text{ and } g = \sum_{l=1}^{M} w^l$$

in which

$$w^l = \prod_{i=1}^{p} \mu_{F_i^l}(x_i) \quad l = 1,...,M$$

These equations lead to the following three-layered architecture for this FLS:

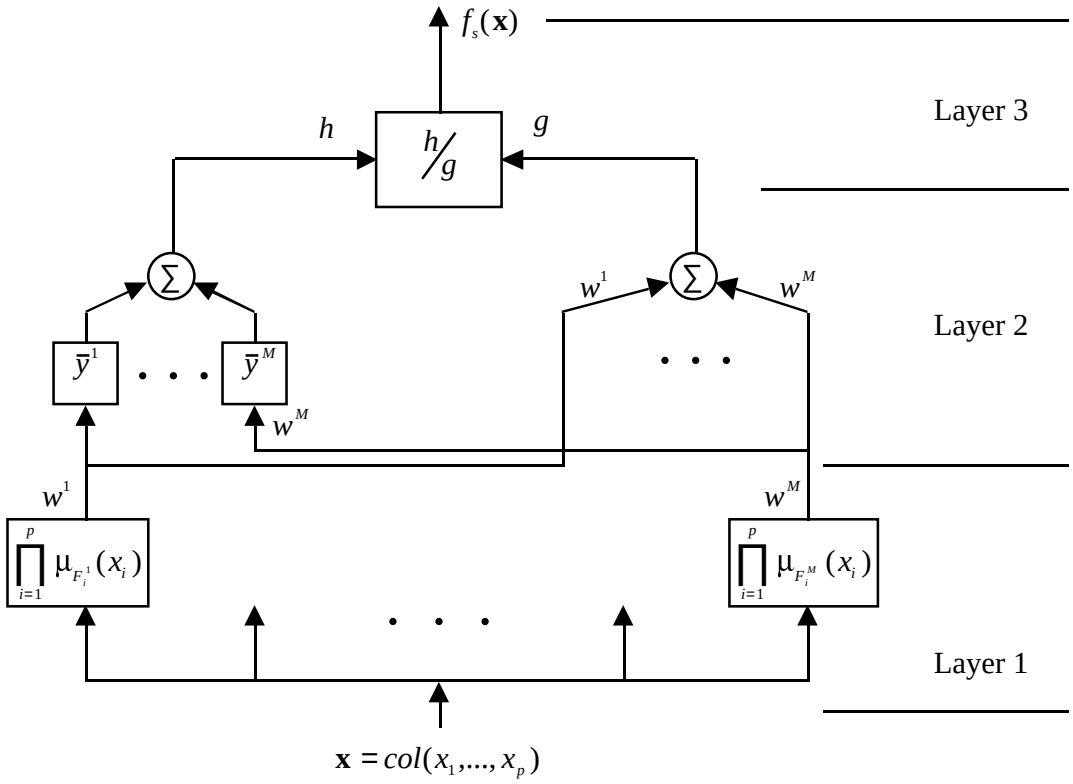


Figure 1: Three-layer architectural interpretation for a type-1 FLS.

■

II. A Very Short Primer on Optimizing a Function Using an Algorithm That Makes Use of First Derivative Information

There are many ways to optimize (i.e., minimize or maximize) a function. Here I will briefly describe a very popular way that uses not only the value of the function being optimized but also its first derivative. Methods that use this information are called *steepest descent algorithms*.

In order to keep the initial discussion as simple as possible, I shall assume that the function being minimized depends only on a single parameter, θ. That function (called an *objective function*) is

denoted $J(\theta)$, and an example of it is depicted in Figure 2. Observe that there are various kinds of *extrema* that can occur—relative maxima, relative minima, global maximum, global minimum, and even inflection points. When our goal is to minimize $J(\theta)$, then we want to determine the value of θ labeled in Figure 2 as θ^*. One of the great challenges to doing this is not to get trapped at a local extremum, e.g., at θ_1^* or θ_2^*. The importance of a good starting value for θ can not be over-stated. If, for example, our initial choice is at θ_0, then it is very likely that an optimization algorithm that is based on derivative information will cause θ to lock-on (converge) to θ_1^* or θ_2^*. On the other hand, if the initial choice is at θ_0', then it is very likely that an optimization algorithm that is based on derivative information will cause θ to converge to the global minimum at θ^*.

One approach to trying to achieve the global minimum is to randomly choose θ_0, solve for the associated minimum of $J(\theta)$, say $J(\theta')$, and to repeat this procedure for a collection of such θ_0 values. One then chooses θ^* as that value of θ' associated with the smallest value of $J(\theta')$. In many practical optimization problems, it may not be essential to compute the overall global minimum of $J(\theta)$. A value of θ that leads to a "small enough" value of $J(\theta)$ may suffice.

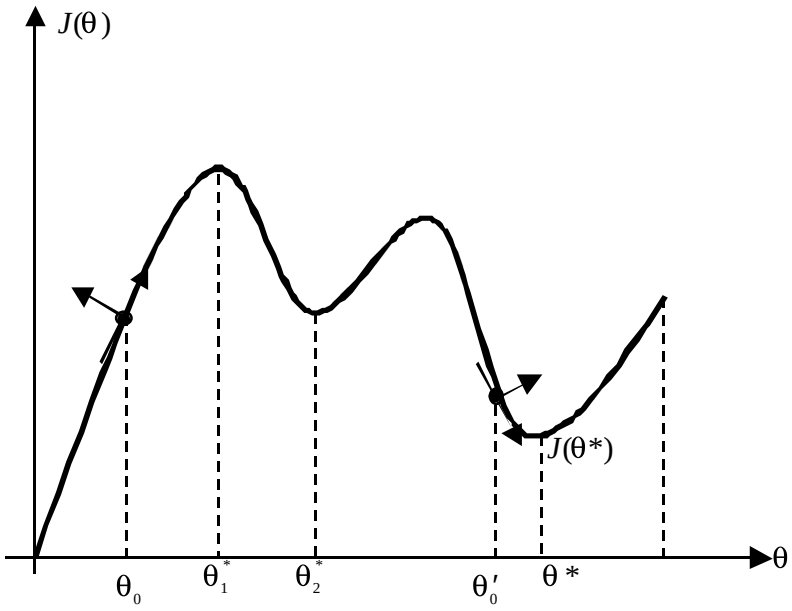


Figure 2: An objective function, $J(\theta)$, that has multiple extrema.

What really makes finding the minimum of $J(\theta)$ very challenging in the design of a FLS is that, even though we have a mathematical formula for $J(\theta)$, we do not know the shape of $J(\theta)$ ahead of time. We have available only a set of training data $[(\mathbf{x}^{(j)}: y^{(j)}), j = 1,2,...,N]$ (see page 158 of the textbook) that contain—we hope—good knowledge about $J(\theta)$.

We do not use all of the data to minimize $J(\theta)$. Instead, we partition the data into two sets, i.e.,

$$\{data\} = \{\{training\ data\},\{testing\ data\}\} \equiv \{D_{TRAIN}, D_{TEST}\} \tag{1}$$

Note that in the textbook I refer to all of training data as the "data," which is then partitioned into a training data *subset* and a testing data *subset*. The idea is to use the training data subset to minimize $J(\theta)$ the best you can, but to then evaluate how well you do this by using the testing data subset. There will be a trade off between over-fitting using the training data subset and generalization using the testing data subset. Usually, over-fitting leads to poor generalization performance.

My goal in the next few paragraphs is to give you a fairly high-level explanation of the construction of a steepest descent algorithm for minimizing objective function $J(\theta)$, where in the discussions below θ now is a vector of design parameters. In order to emphasize the role of the data during the optimization process, as used by the optimization algorithm, I shall denote $J(\theta)$ as $J = J(D,\theta)$. D_{TRAIN} is used by the steepest descent algorithm because that algorithm is based on minimizing $J_{TRAIN} = J(D_{TRAIN},\theta)$. D_{TEST} is used to evaluate the overall optimization results by computing $J_{TEST} = J(D_{TEST},\theta)$ and establishing an overall stopping rule, of the form:

$$\left|J(D_{TEST},\theta_{i+1}) - J(D_{TEST},\theta_i)\right| \le \varepsilon \tag{2}$$

where ε is pre-specified. This is only one example of a stopping rule, but it is one that is frequently used in practice. Another practical stopping rule is to choose a pre-specified maximum number of iterations, and to stop the iterative minimization when that number is reached. This stopping rule is not as effective as the first one because $J_{TEST} = J(D_{TEST},\theta)$ could still be changing by a large amount after the pre-specified number of iterations has been reached.

The general structure of a steepest descent algorithm is:

$$\theta_{i+1} = \theta_i - \alpha \mathbf{g}_{i+1}(D_{TRAIN},\theta_i) \quad i = 0,1,... \tag{3}$$

where $\mathbf{g}$ is a vector of partial derivatives, known as the *gradient vector*, and α is a learning parameter whose choice is part of the *art* of steepest descent. Too small a choice for α can lead to very long convergence times, whereas too large a choice for α can lead to very erratic behavior of θ_{i+1} from one iteration to the next. Of course, in order to start the steepest descent algorithm in (3), an initial value—θ_0—must be specified.

Another way to write (3) is:

$$\theta_{i+1} = \theta_i - \alpha\left[derivatives\ of\ J(D_{TRAIN},\theta)\Big|_{\theta=\theta_i}\right] \quad i = 0,1,... \tag{4}$$

The vertical-bar notation means that after we determine the derivatives of $J(D_{TRAIN},\theta)$ analytically, some or all of them will still be explicit functions of the unknown θ, and those values are then replaced by the best values we have for them, namely θ_i.

In our tuning procedure we use a squared-error function [see (5-47) in the textbook], i.e.

$$J(D_{TRAIN},\theta) = e(D_{TRAIN},\theta_i) \tag{5}$$

where

$$e(D_{TRAIN},\theta) = \tfrac{1}{2}\left[y(D_{TRAIN},\theta) - y^{(j)}(D_{TRAIN})\right]^2 \tag{6}$$

and

$$y(D_{TRAIN},\theta) = f_s(D_{TRAIN},\theta) \tag{7}$$

In (7), $f_s(D_{TRAIN},\theta)$ is the output of a singleton type-1 FLS. Its exact structure depends on the many choices that have to be made by the designer of a FLS. One example of $f_s(D_{TRAIN},\theta)$ is given in (5-46) of the textbook.

It is easy to compute the derivatives of $J(D_{TRAIN},\theta)$, which are needed in (4), using (5)–(7), i.e.

$$\begin{aligned}\frac{\partial J(D_{TRAIN},\theta)}{\partial\theta} &= \frac{\partial e(D_{TRAIN},\theta)}{\partial\theta} = \left[y(D_{TRAIN},\theta) - y^{(j)}(D_{TRAIN})\right]\frac{\partial y(D_{TRAIN},\theta)}{\partial\theta} \\ &= \left[y(D_{TRAIN},\theta) - y^{(j)}(D_{TRAIN})\right]\frac{\partial f_s(D_{TRAIN},\theta)}{\partial\theta}\end{aligned} \tag{8}$$

In order to proceed further, the specific FLS choices mentioned above must be made. Those choices will let us determine analytical formulas for $\partial f_s(D_{TRAIN},\theta)/\partial\theta$. We complete these calculations for a specific set of choices below in Section III.

This completes the high-level overview on optimizing a function using a steepest descent algorithm. Lots of good software already exists for doing this (e.g., The MathWork's Optimization Toolbox), software that has been written by experts who have included lots of the bells and whistles that let a steepest descent algorithm work well. We return to software for doing this in Lesson 14. ■

III. Derivation of the Steepest Descent Algorithms (5-48)–(5-50)

Here we shall derive the steepest descent algorithms that are given in (5-48)–(5-50), for updating the MF parameters. Regardless of whether $\theta = m_{F_k^l}$, $\bar{y}^l$, or $\sigma_{F_k^l}$, certain parts of the calculations of $\text{grad}J(\theta) = \partial J(\theta)/\partial\theta$, where $J(\theta) = e^{(i)} = \frac{1}{2}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)}\right]^2$, are identical, namely:

$$\frac{\partial J(\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\left\{\tfrac{1}{2}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)}\right]^2\right\} = \left[f_s(\mathbf{x}^{(i)}) - y^{(i)}\right]\frac{\partial}{\partial\theta} f_s(\mathbf{x}^{(i)})$$

where

$$f_s(\mathbf{x}^{(i)}) = \sum_{l=1}^{M} \bar{y}^l \varphi_l(\mathbf{x}^{(i)})$$

and

$$\varphi_l(\mathbf{x}^{(i)}) = \frac{\prod_{k=1}^{p} \exp\left[-\frac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma^2_{F_k^l}}\right]}{\sum_{l=1}^{M}\prod_{k=1}^{p} \exp\left[-\frac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma^2_{F_k^l}}\right]} \tag{9}$$

(a) $\theta = \bar{y}^l$: In this case,

$$\frac{\partial}{\partial \bar{y}^l} f_s(\mathbf{x}^{(i)}) = \varphi_l(\mathbf{x}^{(i)})$$

so that

$$\bar{y}^l(i+1) = \bar{y}^l(i) - \alpha_{\bar{y}} \frac{\partial}{\partial \bar{y}^l} J(\bar{y}^l) = \bar{y}^l(i) - \alpha_{\bar{y}}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)}\right]\varphi_l(\mathbf{x}^{(i)})$$

which is (5-49).

(b) $\theta = m_{F_k^l}$: In this case, it is helpful to use the layered architecture interpretation for (5-24) and (5-25) that is depicted above in Figure 1, i.e. we write $f_s(\mathbf{x}^{(i)})$ as

$$f_s = h \big/ g$$

where

$$h = \sum_{l=1}^{M} \bar{y}^l w^l$$

$$g = \sum_{l=1}^{M} w^l$$

and

$$w^l = \prod_{k=1}^{p} \exp\left\{-\tfrac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_{F_k^l}^2}\right\}$$

Then, we use the *chain rule* to compute $\partial f_s / \partial m_{F_k^l}$ as follows:

$$\frac{\partial f_s}{\partial m_{F_k^l}} = \frac{\partial f_s}{\partial w^l}\frac{\partial w^l}{\partial m_{F_k^l}}$$

where

$$\frac{\partial f_s}{\partial w^l} = \frac{g\dfrac{\partial h}{\partial w^l} - h\dfrac{\partial g}{\partial w^l}}{g^2} = \frac{g\bar{y}^l - h}{g^2} = \frac{\bar{y}^l - f_s}{g}$$

and

$$\begin{aligned}
\frac{\partial w^l}{\partial m_{F_k^l}} &= \frac{\partial}{\partial m_{F_k^l}} \prod_{k=1}^{p} \exp\left\{-\tfrac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_{F_k^l}^2}\right\} \\
&= \frac{\partial}{\partial m_{F_k^l}} \exp\left\{-\tfrac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_{F_k^l}^2}\right\} \times \prod_{\substack{k=1 \\ k\neq l}} \exp\left\{-\tfrac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_{F_k^l}^2}\right\} \\
&= \prod_{k=1}^{p} \exp\left\{-\tfrac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_{F_k^l}^2}\right\} \times \frac{\left(x_k^{(i)} - m_{F_k^l}\right)}{\sigma_{F_k^l}^2} = \frac{\left(x_k^{(i)} - m_{F_k^l}\right)}{\sigma_{F_k^l}^2} \times w^l
\end{aligned}$$

Hence

$$\frac{\partial f_s}{\partial m_{F_k^l}} = \frac{\bar{y}^l - f_s}{g} \times \frac{\left(x_k^{(i)} - m_{F_k^l}\right)}{\sigma_{F_k^l}^2} \times w^l$$

and we obtain the following iterative algorithm for updating $m_{F_k^l}$:

$$\begin{aligned}
m_{F_k^l}(i+1) &= m_{F_k^l}(i) - \alpha_m\left[f_s(\mathbf{x}^{(i)}) - y^{(i)}\right]\left.\frac{\partial f_s}{\partial m_{F_k^l}}\right|_i \\
&= m_{F_k^l}(i) - \alpha_m\left[f_s(\mathbf{x}^{(i)}) - y^{(i)}\right] \times \left[\bar{y}^l(i) - f_s(\mathbf{x}^{(i)})\right] \times \frac{\left(x_k^{(i)} - m_{F_k^l}(i)\right)}{\sigma_{F_k^l}^2(i)} \times \frac{w^l(i)}{g(i)}
\end{aligned}$$

Observe, also, from the previous equations for $\varphi_l(\mathbf{x}^{(i)})$, w^l and g, that

$$\frac{w^l(i)}{g(i)} = \varphi_l(\mathbf{x}^{(i)})$$

Substituting this last equation into the one just before it, we reach the steepest descent algorithm for updating $m_{F_k^l}$ that is given in (5-48).

(c) $\theta = \sigma_{F_k^l}$: The derivation of (5-50) is just like the derivation of (5-48). The key steps are summarized in the layered architectural equations given above for f_s, h, g and w^l. We then compute

$$\frac{\partial f_s}{\partial \sigma_{F_k^l}} = \frac{\partial f_s}{\partial w^l}\frac{\partial w^l}{\partial \sigma_{F_k^l}}$$

where $\partial f_s / \partial w^l$ has been computed above, so we only need the new computation of $\partial w^l / \partial \sigma_{F_k^l}$. Because this last computation is just like the one we just carried out for $\partial w^l / \partial m_{F_k^l}$, we leave its details to the reader.

Key Points

- Each training datum can be interpreted as an IF–THEN rule of the form "IF x_1 is F_1^l and $\cdots$ and x_p is F_p^l, THEN y is G^l," where F_i^l are fuzzy sets described by Gaussian (other shapes can be used) MFs. A particular design method establishes how the MF parameters are specified.
- It is good design practice to have fewer FLS design parameters than training pairs; hence, a constraint always exists among the number of training samples, number of rules, and number of antecedents.
- Three high-level designs can be associated with a singleton type-1 FLS, ranging from one in which the data establishes the rules and no tuning is used, to two others in which the training data is used to tune some or all of the antecedent and consequent MF parameters.
- The layered architecture for a type-1 FLS suggests that errors will be *back-propagated* during a steepest descent parameter tuning procedure, just as they are during the steepest-descent design of a feed-forward neural network.
- The two one-pass design methods let the data establish either the parameters of the MFs or the entire rule. Their major drawback is that they lead to a FLS that has too many rules.
- When all of the antecedent parameters are pre-specified, the method of least-squares can be used to design the consequent parameters; doing this leads to a linear system of equations that has to be solved for the consequent parameters. Knowing how to choose the antecedent parameters ahead of time is a major drawback to using this design method.

- When none of the antecedent or consequent parameters are pre-specified, they can all be tuned using the method of steepest descent.
- Calculating the derivative of the objective function, which is required to derive a steepest descent algorithm, requires a careful use of the chain rule; this can be expedited by making use of the three-layer architectural interpretation of a type-1 FLS.

Practice Problem

Complete Exercise 5-10.

Lesson 10–SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–Part 4

Learning Objectives

This lesson is the fourth in a series of four that cover many aspects of a very widely used FLS—a *singleton type-1 FLS*—ranging from analysis to design to applications. The main purpose of this lesson is to see how singleton type-1 FLSs can be designed for our two case studies, forecasting of time-series and knowledge mining using surveys. After completing this lesson you will be able to:

- Demonstrate how one-pass and back-propagation design methods can be applied to forecasting the Mackey-Glass time-series.
- Explain that if we only have access to noisy measurements, then the performance of a singleton type-1 FLS may not be acceptable because it is unable to directly model such uncertainty.
- Describe how to construct MFs for linguistic labels from survey information about interval information for each label.
- Demonstrate three ways in which expert information (i.e., the consequent of survey rules) can be used in a fuzzy logic advisor (FLA).
- Demonstrate the designs of two FLAs and describe their interpretations as judgment decision surfaces.

Reading Assignment

Read pages 169–183 of the textbook.

Key Points

- It is possible to successfully forecast the *perfectly-measured* chaotic Mackey–Glass time series using a FLS with only 16 rules, when the rule's MFs are tuned using a back-propagation procedure.
- Measurement noise severely degrades a singleton type-1 FLS forecaster, because it has not been accounted for during the design of the FLS and cannot be accounted for during the operation of that system.
- Fixing the parameters of type-1 MFs using survey data is difficult to do because uncertainties about the survey data cannot be modeled using type-1 MFs.
- The three possibilities for using consequent results from surveys are: keep the response chosen by the largest number of experts, find a weighted average of the rule consequents for each rule, and preserve the distributions of the expert-responses for each rule.
- A FLA can be visualized as a multi-dimensional surface.

Practice Problems

Complete Exercises 5–14 and 5–15.

Lesson 11–NON-SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS

Learning Objectives

This lesson covers many aspects of another FLS—a *non-singleton type-1 FLS* (which is also a *Mamdani* FLS)—ranging from analysis to design to applications. Because a non-singleton FLS is very similar to a singleton FLS, we only spend one lesson on it. The main purposes of this lesson are to explain how to quantify the input–output operations of the inference mechanism in a non-singleton type-1 FLS, why this quantification is more complicated than in the singleton case, learn how to design non-singleton type-1 FLSs when training data are available, and learn how non-singleton type-1 FLSs can be designed for forecasting of time-series when only noisy measurements are available. After completing this lesson you will be able to:

- Explain why the architecture of a non-singleton type-1 FLS is the same as for a singleton type-1 FLS.
- Describe what is meant by *non-singleton fuzzification*.
- Demonstrate the calculation of the sup-star composition for the case of non-singleton fuzzification and explain why it is more difficult than in the singleton case.
- Explain how a non-singleton FLS can be interpreted as a prefiltering operation on the measurements followed by the inference mechanism.
- Demonstrate pictorial descriptions of the firing of rules and the combining of multiple-fired rules.
- Explain that what is new for a non-singleton type-1 FLS is the need for the designer to choose MFs for the input measurements, something that wasn't necessary for a singleton type-1 FLS.
- Demonstrate the input–output formula for a non-singleton type-1 FLS as a fuzzy basis function (FBF) expansion and explain the differences between this FBF expansion and the FBF for singleton type-1 FLSs.
- Explain how training data can be interpreted as a collection of IF-THEN rules and describe what the difference is between these IF-THEN rules and the ones for a singleton type-1 FLS.
- Enumerate how many design parameters there can be in a specific design and describe the relation of that number to the number of possible rules in the non-singleton type-1 FLS, and how these numbers compare with those for a singleton type-1 FLS.
- Describe four high-level designs that can be associated with a non-singleton type-1 FLSs.
- Describe two high-level approaches to the tuning of a non-singleton FLS.
- Demonstrate that the design methods learned for singleton type-1 FLSs are easily modified for non-singleton type-1 FLSs.
- Demonstrate how one-pass and back-propagation design methods can be applied to forecasting the Mackey-Glass time-series.
- Explain that if we only have access to noisy measurements, then the performance of a non-singleton type-1 FLS outperforms that of a singleton type-1 FLS, but that there is room for further improvements.

Reading Assignment

Read pages 186–192 of the textbook.

Before you read Example 6–2, review Examples 5–1 and 5–2 in Lesson 5.

Read pages 193–209 of the textbook. Omit Sections 6.6.4 and 6.6.5.

Key Points

- A non-singleton type-1 FLS consists of a fuzzifier, inference mechanism and defuzzifier; its rules are the same as those for a singleton type-1 FLS; it differs from a singleton type-1 FLS in the nature of the fuzzifier.
- A non-singleton fuzzifier treats each input as a fuzzy number, i.e. it assigns a MF to each input that has a value equal to one at the measured value of the input and decreases to zero as the input variable gets farther away from the measured input value.
- As in a singleton type-1 FLS, the MF of a fired rule, $\mu_{B^l}(y)$, is given by $\mu_{B^l}(y) = \sup_{\mathbf{x}\ x}\left[\mu_{A_{\mathbf{x}}}(\mathbf{x}) \bigstar \mu_{A^l \to G^l}(\mathbf{x}, y)\right], \quad y \quad Y$; but, for a non-singleton type-1 FLS, the sup operation does not disappear, because $\mu_{A_{\mathbf{x}}}(\mathbf{x})$ has non-zero values over a range of values for each x_i. Except for some simple, but important choices for the MFs (e.g., Gaussian MFs) it is not possible to evaluate this sup-star composition analytically.
- A non-singleton FLS first pre-filters its input $\mathbf{x}$, transforming it into $\mathbf{x}^l_{\max}$. Doing this accounts for the effects of the input measurement uncertainty, and is a direct result of the sup-star composition.
- Only the pictorial description for the input and antecedent operations of a non-singleton type-1 FLS differs from the one for a singleton FLS. The other pictorial descriptions remain the same.
- The only difference between a type-1 non-singleton and a singleton FLS is the *numerical value* of the firing level; for the former, this value includes the effects of input uncertainties, whereas for the latter it does not.
- The same choices must be made to specify or design a non-singleton type-1 FLS as had to be made for a singleton type-1 FLS. In addition, the designer must specify the MFs for the input measurements, which provides new design degrees of freedom to the non-singleton FLS.
- A non-singleton FLS can also be interpreted as a FBF expansion. Input uncertainty may activate more of these FBFs, which means that decisions are more distributed in the non-singleton case than in the singleton case.
- Training data establish exactly the same sort of rules as they did in the singleton FLS case, and a particular design method establishes how the MF parameters are specified, including those for the input MFs.
- The constraint that exists among the number of training samples, number of rules, and number of antecedents is slightly different for a non-singleton type-1 FLS than it is for a singleton type-1 FLS, because of the addition of the input MF parameters.

- Four high-level designs can be associated with a non-singleton type-1 FLS, ranging from one in which the data establishes the rules and no tuning is used, to three in which the training data is used to tune some or all of the antecedent, consequent, and input measurement MF parameters.
- One approach to designing a non-singleton type-1 FLS—the *partially dependent approach*—is to first design the best possible singleton FLS, freeze the common parameters, and only optimize the parameters that are new to the non-singleton type-1 FLS. A second approach—the *totally independent approach*—is to design the best possible non-singleton type-1 FLS regardless of any pre-existing singleton FLS design.
- The one-pass and least-squares design methods developed for a singleton type-1 FLS are essentially the same for a non-singleton type-1 FLS.
- The steepest-descent algorithms are different for a non-singleton type-1 FLS because of the pre-filtering operation performed by the sup-star composition.
- A non-singleton type-1 FLS forecaster is less sensitive to noisy measurements than a singleton type-1 FLS forecaster, but the improvement is modest.
- When the training data are noisy there is no way to account for this in the antecedent and consequent MFs of a type-1 FLS. This represents a limitation of a type-1 FLS.

Practice Problems

Exercise 11–1
Example 6-1 (just as Example 5-1) is one of the most important ones given in Chapter 6, because it provides a geometric interpretation for the operations that occur within the inference engine. In this exercise, I want you to provide the figures that are comparable to the ones given in Figures 6-3, 5-5 and 5-6, but using triangular MFs. Do this for both the minimum and product t-norms.

Complete Exercises 6–5 and 6–7.

Lesson 12–TSK FUZZY LOGIC SYSTEMS

Learning Objectives

This lesson covers many aspects of another (and our last) FLS—a *type-1 TSK FLS*—ranging from analysis to design to applications. The TSK FLS is very popular in control systems applications of FL and is also becoming popular in some signal processing applications of FLSs. After completing this lesson you will be able to:

- Describe the mathematical model for a first-order type-1 TSK FLS.
- Demonstrate connections between type-1 TSK and Mamdani FLSs.
- Explain that TSK FLSs are also universal approximators.
- Enumerate how many design parameters there are in a TSK FLS.
- Demonstrate some design methods that can be used for the design of TSK FLSs, namely: least-squares method, back-propagation (steepest descent method), and an iterative design method.
- Describe what we mean by "forecasting of compressed video traffic."
- Demonstrate how to design type-1 TSK and Mamdani forecasters of compressed video traffic.

Reading Assignment

Read pages 421–428 of the textbook. Omit Section 13.3

Section 13.4 explains how to design both type-1 and type-2 TSK and Mamdani FLSs for the problem of forecasting compressed video traffic. Because the textbook interweaves material about both type-1 and type-2 designs, here we will filter out all of the type-2 design materials (leaving them for the follow-on course *New Directions in Rule-Based Fuzzy Logic Systems: Handling Uncertainties*), i.e. we will guide you through Section 13.4.

Start by reading Section 13.4.1, including Example 13-5, pp. 442–444, but, omit the two paragraphs directly after Example 13-5. Read the last paragraph of Section 13.4.1.

Next, we have extracted materials from Sections 13.4.2–13.5 that focus on the type-1 designs.

Section 13.4.2 Forecasting I frame sizes: General Information

In the rest of this section we focus on the problem of forecasting I frame sizes (i.e., the number of bits/frame) for a specific video product, namely *Jurassic Park*. All of our methodologies for doing this apply as well to forecasting P and B frame sizes and can also be applied to other video products.

Here we examine two designs of FLS forecasters based on the logarithm of the first 1000 I frame sizes of *Jurassic Park*, $s(1)$, $s(2)$, …, $s(1000)$ (see Figure 13-1). Those designs are type-1 TSK FLS and singleton type-1 Mamdani FLS. We used the first 504 data [$s(1)$, $s(2)$, …, $s(504)$] for

tuning the parameters of these forecasters, and the remaining 496 data [$s(505)$, $s(502)$, …, $s(1000)$] for testing after tuning.

Type-1 TSK FLS: The rules of this FLS forecaster are (i = 1, …, M)

$$\begin{aligned} R^i\text{: IF } & s(k-3) \text{ is } F_1^i \text{ and } s(k-2) \text{ is } F_2^i \text{ and } s(k-1) \text{ is } F_3^i \\ & \text{and } s(k) \text{ is } F_4^i \text{ THEN } \hat{s}^i(k+1) = c_0^i + c_1^i s(k-3) \\ & + c_2^i s(k-2) + c_3^i s(k-1) + c_4^i s(k) \end{aligned} \tag{13-53}$$

We initially chose F_j^i to be the same for all i (rules) and j (antecedents), and used a Gaussian membership function for them, one whose initial mean and standard deviation were chosen from the first 500 I frames as (see "Entire segment" row of Table 13-2) $m = 4.7274$ and $\sigma = 0.0954$. According to Table 13-1, the number of design parameters for this type-1 TSK FLS is $(3p+1)M = 13M$.

Singleton type-1 Mamdani FLS: The rules of this FLS forecaster are (i = 1, …, M)

$$\begin{aligned} R^i\text{: IF } & s(k-3) \text{ is } F_1^i \text{ and } s(k-2) \text{ is } F_2^i \text{ and } s(k-1) \text{ is } F_3^i \\ & \text{and } s(k) \text{ is } F_4^i \text{ THEN } \hat{s}^i(k+1) \text{ is } G^i \end{aligned} \tag{13-57}$$

We used height defuzzification. As we did for the type-1 TSK FLS, we initially chose F_j^i to be the same for all i (rules) and j (antecedents), and used a Gaussian membership function for them, one whose initial mean and standard deviation were chosen from the first 500 I frames, as described earlier, as $m = 4.7274$ and $\sigma = 0.0954$. According to Table 13-1, the number of design parameters for this singleton type-1 Mamdani FLS is $(2p+1)M = 9M$.

13.4.3 Forecasting I frame sizes: Using the same number of rules

In this first approach to designing the two FLS forecasters, we fixed the number of rules at five in both of them; i.e., M = 5. Doing this means that the type-1 TSK FLS is described by 65 design parameters and the singleton type-1 Mamdani FLS is described by 45 design parameters. Steepest descent algorithms (as described in Section 5.9.3 for the Mamdani FLS and in Section 13.2.4 for the TSK FLS) were used to tune all of these parameters. In these algorithms, we used step sizes of $\alpha = 0.001$ and $\alpha = 0.01$ for the TSK and Mamdani FLSs, respectively.

We have already explained how we chose initial values for the membership function parameters. All of the remaining parameters were initialized randomly, as follows:

- Consequent parameters, $c_j^i (i = 1,...,5; j = 0,1,...,4)$, of the TSK FLS were each chosen randomly in [0, 0.2] with uniform distribution.
- Consequent parameters, $\bar{y}^i (i = 1,...,5)$, of the Mamdani FLS were chosen randomly in [0, 5] with uniform distribution.

Because we chose the initial values of the consequent parameters randomly, we ran 50 Monte-Carlo realizations for each of the 2 designs.[10] For each realization, each of the two FLSs was tuned for 10 epochs on the 504 training data. All designs were then evaluated on the remaining 496 testing data using the following *RMSE*:

$$RMSE = \sqrt{\frac{1}{496}\sum_{k=504}^{999}\left[s(k+1) - f_{FLS}(\mathbf{s}^{(k)})\right]^2} \tag{13-59}$$

where $\mathbf{s}^{(k)} = [s(k-3), s(k-2), s(k-1), s(k)]^T$. The average value and standard deviations of these *RMSE*s are plotted in Figure 13-2 for each of the 10 epochs. Observe, from Figure 13-2(a), that (pay attention only to the curves for the two type-1 designs):

1. After 10 epochs of tuning, the average RMSE of the 2 FLS forecasters is:

   - Type-1 TSK FLS: 0.0779
   - Singleton type-1 Mamdani FLS: 0.0808

2. In terms of average RMSE and standard deviation of the RMSE, the type-1 TSK FLS outperforms the singleton type-1 Mamdani FLS for epochs 2–10.

13.4.4 Forecasting I frame sizes: Using the same number of design parameters

Because a five-rule TSK FLS always has more parameters (design degrees of freedom) to tune than does a comparable five-rule Mamdani FLS, we modified the previous approach to designing the two FLSs. We did this by fixing the rules used by the TSK FLS at five and by then choosing the number of rules used by the Mamdani FLS so that its total number of design parameters approximately equals the number for the TSK FLS. Doing this led us to use seven rules for the Mamdani FLS. The designs of the resulting two FLSs proceeded exactly as described in the preceding section. All designs were again evaluated using the *RMSE* in (13-59). The average value and standard deviations of these *RMSE*s are plotted in Figure 13-3 for each of the 10 epochs (again, only pay attention to the curves for the two type-1 designs). Observe that:

- The results are similar to the ones depicted in Figure 13-2; so, at least for this example, equalizing the numbers of design parameters in the Mamdani and TSK FLSs does not seem to be so important.

13.4.5/13.5 Conclusion

It is not our intention in this example to recommend one FLS architecture over another. Some people prefer a TSK FLS over a Mamdani FLS or vice-versa. We leave that choice to the designer who, as always, must be guided by a specific application. When both kinds of FLSs are applicable, as in the case of forecasting a random-signal and perfect-measurement time-series, the designer can carry out a comparative performance analysis between the two architectures, as we have just done.

[10] In Chapters 5 and 6 Monte-Carlo simulations were run to average out the effects of additive measurement noise. Here they are run to average out the effects of random initial consequent parameter values.

Key Points

- "TSK" is short for Takagi, Sugeno and Kang, the originators of the TSK FLS.
- To-date only a singleton type-1 TSK FLS has been described in the literature.
- The most widely used type-1 TSK FLS uses *first-order rules*, i.e., rules whose antecedents are type-1 fuzzy sets, and whose consequent is a linear combination of the measured antecedents. The fact that its consequent is a function and not a fuzzy set is the biggest difference between a TSK FLS and a Mamdani FLS.
- The output formula for a type-1 TSK FLS is obtained by combining its rules in a *prescribed way*; it does not derive from the sup-star composition, as does the output of a type-1 Mamdani FLS. This is another big difference between a TSK FLS and a Mamdani FLS.
- Normalized and unnormalized type-1 TSK FLSs have been defined.
- When the consequent function in a TSK rule is a constant, then the normalized type-1 TSK FLS is exactly the same as a type-1 Mamdani FLS that uses either center-of-sums, height, modified height, or center-of-sets defuzzification.
- TSK FLSs are also universal approximators.
- Just as in a type-1 Mamdani FLS, a constraint always exists among the number of training samples, number of rules and number of antecedents in a type-1 TSK FLS. Because the consequent of a TSK rule contains more design parameters than does the consequent of a Mamdani rule, a TSK FLS that uses the same number of rules as a Mamdani FLS always has more design degrees of freedom than a Mamdani FLS.
- Two high-level designs can be associated with a singleton TSK FLS. In one design, the shapes and parameters of all the antecedent MFs are fixed ahead of time and the training data is used to tune only the consequent parameters. In the other design, the training data is used to tune all of the MF and consequent parameters.
- When all of the antecedent parameters are pre-specified, the method of least-squares can be used to design the consequent parameters; doing this leads to a linear system of equations that has to be solved for the consequent parameters. Knowing how to choose the antecedent parameters ahead of time is a major drawback to using this method.
- When none of the antecedent of consequent parameters are pre-specified, they can all be tuned using the method of steepest descent.
- It is possible to interweave the steepest-descent and least squares design methods to obtain a more powerful iterative design method. (This can also be done for the design of a Mamdani FLS.)
- *Forecasting compressed video* means predicting a future value of either an I, P or B frame, directly in the compressed video domain, using a window of previously measured I, P, or B frame values.
- Forecasting of compressed video can be accomplished using either singleton type-1 TSK or Mamdani FLSs. Somewhat better performance is achieved for the TSK forecaster.
- Some people prefer a TSK FLS over a Mamdani FLS or vice-versa. The final choice is left to the designer who, as always, must be guided by a specific application.

Practice Problem

Complete Exercise 13–1. [This exercise is very similar to the calculations that are included in Lesson 9 of this Study Guide. So, you may be wondering why I am asking you to once again carry out derivative calculations. My answer to this rhetorical question is “These calculations require your bringing together all of the equations that are needed to implement a type-1 TSK FLS, and this is a good thing to do.”]

Lesson 13–APPLICATIONS OF TYPE-1 FLSs

Learning Objectives

This lesson will let you explore one-to-three applications of type-1 FLSs, namely: rule-based pattern classification of video traffic, equalization of time-varying non-linear digital communication channels, and fuzzy logic control. The main purpose of the lesson is to let you see how one or more of the FLSs already studied can be used to solve some real-world problems. You must cover at least one of these applications; but, if more than one is of interest to you, then do more. After completing this lesson you will be able to:

1. Demonstrate an application that can be solved using FLSs
2. Demonstrate the versatility of FLSs

Reading Assignment

Read below about one or more of the following three applications:

1. Rule-based pattern classification of video traffic
2. Equalization of time-varying non-linear digital communication channels
3. Fuzzy logic control

I. Rule-Based Classification of Video Traffic

For this self-study course, we focus on the use of type-1 FLSs as rule-based classifiers. Consequently, we have modified Section 14.4 of the textbook as follows:

1. Read the first three paragraphs in Section 14.4 on pp. 458–459.

2. Paragraph 4 of Section 14.4, on p. 459, is modified to:

Given a collection of MPEG-1 compressed movies and sports program videos, we shall use a subset of them to create (i.e., design and test) a rule-based classifier (RBC) in the framework of FL. We shall develop two type-1 classifiers and compare them to see which provides the best performance. Our overall approach is to:

1. Choose appropriate features that act as the antecedents in a RBC
2. Establish rules using the features
3. Optimize the rule design-parameters using a tuning procedure
4. Evaluate the performance of the optimized RBC using testing

The first two steps of this procedure are relatively straightforward. The third step requires that we establish the computational formulas for the FL-based classifiers, in much the same way that we established such formulas for the Mamdani FLSs of Chapters 5 and 6 and the TSK FLS in Chapter 13. We do this following. The fourth step requires that we also baseline our FL classifiers. We do this using the accepted standard of a Bayesian classifier, one whose structure we also explain following.

3. Read Section 14.4.1

4. Omit Section 14.4.2.

5. Section 14.4.3 (**Rules**) is modified to:

Rules for a RBC of compressed video traffic use the three selected features as their antecedents and have one consequent. The antecedents are: logarithm of bits/I frame, logarithm of bits/P frame, and logarithm of bits/B frame. The consequent is +1 if the video is a movie and –1 if it is a sports program. Observe that there is nothing fuzzy about a rule's consequent in rule-based classification; i.e., each rule's consequent is assigned a numerical value, +1 or –1.

Each rule in a *type-1 fuzzy logic rule-based classifier* (FL RBC) has the following structure:[4]

$$R^l\text{: IF I frame is } F_1^l \text{ and P frame is } F_2^l \text{ and B frame is } F_3^l \text{ , THEN the product is a movie (+1) or a sports program (-1)} \tag{14-1}$$

Observe that these rules are a special case of a Mamdani FLS rule, one in which the consequent is a singleton. Such a rule can also be interpreted as a TSK rule.

We use a very small number of rules, namely one per video product, e.g. if our training set contains four movies and four sports programs, we use just eight rules.

6. Omit Section 14.4.4.

7. Section 14.4.5 (**Design parameters in a FL RBC)** is modified to:

In our simulations below we shall design two FL RBCs—singleton type-1 FL RBC and non-singleton type-1 FL RBC. The design results will establish which classifier provides the best performance.

Each antecedent membership function has two design parameters, its mean and standard deviation; hence, there are six design parameters per rule. For the non-singleton type-1 FL RBC there is also one additional design parameter for each measurement—the standard deviation of its Gaussian MF.

Optimum values for all design parameters are determined during a tuning process; but, before such a process can be programmed, we must first establish computational formulas for the FL RBCs.

8. Read Section 14.4.6.

9. Omit Section 14.4.7.

[4]See also Kuncheva (2000) for an excellent introduction to RBCs.

10. Section 14.4.8 (**Optimization of rule design-parameters)** is modified to:

In our simulation results discussed in Section 14.4.10, we begin with five movies and five sports programs and, by way of illustration, design FL RBCs using four movie rules and four sports program rules; i.e. each classifier has eight rules. Each one of the two FL RBCs is optimized using very simple modifications of the tuning procedures that are described in Sections 5.9.3 and 6.6.3. The modifications are due to using an unnormalized output. We leave it to the reader to develop the details of these tuning procedures. The online M-files (see Appendix C) *train_sfls_type1.m* and *train_nsfls_type1.m*, which are for tuning normalized type-1 FLSs, are easily adapted to the present situations.

11. Read Section 14.4.9.

12. Section 14.4.10 (**Results and conclusions**) is modified to:

So as not to get lost in the many details associated with the designs of the 2 FL RBCs, we refer the reader to Liang and Mendel (2000e) for them. Here we focus on one set of results for so-called *out-of-product classification*. "Out-of-product" means that we use *some* of the compressed data from *some* of the *available* video products to establish the rules and to optimize (tune) the resulting classifiers, and we test the classifiers on the unused video products. As mentioned earlier, we used eight video products out of a total of 10 available products—four movies and four sports programs—to design each RBC. The first 24,000 (out of 40,000) compressed frames of each of the eight video products were used to establish and design two eight-rule FL RBCs. The first 37,500 compressed frames of the remaining two videos were then used for testing. An exhaustive study of the 25 possible designs (five movies taken four at a time multiplied by four sports programs taken four at a time equals five times five) was conducted. Average FAR (averaged aver the 25 possible designs) for the two FL RBCs as well as for the Bayesian classifier are:

- singleton type-1 FL RBC: FAR = 9.41%
- non-singleton type-1 FL RBC: FAR = 9.17%
- Bayesian classifier: FAR = 14.29%

From these results, we see that the non-singleton type-1 FL RBC provides the best performance, and has 35.8% fewer false alarms than does the Bayesian classifier. Additional simulation studies that use 20 video products (10 movies and 10 sports programs) have been performed and support these conclusions.

In summary, we have demonstrated that it is indeed possible to perform high-level classification of movies and sports programs working directly with compressed data. Even better performance is possible using type-2 FL RBCs, as will be demonstrated in the follow-on course *New Directions in Rule-Based Fuzzy Logic Systems: Handling Uncertainties* (see, also, the discussion of results in the textbook's Section 14.4.10 on pp. 468–469).

II. Equalization of Time-Invariant Non-linear Digital Communication Channels

For this self-study course, we focus on the use of type-1 FLSs as equalizers—fuzzy adaptive filters (FAFs)—for *time-invariant* non-linear digital communication channels. Consequently, we have modified Section 14.5 of the textbook as follows:

1. Read pp. 469–470, through Figure 14-2.

2. Read Section 14.5.1.

3. Omit Section 14.5.2.

4. Section 14.5.3 (**Designing the FAFs**) is modified to:

Here we illustrate the design of a singleton type-1 FAF for the non-linear time-invariant channel in (14-36). The FAF has eight rules, one per channel state, and the rules have the following structure ($l = 1, \ldots, 8$):

$$R^l\text{: IF } r(k) \text{ is } F_1^l \text{ and } r(k-1) \text{ is } F_2^l \text{, THEN } y^l = w_l \qquad (14\text{-}44)$$

In these rules, w_l is a crisp value of +1 or –1, as determined by (14-39). We used Gaussian membership functions for F_1^l and F_2^l.

Because of the isomorphism between equalization and classification, the computational formulas for type-1 FAFs are easily obtained from Section 14.4.6, as follows:

$$f^l(\mathbf{x}) \cdots \text{use (14-}$$

$$y_{RBC,1}(\mathbf{x}) \to y_{FAF,1}(\mathbf{x}) \cdots \text{use (14-1}$$

$$y^l = w_i = \pm \text{ if } y_{RBC,1}(\mathbf{x}) \to y_{FAF,1}(\mathbf{x})$$

$$\textit{Decision rule} \cdots \text{use (14-1}$$

$$y_{RBC,1}(\mathbf{x}) \to y_{FAF,1}(\mathbf{x})$$

In Karnik et al. (1999) and Liang and Mendel (2000d), the mean-value parameters of all membership functions were estimated using a clustering procedure [Chen et al. (1993a)] that was applied to some training data, because such a procedure is computationally simple. We used this same procedure. An alternative to doing this is to use a tuning procedure.

5. Section 14.5.4 (**Simulations and conclusions**) is modified to:

Here we compare a singleton type-1 FAF and a *K*-nearest neighbor classifier (NNC) [Savazzi et al. (1998)] for equalization of the time-invariant non-linear channel in (14-36). In our simulations, we chose the number of taps of the equalizer, *p*, equal to the number of taps of the channel, $n+1$, where $n=1$; i.e. $p = n+1 = 2$. The number of rules equaled the number of clusters; i.e. $2^{p+n} = 8$. We used a sequence $s(k)$ of length 1000 for our experiments. The first

121 symbols[8] were used for training (i.e. clustering) and the remaining 879 were used for testing. The training sequence established the parameters of the antecedent membership functions, as described in Section 14.5.3. After training, the parameters of the type-1 FAF were fixed and then testing was performed.

The results below do *not* appear in the textbook (the ones in Figures 14-4 and 14-5 are for a time-varying channel). They were created especially for the *Study Guide* by Dr. Qilian Liang.

We ran simulations for nine different *SNR* values, ranging from *SNR* = 10dB to *SNR* = 18dB (at equal increments of 1dB), and we set $d = 0$. We performed 100 Monte-Carlo simulations for each value of *SNR*, where in each realization the AGN was uncertain. The mean values and standard deviations of the bit error rate (BER) for the 100 Monte-Carlo realizations are plotted in Figures 1 and 2 below, respectively. Observe, from these figures that:

4. In terms of the mean values of BER, the type-1 FAF performs better than the NNC (see Figure 1).
5. In terms of the standard deviation of BER, the type-1 FAF is more robust than the NNC (see Figure 2).

These observations suggest that a type-1 FAF, as just designed, looks very promising as a transversal equalizer for time-invariant non-linear channels.

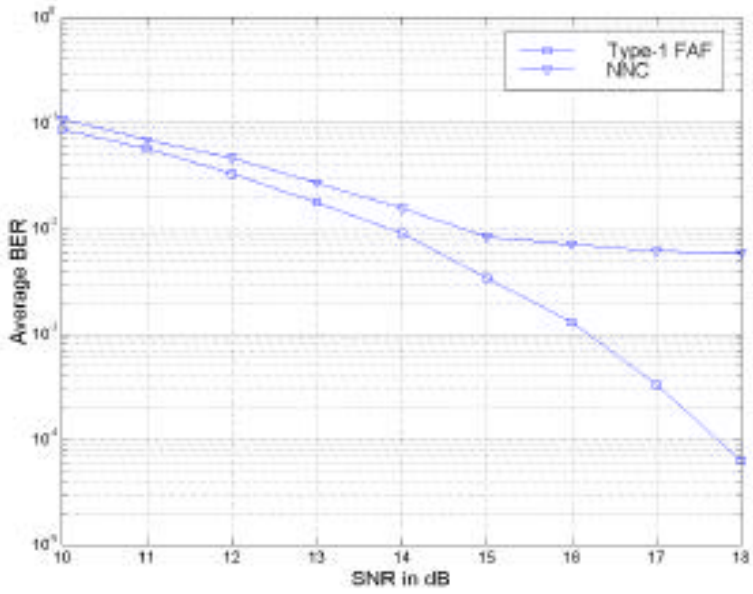


Figure. 1: Average BER of type-1 FAF and nearest neighbor classifier (NNC) versus SNR.

[8]In the *K*-NNC, if the number of training prototypes is *N*, then $K = \sqrt{N}$ is the optimal choice for *K*. It is required that *N* be an odd integer; hence, the choice of $N = 121$.

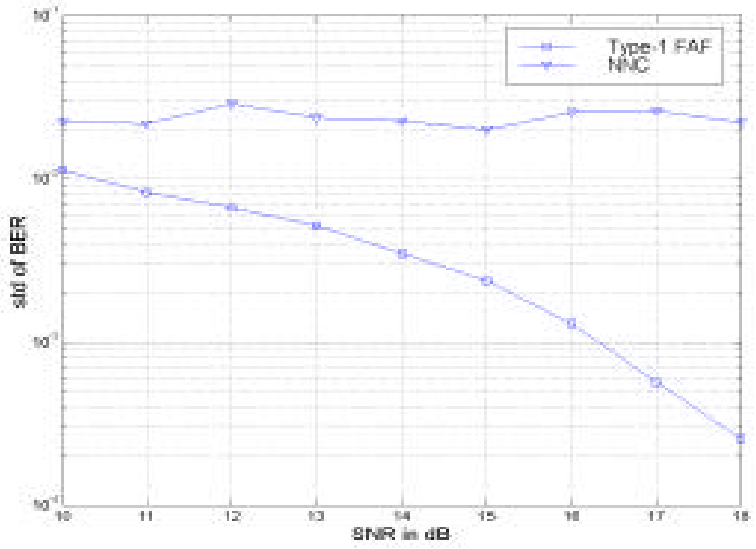


Figure. 2: STD of BER of type-1 FAF and nearest neighbor classifier (NNC) versus SNR.

When uncertainties, such as additive measurement noise or time-varying channel coefficients, are present, then type-2 FAFs outperform their type-1 counterparts, because they are able to model such uncertainties and minimize their effects. This will be demonstrated in the follow-on course *New Directions in Rule-Based Fuzzy Logic Systems: Handling Uncertainties.*

III. Fuzzy Logic Control

The material in this section was prepared by Prof. Li-Xin Wang.

III.A Introduction

A control system consists of two parts: the controller and the plant under control. Therefore, the fuzzy control approaches developed over the years can be best classified according to the structures and assumptions on the controller and the plant. Specifically, the plant can be modeled as linear, non-linear, or fuzzy system models, and these models can be known or unknown to the control-system designer ahead of time. The controller, on the other hand, can be a fixed structure (e.g., TSK or Mamdani FLS) or can be designed according to the model of the plant. The controller can be non-adaptive (i.e., its parameters are determined during the design phase and do not change during the on-line implementation phase) or adaptive (i.e., its parameters are updated on-line during the real-time operation of the overall system). Figure 3 depicts this classification of fuzzy control approaches. Many combinations of plant and controller subclasses result in meaningful fuzzy control systems. In the next two sections, we will summarize the state-of-the-art of non-adaptive and adaptive fuzzy control theory, respectively.

III.B Non-Adaptive Fuzzy Control

In this section, we consider three situations: linear plant with a fuzzy controller, non-linear plant with a fuzzy controller, and fuzzy plant with a fuzzy controller.

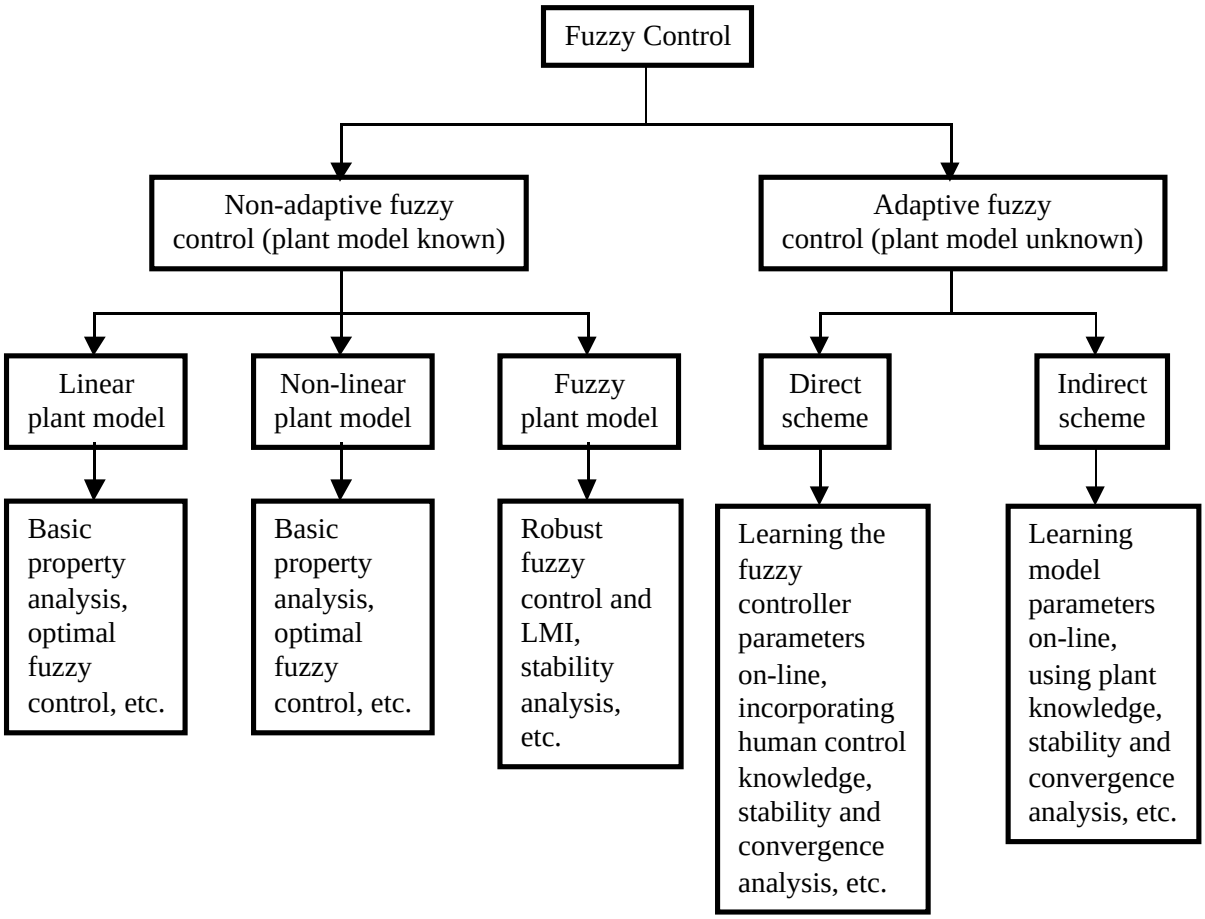


Figure 3: Classification of fuzzy control schemes.

III.B.1 Linear plant plus fuzzy controller

The motivation to study the control of a linear plant using a fuzzy controller is the fact that improved performance can usually be obtained by controlling a linear plant with a non-linear controller, and a fuzzy logic controller is non-linear. The main issues here are how to design the structure and parameters of the fuzzy controller so as to guarantee the stability and robustness of the closed-loop system when the linear plant model is either known or unknown. See Chapters 17 and 18 of Wang (1997) where these issues are addressed. The following reference is a recent approach in which optimal control principles were used to design a fuzzy controller for a linear plant so as to achieve certain optimal performance [Wang, L.-X., "Stable and Optimal Fuzzy Control of Linear Systems," *IEEE Trans. on Fuzzy Systems*, vol. 6, pp. 137-143, 1998].

III.B.2 Non-linear plant plus fuzzy controller

Sliding-mode control is a powerful approach to controlling non-linear and uncertain systems. It is a robust control method and can be applied in the presence of non-linear plant-model uncertainties and plant-parameter disturbances, provided that the bounds of these uncertainties and disturbances are known. In fuzzy sliding-mode control the fuzzy controller is decomposed locally, with each rule responsible for control within a region of the state space that is covered by

that rule. The fuzzy control rules are designed so as to push the system's states to the so-called *sliding surface*. See Chapter 19 of Wang (1997) or Driankov, et al (1996) for detailed discussions about fuzzy sliding-mode control.

III.B.3 Fuzzy plant model plus fuzzy controller

Using a fuzzy model of the plant as well as a fuzzy controller is very popular in recent fuzzy control studies. The plant is modeled using *r* TSK fuzzy logic rules of the form:

$$\text{Plant } R^i\text{: IF } z_1(t) \text{ is } F_{i1} \text{ and } \cdots \text{ and } z_g(t) \text{ is } F_{ig}\text{, THEN } \dot{\mathbf{x}}(t) = \mathbf{A}_i\mathbf{x}(t) + \mathbf{B}_i u(t),\ y_i(t) = \mathbf{C}_i\mathbf{x}(t) \qquad (1)$$

where $i = 1,...,r$. The fuzzy controller is also modeled by *r* TSK fuzzy rules of the form:

$$\text{Controller } R^i\text{: IF } z_1(t) \text{ is } F_{i1} \text{ and } \cdots \text{ and } z_g(t) \text{ is } F_{ig}\text{, THEN } u(t) = \mathbf{K}_i\mathbf{x}(t) \qquad (2)$$

where $i = 1,...,r$. The main advantage of this approach is that, although the plant model and the controller are non-linear, the control law can be designed locally (i.e., for each *i*) using linear control design principles. Specifically, from (1) and (2), we see that for each local region described by "IF $z_1(t)$ is F_{i1} and $\cdots$ and $z_g(t)$ is F_{ig}" the plant model is linear and the controller is also linear. Studies have shown that if all of the local linear controllers are stable, then under certain conditions the global control system is also stable. For detailed discussions about this, see [Tanaka, K., Ikeda, T. and H. O. Wang, "Fuzzy Regulators and Fuzzy Observers: Relaxed Stability Conditions and LMI-Based Designs," *IEEE Trans. on Fuzzy Systems*, vol. 6, pp. 250-256, 1998.]

IV. Adaptive Fuzzy Control

In this section we describe two kinds of adaptive fuzzy control—indirect and direct. In indirect adaptive fuzzy control the fuzzy controller comprises a number of fuzzy systems constructed (initially) form knowledge about the plant, whereas in direct adaptive fuzzy control, the fuzzy controller is a single fuzzy system constructed (initially) from knowledge about the control. It is even possible to combine indirect and direct fuzzy controllers.

IV.A Indirect Adaptive Fuzzy Control

In indirect adaptive fuzzy control, plant non-linearities are unknown and fuzzy systems are used to model them. The parameters of the fuzzy systems are tuned on-line in such a way that the overall output of the fuzzy system model follows the output of the plant. The controller is designed according to the fuzzy system model, which is considered to be the true model of the plant. Since the fuzzy system model is changing on-line, the controller is time-varying and adaptive. More specifically, consider the plant with the structure:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))u(t) \qquad (3)$$

$$y(t) = x_1(t) \qquad (4)$$

where **f** and **g** are unknown non-linear functions. The fuzzy system model for the plant is

$$\dot{\hat{\mathbf{x}}}(t) = \hat{\mathbf{f}}(\mathbf{x}(t) | \theta_f(t)) + \hat{\mathbf{g}}(\mathbf{x}(t) | \theta_g(t))u(t) \quad (5)$$

where $\hat{\mathbf{f}}$ and $\hat{\mathbf{g}}$ are fuzzy systems, and $\theta_f(t)$ and $\theta_g(t)$ are parameters of the respective fuzzy system. These parameters change on-line (which is why they are shown as functions of time) so as to make $\hat{\mathbf{f}}$ and $\hat{\mathbf{g}}$ approximate **f** and **g**, respectively. The adaptation laws for $\theta_f(t)$ and $\theta_g(t)$ have the general forms:

$$\dot{\theta}_f(t) = h_f(\theta_f(t), \theta_g(t), y(t), \hat{\mathbf{x}}(t), u(t)) \quad (6)$$

$$\dot{\theta}_g(t) = h_g(\theta_g(t), \theta_f(t), y(t), \hat{\mathbf{x}}(t), u(t)) \quad (7)$$

The controller, $u(t)$, is designed as if (5) is the true model of the plant in (3). For example, the following controller cancels the non-linearities and then uses a linear control law to make the plant output $x(t)$ follow a desired trajectory, $x_d(t)$, of a first-order dynamical system:

$$u(t) = \frac{1}{\hat{g}(x(t) | \theta_g(t))}\left[-\hat{f}(x(t) | \theta_f(t)) + \dot{x}_d(t) + 0.5\left(x_d(t) - x(t)\right)\right] \quad (8)$$

See Wang (1997, 1994) for the details.

IV.B Direct Adaptive Fuzzy Control

In direct adaptive control, the controller is a single fuzzy system whose parameters are updated on-line so as to make the plant output follow a set-point trajectory. Specifically, suppose the plant structure is still the one in (3), but the controller now is:

$$u(t) = \hat{u}(\mathbf{x}(t) | \theta(t)) \quad (9)$$

where $\hat{u}$ is a standard FLS whose parameters, $\theta(t)$, are up-dated on-line in a similar manner to (6) and (7), so as to force $y(t)$ follow a desired trajectory, $x_d(t)$. See Wang (1997, 1994) for the details.

V. Conclusions

Fuzzy control is an active research field and many new results have appeared in recent years. A good reference that puts many approaches to fuzzy control into a single book is [Farinwata, S. S., Filev, D. and R. Langari, *Fuzzy Control: Synthesis and Analysis*, John Wiley & Sons, Ltd., New York, 2000].

Key Points

I. Rule-Based Classification of Video Traffic

- Direct classification of compressed video traffic can save time and money.
- The three features that are used in RBCs are logarithm of bits per I, P, and B frames.
- I frames have more bits/frame than P frames, which have more bits/frame than B frames.

- Rules for a RBC of compressed video traffic use the three selected features as their antecedents and have one consequent (+1 for a movie or –1 for a sports program).
- Each video product leads to one rule.
- The computational formulas for type-1 FL RBCs follow directly from computational formulas for singleton or non-singleton unnormalized Mamdani type-1 FLSs, in which the MF of the consequent is either 1(for a movie or sports program) or 0 (for anything else).
- Each rule has a small number of design parameters that can be tuned using a training set of video traffic and the steepest descent tuning procedures described in Chapters 5 and 6.
- The performance of the FL RBCs is base-lined against a Bayesian classifier.
- False-alarm rate (FAR) is used as the measure of performance for all classifiers.
- The FLCs outperformed the Bayesian classifier, and the FAR of the non-singleton type-1 FL RBC gave the best results.

II. Equalization of Time-Invariant Non-linear Digital Communication Channels

- When a message gets confused because of the transmitting and receiving media as well as by objects that may interfere with it, there is *inter-symbol interference* (ISI).
- Inter-symbol interference is undone at the receiving end of a digital communication system by *equalization*.
- The goal in channel equalization is to recover the input sequence based on a sequence of measured channel output values without knowing or estimating the channel's coefficients.
- A *transversal* equalizer processes a finite window of past channel output measurements.
- An equalizer of order *p* for a channel of order *n* is characterized by 2^{n+p} *channel states*.
- Equalization of binary signals is equivalent to two-category classification; hence, an *un-normalized output* singleton type-1 FL RBC—a FAF—can be used to implement a Bayesian equalizer for a time-invariant channel.
- The antecedents of FLS rules are the *p* components of the channel state vector; the consequent is a crisp value of either +1 (for a +1 transmitted symbol) or –1 (for a –1 transmitted symbol)
- A type-1 FAF outperforms a nearest neighbor classifier (equalizer), especially at higher SNRs.

III. Fuzzy Logic Control

- Many different kinds of fuzzy logic controllers have been developed.
- At the highest level, we can distinguish between non-adaptive and adaptive fuzzy logic controllers
- Non-adaptive fuzzy control, in which the controller is a FLS, can be further classified by the way in which the plant is modeled: linear plant, non-linear plant, or fuzzy plant
- In non-adaptive fuzzy control, the controller's parameters are determined during the design phase and do not change during the on-line implementation phase
- Adaptive fuzzy control, in which the controller is also a FLS, can be further classified by the knowledge used to construct the fuzzy controller: indirect or direct.

· In adaptive fuzzy control, the controller's parameters are updated during the real-time operation of the overall system.

Review Questions

I. Rule-Based Classification of Video Traffic

1. Circle all of the possible design parameters for a non-singleton type-1 FL RBC, when Gaussian MFs are used:

a. Mean of each antecedent MF
b. Mean of the consequent MF
c. Standard deviation of each antecedent MF
d. Standard deviation of the consequent MF
e. Mean of each measurement MF
f. Standard deviation of each measurement MF
g. Kurtosis of each measurement MF

2. The output of the FLS in a RB FLC:

a. must be normalized
b. does not have to be normalized
c. must come from a Mamdani architecture

3. Normalization of $y_{RBC,1}(\mathbf{x})$ by $\sum_{l=1}^{M} f^l$ in (14-9) does not change the sign of $y_{RBC,1}(\mathbf{x})$ because:

a. $\sum_{l=1}^{M} f^l = 0$
b. $\sum_{l=1}^{M} f^l = 1$
c. $\sum_{l=1}^{M} f^l > 0$ always
d. $\sum_{l=1}^{M} f^l < 0$ always

4. A 10-rule singleton type-1 FL RBC that uses Gaussian MFs has how many design parameters?

a. 50
b. 60
c. 70

5. Suppose that a FL RBC gives the following results for 500 testing elements: 240 movies are correctly classified, 245 sports programs are correctly classified, 10 movies are mis-classified as

sports programs, and 5 sports programs are mis-classified as movies. How many false alarms are there?

a. 5
b. 10
c. 15

II. Equalization of Time-Invariant Non-linear Digital Communication Channels

1. Inter-symbol interference (ISI) occurs when the:

a. Message is sent to a wrong address
b. Receiver becomes inoperative
c. Message gets confused because of the transmitting and receiving media as well as by objects that may interfere with it

2. ISI is undone by a process known as:

a. Deconvolution
b. Equalization
c. Filtering

3. A transversal equalizer for a channel of order n that uses a window of past measurements $r(k), r(k-1), \ldots, r(k-p+1)$ has how many *taps*?

a. p
b. n
c. $n-p+1$

4. A channel of order 4 that is equalized by a transversal equalizer of order 4 has how many states?

a. 2^4
b. 2^8
c. 2^{16}

5. Monte-Carlo simulations in our equalization experiment are needed in order to average out the effects of:

a. Classification errors
b. Channel initial conditions
c. Additive random noise

III. Fuzzy Logic Control

1. How many kinds of fuzzy logic controllers are there?

    a. one
    b. many
    c. six

2. Controller's parameters are determined during the design phase and do not change during the on-line implementation phase in what kind of control?

    a. non-linear
    b. non-adaptive fuzzy control
    c. adaptive fuzzy control
    d. sliding-mode control

3. The motivation to study the control of a linear plant using a fuzzy controller is:

    a. Improved performance can usually be obtained by controlling a linear plant with a non-linear controller, and a FL controller is non-linear
    b. Systems that use a fuzzy logic controller are guaranteed to be stable and robust
    c. They are very simple to design

4. Sliding-model control can be applied in the presence of non-linear plant-model uncertainties and plant-parameter disturbances, provided that the uncertainties and disturbances are:

    a. uncorrelated
    b. unknown
    c. known
    d. stationary

5. The main advantage to using a fuzzy model of the plant as well as a fuzzy controller is:

    a. The plant model and controller are linear
    b. Although the plant model and controller are non-linear, the control law can be designed locally, and for each local region the plant model is linear and the controller is also linear
    c. Although the plant model and controller are non-linear, the control law can be designed locally, and for each local region the plant model and the controller are non-linear

Lesson 14–COMPUTATION

Learning Objectives

This lesson focuses on *computation*, both for implementing a type-1 FLS during its operation and for the design of the FLS. The purposes of this lesson are to enumerate all computations for singleton and non-singleton type-1 Mamdani FLSs and for a singleton Type-1 TSK FLS, and to overview on-line software that is available for these computations. This lesson will let you see the forest from the trees (so-to-speak). After completing this lesson you will be able to:

- Describe the nature of and the order of all computations needed to *implement* the three type-1 FLSs studied in this course.
- Describe the nature of and the order of all computations needed to *design* the three type-1 FLSs studied in this course.
- Explain what *software* is available to implement and design the three type-1 FLSs studied in this course.

Reading Assignment

All of the reading material for this lesson is in this Study Guide.

I. Implementation of Type-1 Mamdani FLSs

In this section we collect all of the equations that are needed to implement singleton and non-singleton type-1 Mamdani FLSs. These equations require the designer to make many choices (see Figure 5–9) and will change if the choices are different from the ones we make.

I.A Singleton type-1 Mamdani FLS

General equations for inference engine [see (5–10)]:

$$\mu_{B^l}(y) = \mu_{G^l}(y) \star \left[\mu_{F_1^l}(x_1') \star \cdots \star \mu_{F_p^l}(x_p') \right], \quad y \quad Y \tag{1}$$

Input–output equation for the FLS: This requires specific choices to be made, e.g. max-product composition and product implication [which together mean we use the product t-norm in (1)], and height defuzzification, so that [see (5–24) and (5–25)]:

$$y(\mathbf{x}) = f_s(\mathbf{x}) = \sum_{l=1}^{M} \bar{y}^l \phi_l(\mathbf{x}) \tag{2}$$

$$\phi_l(\mathbf{x}) = \frac{\prod_{i=1}^{p} \mu_{F_i^l}(x_i)}{\sum_{l=1}^{M} \prod_{i=1}^{p} \mu_{F_i^l}(x_i)} \qquad l = 1,\ldots,M \tag{3}$$

Final implementation of input–output equation for the FLS: This requires choices to be made about the MFs, e.g. Gaussian antecedent MFs [see (5–33)]

$$\mu_{F_i^l}(x_i) = \exp\left[-\tfrac{1}{2}\left(\frac{x_i - m_{F_i^l}}{\sigma_{F_i^l}}\right)^2\right] \qquad i = 1,...,p \text{ and } l = 1,...,M \tag{4}$$

Equations (2)–(4) implement a singleton type-1 Mamdani FLS.

I.B Non-Singleton type-1 Mamdani FLS

General equations for inference engine [although Equations (5) and (6) do not appear in the textbook, they are an explicit restatement of (6–2) and the sentence in which it is embedded]:

$$\mu_{Q_k^l}(x_{k,\max}^l) = \sup_{x_k \quad X_k} \mu_{X_k}(x_k) \bigstar \mu_{F_k^l}(x_k) \tag{5}$$

where

$$x_{k,\max}^l = \arg\left[\sup_{x_k \quad X_k} \mu_{X_k}(x_k) \bigstar \mu_{F_k^l}(x_k)\right] \tag{6}$$

so that [see (6–3)]

$$\mu_{B^l}(y) = \mu_{G^l}(y) \bigstar \left[T_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l)\right] \tag{7}$$

Input–output equation for the FLS: This requires specific choices to be made, e.g. max-product composition and product implication [which together mean we use the product t-norm in (7)], and height defuzzification, so that [see (6–17) and (6–18)]:

$$y(\mathbf{x}) = f_{ns}(\mathbf{x}) = \sum_{l=1}^{M} \bar{y}^l \phi_l(\mathbf{x}) \tag{8}$$

$$\phi_l(\mathbf{x}) = \frac{\prod_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l)}{\sum_{l=1}^{M} \prod_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l)} \tag{9}$$

Final implementation of input–output equation for the FLS: This requires choices to be made about the MFs, e.g. Gaussian antecedent MFs and Gaussian input MFs [see (6–24) and (6-25)]

$$\mu_{F_i^l}(x_i) = \exp\left[-\tfrac{1}{2}\left(\frac{x_i - m_{F_i^l}}{\sigma_{F_i^l}}\right)^2\right] \qquad i = 1,...,p \text{ and } l = 1,...,M \tag{10}$$

$$\mu_{X_k}(x_k) = \exp\left[-\frac{1}{2}\left(\frac{x_k - m_{X_k}}{\sigma_{X_k}}\right)^2\right] \qquad k = 1,\ldots,p \tag{11}$$

so that [see (6–7) and (6–8)]

$$x_{k,\max}^l = \frac{\sigma_{X_k}^2 m_{F_k^l} + \sigma_{F_k^l}^2 m_{X_k}}{\sigma_{X_k}^2 + \sigma_{F_k^l}^2} \tag{12}$$

$$\mu_{Q_k^l}(x_{k,\max}^l) = \exp\left[-\frac{1}{2}\frac{\left(m_{X_k} - m_{F_k^l}\right)^2}{\sigma_{X_k}^2 + \sigma_{F_k^l}^2}\right] \tag{13}$$

Equations (8), (9) and (13) implement a non-singleton type-1 Mamdani FLS.

II. Implementation of Type-1 TSK FLSs

In this section we collect all of the equations that are needed to implement singleton normalized and unnormalized type-1 TSK FLSs. These equations also require the designer to make many choices and will change if the choices are different from the ones we make.

II.A First-order normalized type-1 TSK FLS

General equations [see (13–2) and (13–3)]: Using product t-norm,

$$y_{TSK,1}(\mathbf{x}) = \frac{\sum_{i=1}^{M} f^i(\mathbf{x})\left(c_0^i + c_1^i x_1 + c_2^i x_2 + \cdots + c_p^i x_p\right)}{\sum_{i=1}^{M} f^i(\mathbf{x})} \tag{14}$$

$$f^i(\mathbf{x}) = T_{k=1}^{p} \mu_{F_k^i}(x_k) \tag{15}$$

Final implementation of input–output equation for the normalized TSK FLS: This requires choices to be made about the MFs, e.g. Gaussian antecedent MFs [see (13–6)]

$$\mu_{F_i^l}(x_i) = \exp\left[-\frac{1}{2}\left(\frac{x_i - m_{F_i^l}}{\sigma_{F_i^l}}\right)^2\right] \qquad i = 1,\ldots,p \text{ and } l = 1,\ldots,M \tag{16}$$

Equations (14)–(16) implement a first-order normalized type-1 TSK FLS.

II.B First-order unnormalized type-1 TSK FLS

General equations [see (13-2)–(13-4)]: Using product t-norm,

$$y_{TSK,1}(\mathbf{x}) = \sum_{i=1}^{M} f^i(\mathbf{x}) y^i(\mathbf{x}) = \sum_{i=1}^{M} f^i(\mathbf{x}) \left(c_0^i + c_1^i x_1 + c_2^i x_2 + \cdots + c_p^i x_p \right) \tag{17}$$

$$f^i(\mathbf{x}) = T_{k=1}^{p} \mu_{F_k^i}(x_k) \tag{18}$$

Final implementation of input–output equation for the unnormalized TSK FLS: This requires choices to be made about the MFs, e.g. Gaussian antecedent MFs [see (13–6)]

$$\mu_{F_i^l}(x_i) = \exp\left[-\frac{1}{2} \left(\frac{x_i - m_{F_i^l}}{\sigma_{F_i^l}} \right)^2 \right] \quad i = 1,...,p \text{ and } l = 1,...,M \tag{19}$$

Equations (17)–(19) implement a first-order unnormalized type-1 TSK FLS.

III. Designs of Mamdani FLSs Using a Back Propagation (Steepest Descent) Design Procedure

In this section we collect all of the equations that are needed to design singleton and non-singleton type-1 Mamdani FLSs using the back-propagation (steepest descent) method. These equations require the designer to make many choices (see Figure 5–9) and will change if the choices are different from the ones we make.

III.A Singleton type-1 Mamdani FLS

The key design equations are described in Section 5.9.3 [see (5–48)–(5–50)]:

$$m_{F_k^l}(i+1) = m_{F_k^l}(i) - \alpha_m [f_s(\mathbf{x}^{(i)}) - y^{(i)}] \left[\bar{y}^l(i) - f_s(\mathbf{x}^{(i)}) \right] \times \frac{\left[x_k^{(i)} - m_{F_k^l}(i) \right]}{\sigma_{F_k^l}^2(i)} \phi_l(\mathbf{x}^{(i)}) \tag{20}$$

$$\bar{y}^l(i+1) = \bar{y}^l(i) - \alpha_{\bar{y}} [f_s(\mathbf{x}^{(i)}) - y^{(i)}] \phi_l(\mathbf{x}^{(i)}) \tag{21}$$

$$\sigma_{F_k^l}(i+1) = \sigma_{F_k^l}(i) - \alpha_\sigma [f_s(\mathbf{x}^{(i)}) - y^{(i)}] \left[\bar{y}^l(i) - f_s(\mathbf{x}^{(i)}) \right] \times \frac{\left[x_k^{(i)} - m_{F_k^l}(i) \right]^2}{\sigma_{F_k^l}^3(i)} \phi_l(\mathbf{x}^{(i)}) \tag{22}$$

where $\phi_l(\mathbf{x}^{(i)})$ and $f_s(\mathbf{x}^{(i)})$ are computed using (2)–(4) in which $\bar{y}^l = \bar{y}^l(i)$, $m_{F_k^l} = m_{F_k^l}(i)$ and $\sigma_{F_k^l} = \sigma_{F_k^l}(i)$.

III.B Non-singleton type-1 Mamdani FLS

The key design equations are described in Section 6.6.3 [see (6–30)–(6–33)]:

$$m_{F_k^l}(i+1) = m_{F_k^l}(i) - \alpha_m [f_{ns}(\mathbf{x}^{(i)}) - y^{(i)}]\left[\bar{y}^l(i) - f_{ns}(\mathbf{x}^{(i)})\right] \times \left[\frac{x_k^{(i)} - m_{F_k^l}(i)}{\sigma_X^2(i) + \sigma_{F_k^l}^2(i)}\right] \phi_l(\mathbf{x}^{(i)}) \tag{23}$$

$$\bar{y}^l(i+1) = \bar{y}^l(i) - \alpha_{\bar{y}} [f_{ns}(\mathbf{x}^{(i)}) - y^{(i)}]\phi_l(\mathbf{x}^{(i)}) \tag{24}$$

$$\sigma_{F_k^l}(i+1) = \sigma_{F_k^l}(i) - \alpha_\sigma [f_{ns}(\mathbf{x}^{(i)}) - y^{(i)}]\left[\bar{y}^l(i) - f_{ns}(\mathbf{x}^{(i)})\right] \times \sigma_{F_k^l}(i)\left[\frac{x_k^{(i)} - m_{F_k^l}(i)}{\sigma_X^2(i) + \sigma_{F_k^l}^2(i)}\right]^2 \phi_l(\mathbf{x}^{(i)}) \tag{25}$$

$$\sigma_X(i+1) = \sigma_X(i) - \alpha_X [f_{ns}(\mathbf{x}^{(i)}) - y^{(i)}]\left[\bar{y}^l(i) - f_{ns}(\mathbf{x}^{(i)})\right] \times \sigma_X(i)\left[\frac{x_k^{(i)} - m_{F_k^l}(i)}{\sigma_X^2(i) + \sigma_{F_k^l}^2(i)}\right]^2 \phi_l(\mathbf{x}^{(i)}) \tag{26}$$

where $\phi_l(\mathbf{x}^{(i)})$ and $f_{ns}(\mathbf{x}^{(i)})$ are computed using (8), (9) and (13) in which $\bar{y}^l = \bar{y}^l(i)$, $m_{F_k^l} = m_{F_k^l}(i), \sigma_{F_k^l} = \sigma_{F_k^l}(i)$ and $\sigma_X = \sigma_X(i)$.

IV. Designs of TSK FLSs Using a Back Propagation (Steepest Descent) Design Procedure

In this section we collect all of the equations that are needed to design singleton normalized and unnormalized type-1 TSK FLSs using the back-propagation (steepest descent) method. These equations also require the designer to make many choices and will change if the choices are different from the ones we make.

IV.A First-order normalized type-1 TSK FLSs

The key design equations have been worked out by you in Lesson 12, Exercise 13–1 [see (5) and (15) in the solution to Exercise 13–1]:

$$c_j^i(n+1) = c_j^i(n) - \alpha_c \left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right] \times g_j^i\left(\mathbf{x}^{(t)}\right) \tag{27}$$

$$m_{F_k^i}(n+1) = m_{F_k^i}(n) - \alpha_m \left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right] \times \left[\sum_{j=0}^{p} x_j^{(t)} c_j^i(n) - y_{TSK,1}\left(\mathbf{x}^{(t)}\right)\right] \times \frac{\left[x_k^{(t)} - m_{F_k^i}(n)\right]}{\sigma_{F_k^i}^2(n)} \times \frac{w^i(n)}{g(n)} \tag{28}$$

where

$$w^i(n) = \prod_{k=1}^{p} \exp\left[-\frac{1}{2}\frac{\left(x_k^{(t)} - m_{F_k^i}(n)\right)^2}{\sigma^2_{F_k^i}(n)}\right] \tag{29}$$

$$g(n) = \sum_{i=1}^{M} w^i(n) \tag{30}$$

and $y_{TSK,1}\left(\mathbf{x}^{(t)}\right)$ is computed using (14)–(16) in which $c_j^i = c_j^i(n)$, $m_{F_k^l} = m_{F_k^l}(n)$ and $\sigma_{F_k^l} = \sigma_{F_k^l}(n)$. We have not included the equation for $\sigma_{F_k^l}(n+1)$, leaving it to for you to derive.

IV.B First-order unnormalized type-1 TSK FLSs

We leave it for you to derive the steepest descent formulas for $c_j^i(n+1)$, $m_{F_k^i}(n+1)$ and $\sigma_{F_k^l}(n+1)$.

V. M-Files for Type-1 FLSs

Although no MATLAB M-files are packaged with the textbook, eight are available for type-1 FLSs as freeware on the Internet at the following URL: **http://sipi.usc.edu/~mendel/software**. Brief descriptions of the M-files appear at the end of each chapter for which the M-file is most applicable. In this section we collect all of the type-1 FLS M-files together as they are organized on the Internet in the folder *type-1 fuzzy logic systems*.

V.A Singleton Mamdani Type-1 FLS

sfls_type1.m: Compute the output(s) of a singleton type-1 FLS when the antecedent membership functions are Gaussian.

train_sfls_type1.m: Tune the parameters of a singleton type-1 FLS when the antecedent membership functions are Gaussian, using some input–output training data.

svd_qr_sfls_type1.m: Rule-reduction of a singleton type-1 FLS when the antecedent membership functions are Gaussian, using some input–output training data.

V.B Non-Singleton Mamdani Type-1 FLS

nsfls_type1.m: Compute the output(s) of a non-singleton type-1 FLS when the antecedent membership functions are Gaussian and the input sets are Gaussian.

train_nsfls_type1.m: Tune the parameters of a non-singleton type-1 FLS when the antecedent membership functions are Gaussian, and the input sets are Gaussian, using some input–output training data.

svd_qr_nsfls_type1.m: Rule-reduction of a non-singleton type-1 FLS when the antecedent membership functions are Gaussian, and the input sets are Gaussian, using some input–output training data.

V.C Normalized TSK FLS

tsk_type1.m: Compute the output(s) of a type-1 TSK FLS (type-1 antecedents and type-0 consequent) when the antecedent membership functions are Gaussian.

train_tsk_type1.m: Tune the parameters of a type-1 TSK FLS (type-1 antecedents and type-0 consequent) when the antecedent membership functions are Gaussian, using some input–output training data.

V.D Unnormalized TSK FLS

Although no M-files are available for unnormalized TSK FLSs, they can easily be constructed using the structure of the M-files that are available for a normalized TSK FLS.

Key Points

- The equations needed to implement singleton type-1 and non-singleton type-1 Mamdani FLSs and singleton normalized and unnormalized type-1 TSK FLSs have been collected in one place.
- The equations needed to design [using the back-propagation (steepest descent) design method] singleton type-1 and non-singleton type-1 Mamdani FLSs and singleton normalized and unnormalized type-1 TSK FLSs have been collected in one place. These equations also make use of the ones for implementing their respective type-1 FLSs.
- On-line (free) software for implementation and design of FLSs—singleton type-1 and non-singleton type-1 Mamdani FLSs and normalized type-1 TSK FLSs—are available on the Internet at: http://sipi.usc.edu/~mendel/software.

Practice Problems

Exercise SG 14-1

Suppose that the t-norm used for implementation of a singleton type-1 Mamdani FLS is the minimum. How do the *implementation* equations change for that FLS?

Exercise SG 14-2

Suppose that the t-norm used for implementation of a non-singleton type-1 Mamdani FLS is the minimum. How do the *implementation* equations change for that FLS?

Exercise SG 14-3

Suppose that the t-norm used for implementation of a singleton normalized type-1 TSK FLS is the minimum. How do the *implementation* equations change for that FLS?

Lesson 15–OPEN ISSUES WITH TYPE-1 FLSs

Learning Objectives

This lesson focuses on the shortcomings of type-1 FLSs and how they can be overcome. The main purposes of this lesson are to introduce you to different kinds of uncertainties that can occur in a type-1 FLS, to explain where such uncertainties can occur in the applications studied in this course, why they can not be handled by a type-1 FLS, and what can be done about this situation. After completing this lesson you will be able to:

- Describe three *general* types of uncertainty.
- Describe the four kinds of uncertainties that can occur in a rule-based FLS.
- Demonstrate what the phrase "Words mean different things to different people" means.
- Describe where the four kinds of uncertainties can occur in time-series forecasting, knowledge mining from surveys, rule-based pattern classification, equalization of time-varying non-linear digital communication channels, and fuzzy logic control.
- Describe why a type-1 FLS can not handle (i.e., directly model and minimize the effects of) the uncertainties.
- Explain why an expanded and richer FL is needed to do this, and that it already exists and has led to *type-2 FLSs*.

Reading Assignment

Read pages 66-78 of the textbook. Then read the following new material.

I. Uncertainties in Our Applications

Here I explain where uncertainties can occur in the applications that have been included as part of this course.

I.A Forecasting of time series (Lesson 6)

Since rules are extracted from numerical data, if the data are corrupted by additive noise then the rule antecedents and the rule consequent are uncertain. Uncertainty also affects the tuning of the FLS parameters because noisy measurements are used. Finally, if only noisy measurements are available to activate the FLS, then uncertainty also affects the inputs to the FLS. In this application, all four sources of uncertainty that are listed in the first paragraph on p.68 of the textbook can be present.

I.B Knowledge mining using surveys (Lesson 6)

In Chapter 2, we saw that words can mean different things to different people, so rule antecedents are uncertain because they use words. Surveys collected from a group of experts lead to a histogram for the consequent of each rule; hence, there is uncertainty about a rule's consequent. There is no tuning of a FLA, so this kind of uncertainty is not present in a FLA. Activating a FLA can be done using words. In this case, there is the usual uncertainty about

words associated with this activation. In this application, only three kinds of uncertainty are present because data is not used to tune the FLA.

I.C Rule-based classification of video traffic (Lesson 13)
Example 13-5 in the textbook demonstrates that the logarithm of I, P, or B frame sizes are more appropriately modeled as Gaussians each of whose mean is a constant, but whose standard deviation varies. This suggests that we should use a Gaussian MF with a fixed mean and an uncertain standard deviation to model each frame of the compressed video. Hence, rule antecedents are uncertain. Rule consequents in a RBC are certain because they correspond to a class (e.g. ±1, movies or sports programs). If the parameters of a FL RBC are tuned using a training sample, then the just-described uncertainties also affect the tuning. Measurements that activate the FL RBC will also be uncertain because they are logarithms of I, P, or B frame sizes as computed over a window of measurements. In this application, only three sources of uncertainty will be present, because the uncertainty about "words" does not affect a rule's consequent.

I.D Equalization of Time-Invariant Non-linear Digital Communication Channels
(Lesson 13)
Because equalization using a rule-based FLS—a FAF—is equivalent to rule-based classification, there can be three sources of uncertainty present, namely: uncertainty about a rule's antecedents (but not about a rule's consequent), uncertainty about the data used to tune the parameters of the FAF, and uncertainty about the measurements used to activate the FAF. When measurements are very accurate, then all of these uncertainties disappear. However, if the communication system is in a time-varying environment (e.g., as in mobile communications), then channel coefficients will be time-varying, and rule antecedents become uncertain (e.g., see Example 14-2).

I.E Fuzzy Logic Control
Because of the vast scope of FL control, and our very brief coverage of FL control in Lesson 13, we can only provide a very cursory discussion here about where uncertainties can occur in FL control. To be as specific as possible, we focus first on non-adaptive fuzzy control in which a fuzzy model is used for both the plant as well as the controller [see Equations (1) and (2) in Lesson 13]. Uncertainty can occur in rule antecedents, and may also be present in each rule's consequent if only noisy measurements are available, or if the control cannot be implemented perfectly. If control parameters are tuned during an off-line design phase using noisy data, then that kind of uncertainty is also present. It would seem, therefore, that all four sources of uncertainty could be present in this kind of non-adaptive fuzzy controller problem.

Next, focus on indirect adaptive fuzzy control, as described by Equations (3)–(7) in Lesson 13. The fuzzy system models for $\hat{\mathbf{f}}$ and $\hat{\mathbf{g}}$ will use IF-THEN rules. If they are Mamdani rules, then uncertainties may be present in both antecedent and consequent words. If they use TSK rules, then uncertainties may be present just in the antecedent words. Uncertain antecedents or consequents can be used to model the lack of knowledge about the true non-linearities, $\mathbf{f}$ and $\mathbf{g}$. Observe, in the adaptation laws (6) and (7), that fuzzy system parameters are updated using measurements, so if only noisy measurements are available to do this, then data uncertainties will also be present. It would seem, therefore, that all four sources of uncertainty can also be present in this kind of adaptive fuzzy controller problem.

II. Why Type-1 FLSs Cannot Handle Uncertainties

The original FL, founded by Lotfi Zadeh, has been around for more than 35 years, as of the year 2000, and yet it is unable to handle uncertainties. By *handle*, I mean *to model and minimize the effect of*. That the original FL—type-1 FL—cannot handle uncertainties sounds paradoxical because the word *fuzzy* has the connotation of uncertainty. Type-1 FL handles uncertainties by using *precise* membership functions (MFs) that the user believes capture the uncertainties. Once the type-1 MFs are chosen, all uncertainty disappears, because type-1 MFs are totally precise.

III. An Expanded and Richer FL

An expanded FL—type-2 FL—is able to handle uncertainties because it can model them and minimize their effects. And, if all uncertainties disappear, type-2 FL reduces to type-1 FL, in much the same way that if randomness disappears, probability reduces to determinism.

As you now know, FL is all about IF-THEN rules in which *antecedents* and *consequent* are modeled as *fuzzy sets*. And, rules are described by the MFs of these fuzzy sets. In type-1 FL, the antecedents and consequent are all described by the MFs of type-1 fuzzy sets. In type-2 FL, some or all of the antecedents and consequent are described by the MFs of type-2 fuzzy sets.

Good news, the rules do not change as we go from a type-1 to a type-2 FLS. Paraphrasing Gertrude Stein, "A rule is a rule is a rule …." What does change is the way in which we model a rule's antecedent and consequent fuzzy sets. In type-1 FL, they are all modeled as type-1 fuzzy sets, whereas in type-2 FL some or all are modeled as type-2 fuzzy sets.

The term "fuzzy set" is general and includes type-1 and type-2 fuzzy sets (and even higher-type fuzzy sets). All fuzzy sets are characterized by MFs. A "type-1 fuzzy set" is characterized by a two-dimensional MF, whereas a "type-2 fuzzy set" is characterized by a *three-dimensional* MF.

As an example, suppose the variable of interest is *eye contact*, which we denote as x. Let's put *eye contact* on a scale of values 0–10. One of the terms that might characterize the amount of perceived *eye contact* (e.g. during flirtation) is "some eye contact." Suppose that we surveyed 100 men and women, and asked them to locate the ends of an interval for *some eye contact* on the scale 0–10. In Chapter 2 of the textbook, we have already seen that we do not get the same interval end-points from all of them, because *words mean different things to different people*.

One approach to using the 100 sets of two end-points is to average the end-point data and to use the average values for the interval associated with *some eye contact*. We could then construct a triangular (other shapes could be used) MF whose base end-points (on the x-axis) are at the two average values and whose apex is midway between the two end-points. This type-1 triangle MF can be displayed in two-dimensions and is expressed mathematically as $\mu_F(x), x \quad X$. Unfortunately, this MF has completely ignored the uncertainties associated with the two end-points.

A second approach is to make use of the average values *and* the standard deviations for the two end-points. By doing this we are blurring the location of the two end-points along the x-axis.

Now locate triangles so that their base end-points can be anywhere in the intervals along the *x*-axis associated with the blurred average end-points. Doing this leads to a continuum of triangular MFs sitting on the *x*-axis, e.g. picture a whole bunch of triangles all having the same apex point but different base points, as in Figure 1.

For purposes of this discussion, suppose there are exactly *N* such triangles. Then at each value of *x*, there can be up to *N* MF values, $MF_1(x)$, $MF_2(x)$,…, $MF_N(x)$. Let's assign a weight to each of the possible MF values, say w_{x1}, w_{x2},…, w_{xN} (see the insert on Figure 1). We can think of these weights as the *possibilities* associated with each triangle at this value of *x*. The resulting type-2 MF can be expressed mathematically as

$$\{(x, \{(\, MF_i(x), w_{xi})|\ i = 1, \ldots, N\}|\ x \text{ an element of } X\}$$

Another way to write this is:

$$\{(x, MF(x, w)|\ x \text{ an element of } X \text{ and } w \text{ an element of } J_x\}$$

MF(*x*, *w*) is a type-2 MF. It is *three-dimensional* because MF(*x*, *w*) depends on two variables, *x* and *w*.

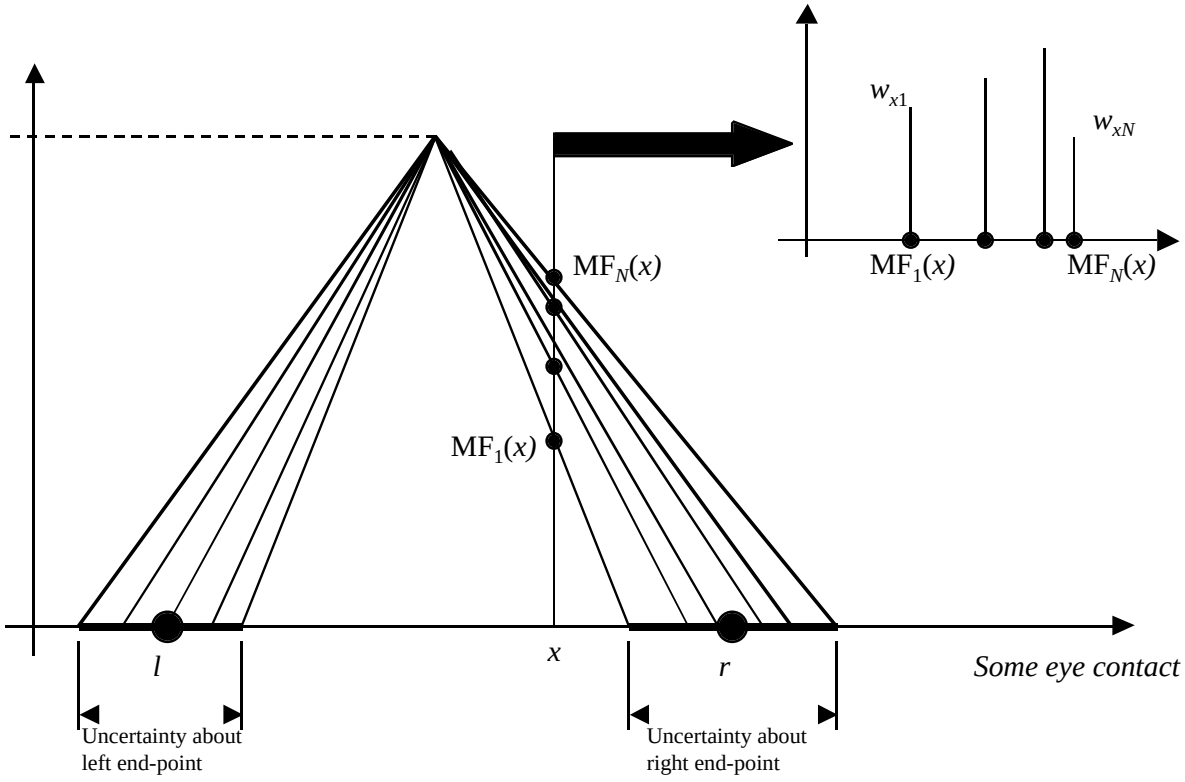


Figure 1: Triangular MFs when base end-points (*l* and *r*) have uncertainty intervals associated with them.

A type-1 FLS only uses type-1 fuzzy sets whereas a type-2 FLS uses at least one type-2 fuzzy set. The diagram for a type-2 FLS is the same as for a type-1 FLS (see Figure 1-1 in the

textbook). The inference engine of a type-1 FLS maps type-1 input fuzzy sets into type-1 output fuzzy sets, whereas the inference engine of a type-2 FLS maps type-2 and/or type-1 fuzzy sets into type-2 fuzzy sets. The output processor for a type-1 FLS transforms a type-1 fuzzy set into a number (i.e. a type-0 fuzzy set), and is the familiar *defuzzifier*. The output processor for a type-2 FLS has two components to it: (1) a *type-reducer* that transforms a type-2 fuzzy set into a type-1 fuzzy set (a two-dimensional *type- reduced set*), followed by (2) a defuzzifier that transforms the resulting type-1 fuzzy set into a number. A type-reduced set is like a confidence interval. The more uncertainty that is present, then the larger is the type-reduced set, and vice-versa.

Type-2 FLSs have been developed that satisfy the following fundamental design requirement: *when all sources of uncertainty disappear, a type-2 FLS must reduce to a comparable type-1 FLS*. This design requirement is analogous to what happens to a probability density function when random uncertainties disappear. In that case, the variance of the pdf goes to zero, and a probability analysis reduces to a deterministic analysis. So, just as the capability for a deterministic analysis is embedded within a probability analysis, the capability for a type-1 FLS is embedded within a type-2 FLS.

Type-2 FLSs are described by type-2 membership functions (MFs) that are characterized by more parameters than are MFs for type-1 FLSs. During the designs of type-1 and type-2 FLSs, MF parameters are optimized using some training data. Because type-2 FLSs are characterized by more design parameters than are type-1 FLSs (i.e., they have more design degrees of freedom), type-2 FLSs have the potential to outperform type-1 FLSs.

Of course, one way to introduce more design degrees of freedom into a type-1 FLS is to add more rules to it. Unfortunately, additional rules do not let a type-1 FLS account for uncertainties, because uncertainties cannot be modeled by type-1 fuzzy sets. And, in all fairness, the additional rules should also be provided to the type-2 FLS, especially if we require that a type-2 FLS must reduce to a type-1 FLS when all sources of uncertainty disappear.

Some specific situations where we have found that type-2 FLSs outperform type-1 FLSs are: (1) *Measurement noise is non-stationary*, but the nature of the non-stationarity cannot be expressed ahead of time mathematically (e.g. variable SNR measurements); (2) A *data-generating mechanism is time-varying*, but the nature of the time-variations cannot be expressed ahead of time mathematically (e.g. equalization of non-linear and time-varying digital communication channels); (3) *Features are described by statistical attributes that are non-stationary*, but the nature of the non-stationarity cannot be expressed ahead of time mathematically (e.g. rule-based classification of video traffic); and, (4) *Knowledge is mined* from experts using IF–THEN questionnaires (e.g. connection admission control for ATM networks).

Type-2 fuzzy sets and FLSs are covered in the textbook that came with this course and are the subject of the follow-on (to this course) IEEE Self-Study Course *New Directions in Rule-Based Fuzzy Logic Systems: Handling Uncertainties*.

Key Points

- There are three *general* types of uncertainties—*fuzziness*, *non-specificity*, and *strife*. Fuzziness results from the imprecise boundaries of fuzzy sets; nonspecificity is connected with the sizes of relevant sets of alternatives; and strife expresses conflicts among the various sets of alternatives.
- There are four sources of uncertainty that can occur in a FLS: meanings of words used in rules, (histograms of) consequents that occur in rules, measurements that activate rules, and data that are used to tune the parameters of a FLS.
- Surveys demonstrate that there is uncertainty associated with intervals used to describe words/phrases; hence, *words mean different things to different people*.
- Rule reduction can be achieved by including uncertainties about words.
- Some or all of the four sources of uncertainty can occur in all of the applications studied in this course.
- Type-1 fuzzy sets can not handle uncertainties because they cannot directly model them.
- Type-2 fuzzy sets and FLSs can handle the four kinds of uncertainties because they can model them and minimize their effects.
- Type-2 fuzzy sets are described by three-dimensional MFs, whereas type-1 fuzzy sets are described by two-dimensional MFs. It is the new third dimension of type-2 fuzzy sets that provides them with the ability to handle the uncertainties.

Solutions

Lesson 2: Practice Problem Solutions

Exercise 1-2

(a) Real numbers close to 10

Examples of formulas for Gaussian, triangular, and even an unnamed MF are given on p.188 of the textbook. Any one of these could be used. One could also use a trapezoidal MF. How to choose the *width* of these MFs is unclear because the word *close* can mean different things to different people. If I were to choose a Gaussian MF, then I'd use $\sigma = 1$ so that

$$\mu_{close\ to\ 10}(x) = \exp\left[-\frac{(x-10)^2}{2}\right] \quad x \quad R$$

(b) Real numbers approximately equal to 6

Again, examples of formulas for Gaussian, triangular, and an unnamed MF are given on p.188 of the textbook, and, any one of these could be used (as could a trapezoidal MF). And, as in (a), how to choose the *width* of these MFs is unclear because *approximately equal to* can mean different things to different people. To me, what is clear is that *approximately equal to* should be associated with a MF that is much narrower than the MF for *close to*. So, if I were to choose a Gaussian MF, then I'd use $\sigma << 1$ (e.g., $\sigma = 0.05$) so that

$$\mu_{approximately\ \ equal\ to\ 6}(x) = \exp\left[-\frac{(x-6)^2}{0.005}\right] \quad x \quad R$$

(c) Integers very far from 10

One choice for $\mu_{very\ far\ from\ 10}(x)$ might be:

$$\mu_{very\ far\ from\ 10}(x) = 0.1/\ 50 + 0.1/51 + \cdots + 0.2/60 + \cdots + 0.3/70 + \cdots$$
$$+ 0.4/80 + \cdots + 0.5/90 + \cdots + 1/200 + \cdots$$

This choice for the MF points out a number of interesting points:

1. Again the choice of the MF is not unique, because of our interpretation of the phrase *very far from.*
2. It is difficult to express $\mu_{very\ far\ from\ 10}(x)$ as we have tried to do because integers go on indefinitely. A closed-form formula would be a better representation, e.g.,

$$\mu_{very\ far\ from\ 10}(x) = \begin{cases} a(x) & x < 200 \quad x \quad I \\ 1 & x \geq 200 \quad x \quad I \end{cases}$$

There is no unique choice for $a(x)$ and the choice of 200 is also arbitrary.

(d) Complex numbers near the origin

Let $x = a + jb$ so that $|x| = \sqrt{a^2 + b^2}$. We can interpret *complex numbers near the origin* as those numbers for which $|x|$ is very small. In this case $|x|$ must be a positive real number. By multiplying a MF like any of the three given on p. 188 of the textbook by a unit step function, we can obtain the desired MF, e.g.

$$\mu_{close-to-origin}(|x|) = \left[\exp\left(-\frac{x^2}{2\sigma^2}\right)\right] u_{-1}(|x|)$$

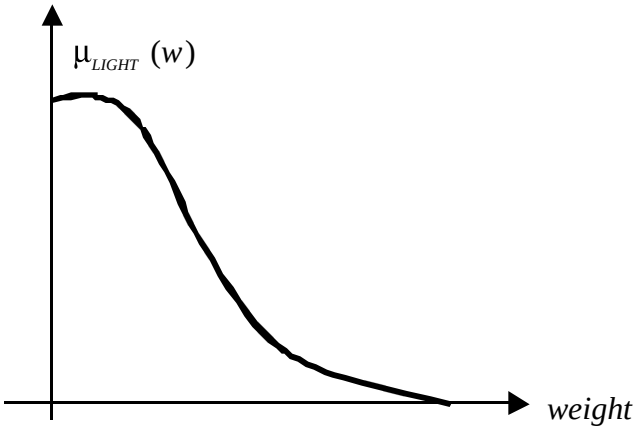
where σ <<.

(e) light (weight)

The MF for *light (weight)*, $\mu_{LIGHT}(w)$, can look like the one shown below. Any number of mathematical functions can be used to represent this MF, e.g.

$$\mu_{LIGHT}(w) = \frac{2e^{-aw}}{1 + e^{-aw}} \qquad w \geq 0$$

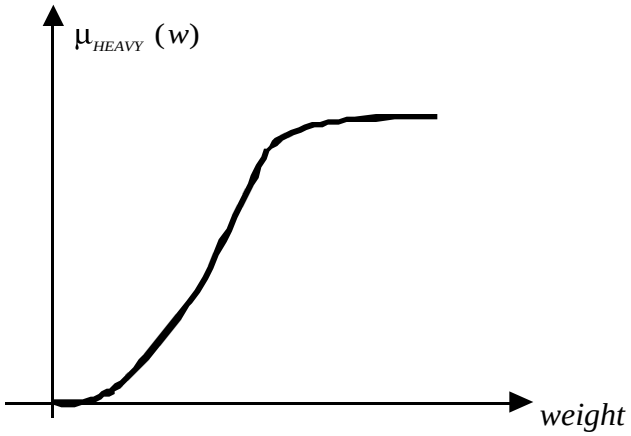
that is related to the sigmoidal function $(1 - e^{-aw})/(1 + e^{-aw})$ (a shifted version of which is widely used in neural networks). A more general s-curve that can be used for $\mu_{LIGHT}(w)$ is given in Cox (1994, pp. 51–53).



(f) heavy (weight)

The MF for *heavy (weight)*, $\mu_{HEAVY}(w)$, can look like the one shown below. As in (e), any number of mathematical functions can be used to represent this MF, e.g. the sigmoidal function

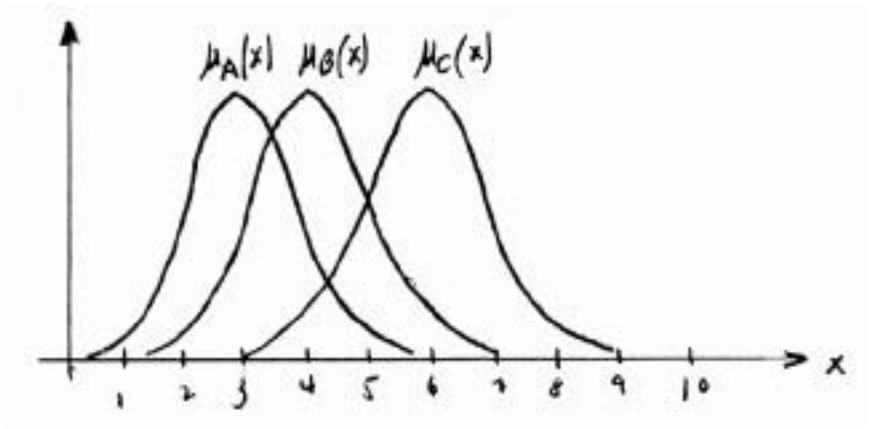
$$\mu_{HEAVY}(w) = \frac{1 - e^{-aw}}{1 + e^{-aw}} \qquad w \geq 0$$


μ_{HEAVY} (*w*)
weight

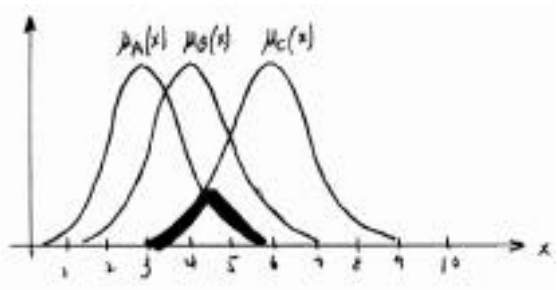
Lesson 3: Practice Problem Solutions

Exercise 1-9 The solution is given in the following table.

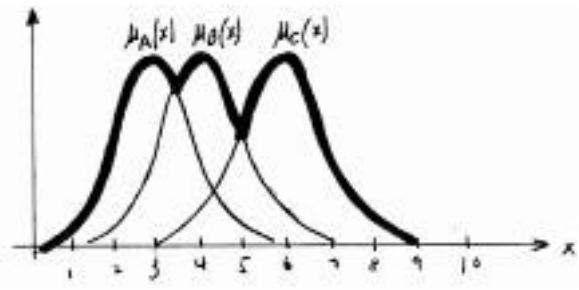
Three membership functions



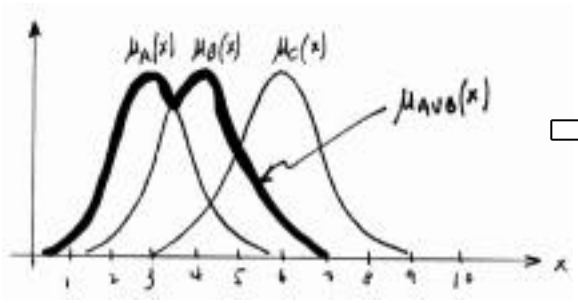
(a) $\mu_{A \cap B \cap C}(x)$



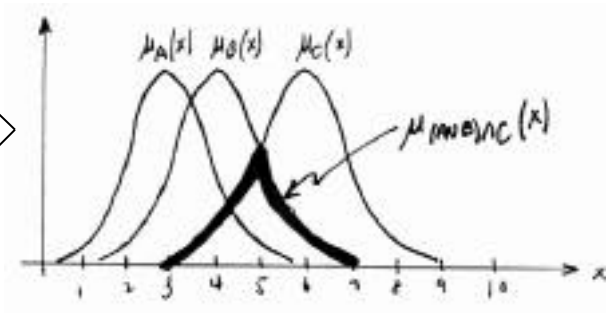
(b) $\mu_{A \quad B \quad C}(x)$



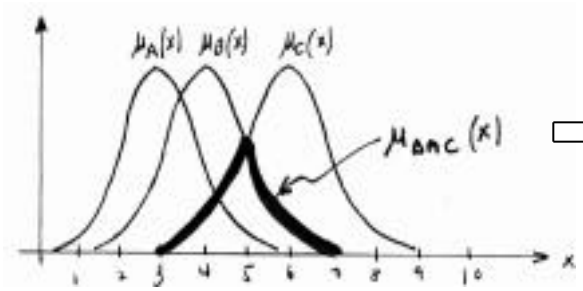
(c-1) $\mu_{(A \quad B) \cap C}(x)$



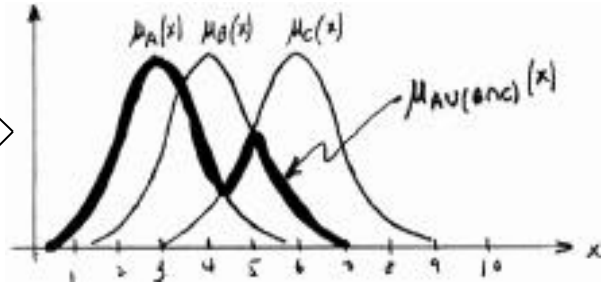
(c-1) $\mu_{(A \quad B) \cap C}(x)$



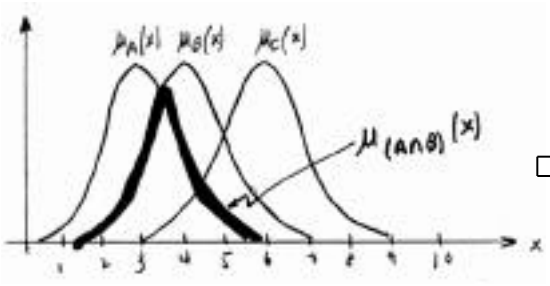
(c-2) $\mu_{A\ (B\cap C)}(x)$



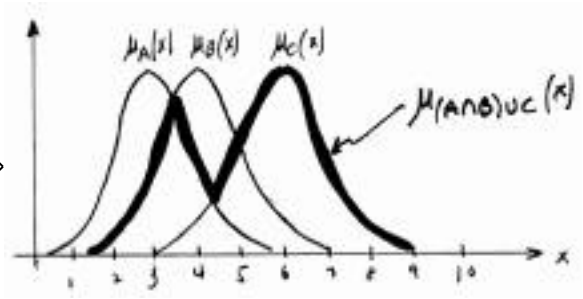
(c-2) $\mu_{A\ (B\cap C)}(x)$



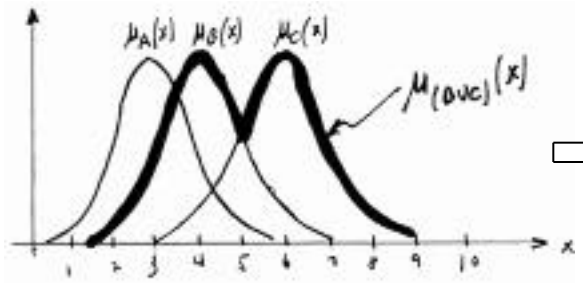
(d-1) $\mu_{(A\cap B)\ C}(x)$



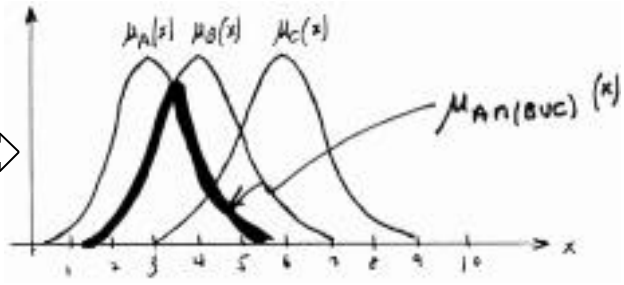
(d-1) $\mu_{(A\cap B)\ C}(x)$



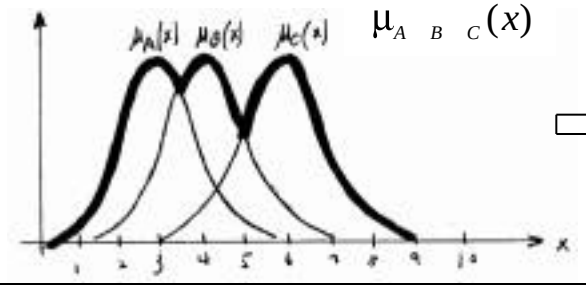
(d-2) $\mu_{A\cap (B\ C)}(x)$



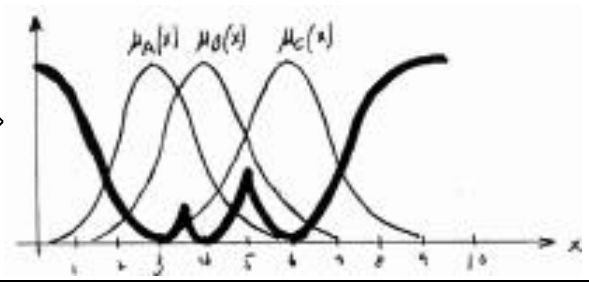
(d-2) $\mu_{A\cap (B\ C)}(x)$



(e) $\mu_{\overline{A\ B\ C}}(x)$



(e) $\mu_{\overline{A\ B\ C}}(x)$



Exercise 1-11

We restate (1-31) using the maximum t-norm:

$$\mu_{c\ s}(u_i, v_j) = \mu_c(u_i, v_j) \quad \mu_s(u_i, v_j) = \max\left[\mu_c(u_i, v_j), \mu_s(u_i, v_j)\right]$$

It follows that:

$$\begin{aligned}
&\mu_{c\ s}(1,1) = \max(0.9,0) = 0.9\\
&\mu_{c\ s}(1,2) = \max(0.4,0.6) = 0.6\\
&\mu_{c\ s}(1,3) = \max(0.1,1) = 1\\
&\mu_{c\ s}(2,1) = \max(0.1,0) = 0.1\\
&\mu_{c\ s}(2,2) = \max(0.4,0) = 0.4\\
&\mu_{c\ s}(2,3) = \max(0.9,0.3) = 0.9
\end{aligned}$$

We restate (1-32) using the minimum t-norm:

$$\mu_{c\cap s}(u_i, v_j) = \mu_c(u_i, v_j) \quad \mu_s(u_i, v_j) = \min\left[\mu_c(u_i, v_j), \mu_s(u_i, v_j)\right]$$

It follows that:

$$\begin{aligned}
&\mu_{c\cap s}(1,1) = \min(0.9,0) = 0\\
&\mu_{c\cap s}(1,2) = \min(0.4,0.6) = 0.4\\
&\mu_{c\cap s}(1,3) = \min(0.1,1) = 0.1\\
&\mu_{c\cap s}(2,1) = \min(0.1,0) = 0\\
&\mu_{c\cap s}(2,2) = \min(0.4,0) = 0\\
&\mu_{c\cap s}(2,3) = \min(0.9,0.3) = 0.3
\end{aligned}$$

Exercise 1-19a

Let *very likely* ≡ *VL*. Then, according to the concept of *concentration*, $\mu_{VL}(x) = \left(\mu_L(x)\right)^2$. We use $\mu_L(x)$ from (1-56) to compute $\mu_{VL}(x)$, i.e.

$$\begin{aligned}
\mu_{VL}(x) = \left(\mu_L(x)\right)^2 = 1/\ &1 + 1\ /0.9 + 1\ /0.8 + 0.64/0.7 + 0.36\ /0.6 + 0.25/0.5\\
&+ 0.09/0.4 + 0.04/0.3
\end{aligned}$$

Observe the higher concentration of $\mu_{VL}(x)$ MF values for high values of probability (x), which seems sensible for the term *very likely*. ■

Lesson 4: Practice Problem Solutions

Exercise 1-16

For completeness, we repeat Equations (1-29) and (1-46) which provide the MFs for $\mu_c(u,v)$ and $\mu_{mb}(v,w)$, respectively:

$$\mu_c(u,v) = \begin{array}{c} \\ u_1 \\ u_2 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{pmatrix} 0.9 & 0.4 & 0.1 \\ 0.1 & 0.4 & 0.9 \end{pmatrix} \end{array} \qquad \mu_{mb}(v,w) = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{cc} w_1 & w_2 \end{array} \\ \begin{pmatrix} 0 & 0 \\ 0.6 & 0 \\ 1 & 0.7 \end{pmatrix} \end{array}$$

In this Exercise, $\equiv$ product and $\equiv$ maximum. The four elements of $\mu_{c\circ mb}(u,v)$ are computed using the max-product composition shortcuts that are described on p. 42, as:

$$\mu_{c\circ mb}(u_1,v_1) = \begin{pmatrix} 0.9 & 0.4 & 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0.6 \\ 1 \end{pmatrix} = 0.9\times 0 + 0.4 \times 0.6 + 0.1 \times 1 = \max(0,0.24,0.1) = 0.24$$

$$\mu_{c\circ mb}(u_1,v_2) = \begin{pmatrix} 0.9 & 0.4 & 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0.7 \end{pmatrix} = \max(0,0,0.07) = 0.07$$

$$\mu_{c\circ mb}(u_2,v_1) = \begin{pmatrix} 0.1 & 0.4 & 0.9 \end{pmatrix} \begin{pmatrix} 0 \\ 0.6 \\ 1 \end{pmatrix} = \max(0,0.24,0.9) = 0.9$$

$$\mu_{c\circ mb}(u_2,v_2) = \begin{pmatrix} 0.1 & 0.4 & 0.9 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0.7 \end{pmatrix} = \max(0,0,0.63) = 0.63$$

so that

$$\mu_{c\circ mb}(u,v) = \begin{pmatrix} 0.24 & 0.07 \\ 0.9 & 0.63 \end{pmatrix} \text{ when } \equiv \text{ product and } \equiv \text{maximum}$$

Next, we compare this result for $\mu_{c\circ mb}(u,v)$ with the result in (1-49), which we repeat here for the convenience of the reader:

$$\mu_{comb}(u,v) = \begin{pmatrix} 0.4 & 0.1 \\ 0.9 & 0.7 \end{pmatrix} \quad \text{when} \quad \equiv \text{minimum and} \quad \equiv \text{maximum}$$

Observe that the two MF matrices are very similar, with the biggest difference between the two occurring in the 1-1 element. ■

Exercise 1-20 (b)

The calculations for $\mu_B(y)$ are given in the following table. We used the Extension Principle that is stated in (1-60).

x	$\mu_A(x)$	$y = \lvert x \rvert$	$\mu_B(y)$
-5	0.2	5	max{0.2, 0.1} = 0.2
-4	0.4	4	max{0.4, 0.5} = 0.5
-3	0.4	3	max{0.4, 0.8} = 0.8
-2	0.5	2	max{0.5, 1} = 1
-1	0.5	1	max{0.5, 0.9} = 0.9
0	0.6	0	max{0.6} = 0.6
1	0.9	1	max{0.5, 0.9} = 0.9
2	1	2	max{0.5, 1} = 1
3	0.8	3	max{0.4, 0.8} = 0.8
4	0.5	4	max{0.4, 0.5} = 0.5
5	0.1	5	max{0.2, 0.1} = 0.2

From the last two columns of this table, we conclude that

$$B = 0.6/0 + 0.9/1 + 1/2 + 0.8/3 + 0.5/4 + 0.2/5$$

■

Lesson 5 Practice Problem Solutions

Exercise 1-23 (c)

We set up the following truth table in order to prove that $((p \quad q) \rightarrow r) \leftrightarrow (p \rightarrow r) \quad (q \rightarrow r)$ is indeed a tautology.

p	q	r	$p \quad q$	$(p \quad q) \rightarrow r$	$p \rightarrow r$	$q \rightarrow r$	$(p \rightarrow r) \quad (q \rightarrow r)$
T	T	T	T	T	T	T	T
T	T	F	T	F	F	F	F
T	F	T	F	T	T	T	T
T	F	F	F	T	F	T	T
F	T	T	F	T	T	T	T
F	T	F	F	T	T	F	T
F	F	T	F	T	T	T	T
F	F	F	F	T	T	T	T

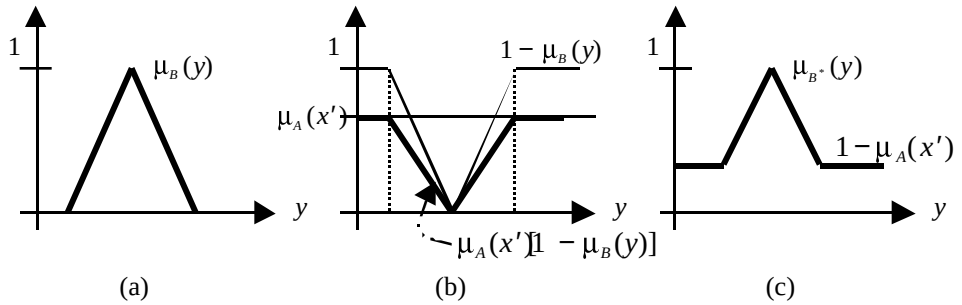
Observe equality in the columns for $(p \quad q) \rightarrow r$ and $(p \rightarrow r) \quad (q \rightarrow r)$. This tautology suggests that a two-antecedent rule can be decomposed into the union of two single-antecedent rules, something that has already been developed by W. E. Combs and J. E. Andrews, in "Combinatorial rule explosion eliminated by a fuzzy rule configuration," *IEEE Trans. on Fuzzy Systems*, vol. 6, pp. 1–11, Feb. 1998.

Exercise 1-26

Beginning with the implication MF in (1-73), namely $\mu_{A \rightarrow B}(x,y) = 1 - \mu_A(x)[1 - \mu_B(y)]$, (1-76) can be expressed as:

$$\mu_{B^*}(y) = \mu_{A \rightarrow B}(x',y) = 1 - \mu_A(x')[1 - \mu_B(y)]$$

The three figures below provide our construction of $\mu_{B^*}(y)$. Observe that the result in part (c) is identical to the result in part (c) of Figure 1-10; hence, conclusions drawn at the end of Example 1-19 apply here as well.



(a) (b) (c)

Lesson 6 Practice Problem Solution

Lesson 6 Exercise

Regardless of the numerical values you chose for the end points of the five labels, there is the following interesting question: "How do you update the mean and standard deviation values that are given in Table 4-2 to account for your new values?" Consider the sample mean first.

Suppose we are given a collection of i measured values of a quantity X, that is, $x(1)$, $x(2)$, …, $x(i)$. The sample mean of these measurements, $\bar{x}(i)$, is

$$\bar{x}(i) = \frac{1}{i}\sum_{j=1}^{i} x(j)$$

A recursive formula for the sample mean lets us fold in a new measurement, $x(i+1)$, into this formula to compute $\bar{x}(i+1)$. It is obtained as follows:

$$\bar{x}(i+1) = \frac{1}{i+1}\sum_{j=1}^{i+1} x(j) = \frac{1}{i+1}\left[\sum_{j=1}^{i} x(j) + x(i+1)\right]$$

$$\bar{x}(i+1) = \frac{i}{i+1}\bar{x}(i) + \frac{1}{i+1}x(i+1)$$

In our case, $i = 47$; hence, we use the formula

$$\bar{x}(48) = \frac{47}{48}\bar{x}(47) + \frac{1}{48}x(48)$$

Next, consider the standard deviation. We update the standard deviation by first updating the variance, $\sigma^2(i)$, and then taking its positive square root. Recall that the sample variance is given as

$$\sigma^2(i) = \frac{1}{i}\sum_{j=1}^{i}\left[x(j) - \bar{x}(j)\right]^2$$

Following exactly the same procedure as above, we find that

$$\sigma^2(i+1) = \frac{i}{i+1}\sigma^2(i) + \frac{1}{i+1}\left[x(i+1) - \bar{x}(i+1)\right]^2$$

In our case, we use the formula

$$\sigma^2(48) = \frac{47}{48}\sigma^2(47) + \frac{1}{48}\left[x(48) - \bar{x}(48)\right]^2$$

Observe that to compute $\sigma^2(48)$ we must first compute $\bar{x}(48)$.

Lesson 7 Practice Problem Solution

Lesson 7 Exercise

Follow the discussion in Example 5-1 for an explanation of how to construct the three figures below.

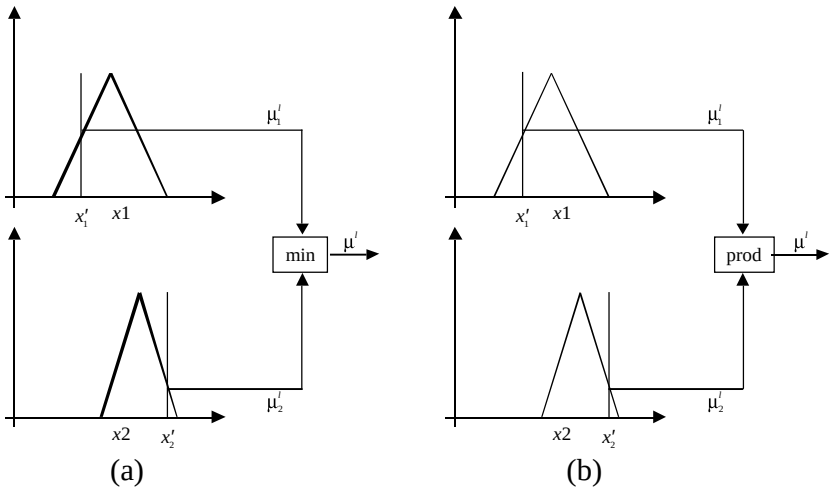


Figure 5-4: Pictorial description of input and antecedent operations for a type-1 FLS that uses triangular MFs. (a) Singleton fuzzification with minimum t-norm, and (b) singleton fuzzification with product t-norm.

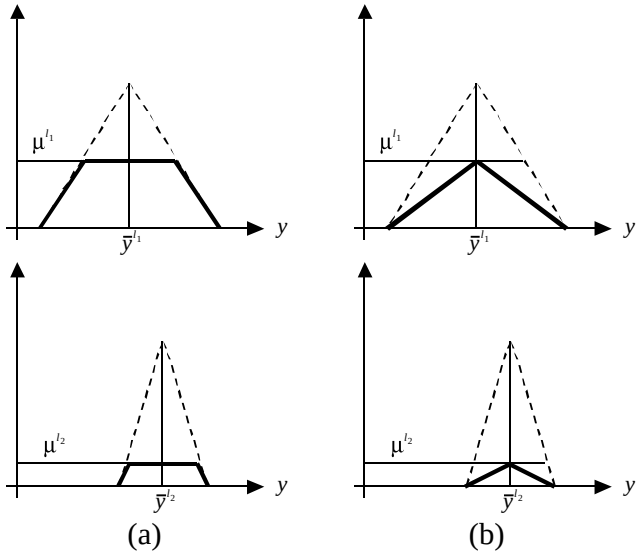


Figure 5-5: Pictorial description of consequent operations for a type-1 FLS when consequent fuzzy set MFs are triangles. (a) Fired output sets with minimum t-norm, and (b) fired output sets with product t-norm.

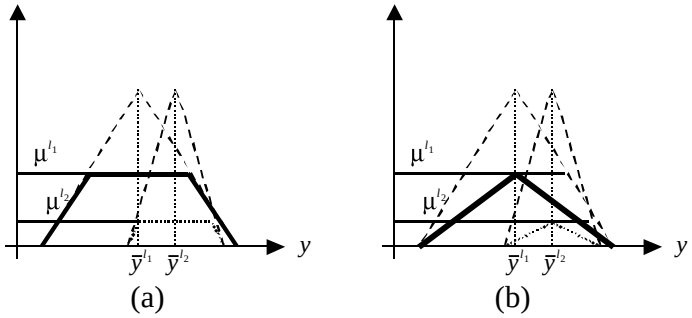


Figure 5-6: Pictorial description of (a) combined output sets for the two fired output sets depicted in Figure 5-5 (a), and (b) combined output sets for the two fired output sets depicted in Figure 5-5 (b). Observe that the maximum of the MFs for the two fired-sets coincides with the MF for the first fired output set.

Note that it is purely by coincidence that the second fired rule makes no contribution to the combined output sets for the two fired output sets. Your solution to this exercise might have led to a maximum operation in which the two fired output sets contributed to the final output set.

Lesson 8 Practice Problem Solutions

Exercise 5-4:

(a) *Features of a fired-rule consequent set used by a center-of-sums defuzzifier*:

Under product implication and product t-norm [see (5-10)], $y_a(\mathbf{x})$ is given in (5-15), from which we see that we use the centroid and area of each consequent set. For other kinds of implications (e.g., minimum) we don't use specific features of each $G^l(y)$. Instead, $y_a(\mathbf{x})$ must be computed using (5-14) which involves features of the output set $B^l(y)$, namely its centroid, c_{B^l}, and area, a_{B^l}.

(b) *Features of a fired-rule consequent set used by a height defuzzifier*:

$\bar{y}^l$—the point having maximum membership in $\mu_{B^l}(y)$.

(c) *Features of a fired-rule consequent set used by a center-of-sets defuzzifier*:

c^l—the centroid of the *l*th consequent set B^l. ■

Exercise 5-6:

When triangles are used for the interior MFs and piecewise linear functions are used for the two shoulder (exterior) MFs, the *design parameters* are:

1. Shoulder MFs: break point and slope of leg (or location of base point)—2 parameters/MF.
2. Interior triangles: center location and length of base [assume that the triangle is symmetrical (for non-symmetrical triangles, a third parameter is needed, e.g., slopes of both legs, or left-end and right-end base points)]— 2 parameters/MF.

Assume *L* fuzzy sets for each antecedent and consequent. Total antecedent/consequent MF design parameters—2*L*. ■

Lesson 9 Practice Problem Solution

Exercise 5-10:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} e^{(i)}$$

where

$$e^{(i)} = \tfrac{1}{2}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)} \right]^2 \quad \ldots (5\text{-}47)$$

Hence,

$$grad_{\theta} J(\theta) = \frac{\partial J(\theta)}{\partial \theta} = \frac{\partial}{\partial \theta}\left[\frac{1}{N}\sum_{i=1}^{N} e^{(i)}\right] = \frac{1}{2N}\sum_{i=1}^{N} \frac{\partial}{\partial \theta}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)} \right]^2$$
$$= \frac{1}{N}\sum_{i=1}^{N}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)} \right] \times \frac{\partial}{\partial \theta} f_s(\mathbf{x}^{(i)})$$

In this last equation, note that the summation also acts on $\partial f_s(\mathbf{x}^{(i)}) / \partial \theta$. Calculations of $\partial f_s(\mathbf{x}^{(i)}) / \partial \theta$ are exactly the same as in Exercise 5-9. See its solution given in this *Study Guide*. Here we just need to include the $\sum_{i=1}^{N}$ in its proper places, e.g. when $\theta = \bar{y}^l$

$$\frac{\partial}{\partial \bar{y}^l} f_s(\mathbf{x}^{(i)}) = \varphi_l(\mathbf{x}^{(i)})$$

and

$$\bar{y}^l(i+1) = \bar{y}^l(i) - \alpha_{\bar{y}} \frac{\partial}{\partial \bar{y}^l} J(\bar{y}^l) = \bar{y}^l(i) - \alpha_{\bar{y}} \frac{1}{N}\sum_{i=1}^{N}\left[f_s(\mathbf{x}^{(i)}) - y^{(i)} \right] \times \varphi_l(\mathbf{x}^{(i)})$$

Compare this equation with its counterpart in Equation (5-49).

The steepest descent algorithms for $m_{F_k^l}(i+1)$ and $\sigma_{F_k^l}(i+1)$, given in (5-48) and (5-50), are structurally the same, except that each has $\sum_{i=1}^{N}$ in front of its second term. ■

Lesson 10 Practice Problem Solutions

Exercise 5-14:

Our goal is to compute $y_{c1}(2,4)$. When $x_1 = 2$ two subsets are fired, *NVL* and *S*, and their firing degrees (picked off of Figure 5-13) are 1 and 0.182, respectively. When $x_2 = 4$ two subsets are fired, *S* and *MOA*, and their firing degrees (also picked off of Figure 5-13) are 1 and 0.545, respectively. It follows, therefore, that the rules whose antecedent pairs are

$$(NVL, S),\ (NVL, MOA),\ (S, S),\ \text{and}\ (S, MOA)$$

are the ones fired. From Table 5-6, we see that these are rules 2, 3, 7 and 8. The firing degree for each of these rules is obtained by multiplying the rule's respective antecedent firing degrees, e.g. the firing degree for R^2 is $1 \times 1 = 1$. Consequently, we can compute $y_{c1}(2,4)$ using (5-63) and the results given in the last column of Table 5-6, as:

$$\begin{aligned} y_{c1}(2,4) &= \left[\frac{1}{(1\times 1 + 1\times 0.545 + 0.182\times 1 + 0.182\times 0.545)}\right] \\ &\quad \times [2.099\times 1 + 4.3204\times 0.545 + 3.1601\times 0.182 + 5.1566\times (0.182\times 0.545)] \\ &= \left[\frac{1}{1.8268}\right]\times [2.099 + 2.3546 + 0.5751 + 0.5115] \\ &= 5.5402 \big/ 1.8262 = 3.0337 \end{aligned}$$

■

Exercise 5-15:

Single-antecedent rules: Using Figure 5-13, project upwards from the horizontal axis and observe that there can be either one, two or three intersections with MFs. Hence, a single-antecedent FLS that uses these MFs can fire one, two or three rules.

Two-antecedent rules: If each antecedent can intersect one, two or three MFs, then we take all possible combinations of products of (1, 2, 3) and (1, 2, 3) to obtain (1, 2, 3, 4, 6, 9). Hence, we conclude that a two-antecedent FLS that uses these MFs can fire one, two, three, four, six or nine rules. ■

Lesson 11 Practice Problem Solutions

Exercise 11–1:

Follow the discussion in Example 6-1 for an explanation of how to construct the pictorial description of input and antecedent operations for a non-singleton type-1 FLS, one that now uses triangular MFs. The results are depicted in Figure 1 below. In part (b) of the figure, although we show the product as two heavy triangles (they are actually quadratics) their exact shape is unimportant, because we only use the location of the maximum of the product, which occurs at the value of x_1 and x_2 where the input MFs equal one (i.e., at $x_1 = x_1'$ and $x_2 = x_2'$). The latter occurs at the apex of the heavy figures regardless of their shape.

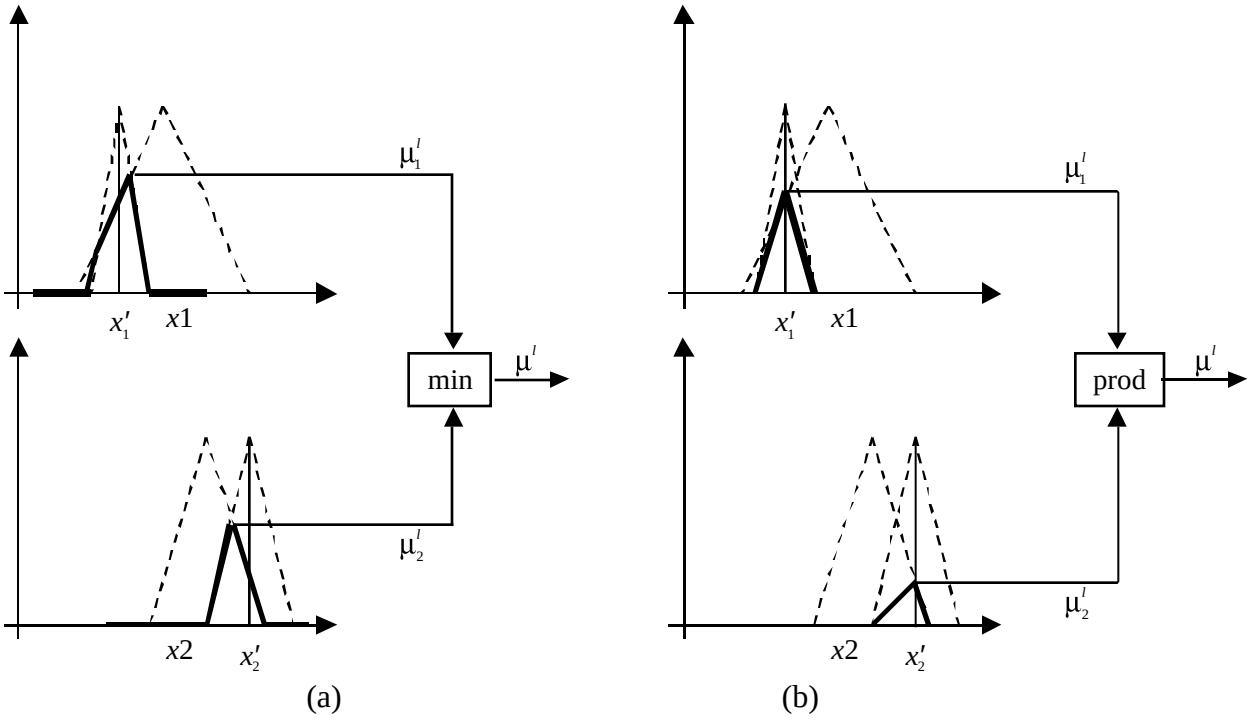


Figure 1: Pictorial description of input and antecedent operations for a non-singleton type-1 FLS that uses triangular MFs. (a) Singleton fuzzification with minimum t-norm, and (b) singleton fuzzification with product t-norm.

Figures comparable to Figures 5-5 and 5-6 have already been created by you in Lesson 7, and they do not change. What does change are the numerical values for μ^{l_1} and μ^{l_2}. ■

Exercise 6-5:

Regardless of whether $\theta = m_{F_k^l}$, $\bar{y}^l$, $\sigma_{F_k^l}$ or σ_X, certain parts of the calculations of $\text{grad}J(\theta) = \partial J(\theta)/\partial\theta$, where $J(\theta) = e^{(i)} = \frac{1}{2}\left[f_{ns}(\mathbf{x}^{(i)}) - y^{(i)} \right]^2$, are identical, namely:

$$\frac{\partial J(\theta)}{\partial \theta} = \frac{\partial}{\partial \theta}\left\{\tfrac{1}{2}\left[f_{ns}(\mathbf{x}^{(i)}) - y^{(i)} \right]^2\right\} = \left[f_{ns}(\mathbf{x}^{(i)}) - y^{(i)} \right]\frac{\partial}{\partial \theta} f_{ns}(\mathbf{x}^{(i)}) \tag{1}$$

where

$$f_{ns}(\mathbf{x}^{(i)}) = \sum_{l=1}^{M} \bar{y}^{l} \varphi_l(\mathbf{x}^{(i)}) \tag{2}$$

and

$$\varphi_l(\mathbf{x}^{(i)}) = \frac{\prod_{k=1}^{p} \exp\left\{-\frac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_X^2 + \sigma_{F_k^l}^2}\right\}}{\sum_{l=1}^{M}\prod_{k=1}^{p} \exp\left\{-\frac{1}{2}\frac{\left(x_k^{(i)} - m_{F_k^l}\right)^2}{\sigma_X^2 + \sigma_{F_k^l}^2}\right\}} \tag{3}$$

Comparing the first two equations in Section III of Lesson 9 (in the *Study Guide*) with Equations (1) and (2) above, we see that they are identical. Comparing Equation (10) in Section III of Lesson 9 with Equation (3) above, we see that they are identical when

$$\sigma_{F_k^l}^2\Big|_{(5-9)} \rightarrow \left(\sigma_X^2 + \sigma_{F_k^l}^2\right)\Big|_{(6-5)} \tag{4}$$

Hence, we do not need to repeat the derivation of the steepest descent algorithms for $m_{F_k^l}$, $\bar{y}^l$, and $\sigma_{F_k^l}$. What we conclude is that

$$(6-30) = (5-48)\Big|_{(4\text{ above})}$$

$$(6-31) = (5-49)\Big|_{(4\text{ above})}$$

$$(6-32) = (5-50)\Big|_{(4\text{ above})}$$

We did not derive a steepest descent algorithm for σ_X in Chapter 5, because $\sigma_X = 0$ in that chapter. In (1)–(3), σ_X only appears in (3). Comparing (3) above with Equation (10) in Lesson 9 of this *Study Guide*, we see (as mentioned in the textbook) that, in Chapter 6, $\sigma_X^2 + \sigma_{F_k^l}^2$ plays the role of Chapter 5's $\sigma_{F_k^l}^2$. We leave it to the reader to show that the steepest descent algorithm for σ_X—in (6-33)—is the same as the steepest descent algorithm for $\sigma_{F_k^l}$—in (6-32)—when

$$\sigma_{F_k^I}\Big|_{(6-32)} \rightarrow \sigma_X\Big|_{(6-33)}$$

Set $\sigma_X = 0$ in this chapter's steepest-descent algorithms to show that they reduce to their singleton counterparts in (5-48)–(5-50). ■

Exercise 6-7:

1. *Fix the shapes and parameters of all the antecedent, consequent and input measurement membership functions ahead of time. The data establishes the rules and the standard deviation of the measurements, and no tuning is used.*

Either of our two one-pass methods can be used.

2. *Fix the shapes and parameters of the antecedent and input measurement membership functions ahead of time. Use the training data to tune the consequent parameters.*

Use the least-squares method to do this.

3. *Fix the shapes and parameters of all the antecedent and consequent membership functions ahead of time. Fix the shape but not the parameter(s) of the input measurement membership function(s) ahead of time. Use the training data to tune the parameter(s) of the input measurement membership function(s).*

Use a back-propagation (steepest descent) method to do this.

4. *Fix the shapes of all the antecedent, consequent and input measurement membership functions ahead of time. Use the training data to tune the antecedent, consequent and input measurement parameters.*

Use a back-propagation (steepest descent) method to do this.

Lesson 12 Practice Problem Solution

Exercise 13-1:

Regardless of whether $\theta = m_{F_k^l}, c_i$, or $\sigma_{F_k^l}$, certain parts of the calculations of $\partial J(\theta)/\partial\theta$, where $J(\theta) = e^{(t)} = \frac{1}{2}\left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right]^2$, are identical, namely:

$$\frac{\partial J(\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\left\{\frac{1}{2}\left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right]^2\right\} = \left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right] \times \frac{\partial}{\partial\theta} y_{TSK,1}\left(\mathbf{x}^{(t)}\right) \tag{1}$$

where [see (13-9)]

$$y_{TSK,1}\left(\mathbf{x}^{(t)}\right) = \sum_{i=1}^{M}\sum_{j=0}^{p} g_j^i\left(\mathbf{x}^{(t)}\right)c_j^i \tag{2}$$

and [see (13-16)]

$$g_j^i\left(\mathbf{x}^{(t)}\right) = \frac{x_j^{(t)}\prod_{k=1}^{p}\exp\left\{-\frac{1}{2}\frac{\left(x_k^{(t)} - m_{F_k^i}\right)^2}{\sigma^2_{F_k^i}}\right\}}{\sum_{i=1}^{M}\prod_{k=1}^{p}\exp\left\{-\frac{1}{2}\frac{\left(x_k^{(t)} - m_{F_k^i}\right)^2}{\sigma^2_{F_k^i}}\right\}} \tag{3}$$

(1) $\theta = c_j^i$: In this case,

$$\frac{\partial}{\partial c_j^i} y_{TSK,1}\left(\mathbf{x}^{(t)}\right) = g_j^i\left(\mathbf{x}^{(t)}\right) \tag{4}$$

Hence,

$$c_j^i(n+1) = c_j^i(n) - \alpha_c \frac{\partial}{\partial c_j^i} J(c_j^i) = c_j^i(n) - \alpha_c\left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right] \times g_j^i\left(\mathbf{x}^{(t)}\right) \tag{5}$$

where j = 0, 1, …, p and i= 1, …, M.

(2) $\theta = m_{F_k^i}$: In this case, we write $y_{TSK,1}$ in (2) as

$$y_{TSK,1} = h/g \tag{6}$$

where [see (2) and (3)]

$$h = \sum_{i=1}^{M} \sum_{j=0}^{p} c_j^i w^i x_j^{(t)} \tag{7}$$

$$g = \sum_{i=1}^{M} w^i \tag{8}$$

and

$$w^i = \prod_{k=1}^{p} \exp\left\{ -\tfrac{1}{2} \frac{\left(x_k^{(t)} - m_{F_k^i} \right)^2}{\sigma_{F_k^i}^2} \right\} \tag{9}$$

We use the chain rule to compute $\partial y_{TSK,1} / \partial m_{F_k^i}$ as follows:

$$\frac{\partial y_{TSK,1}}{\partial m_{F_k^i}} = \frac{\partial y_{TSK,1}}{\partial w^i} \frac{\partial w^i}{\partial m_{F_k^i}} \tag{10}$$

where

$$\frac{\partial y_{TSK,1}}{\partial w^i} = \frac{g \dfrac{\partial h}{\partial w^i} - h \dfrac{\partial g}{\partial w^i}}{g^2} = \frac{g \sum_{j=0}^{p} x_j^{(t)} c_j^i - h}{g^2} = \frac{\sum_{j=0}^{p} x_j^{(t)} c_j^i - y_{TSK,1}}{g} \tag{11}$$

$$\frac{\partial w^i}{\partial m_{F_{k'}^i}} = \frac{\partial}{\partial m_{F_{k'}^i}} \prod_{k=1}^{p} \exp\left\{ -\tfrac{1}{2} \frac{\left(x_k^{(t)} - m_{F_k^i} \right)^2}{\sigma_{F_k^i}^2} \right\} = \frac{\partial}{\partial m_{F_{k'}^i}} \exp\left\{ -\tfrac{1}{2} \frac{\left(x_{k'}^{(t)} - m_{F_{k'}^i} \right)^2}{\sigma_{F_{k'}^i}^2} \right\} \times \prod_{\substack{k=1 \\ k \neq k'}}^{p} \exp\left\{ -\tfrac{1}{2} \frac{\left(x_k^{(t)} - m_{F_k^i} \right)^2}{\sigma_{F_k^i}^2} \right\}$$

$$\frac{\partial w^i}{\partial m_{F_k^i}} = \prod_{k=1}^{p} \exp\left\{ -\tfrac{1}{2} \frac{\left(x_k^{(t)} - m_{F_k^i} \right)^2}{\sigma_{F_k^i}^2} \right\} \times \frac{\left(x_{k'}^{(t)} - m_{F_k^i} \right)}{\sigma_{F_k^i}^2} \tag{12}$$

so that

$$\frac{\partial w^i}{\partial m_{F_k^i}} = \frac{\left(x_k^{(t)} - m_{F_k^i} \right)}{\sigma_{F_k^i}^2} \times w^i \tag{13}$$

Hence, substituting (11) and (13) into (10), we find that

$$\frac{\partial y_{TSK,1}}{\partial m_{F_k^i}} = \frac{\sum_{j=0}^{p} x_j^{(t)} c_j^i - y_{TSK,1}}{g} \times \frac{\left(x_k^{(t)} - m_{F_k^i}\right)}{\sigma_{F_k^i}^2} \times w^i \tag{14}$$

and, we obtain the following iterative algorithm for updating $m_{F_k^i}$:

$$m_{F_k^i}(n+1) = m_{F_k^i}(n) - \alpha_m \left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right] \frac{\partial y_{TSK,1}}{\partial m_{F_k^i}}\bigg|_n$$

$$m_{F_k^i}(n+1) = m_{F_k^i}(n) - \alpha_m \left[y_{TSK,1}\left(\mathbf{x}^{(t)}\right) - y^{(t)}\right] \times \left[\sum_{j=0}^{p} x_j^{(t)} c_j^i(n) - y_{TSK,1}\left(\mathbf{x}^{(t)}\right)\right] \times \frac{\left[x_k^{(t)} - m_{F_k^i}(n)\right]}{\sigma_{F_k^i}^2(n)} \times \frac{w^i(n)}{g(n)} \tag{15}$$

(3) $\theta = \sigma_{F_k^l}$: The derivation of the back-propagation algorithm for $\sigma_{F_k^l}$ is just like the derivation of (15). The key steps are (6)–(9). We then compute

$$\frac{\partial y_{TSK,1}}{\partial \sigma_{F_k^i}} = \frac{\partial y_{TSK,1}}{\partial w^i} \frac{\partial w^i}{\partial \sigma_{F_k^i}} \tag{16}$$

where $\partial y_{TSK,1} / \partial w^i$ is in (11), and we only need to compute $\partial w^i / \partial \sigma_{F_k^i}$. Because this last computation is just like the one for $\partial w^i / \partial m_{F_k^i}$, we leave its details to the reader.

Lesson 13 Review Question Solutions

I. Rule-Based Classification of Video Traffic

1. a, c, f
2. b
3. c
4. b
5. c

II. Equalization of Time-Invariant Non-linear Digital Communication Channels

1. c
2. b
3. a
4. b
5. c

III. Fuzzy Logic Control

1. b
2. b
3. a
4. c
5. b

Lesson 14 Practice Problem Solutions

Exercise SG 14-1:

Only Equation (3) changes to [see (5-26)] $\phi_l(\mathbf{x}) = \dfrac{\min_{i=1,2,\ldots,p}\left\{\mu_{F_i^l}(x_i)\right\}}{\sum_{l=1}^{M}\min_{i=1,2,\ldots,p}\left\{\mu_{F_i^l}(x_i)\right\}} \quad l = 1,\ldots,M.$

Equations (2) and (4) and this new equation for $\phi_l(\mathbf{x})$ implement a singleton type-1 Mamdani FLS under minimum t-norm. ■

Exercise SG 14-2:

Equation (9) changes to [see (6-19)] $\phi_l(\mathbf{x}) = \dfrac{\min_{k=1,2,\ldots,p}\left\{\mu_{Q_k^l}(x_{k,\max}^l)\right\}}{\sum_{l=1}^{M}\min_{k=1,2,\ldots,p}\left\{\mu_{Q_k^l}(x_{k,\max}^l)\right\}} \quad l = 1,\ldots,M;$

Equation (12) changes to [see (6-13)] $x_{k,\max}^l = \dfrac{\sigma_{X_k} m_{F_k^l} + \sigma_{F_k^l} m_{X_k}}{\sigma_{X_k} + \sigma_{F_k^l}}$; and, Equation (13) changes to [see Lesson 5, Example 5-2, Part (c), in which we make the appropriate substitutions for $x_{\max}$, x', m_A, σ_{A^*}, and σ_A] $\mu_{Q_k^l}(x_{k,\max}^l) = \exp\left[-\frac{1}{2}\left(\dfrac{m_{X_k} - m_{F_k^l}}{\sigma_{X_k} + \sigma_{F_k^l}}\right)^2\right]$. Equation (8) and the new equations for $\phi_l(\mathbf{x})$ and $\mu_{Q_k^l}(x_{k,\max}^l)$ implement a non-singleton type-1 Mamdani FLS under minimum t-norm. ■

Exercise SG 14-3:

Only Equation (15) changes to $f^i(\mathbf{x}) = \min_{k=1,2,\ldots,p}\left\{\mu_{F_k^i}(x_k)\right\}$. Equation (14), this new equation for $f^i(\mathbf{x})$, and Equation (16) implement a singleton, normalized type-1 TSK FLS under minimum t-norm. ■

The inventor of fuzzy logic persisted despite decades of opposition

LOTFI A. ZADEH

TEKLA S. PERRY
Senior Editor

THE DENUNCIATIONS were sometimes extreme. "Fuzzy theory is wrong, wrong, and pernicious," said William Kahan, a highly regarded professor of computer sciences and mathematics at the University of California at Berkeley in 1975. "The danger of fuzzy theory is that it will encourage the sort of imprecise thinking that has brought us so much trouble."

Another berated the theory's scientific laxity. "No doubt professor Zadeh's enthusiasm for fuzziness has been reinforced by the prevailing political climate in the United States—one of unprecedented permissiveness," said R. E. Kalman in 1972, who is now a professor at Florida State University in Tallahassee. "Fuzzification is a kind of scientific permissiveness; it tends to result in socially appealing slogans unaccompanied by the discipline of hard scientific work."

A multitude of other outspoken critics also disputed the theory of fuzzy logic, developed by Lotfi A. Zadeh in the mid-1960s. Some 20 years were to pass before the theory became widely accepted—capped by this year's award of the IEEE Medal of Honor to Zadeh "for pioneering development of fuzzy logic and its many diverse applications." Even today some critics remain. But Zadeh never wavered. He had found himself alone in his scientific opinions on several earlier occasions.

"There is a picture of me in my study, taken when I was a student at the University of Tehran," Zadeh told *IEEE Spectrum*. "I sit at a table, and above the table is a sign in Russian: ODIN, which means 'alone.' It was a proclamation of my independence."

Child of privilege

Perhaps the confidence Zadeh had in his judgment despite some tough opposition, and his willingness to stand apart from the crowd, originated in a childhood of privilege. He was born in 1921 in Azerbaijan, then part of the Soviet Union, and moved to Iran at age 10. His parents—his father a businessman and newspaper correspondent, his mother a doctor—were comfortably well off. As a child, Zadeh was surrounded by governesses and tutors, while as a young adult, he had a personal servant.

His career goal, for as long as he can remember, was to be an engineering professor. He never considered going into industry, he said, because money was no problem. Rather, he thought of scientific and engineering research as a type of religion, practiced at universities.

Zadeh received an electrical engineering degree from the University of Teheran in 1942. But instead of taking the comfortable route—becoming a professor in Iran—he emigrated to the United States.

"I could have stayed in Iran and become rich, but I felt that I could not do real scientific work there," he told *Spectrum*. "Research in Iran was nonexistent."

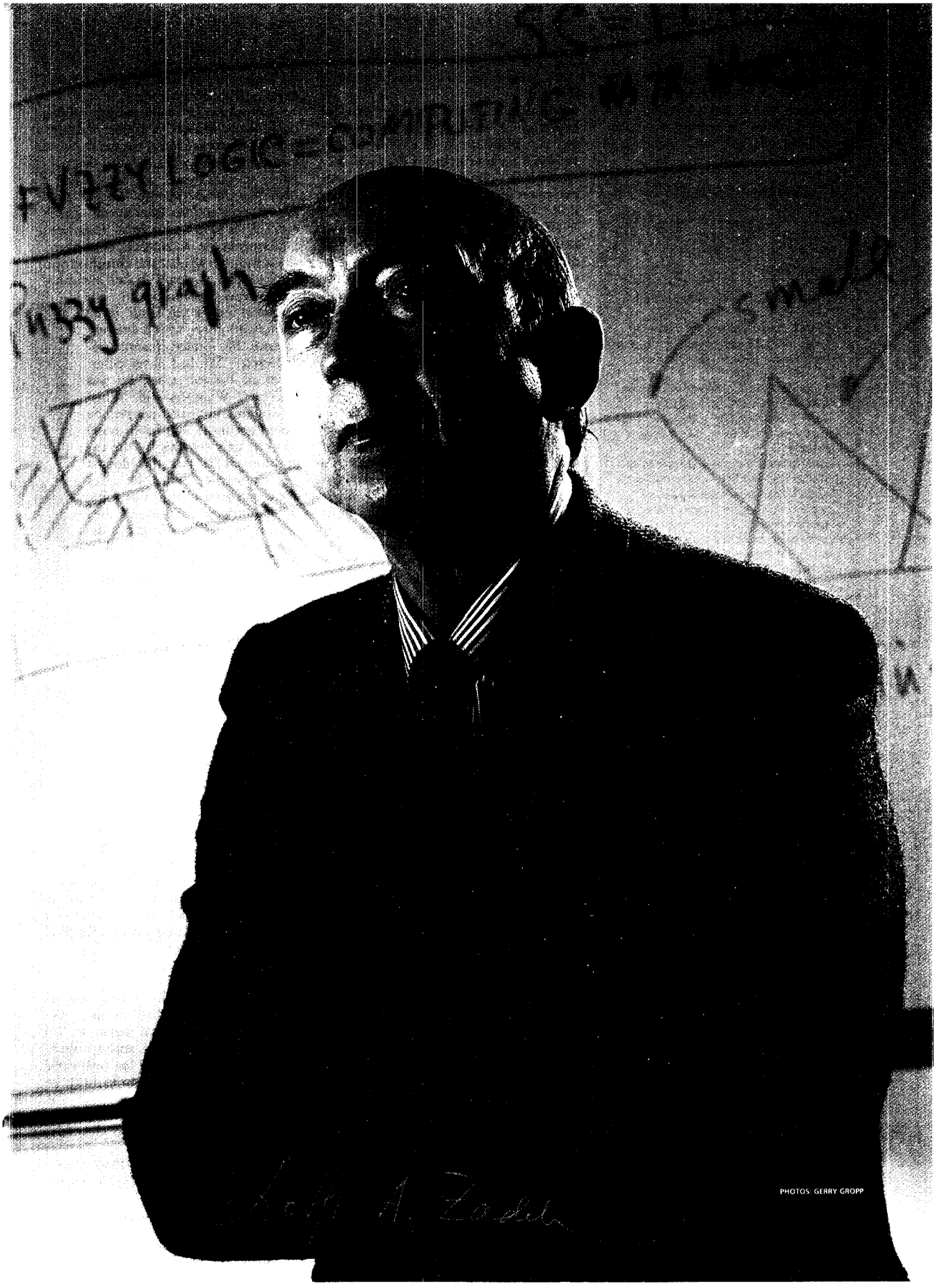
After graduation, Zadeh had a business association with the U.S. Army Persian Gulf Command. That enabled him to be financially independent when he came to the United States to enroll in graduate school at the Massachusetts Institute of Technology (MIT) in Cambridge. "MIT didn't have many graduate students at the time," Zadeh recalled, "so it was fairly easy to get in, even though the University of Teheran had no track record."

MIT, it turned out, was an easy ride after the demanding course work Zadeh had faced in Teheran. His choice of subject for his master's thesis, though, marked one of the first times he would sail against the prevailing technical winds. He chose to study helical antennas, a subject deemed unreasonable by the professor who had taught him antenna theory. Undaunted, Zadeh found another professor to supervise his work.

"I felt that my judgment was correct, and the judgment of people who supposedly knew much more about the subject than I did was not correct," Zadeh said. "This was one of many such situations. Helical antennas came into wide use in the '40s and '50s, and my judgment was vindicated."

By the time Zadeh received his master's degree in 1946, his parents had moved from Teheran to New York City. So instead of continuing at MIT, he searched out a post as an instructor at New York City's Columbia University and began his Ph.D.

0018-9235/95/$4.00©1995 IEEE



PHOTOS: GERRY GROPP

studies there. His thesis on the frequency analysis of time-varying networks considered ways of analyzing systems that change in time. "It was not a breakthrough," he recalled, "but it did make an impact and opened a certain direction in its field."

Vital statistics

Name: Lotfi A. Zadeh
Date of birth: Feb. 4, 1921
Birthplace: Baku, Azerbaijan
Height: 178 cm
Weight: 63.5 kg
Family: wife, Fay; children, Stella and Norman
Education: BSEE, University of Teheran, 1942; MSEE, Massachusetts Institute of Technology, 1946; Ph.D., Columbia University, 1949
First job: design and analysis of defense systems, International Electronics Corp., New York City, summer of 1944
Patents: one U.S. patent, two Iranian patents
Favorite books: "I made a conscious decision to stop reading fiction at age 15, when I was a voracious reader. I now read scientific books and other nonfiction only."
Favorite periodicals: Four newspapers daily (*The New York Times, San Francisco Chronicle, San Francisco Examiner, The Wall Street Journal* or *San Jose Mercury News*), *Business Week, The Economist*
Favorite kind of music: classical and electronic
Favorite composers: Sergey Prokofiev, Dimitry Shostakovich
Computer: a Hewlett-Packard workstation, which is used "only to print my e-mail and dictate all my answers to my secretary."
Favorite television show: "MacNeil/Lehrer Newshour"
Least favorite food: any kind of shellfish
Favorite restaurant: Three Cs Cafe, an inexpensive creperie in Berkeley, Calif.
Favorite expression: "No matter what you are told, take it as a compliment."
Favorite city: Berkeley, Calif.
Leisure activities: portrait photography (has photographed U.S. Presidents Richard Nixon and Harry Truman, as well as other notables), high-fidelity audio, garage sales
Car: Nissan Quest Minivan
Languages spoken: English, Russian, Iranian, French
Airline mileage: two million miles in past 10 years on American and United airlines alone, uncounted mileage on other airlines
Key organizational memberships: the IEEE, Association for Computing Machinery, International Fuzzy Systems Association, American Association for Artificial Intelligence
Top awards: the IEEE Medal of Honor (1995) and the Japan Honda Prize (1989)

What he views as his first technical breakthrough came in 1950, when, as an assistant professor at Columbia, he coauthored a paper with his doctoral thesis advisor, John R. Ragazzini, on "An extension of Wiener's theory of prediction." This analysis of prediction of time series is often cited as an early classic in its field. This thesis introduced the use of a finite, rather than an infinite, preceding time interval of observation for subsequent smoothing and prediction in the presence of multiple signals and noises. This, and Zadeh's other work while he was at Columbia, made him a well-known figure in the analysis of analog systems.

Berkeley beckons

As Zadeh was pretty much entrenched at Columbia, he surprised his colleagues when he packed up in 1959 and moved to the University of California at Berkeley.

"I had not been looking for another position," Zadeh said, "so the offer from Berkeley was unexpected." It came from electrical engineering department chairman John Whinnery, who called him at home over the weekend and offered him a position. "If my line had been busy, I believe I would still be at Columbia," Zadeh told *Spectrum*.

Whinnery recalls it slightly differently. He had heard from a colleague that Zadeh had been toying with the idea of leaving Columbia. Minutes later, Whinnery picked up the phone and called him, arranged to meet in him New York City for dinner, and soon afterward hired him. Berkeley was then growing rapidly, and Whinnery was on the lookout for young scholars who were considered brilliant in their fields. Zadeh fit the bill.

For Zadeh, moving to Berkeley was a simple decision to make: "I was happy at Columbia, but the job was too soft. It was a comfortable, undemanding environment; I was not challenged internally. I realized that at Berkeley my life would not be anywhere near as comfortable, but I felt that it would be good for me to be challenged."

Zadeh has never regretted the decision. To this day he remains at Berkeley, although by now as professor emeritus.

At Berkeley, Zadeh initially continued his work in linear, nonlinear, and finite-state systems analysis. But before long he became convinced that digital systems would grow in importance. Appointed as chairman of the electrical engineering department, he decided to act on that conviction, and immediately set about strengthening the role of computer science in the department's curriculum. He also lobbied the electrical engineering community nationwide to recognize the importance of computer science.

Once again, he found himself fighting conventional wisdom. A number of departmental colleagues felt that the trend toward computer science was a fad, and that consumer science should not be assigned a high departmental priority. "They accused me of being an Yves St. Laurent," Zadeh recalled, "a follower of fads." Elsewhere, professors in the mathematics department, along with the head of the computer center, were lobbying to set up their own computer science department.

Zadeh fought this battle as he has fought others, with polite persistence, his former chairman recollected. "We had many differences of opinion when he was chairman," Whinnery said. "When he couldn't convince people, he would get upset, but [even now] you can only tell this by the expression on his face. He doesn't yell or scream. Then he goes ahead and does what he was going to do anyway. And mostly he's been right, particularly about the importance of computers in electrical engineering."

Said Earl Cox, chief executive officer of the Metus Systems Group, Chappaqua, N.Y., who has known Zadeh since the '70s: "I've never seen him anger anybody, even though he prides himself in going his own way, in thinking his own thoughts." (Zadeh is also known for encouraging others to be independent. He insists his graduate students publish in their own name, noted former student Chin L. Chang, who is now president of Nicesoft Corp., Austin, Texas. That practice goes against custom.)

Zadeh finally got his way in 1967: the name of the department was changed to electrical engineering and computer science (EECS). A separate computer science department was also established in Berkeley's College of Letters, but after a few years it folded and became absorbed into EECS.

Fuzzy is born

While he was focusing on systems analysis, in the early 1960s, Zadeh began to feel that traditional systems analysis techniques were too precise for real-world problems. In a paper written in 1961, he mentioned that a new technique was needed, a "fuzzy" kind of mathematics. At the time, though, he had no clear idea how this would work.

That idea came in July 1964. Zadeh was in New York City visiting his parents,

and planned to leave soon for Southern California, where he would spend several weeks at Rand Corp. working on pattern recognition problems. With this upcoming work on his mind, his thoughts often turned to the use of imprecise categories for classification.

"One night in New York," Zadeh recalled, "I had a dinner engagement with some friends. It was canceled, and I spent the evening by myself in my parents' apartment. I remember distinctly that the idea occurred to me then to introduce the

concept of grade of membership [concepts that became the backbone of fuzzy set theory]. So it is quite possible that if that dinner engagement had not been canceled, the idea would not have occurred to me."

Fuzzy technology, Zadeh explained, is a means of computing with words—*bigger, smaller, taller, shorter.* For example, *small* can be multiplied by a *few* and added to *large,* or *colder* can be added to *warmer* to get something in between.

Once the issue of classification had been solved, Zadeh could develop the theory of fuzzy sets quickly. Two weeks later he had a fairly fleshed-out group of concepts to present to his collaborator at Rand, Richard Bellman. "His response was enthusiastic," Zadeh said, "and that was a source of encouragement to me—though had he been very critical, I wouldn't have changed my mind."

Since he was Berkeley's electrical engineering department chairman at the time, and engaged in his struggle over the place of computer science at the university, Zadeh had little time to work on his new theory of fuzzy sets. He published his first paper in 1965, convinced that he was onto something important, but wrote only sparingly on the topic until after he left the department chairmanship in 1968.

Since then, fuzzy sets have been his full-time occupation. "I continue to be an active player," he said. "I am not merely an elder statesman who rests on his laurels. I give many talks, and this puts me under pressure. I must constantly think of new ideas to talk about and keep up with what others are doing."

The Golden Fleece

Acceptance of fuzzy set theory by the technical community was slow in coming. Part of the problem was the name—"fuzzy" is hardly proper terminology. And Zadeh knew it.

"I was cognizant of the fact that it would be controversial, but I could not think of any other, respectable term to describe what I had in mind, which was classes that do not have sharp boundaries, like clouds," he said. "So I decided to do what I thought was right, regardless of how it might be perceived. And I've never regretted the name. I think it is better to be visible and provocative than to be bland."

And, as expected, fuzzy theory did cause controversy. Some people rejected it outright because of the name, without knowing the content. Others rejected it because of the theory's focus on imprecision.

In the late 1960s, it even garnered the passing attention of Congress as a prime example of the waste of government funds (much of Zadeh's research was being funded by the National Science Foundation). Former Senator William Proxmire (D-Wis.), the force behind the Golden Fleece Awards that honored such government boondoggles as $600 toilet seats, sent a letter to the foundation suggesting that such "fuzzy" garbage they were supporting should earn a Golden Fleece nomination. A flurry of correspondence from Zadeh and the foundation emerged in defense of the work.

Zadeh remembers the challenge of developing his theories "in the face of opposition, even hostility. Someone with a thinner skin would have been traumatized," he said. And Cox remarked, "He meets people who have written some really nasty things, and he's nice to them."

But, observed Berkeley's Whinnery, "I do think this lack of acceptance bothered him, although he now describes it with some humor."

Eventually, fuzzy theory was taken seriously—by the Japanese. And their implementations of it surprised even Zadeh.

He at first had expected fuzzy sets to apply to fields in which conventional analytic techniques had been ineffectual, for work outside of the hard sciences, for work in philosophy, psychology, linguistics, biology, and so on. He also thought that the theory might apply to control systems, in engine control, for example. But he never expected it to be used in consumer products, which today is perhaps its biggest application, thanks to Japanese electronics companies.

Matsushita Electric Industrial Co. was the first to apply fuzzy theory to a consumer product, a shower head that controlled water temperature, in 1987. Now numerous Japanese consumer products—dishwashers, washing machines, air conditioners, microwave ovens, cameras, camcorders, television sets, copiers, and even automobiles—quietly apply fuzzy technology.

These products make use of fuzzy logic combined with sensors to simplify control. For example, cameras have several focusing spots and use fuzzy's IF-THEN rules to calculate the optimal focus; camcorders use fuzzy logic for image stabilization; and washing machines use sensors to detect how dirty the water is and how quickly it is clearing to determine the length of wash cycles.

The introduction of fuzzy products by the Japanese riveted press attention on this apparently "new" technology (some two decades after Zadeh had developed the theory). Growing acknowledgment of the theory by his colleagues followed, although some still reject it.

Acceptance, colleagues say, has somewhat changed Zadeh. "Since fuzzy logic has turned into something with so much panache, and he has finally come into his own after being ignored for so many years, I think Lotfi has come out of his shell," said Cox.

To date, hundreds of books have been published on the topic, and some 15 000 technical papers have been written (most, it seems, piled around his office, where stacks of papers leave only a narrow path from the door to his desk). Zadeh is now known as the Father of Fuzzy.

"Had I not launched that theory," said Zadeh, "I would fall into the same category as many professors—be reasonably well known, have attained a certain level of recognition, and written some books and papers, but not have made a long-lasting impact. So I consider myself to have been lucky that this thing came about.

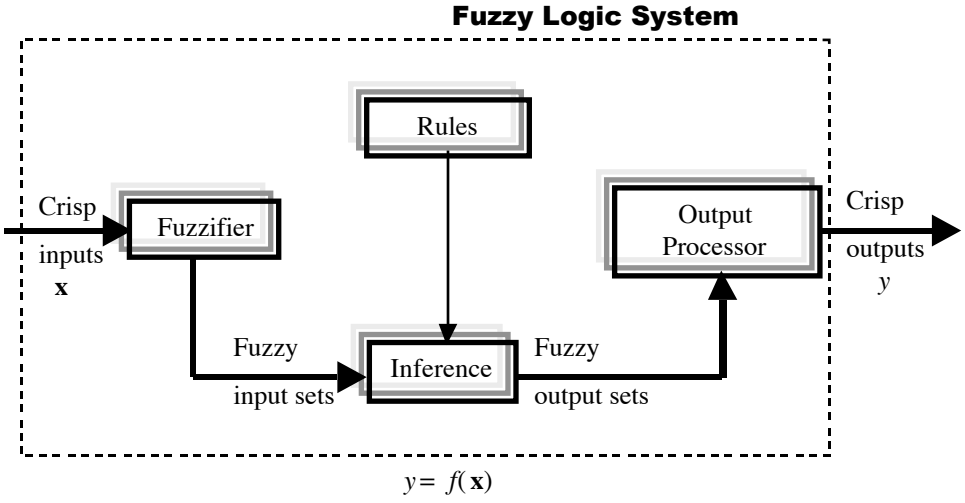
"The important criterion of your impact is: has what you have done generated a following? With fuzzy sets, I can definitely say, 'Yes.' " ◆

Final Examination for

Introduction to Rule-Based Fuzzy Logic Systems

by Jerry M. Mendel

University of Southern California



Lesson 1: INTRODUCTION AND OVERVIEW

1. Circle the four elements that comprise a fuzzy logic system:
   a. inference engine
   b. encoder
   c. demodulator
   d. fuzzifier
   e. equalizer
   f. output processor
   g. sampler
   h. rules

2. A neural network that incorporates fuzzy sets or fuzzy logic is called a:
   a. neural fuzzy system
   b. fuzzy logic system
   c. fuzzy neural system

3. The founder of fuzzy logic is:
   a. Aristotle
   b. Lotfi Zadeh
   c. Bertrand Russell

4. Circle the two fundamental laws of classical logic that are usually broken in fuzzy logic:
   a. De Morgan's Laws
   b. Law of Excluded Middle
   c. Transitive Law
   d. Distributive Law
   e. Law of Contradiction
   f. Idempotent Law

5. Fuzzy logic:
   a. includes classical logic as a special case
   b. is three-valued logic
   c. includes true, false, could be and maybe

6. During the operation of a rule-based fuzzy logic system, it:
   a. is implemented using neural networks
   b. involves three of its four elements: fuzzifier, inference engine and output processor
   c. involves all four of its elements: rules, fuzzifier, inference engine and output processor

7. Applications of fuzzy logic:
   a. are confined to control systems
   b. are non-existent
   c. occur only in Japanese products
   d. abound in many fields

Lesson 2: FUZZY SETS–PART 1

8. Circle all of the ways in which a fuzzy set can be described:
   a. membership function
   b. listing all of its elements
   c. specifying a condition or conditions for which an element is a member of the set

9. Linguistic variables are variables whose values are:
   a. numbers
   b. algebraic relations
   c. words or sentences in a natural or artificial language

10. A *normal* Gaussian MF has how many design degrees of freedom?
   a. one
   b. two
   c. three

11. Membership functions for fuzzy sets:
   a. are unique
   b. can be of shapes that are chosen by the designer
   c. are always chosen as triangles

12. A MF for *integers* close to 10 is:
   a. $\mu_{close\ to\ 10}(x) = 0.3/7 + 0.6/8 + 1/9 + 1/10 + 1/11 + 0.6/12 + 0.3/13$
   b. $\mu_{close\ to\ 10}(x) = \begin{cases} x-10, & x < 10 \\ 1, & x \geq 10 \end{cases}$
   c. $\mu_{close\ to\ 10}(x) = \exp\left[-\frac{(x-10)^2}{2}\right] \quad x \in R$

Lesson 3: FUZZY SETS–PART 2

13. The basic fuzzy set theoretic operations of union, intersection and complement can be computed using:
   a. crisp MFs whose values are 0 or 1
   b. fuzzy MFs whose values are in the closed interval [0, 1]
   c. fuzzy MFs whose values are in the interval (0, 1)

14. Circle all of the following operators that are t-norms:
   a. $x \bigstar y$
   b. $\min(x, y)$
   c. $x + y$
   d. $x \times y$
   e. $\min(1, x + y)$
   f. $\max(0, x + y - 1)$

15. The MFs for two fuzzy sets are depicted in Figure 15a. Which of the other 3 figures represents $\mu_{A \cup B}(x)$ under maximum t-norm?
   a. Figure 15-b
   b. Figure 15-c
   c. Figure 15-d

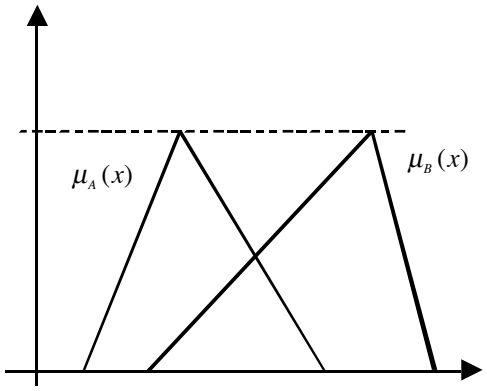


Figure 15-a

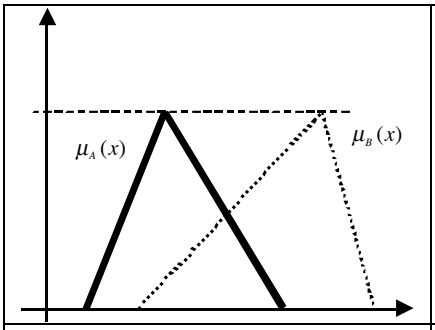


Figure 15-b

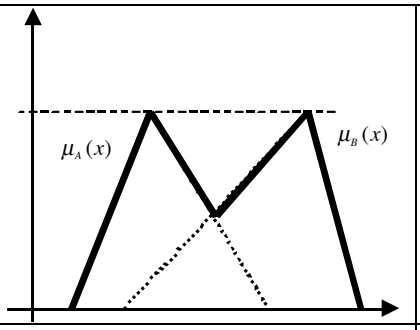


Figure 15-c

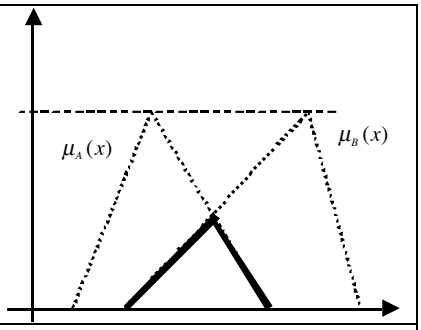


Figure 15-d

16. Given two fuzzy relations on the same product space, $R(U,V)$ and $S(U,V)$. Which of the following is the correct expression for the intersection of $R(U,V)$ and $S(U,V)$?
    a. $\mu_{R\cap S}(x,y) = \mu_R(x,y)\mu_S(x,y) \quad \forall x \in U$ and $\forall y \in V$
    b. $\mu_{R\cap S}(x,y) = \max\left[\mu_R(x,y)\mu_S(x,y)\right] \quad \forall x \in U$ and $\forall y \in V$
    c. $\mu_{R\cap S}(x,y) = 1 - \mu_R(x,y)\mu_S(x,y) \quad \forall x \in U$ and $\forall y \in V$

17. The intersection and union of two fuzzy relations on the same product space are called _____ of the fuzzy relations.
    a. Cartesian products
    b. compositions
    c. membership functions

18. A linguistic hedge is an:
    a. even bet
    b. operator that acts on a fuzzy set's MF converting it into a crisp MF
    c. operation that modifies the meaning of a term

19. The MF for very very unlikely is:
    a. $1 - \mu^4_{LIKELY}(x)$
    b. $\left[1 - \mu_{LIKELY}(x)\right]^2$
    c. $\left[1 - \mu_{LIKELY}(x)\right]^4$

Lesson 4: FUZZY SETS–PART 3

20. In the sup-star composition, the "star" refers to a:
    a. linear operator
    b. t-co-norm operator
    c. t-norm operator

21. Which of the following are correct sup-star compostions?
    a. $\max_{y\in V}[\mu_R(x,y)\mu_S(y,z)]$
    b. $\max_{y\in V}\left[\min\left[\mu_R(x,y),\mu_S(y,z)\right]\right]$
    c. $\max_{x\in U}[\mu_R(x,y)\mu_S(y,z)]$
    d. $\min_{y\in V}\left[\max\left[\mu_R(x,y),\mu_S(y,z)\right]\right]$
    e. $\max_{y\in V}\left[\max\left[0,\mu_R(x,y)+\mu_S(y,z)-1\right]\right]$

22. The (2, 1)-element, $\mu_{a\circ b}(u_2,w_1)$, in the max-min composition of $\mu_a(u,v)=\begin{pmatrix}0.7 & 0.2 & 0.4\\ 0.1 & 0.3 & 0.5\end{pmatrix}$ and $\mu_b(u,w)=\begin{pmatrix}0 & 0.2\\ 0.8 & 1\\ 0.3 & 0.4\end{pmatrix}$ is:
    a. $\mu_{a\circ b}(u_2,w_1)=0$
    b. $\mu_{a\circ b}(u_2,w_1)=0.3$
    c. $\mu_{a\circ b}(u_2,w_1)=0.5$

23. The Extension Principle lets us:
    a. compute the sup-star composition for continuous-valued fuzzy sets
    b. extend mathematical relations between one-dimensional fuzzy variables to multi-dimensional fuzzy variables
    c. extend mathematical relationships between non-fuzzy variables to fuzzy variables

24. Given $A = small = \mu_A(x)$, in which one of the situations below would you use the Extension Principle?
    a. Find the MF of $C = not\ small = \mu_C(x)$
    b. Find the MF of $B = very\ small = \mu_B(x)$
    c. Find the MF of $D = (very\ small)^3$

25. Which of the Extension Principles stated below is the correct one to use for a one-to-many multi-variable mapping?
    a. $\mu_B(y)=\max_{x\in f^{-1}(y)}\mu_A(x)\quad y\in V$
    b. $\mu_{f(A_1,A_2)}(y)\equiv\mu_B(y)=\begin{cases}\sup\limits_{(x_1,x_2)\in f^{-1}(y)}\min\{\mu_{A_1}(x_1),\mu_{A_2}(x_2)\}\\ 0 \text{ if } f^{-1}(y)=\varnothing\end{cases}$
    c. $B=f(A)=f\left(\sum_{x\in U}\mu_A(x)/x\right)=\mu_A(x_1)/y_1+\mu_A(x_2)/y_2+\cdots\mu_A(x_N)/y_N\equiv\mu_B(y)$

Lesson 5: FUZZY LOGIC

26. Which two tautologies are for an implication?
    a. $(p \to q) \leftrightarrow (\sim p) \vee q$
    b. $[(p \to q) \wedge (r \to s) \wedge (p \vee r)] \to (q \vee s)$
    c. $(p \wedge q) \to (p \vee q)$
    d. $(p \to q) \leftrightarrow \sim [p \wedge (\sim q)]$

27. The importance of the tautologies for implication is that they let us:
    a. establish MFs for the implication operator
    b. prove the truth of implication
    c. apply set theoretic operations to implication

28. The following is a tautology, $(p \wedge q) \to (p \vee q)$
    a. true
    b. false

29. The importance of set theory, logic and Boolean Algebra being mathematically equivalent is:
    a. set theory can be replaced by logic
    b. set theory and logic can be replaced by Boolean Algebra
    c. any statement that is true in one system becomes true in the other simply by making some changes in notation

30. The transition from crisp logic to FL is done by replacing crisp logic's MFs by fuzzy MFs, and Modus Ponens by Generalized Modus Ponens, which means that the fuzzy MF of a fired rule is:
    a. non-unique
    b. unique
    c. zero or one

31. Suppose that antecedent and consequent MFs are triangles, singleton fuzzification occurs, minimum t-norm is used, and Mamdani minimum implication is used. Then the MF of a fired rule is:
    a. a scaled version of the consequent's triangular MF
    b. a clipped version of the consequent's triangular MF
    c. a clipped version of the product of the antecedent's MFs

32. When the antecedent MF is Gaussian, ($\mu_A(x) = \exp\left\{-\frac{1}{2}\left[\frac{(x-m_A}{\sigma_A}\right]^2\right\}$), the input is modeled as a Gaussian fuzzy number ($\mu_{A^*}(x) = \exp\left\{-\frac{1}{2}\left[\frac{(x-x'}{\sigma_{A^*}}\right]^2\right\}$), product implication and t-norm are used, then:
    a. $\sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\right]$ occurs at $x = x_{\max} = \left(\sigma_{A^*}^2 m_A + \sigma_A x'\right) \Big/ \left(\sigma_{A^*}^2 + \sigma_A\right)$
    b. $\sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\right]$ occurs at $x = x_{\max} = \left(\sigma_{A^*} m_A + \sigma_A x'\right) \Big/ \left(\sigma_{A^*} + \sigma_A\right)$
    c. $\sup_{x \in X}\left[\mu_{A^*}(x)\mu_A(x)\right]$ occurs at $x = x_{\max} = \left(\sigma_{A^*}^2 m_A + \sigma_A^2 x'\right) \Big/ \left(\sigma_{A^*}^2 + \sigma_A^2\right)$

Lesson 6: CASE STUDIES

33. *Forecasting* means:
    a. looking into the past
    b. looking into the future
    c. looking at the present

34. During the design of a FLS forecaster data is partitioned into which two subsets?
    a. testing
    b. universe of discourse
    c. training
    d. validation

35. Rules can be extracted from data in:
    a. only one way
    b. exactly three ways
    c. at least three ways

36. Chaotic behavior is:
    a. the same as random behavior
    b. deterministic and therefore repeatable
    c. sensitivity to parameter variations

37. A fuzzy logic advisor is a rule-based FLS whose rules are extracted from:
    a. people about controlling dynamical systems
    b. numerical data
    c. people about a judgment

38. People do not like to answer questions that have more than two antecedents because:
    a. they have a short memory
    b. correlating more than two things is very difficult to do
    c. it is boring to do

39. If more than two indicators of a judgment are important, then it is advisable to:
    a. Use a FLA that is comprised of sub-advisors
    b. Use all of them in a single FLA
    c. Use a neural network

Lesson 7: SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–PART 1

40. A singleton type-1 FLS is comprised of which elements?
    a. defuzzifier
    b. hedge controller
    c. inference mechanism
    d. fuzzifier
    e. rules
    f. type-reduction

41. The MF of a fired rule, $\mu_{B^l}(y)$, is given by:
    a. $\mu_{B^l}(y) = \sup_{\mathbf{x}\in\mathbf{X}}\left[\mu_{A_{\mathbf{x}}}(\mathbf{x}) \oplus \mu_{A^l\to G^l}(\mathbf{x},y)\right], \quad y \in Y$
    b. $\mu_{B^l}(y) = \sup_{\mathbf{x}\in\mathbf{X}}\left[\mu_{A^l}(\mathbf{x}) \bigstar \mu_{A^l\to G^l}(\mathbf{x},y)\right], \quad y \in Y$

c. $\mu_{B^l}(y) = \sup_{\mathbf{x}\in \mathbf{X}}\left[\mu_{A_{\mathbf{x}}}(\mathbf{x}) \bigstar \mu_{A^l \to G^l}(\mathbf{x}, y)\right], \quad y \in Y$

42. Singleton fuzzification means that a MF:
    a. is singular at the measurement point
    b. is non-zero at only one point, $x_i = x_i'$
    c. has non-zero values at a few single points

43. The importance of singleton fuzzification is it greatly simplifies the computation of the sup-star composition, in that it eliminates having to perform the:
    a. t-norm operation
    b. supremum operation
    c. both the t-norm and supremum operations

44. For singleton fuzzification, the MF of a fired two-antecedent rule, $\mu_{B^l}(y)\ y \in Y$, is:
    a. $\mu_{B^l}(y) = \mu_{G^l}(y) \bigstar \left[\mu_{F_1^l}(x_1') \bigstar \mu_{F_2^l}(x_2')\right], \quad y \in Y$
    b. $\mu_{B^l}(y) = \mu_{F_1^l}(x_1') \bigstar \mu_{F_2^l}(x_2'), \quad y \in Y$
    c. $\mu_{B^l}(y) = \mu_{G^l}(y) \bigstar \left[\mu_{F_1^l}(x_1') \bigstar \mu_{F_2^l}(x_2')\right], \quad y = \{y_1, y_2\}$

45. For singleton fuzzification, when all MFs are trapezoids and the t-norm is minimum or product, then a pictorial description of consequent operations shows that the fired output set for each rule is *another trapezoid* for which the one for the product t-norm is _____ the one for the minimum t-norm:
    a. larger than
    b. smaller than
    c. the same size as

Lesson 8: SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–PART 2

46. The defuzzifier that is the simplest to compute is:
    a. centroid
    b. center-of-sums
    c. height
    d. modified height
    e. center-of-sets

47. Outputs of different defuzzifiers are usually:
    a. different
    b. the same
    c. fuzzy sets

48. A singleton FLS has many design degrees of freedom. This means that:
    a. there is one unique type-1 FLS
    b. there can be many different type-1 FLSs, but their outputs will all be the same
    c. there can be many different type-1 FLSs, whose outputs will all be different

49. That a FLS can be interpreted as a fuzzy basis function (FBF) expansion means that the output of the FLS is:
    a. a nonlinear function of those FBFs
    b. a finite series which is a linear function of those FBFs
    c. an infinite series which is a linear function of those FBFs

50. The number of FBFs in a singleton type-1 FLS equals the:
    a. number of rule antecedent MF parameters multiplied by the number of consequent MF parameters

b. sum of all antecedents and consequents multiplied by the number of rules
c. number of rules

51. FBFs are:
a. orthogonal
b. coupled
c. uncoupled

52. The fact that a FLS is a universal approximator:
a. helps to explain why a FLS is so successful in engineering applications
b. tells us exactly how to design a FLS
c. means that a FLS can uniformly approximate any real discontinuous non-linear function to arbitrary degree of accuracy

53. Rule explosion:
a. refers to rules that are too hot to handle
b. is no problem in a FLS
c. refers to rapid growth in the maximum number of rules that may be required in a FLS.

Lesson 9: SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–PART 3

54. Referring to Example 5-6, if each rule has 4 antecedents and there are 30 rules, then the maximum number of design parameters for a singleton type-1 FLS is:
a. 240
b. 270
c. 290

55. Again, referring to Example 5-6, if each rule has 4 antecedents and we have 500 training pairs, then the maximum number of rules is:
a. 50
b. 30
c. 55
d. 60

56. The layered architecture for a FLS:
a. lets us implement the FLS in hardware as such an architecture
b. suggests the possibility of back-propagating errors from the output of the FLS to earlier layers, in analogy with back-propagation in a feed-forward neural network
c. lets us replace a FLS with a feed-forward neural network

57. An algorithm that makes use of first derivative information of an objective function:
a. is guaranteed to maximize the objective function
b. will find the global extremum of the objective function
c. will find a local extremum of the objective function

58. Derivations of steepest descent algorithms make very heavy use of the:
a. chain rule
b. Extension Principle
c. Mean-Value Theorem

59. One-pass design methods:
a. have a small number of rules
b. choose the design parameters using the method of least-squares
c. let the data establish either the parameters of the MFs or the entire rule

60. The least-squares design method is used when:
    a. all of the antecedent MF parameters are specified and only the consequent parameters need to be determined from the data
    b. all of the consequent MF parameters are specified and only the antecedent parameters need to be determined from the data
    c. both the antecedent and consequent parameters have to be determined from the data

Lesson 10: SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS–PART 4

61. In forecasting the Mackey-Glass time series, suppose that we used six antecedents instead of four, and we still used two fuzzy sets for each antecedent. How many rules would there be?
    a. 16
    b. 64
    c. 32

62. Measurement noise degrades a singleton type-1 FLS forecaster because:
    a. although it has been accounted for during the design of the FLS, it cannot be accounted for during its operation
    b. although it can be accounted for during the operation of the FLS, it has not been accounted for during its design
    c. it cannot be accounted for during both the design and operation of the FLS

63. A three-antecedent FLS can be visualized as a _____ surface:
    a. four-dimensional
    b. three-dimensional
    c. two-dimensional

64. Suppose that a type-1 FLS is described by two-antecedent rules and that the MFs for each antecedent are the ones shown in Figure 3-1 of the textbook. The numbers of rules that can be fired for such a FLS are:
    a. 1, 2, 3, or 4
    b. 1, 2, or 3
    c. 1, 2 or 4

65. If uncertainties were present about the consequents of a FLA—e.g., they are collected from a group of experts who do not all agree—then which of the possibilities for using the consequents listed below totally *ignores* the uncertainties?
    a. keep the response chosen by the largest number of experts
    b. find a weighted average of the rule consequents for each rule
    c. preserve the distributions of the expert-responses for each rule

Lesson 11: NON-SINGLETON TYPE-1 FUZZY LOGIC SYSTEMS

66. Non-singleton fuzzification means that each input to the FLS is modeled as a:
    a. spike MF
    b. flat MF
    c. MF that has a value equal to one at the measured value of the input and decreases to zero as the input variable gets farther away from the measured input value

67. For a non-singleton type-1 FLS, the sup operation in the sup-star composition:
    a. is the same computation as for a singleton type-1 FLS
    b. does not disappear, because $\mu_{A_{\mathbf{x}}}(\mathbf{x})$ has non-zero values over a range of values for each x_i

c. can be computed in closed form for all MFs

68. An informative way to interpret a non-singleton type-1 FLS is as a:
    a. pre-filter of the input **x** that transforms **x** into $\mathbf{x}_{\max}^{l}$ ($l = 1, \ldots, M$), after which the remaining operations of the FLS are the same as those of a singleton FLS
    b. pre-filter of the input **x** that transforms **x** into $\mathbf{x}_{\max}^{l}$ ($l = 1, \ldots, M$), after which the remaining operations of the FLS are different from those of a singleton FLS
    c. an inference mechanism followed by post-filtering

69. The firing level for a non-singleton type-1 FLS is:
    a. the same as the firing level of a singleton type-1 FLS
    b. smaller than the firing level of a singleton type-1 FLS
    c. different than the firing level of a singleton type-1 FLS

70. The new design degrees of freedom for a non-singleton type-1 FLS, as compared to those in a singleton type-1 FLS, are associated with parameters in the:
    a. consequent MF
    b. antecedent MF
    c. input MF

71. The rules of a non-singleton type-1 FLS are _____ the rules of a singleton type-1 FLS:
    a. different from
    b. the same as
    c. fewer than

72. In general, the *totally independent design approach* should provide _____ performance than the *partially independent design approach*:
    a. better
    b. the same
    c. worse

73. Which design method can be used to optimize all the parameters of a non-singleton type-1 FLS?
    a. one-pass
    b. least-squares
    c. back-propagation

74. Although a non-singleton type-1 FLS can model uncertain measurements, this is usually insufficient to achieve significantly improved performance over a singleton type-1 FLS because the:
    a. rules of the two kinds of FLSs are the same
    b. uncertainty contained in noisy training data is accounted for by the antecedent MFs of a type-1 FLS
    c. uncertainty contained in noisy training data cannot be accounted for by the antecedent and consequent MFs of a type-1 FLS

Lesson 12: TSK FUZZY LOGIC SYSTEMS

75. The reason that to-date only a singleton TSK FLS has been described in the literature is:
    a. the sup-star composition for the TSK FLS does not lead to a pre-filtering effect
    b. the pre-filtering effect of a non-singleton fuzzifier in a Mamdani FLS is established through the sup-star composition, but in a TSK FLS there is no sup-star composition
    c. TSK and Mamdani FLSs are the same, so it doesn't have to be described

76. Choose the two biggest differences between type-1 TSK and Mamdani FLSs:
    a. their antecedents are different

b. the consequent function of a type-1 TSK FLS is a function and not a fuzzy set
c. the consequent function of a type-1 TSK FLS is a fuzzy set and not a function
d. the output formula for a type-1 TSK FLS is obtained using a sup-star composition
e. the output formula for a type-1 TSK FLS is obtained by combining its rules in a prescribed way and does not derive from the sup-star composition

77. Under which condition is the normalized type-1 TSK FLS exactly the same as a type-1 Mamdani FLS?
a. the consequent function in a TSK rule is a linear function of the antecedent variables
b. the consequent function in a TSK rule is a quadratic function of the antecedent variables
c. the consequent function in a TSK rule is a constant

78. A TSK FLS that uses the same number of rules as a Mamdani FLS always has _____ design degrees of freedom than a Mamdani FLS:
a. fewer
b. the same number
c. more

79. When all of the antecedent parameters of a type-1 TSK FLS are pre-specified, which design method can be used to design the consequent parameters?
a. least-squares
b. one-pass
c. back-propagation

80. A TSK FLS can outperform a Mamdani FLS because it:
a. has more design degrees of freedom when both use the same number of rules
b. is a universal approximator
c. does not require the use of the sup-star composition

81. Within the general class of time-series forecasting problems, the problem of forecasting compressed video falls into which category?
a. deterministic-signal and noisy-measurement case
b. random-signal and noisy-measurement case
c. random-signal and perfect-measurement case

Lesson 13: APPLICATIONS OF TYPE-1 FLSs

Choose one application and answer its five questions. Do not answer the questions for the other applications.

Check-off the application you are answering the five questions for:

a. Rule-Based Classification of Video Traffic _____
b. Equalization of Time-Invariant Non-linear Digital Communication Channels _____
c. Fuzzy Logic Control _____

Now answer the respective five questions:

a. Rule-Based Classification of Video Traffic

82a. The features of a FL RBC for video traffic classification are:
a. bits per I, P and B frames
b. logarithm of bits per I, P and B frames
c. exponential of bits per I, P and B frames

83a. The overall approach to designing a RBC involves four steps performed in a correct order, taken from the following candidate steps:

1.a Establish rules using the features
1.b Evaluate the performance of the optimized RBC using testing
1.c Choose appropriate features that act as the antecedents in a RBC
1.d Optimize the rule design-parameters using a tuning procedure
1.e Cluster the features in feature space

Which of the following is the correctly ordered four steps?

a. 1.a, 1.b, 1.c, 1.d
b. 1.a, 1.e, 1.c, 1.b
c. 1.e, 1.d, 1.c, 1.b
d. 1.c, 1.a, 1.d, 1.b
e. 1.c, 1.e, 1.a, 1.d

84a. The rules for a RBC of compressed video traffic have:
a. Two antecedents and one consequent
b. Three antecedents and two consequents
c. Three antecedents and one consequent

85a. The consequent of a RB FLC for classification of video traffic is:
a. a fuzzy set
b. a linear function of its antecedent variables
c. ±1

86a. Suppose that a FL RBC gives the following results for 600 testing elements: 370 movies are correctly classified, 210 sports programs are correctly classified, 12 movies are mis-classified as sports programs, and 8 sports programs are mis-classified as movies. How many false alarms are there?
a. 12
b. 8
c. 20

b. Equalization of Time-Invariant Non-linear Digital Communication Channels

82b. The goal in channel equalization is to recover the:
a. input sequence based on a sequence of measured channel output values without knowing or estimating the channel's coefficients
b. input sequence based on a sequence of measured channel output values knowing the channel's coefficients
c. channel's coefficients based on a sequence of measured channel output values

83b. What kind of equalizer processes a finite window of past channel output measurements?
a. decision-feedback
b. transversal
c. blind

84b. A transversal equalizer for a channel of order 6 that uses a window of past measurements $r(k), r(k-1), r(k-2)$ has how many taps?
a. 2
b. 3
c. 6

85b. A channel of order 6 that is equalized by a transversal equalizer of order 4 has how many states?
a. 2^4
b. 2^{24}
c. 2^{10}

86b. For a binary input sequence, equalization is equivalent to:
a. The output of an unnormalized TSK FLS
b. The centroid defuzzified output of a Mamdani FLS
c. Two category classification

c. Fuzzy Logic Control

82c. At the highest level, we can distinguish between:
a. adaptive and non-adaptive FL controllers
b. indirect or direct FL controllers
c. linear, non-linear, or fuzzy control

83c. Controller's parameters are updated during the real-time operation of the overall system in what kind of control?
a. non-linear
b. non-adaptive fuzzy control
c. adaptive fuzzy control
d. sliding-mode control

84c. Indicate the three ways in which non-adaptive fuzzy control can be further classified:
a. linear plant
b. time-varying plant
c. sliding mode
d. non-linear plant
e. chaotic plant
f. fuzzy plant
g. constant-coefficient plant

85c. When using a fuzzy model of the plant as well as a fuzzy controller, the plant is modeled using which kind of rules?
a. R^i: IF $z_1(t)$ is F_{i1} and $\cdots$ and $z_g(t)$ is F_{ig}, THEN $\dot{\mathbf{x}}(t) = \mathbf{A}_i\mathbf{x}(t) + \mathbf{B}_i u(t),\ u(t) = \mathbf{K}_i\mathbf{x}(t)$
b. R^i: IF $z_1(t)$ is F_{i1} and $\cdots$ and $z_g(t)$ is F_{ig}, THEN $u(t) = \mathbf{K}_i\mathbf{x}(t)$
c. R^i: IF $z_1(t)$ is F_{i1} and $\cdots$ and $z_g(t)$ is F_{ig}, THEN $\dot{\mathbf{x}}(t) = \mathbf{A}_i\mathbf{x}(t) + \mathbf{B}_i u(t),\ y_i(t) = \mathbf{C}_i\mathbf{x}(t)$

86c. In indirect adaptive fuzzy control, fuzzy systems are used to model what kinds of plant non-linearities?
a. known
b. unknown
c. discontinuous
d. continuous

Lesson 14: COMPUTATION

87. Equations (5-24) and (5-25) implement a singleton:
    a. Mamdani FLS with minimum implication and t-norm
    b. TSK FLS
    c. Mamdani FLS with product implication and t-norm and Gaussian MFs
    d. Mamdani FLS with product implication and t-norm and arbitrary MFs

88. Equations (5-24), (5-25) and (5-33) implement a:
    a. non-singleton type-1 FLS
    b. singleton Mamdani FLS with product implication and t-norm and Gaussian MFs
    c. singleton Mamdani FLS with product implication and t-norm and arbitrary MFs

89. Equations (6-17), (6-18) and (13) of the Study Guide implement a non-singleton:
    a. Mamdani FLS with product implication and t-norm and Gaussian MFs
    b. Mamdani FLS with minimum implication and t-norm and Gaussian MFs
    c. TSK FLS

90. Equations (13-2), (13-3) and (13-6) implement:
    a. an un-normalized type-1 TSK FLS with Gaussian MFs
    b. a normalized second-order type-1 TSK FLS with Gaussian MFs
    c. a normalized first-order type-1 TSK FLS with Gaussian MFs
    d. a normalized first-order type-1 TSK FLS with arbitrary MFs

91. The steepest descent design equations for a non-singleton type-1 Mamdani FLS are:
    a. exactly the same as the ones for a singleton type-1 Mamdani FLS
    b. different from the ones for a singleton type-1 Mamdani FLS, but include the same number of design equations
    c. different from the ones for a singleton type-1 Mamdani FLS, and include at least one additional design equation
    d. the same as the ones for a first-order normalized type-1 TSK FLS

92. How many MATLAB M-files for implementing and designing the type-1 FLSs that are covered in this course are available on-line?
    a. more than 8
    b. 8
    c. 30

93. When the t-norm used to implement a singleton or non-singleton type-1 FLS is the minimum instead of the product, then:
    a. we can use exactly the same equations to implement such a FLS as we use to implement a type-1 FLS that uses the product t-norm
    b. a singleton type-1 TSK FLS cannot be used at all
    c. some of the equations change and the changes depend on the nature of the MFs
    d. some of the equations change and the changes do not depend on the nature of the MFs

Lesson 15: OPEN ISSUES WITH TYPE-1 FLSs

94. Fuzziness results from:
    a. conflicts among a various set of alternatives
    b. imprecise boundaries of fuzzy sets
    c. sizes of relevant sets

95. Strife results from:
    a. conflicts among a various set of alternatives
    b. imprecise boundaries of fuzzy sets
    c. sizes of relevant sets

96. Non-specificity results from:
    a. conflicts among a various set of alternatives
    b. imprecise boundaries of fuzzy sets
    c. sizes of relevant sets

97. Circle the four sources of uncertainty that can occur in a FLS:
    a. data that are used to tune the parameters of a FLS
    b. choosing triangular MFs rather than Gaussian MFs
    c. using height defuzzification rather than centroid defuzzification
    d. meanings of words used in rules
    e. measurements that activate rules
    f. choosing minimum implication rather than product implication
    g. consequents that occur in rules
    h. choosing product t-norm instead of minimum t-norm

98. Words:
    a. mean the same thing to everyone
    b. have no uncertainty associated with them
    c. mean different things to different people

99. Type-1 fuzzy sets:
    a. cannot handle uncertainties because they cannot directly model them
    b. can model all sorts of uncertainties
    c. can provide a worst-case design in which uncertainties can be modeled

100. Which kinds of sets can model the four kinds of uncertainties that can occur in a FLS and can therefore minimize their effects?
    a. crisp sets
    b. type-2 fuzzy sets
    c. type-1 fuzzy sets

SSC#1: Final Examination Solutions

1. A, D, F, H
2. C
3. B
4. B, E
5. A
6. B
7. D
8. A
9. C
10. B
11. B
12. A
13. B
14. A, B, D, F
15. B
16. A
17. B
18. C
19. C
20. C
21. A, B, E
22. B
23. C
24. C
25. B
26. A, D
27. A
28. A
29. C
30. A
31. B
32. C
33. B
34. A, C
35. C
36. B
37. C
38. B
39. A
40. A, C, D, E
41. C
42. B
43. B
44. A
45. B
46. C
47. A
48. B
49. B
50. C
51. B
52. A
53. C
54. B
55. C
56. B
57. C
58. A
59. C
60. A
61. B
62. B
63. A
64. C
65. A
66. C
67. B
68. A
69. C
70. C
71. B
72. A
73. C
74. C
75. B
76. B, E
77. C
78. C
79. A
80. A
81. C

a	b	c
82a. B	82b. A	82c. A
83a. D	83b. B	83c. C
84a. C	84b. B	84c. A, D, F
85a. C	85b. C	85c. C
86a. C	86b. C	86c. A

87. D
88. B
89. A
90. C
91. C
92. B
93. C
94. B
95. A
96. C
97. A, D, E, G
98. C
99. A
100. B