Type-2 Fuzzy Logic Systems

Nilesh N. Karnik, Jerry M. Mendel, *Fellow, IEEE*, and Qilian Liang

***Abstract*— We introduce a type-2 fuzzy logic system (FLS), which can handle rule uncertainties. The implementation of this type-2 FLS involves the operations of fuzzification, inference, and output processing. We focus on "output processing," which consists of type reduction and defuzzification. Type-reduction methods are extended versions of type-1 defuzzification methods. Type reduction captures more information about rule uncertainties than does the defuzzified value (a crisp number), however, it is computationally intensive, except for interval type-2 fuzzy sets for which we provide a simple type-reduction computation procedure. We also apply a type-2 FLS to time-varying channel equalization and demonstrate that it provides better performance than a type-1 FLS and nearest neighbor classifier.**

***Index Terms*—Channel equalization, fuzzy logic systems, interval sets, type reduction, type-2 fuzzy sets, uncertainties.**

I. Introduction

In this paper, we introduce a new class of fuzzy logic systems—*type-2 fuzzy logic systems*—in which the antecedent or consequent membership functions are type-2 fuzzy sets. The concept of a *type-2 fuzzy set* was introduced by Zadeh [44] as an extension of the concept of an ordinary fuzzy set (henceforth called a *type-1 fuzzy set*). Such sets are fuzzy sets whose membership grades themselves are type-1 fuzzy sets; they are very useful in circumstances where it is difficult to determine an exact membership function for a fuzzy set; hence, they are useful for incorporating uncertainties.

Quite often, the knowledge used to construct rules in a fuzzy logic system (FLS) is uncertain. This uncertainty leads to rules having uncertain antecedents and/or consequents, which in turn translates into uncertain antecedent and/or consequent membership functions. For example:

1) A fuzzy logic modulation classifier described in [41] centers type-1 Gaussian membership functions at constellation points on the in-phase/quadrature plane. In practice, the constellation points drift. This is analogous to the situation of a Gaussian membership function (MF) with an uncertain mean. A type-2 formulation can capture this drift.
2) Previous applications of FL to forecasting do not account for noise in training data. In forecasting, since antecedents and consequents are the same variable, the uncertainty during training exists on both the antecedents and consequents. If we have information about the level of uncertainty, it can be used when we model antecedents and consequents as type-2 sets.
3) When rules are collected by surveying experts, if we first query the experts about the locations and spreads of the fuzzy sets associated with antecedent and consequent terms, it is very likely that we will get different answers from each expert. This leads to statistical uncertainties about locations and spreads of antecedent and consequent fuzzy sets. Such uncertainties can be incorporated into the descriptions of these sets using type-2 membership functions. In addition, experts often give different answers to the same rule-question; this results in rules that have the same antecedents but different consequents. In such a case, it is also possible to represent the output of the FLS built from these rules as a fuzzy set rather than a crisp number. This can also be achieved within the type-2 framework.

Recall that our accepted probabilistic modeling of random uncertainty focuses to a large extent on methods that use *at least* the first two moments of a probability density function (pdf)—the mean and the variance. To just use the first-order moments would not be very useful, because random uncertainty requires an understanding of dispersion about the mean and this information is provided by the variance. In fuzzy logic (FL), we may view computing the defuzzified output of a type-1 FLS as analogous to computing the mean of a pdf. Just as variance provides a measure of dispersion about the mean and is almost always used to capture more about probabilistic uncertainty in practical statistical-based designs, FLS's also need some measure of dispersion to capture more about rule uncertainties than just a single number. Type-2 FL provides this measure of dispersion and seems to be as fundamental to the design of systems that include linguistic and/or numerical uncertainties that translate into rule uncertainties as variance is to the mean. Just as one can work with higher than second-order moments in probabilistic modeling, we can also use higher than type-2 sets in fuzzy modeling; but, as we go on to higher types, the complexity of the system increases rapidly. So, in this work we deal just with type-2 sets.

Type-2 fuzzy sets allow us to handle linguistic uncertainties, as typified by the adage "words can mean different things to different people [27]." A fuzzy relation of higher type (e.g., type-2) has been regarded as one way to increase the fuzziness of a relation and, according to Hisdal [11], "increased fuzziness in a description means increased ability to handle inexact information in a logically correct manner." According to John [12], "Type-2 fuzzy sets allow for linguistic grades of membership, thus assisting in knowledge representation and they also offer improvement on inferencing with type-1 sets."

Manuscript received August 2, 1999. This work was supported by the National Science Foundation under Grant MIP 9419386.

The authors are with the Signal and Image Processing Institute, Department of Electrical Engineering-Systems, University of Southern California, Los Angeles, CA 90089-2564 USA.

Publisher Item Identifier S 1063-6706(99)09882-3.

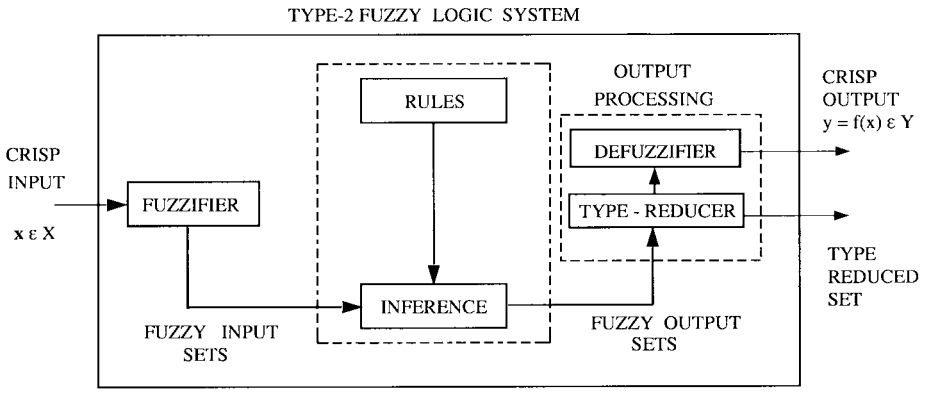


Fig. 1. The structure of a type-2 FLS. In order to emphasize the importance of the type-reduced set we have shown two outputs for the type-2 FLS, the type-reduced set and the crisp defuzzified value.

Mizumoto and Tanaka [29] studied the set theoretic operations of type-2 sets, properties of membership grades of such sets and, examined the operations of algebraic product and algebraic sum for them [30]. More details about algebraic structure of type-2 sets are given in [32]. Dubois and Prade [6]–[8] discussed fuzzy valued logic and give a formula for the composition of type-2 relations as an extension of the type-1 sup-star composition, but this formula is only for minimum t norm. In this paper, we will give a general formula for the extended sup-star composition of type-2 relations.

To date, type-2 sets and FLS's have been used in decision making [1], [43] solving fuzzy relation equations [38], survey processing [17], [16], time-series forecasting [18], [28], function approximation [16], control of mobile robots [42], and preprocessing of data [13]. We believe that other promising areas in which type-2 FLS's may be advantageous over type-1 FLS's include mobile communications, communication networks, pattern recognition, and robust control, because frequently the information to be processed in these areas is uncertain. For example, in mobile communications, the fading channel coefficients are uncertain during adjacent training periods. Existing algorithms treat them as certain. In communication networks, Tanaka and Hosaka [37] observed the difficulties of obtaining appropriate MF's for efficient network control, which suggests that type-2 MF's will be a better way to represent the uncertainty in a network. Additionally, present communication networks are dynamic and there is great deal of uncertainty associated with the input traffic and other environment parameters [10], which suggests that type-2 FLS's will also be useful for them. In this paper, we apply a type-2 FLS to time-varying nonlinear channel equalization.

In the sequel, we use the following notation and terminology: A is a type-1 fuzzy set and the membership grade (a synonym for the degree of membership) of $x \in X$ in A is $\mu_{\mathrm{A}}(x)$, which is a crisp number in $[0,1]$; a type-2 fuzzy set in X is $\tilde{\mathrm{A}}$ and the membership grade of $x \in X$ in $\tilde{\mathrm{A}}$ is $\mu_{\tilde{\mathrm{A}}}(x)$, which is a type-1 fuzzy set in $[0,1]$; the elements of the domain of $\mu_{\tilde{\mathrm{A}}}(x)$ are called *primary memberships* of x in $\tilde{\mathrm{A}}$ and the memberships of the primary memberships in $\mu_{\tilde{\mathrm{A}}}(x)$ are called *secondary memberships* of x in $\tilde{\mathrm{A}}$; the latter defines the possibilities for the primary membership, e.g., the type-2 fuzzy set $\tilde{\mathrm{A}} \in X$ has fuzzy grades (fuzzy sets in $J_x \subseteq [0,1]$), $\mu_{\tilde{\mathrm{A}}}(x)$ that can be represented for each $x \in X$ as

$$\mu_{\tilde{\mathrm{A}}}(x) = f_x(u_1)/u_1 + f_x(u_2)/u_2 + \cdots + f_x(u_m)/u_m = \sum_i f_x(u_i)/u_i, \qquad u_i \in J_x$$

the primary membership having secondary membership equal to one is called the *principal* MF; when the secondary MF's of a type-2 fuzzy set are type-1 Gaussian MF's we call the type-2 fuzzy set a *Gaussian type-2 set* (regardless of the shape of the primary MF); when the secondary MF's are type-1 interval sets we call the type-2 set an *interval type-2 set*; $\sqcap$ denotes *meet* operation; and $\sqcup$ denotes *join* operation. Meet and join are defined and explained in great detail in [16], [19].

Section II presents a comparison of type-1 and type-2 FLSs; Section III describes the inference in a type-2 FLS; type reduction is discussed in detail in Section IV; defuzzification is described in Section V; a simple type-reduction procedure for interval type-2 FLS's is described in Section VI; an application of type-2 FLS's to time-varying channel equalization is presented in Section VII and, conclusions and future work are given in Section VIII.

II. Type-2 FLSs: Overview

Fig. 1 shows the structure of a type-2 FLS. It is very similar to the structure of a type-1 FLS [26]. For a type-1 FLS, the *output processing* block only contains the defuzzifier. We assume that the reader is familiar with type-1 FLS's, so that here we focus only on the similarities and differences between the two FLS's.

The fuzzifier maps the crisp input into a fuzzy set. This fuzzy set can, in general, be a type-2 set, however, in this paper, we consider only *singleton* fuzzification, for which the input fuzzy set has only a single point of nonzero membership.

In the type-1 case, we generally have "IF-THEN" rules, where the lth rule has the form "R^l: IF x_1 is F_1^l and x_2 is F_2^l and $\cdots$ and x_p is F_p^l, THEN y is G^l," where: x_is are inputs; F_i^ls are antecedent sets ($i = 1, \cdots, p$); y is the output; and G^ls are consequent sets. The distinction between type-1 and type-2 is associated with the nature of the membership functions, which is not important while forming rules; hence,

the structure of the rules remains exactly the same in the type-2 case, the only difference being that now some or all of the sets involved are of type-2; so, the lth rule in a type-2 FLS has the form "R^l: IF x_1 is $\tilde{\mathrm{F}}_1^l$ and x_2 is $\tilde{\mathrm{F}}_2^l$ and $\cdots$ and x_p is $\tilde{\mathrm{F}}_p^l$, THEN y is $\tilde{\mathrm{G}}^l$." It is not necessary that all the antecedents and the consequent be type-2 fuzzy sets. As long as one antecedent or the consequent set is type-2, we will have a type-2 FLS.

In a type-1 FLS, the inference engine combines rules and gives a mapping from input type-1 fuzzy sets to output type-1 fuzzy sets. Multiple antecedents in rules are connected by the t-norm (corresponding to intersection of sets). The membership grades in the input sets are combined with those in the output sets using the *sup-star* composition. Multiple rules may be combined using the t-conorm operation (corresponding to union of sets) or during defuzzification by weighted summation. In the type-2 case, the inference process is very similar. The inference engine combines rules and gives a mapping from input type-2 fuzzy sets to output type-2 fuzzy sets. To do this one needs to find unions and intersections of type-2 sets, as well as compositions of type-2 relations.

In a type-1 FLS, the defuzzifier produces a crisp output from the fuzzy set that is the output of the inference engine, i.e., a type-0 (crisp) output is obtained from a type-1 set. In the type-2 case, the output of the inference engine is a type-2 set; so we use "extended versions" (using Zadeh's extension principle [8], [16], [44]) of type-1 defuzzification methods. This extended defuzzification gives a type-1 fuzzy set. Since this operation takes us from the type-2 output sets of the FLS to a type-1 set, we call this operation "type reduction" and the type-reduced set so obtained a "type-reduced set."

To obtain a crisp output from a type-2 FLS, we can defuzzify the type-reduced set. The most natural way of doing this seems to be by finding the centroid of the type-reduced set, however, there exist other possibilities like choosing the highest membership point in the type-reduced set.

From our discussions so far, we see that in order to develop a type-2 FLS, one needs to be able to: 1) perform the set theoretic operations of *union*, *intersection*, and *complement* on type-2 sets [19]; 2) know properties (e.g., *commutativity*, *associativity*, *identity laws*) of membership grades of type-2 sets [19]; 3) deal with type-2 fuzzy relations and their compositions [19]; and 4) perform type reduction and defuzzification to obtain a set-valued or crisp output from the FLS [16], [15].

III. Inference in a Type-2 FLS

Consider a type-2 FLS having p inputs, $x_1 \in X_1$, $x_2 \in X_2, \cdots, x_p \in X_p$, and one output $y \in Y$. Let us suppose that it has M rules where the lth rule has the form

$$R^l: \quad \text{IF } x_1 \text{ is } \tilde{\mathrm{F}}_1^l \text{ and } x_2 \text{ is } \tilde{\mathrm{F}}_2^l \text{ and } \cdots \text{and } x_p \text{ is } \tilde{\mathrm{F}}_p^l, \text{THEN } y \text{ is } \tilde{\mathrm{G}}^l. \tag{1}$$

This rule represents a type-2 fuzzy relation between the input space $X_1 \times X_2 \times \cdots \times X_p$ and the output space Y of the FLS. We denote the membership function of this type-2 relation as $\mu_{\tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l \to \tilde{\mathrm{G}}^l}(\mathbf{x}, y)$, where $\tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l$ denotes the Cartesian product of $\tilde{\mathrm{F}}_1^l, \tilde{\mathrm{F}}_2^l, \cdots, \tilde{\mathrm{F}}_p^l$, and $\mathbf{x} = \{x_1, x_2, \cdots, x_p\}$.

When an input $\mathbf{x}'$ is applied, the composition of the fuzzy set $\tilde{\mathbf{X}}'$ to which $\mathbf{x}'$ belongs and the rule R^l is found by using the extended sup-star composition [19], [14], [16]

$$\mu_{\tilde{\mathbf{X}}' \circ \tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l \to \tilde{\mathrm{G}}^l}(y) = \sqcup_{\mathbf{x} \in \tilde{\mathbf{X}}'} [\mu_{\tilde{\mathbf{X}}'}(\mathbf{x}) \sqcap \mu_{\tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l \to \tilde{\mathrm{G}}^l}(\mathbf{x}, y)]. \tag{2}$$

We use singleton fuzzification, hence, the fuzzy set $\tilde{\mathbf{X}}'$ is such that it has a membership grade 1 corresponding to $\mathbf{x} = \mathbf{x}'$ and has zero membership grades for all other inputs; therefore, (2) reduces to

$$\mu_{\tilde{\mathbf{X}}' \circ \tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l \to \tilde{\mathrm{G}}^l}(y) = \mu_{\tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l \to \tilde{\mathrm{G}}^l}(\mathbf{x}', y). \tag{3}$$

We denote $\tilde{\mathbf{X}}' \circ \tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l \to \tilde{\mathrm{G}}^l$ as $\tilde{\mathrm{B}}^l$, the output set corresponding to the lth rule. The right-hand side (RHS) of (3) is computed using the implication membership function. Since engineering applications generally use product or minimum implication (see [26] for more details), corresponding to the *meet* operation with product or minimum t-norm, (3) can be rewritten as

$$\mu_{\tilde{\mathrm{B}}^l}(y) = \mu_{\tilde{\mathrm{F}}_1^l \times \cdots \times \tilde{\mathrm{F}}_p^l}(\mathbf{x}') \sqcap \mu_{\tilde{\mathrm{G}}^l}(y). \tag{4}$$

The membership function for a Cartesian product of sets is computed by finding the *meet* between the membership functions of individual sets [19], [16]; hence, (4) can be rewritten as

$$\begin{aligned} \mu_{\tilde{\mathrm{B}}^l}(y) &= \mu_{\tilde{\mathrm{F}}_1^l}(x_1) \sqcap \mu_{\tilde{\mathrm{F}}_2^l}(x_2) \sqcap \cdots \sqcap \mu_{\tilde{\mathrm{F}}_p^l}(x_p) \sqcap \mu_{\tilde{\mathrm{G}}^l}(y) \\ &= \mu_{\tilde{\mathrm{G}}^l}(y) \sqcap [\sqcap_{i=1}^p \mu_{\tilde{\mathrm{F}}_i^l}(x_i)] \end{aligned} \tag{5}$$

where we are using the same operation for the t-norm and the inference $\cdots$ product or minimum and have used the fact that type-2 membership functions commute for minimum or product t-norms [19], [16].

Example 3.1: Fig. 2 shows an example of product and minimum inference for an arbitrary single-input single-output type-2 FLS using Gaussian type-2 sets. Uncertainty in the primary membership grades of a type-2 MF consists of a dark region that we call the *footprint of uncertainty* of a type-2 MF. The footprint of uncertainty represents the union of all primary memberships. Darker areas indicate higher secondary memberships. The *principal* membership function, i.e., the set of primary memberships having secondary membership equal to one, is indicated with a thick line. The product inference function (5) in Fig. 2(c) was obtained by finding the *meet* [16], [19] under product t-norm of the membership grade of $x = 4$ with the membership grade of every point of the consequent function in Fig. 2(b). Similarly, the minimum inference function (5) in Fig. 2(d) was obtained by finding the *meet* under minimum t-norm [19], [16] of the membership grade of $x = 4$ with the membership grade of every point of the consequent function. Observe that in both cases, the result of the inference is a type-2 set [Fig. 2(c) and (d)]. We can interpret the banded behavior about the output's principal membership function as an indication of combined antecedent and consequent uncertainties. □

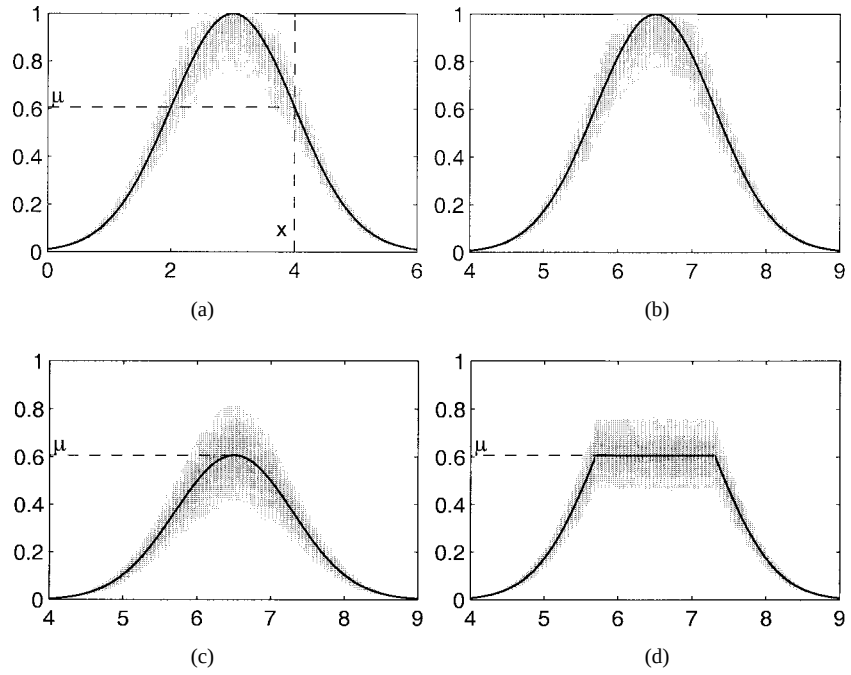


Fig. 2. Illustrations of product and minimum inference in the type-2 case. (a) Gaussian type-2 antecedent set for a single input system. The membership of a certain input $x = 4$ in the principal membership function is also shown, equal to μ. (b) Consequent set corresponding to the antecedent set shown in (a). (c) Scaled consequent set for $x = 4$ using product inference. Observe that the secondary membership functions of the consequent set also change depending upon the standard deviation of the membership grade of x. (d) Clipped consequent set for $x = 4$ using minimum inference.

IV. Type Reduction

In a type-1 FLS the output corresponding to each fired rule is a type-1 set in the output space. The defuzzifier combines the output sets corresponding to all the fired rules in some way to obtain a single output set and then finds a crisp number that is representative of this combined output set, e.g., the centroid defuzzifier finds the union of all the output sets and uses the centroid of the union as the crisp output. *In all the defuzzifiers of interest to us, the crisp number is obtained as the centroid of some combined output set.*

For type-1 defuzzification methods, we assume that the FLS is type-1 (i.e., all the antecedent and consequent sets in (1) are assumed type-1); and for type-reduction methods, we assume that the FLS is type-2 (i.e., some or all of the antecedent and consequent sets in (1) are assumed type-2). In each case, the antecedent and consequent membership grades are assumed to have a continuous domain and integrals indicate logical union.

The output set corresponding to each rule in (1) of the type-2 FLS is a type-2 set (5). The type-reducer combines all these output sets in some way (just like a type-1 defuzzifier combines the type-1 rule output sets) and then performs a centroid calculation on this type-2 set, which leads to a type-1 set that we call the "type-reduced" set.

The centroid of a type-1 set A, whose domain is discretized into N points is given as

$$C_{\mathrm{A}} = \frac{\sum_{i=1}^{N} x_i \mu_{\mathrm{A}}(x_i)}{\sum_{i=1}^{N} \mu_{\mathrm{A}}(x_i)}. \tag{6}$$

Similarly, the centroid of a type-2 set $\tilde{\mathrm{A}}$, whose domain is discretized into N points, can be defined using the extension principle (see [16] and [20] for more details) as follows. We let $\mathrm{D}_i = \mu_{\tilde{\mathrm{A}}}(x_i)$, so that

$$C_{\tilde{\mathrm{A}}} = \int_{\theta_1} \cdots \int_{\theta_N} [\mu_{\mathrm{D}_1}(\theta_1) \star \cdots \star \mu_{\mathrm{D}_N}(\theta_N)] \Big/ \frac{\sum_{i=1}^{N} x_i \theta_i}{\sum_{i=1}^{N} \theta_i} \tag{7}$$

where $\theta_i \in \mathrm{D}_i$. Equation (7) can be described in words as follows. Each point x_i of $\tilde{\mathrm{A}}$ has a type-1 fuzzy membership grade $\mathrm{D}_i = \mu_{\tilde{\mathrm{A}}}(x_i)$ associated with it. To find the centroid, we consider every possible combination $\{\theta_1, \cdots, \theta_N\}$ such that $\theta_i \in \mathrm{D}_i$. For every such combination, we perform the type-1 centroid calculation in (6) by using θ_i in place of $\mu_{\tilde{\mathrm{A}}}(x_i)$; and to each point in the centroid we assign a membership grade equal to the t-norm of the membership grades of the θ_i in the D_i. If more than one combination of θ_i gives us the same point in the centroid, we keep the one with the largest membership grade. If we let $\Sigma_{i=1}^{N}\ x_i\theta_i / \Sigma_{i=1}^{N}\ \theta_i = x$, then (7) can also be written as

$$C_{\tilde{\mathrm{A}}} = \int_{x} \sup_{\{\theta_1, \cdots, \theta_N\}} [\mu_{\mathrm{D}_1}(\theta_1) \star \cdots \star \mu_{\mathrm{D}_N}(\theta_N)]/x \tag{8}$$

where $\{\theta_1, \cdots, \theta_N\}$ are such that $\Sigma_{i=1}^{N}\ x_i\theta_i / \Sigma_{i=1}^{N}\ \theta_i = x$, and $\star$ indicates the chosen t-norm.

Every combination $\{\theta_1, \cdots, \theta_N\}$ can be thought to form the membership function of some type-1 set A', which has the same domain as $\tilde{\mathrm{A}}$. We call A' an *embedded type-1 set*

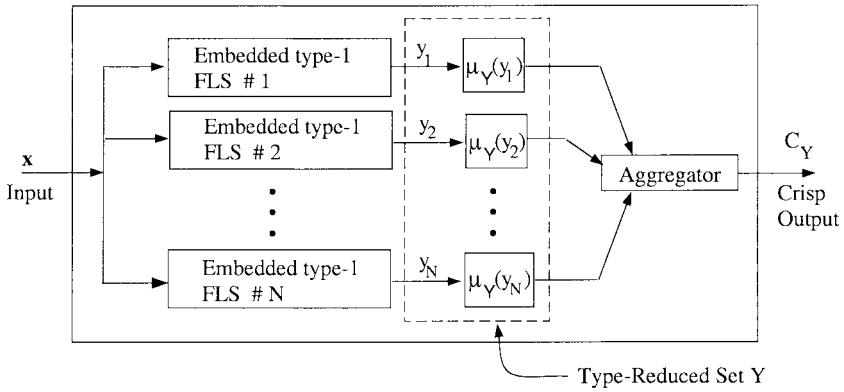


Fig. 3. A type-2 FLS can be thought of as a collection of a large number of type-1 FLS's. The type-reduced set Y is a collection of the outputs of all these embedded type-1 FLS's. When the antecedent and consequent membership grades in the type-2 FLS have a continuous domain, the number of embedded type-1 FLS's is uncountable. This figure is drawn assuming that the membership grades have discrete (or discretized) domains. The memberships in the type-reduced set represent the level of uncertainty associated with each embedded type-1 FLS. A crisp output can be obtained by aggregating the outputs of all the embedded type-1 FLS's by, for example, finding the centroid of the type-reduced set (see Section V).

in $\tilde{\mathrm{A}}$. The centroid $C_{\tilde{\mathrm{A}}}$ is a type-1 set whose elements are the centroids of all the embedded type-1 sets in $\tilde{\mathrm{A}}$. The membership of an embedded set centroid in $C_{\tilde{\mathrm{A}}}$ is calculated as the t-norm of the secondary memberships corresponding to $\{\theta_1, \cdots, \theta_N\}$ that make up that embedded set.

Observe that if the domain of $\tilde{\mathrm{A}}$ and/or $\mu_{\tilde{\mathrm{A}}}(x)$ $(x \in \tilde{\mathrm{A}})$ is continuous, the domain of $C_{\tilde{\mathrm{A}}}$ is also continuous. The number of all the embedded type-1 sets in $\tilde{\mathrm{A}}$ in this case is uncountable; therefore, the domains of $\tilde{\mathrm{A}}$ and each $\mu_{\tilde{\mathrm{A}}}(x)$ $(x \in \tilde{\mathrm{A}})$ have to be discretized for the calculation of $C_{\tilde{\mathrm{A}}}$ (as explained in Appendix A, we always use minimum t-norm for the centroid calculation of a type-2 set having a continuous domain even though we use product t-norm everywhere else). Observe from (7) that if the domain of each D_l is discretized into M points, the number of possible $\{\theta_1, \cdots, \theta_N\}$ combinations is M^N, which can be very large even for small M and N. If, however, the membership functions of D_l's have a regular structure (e.g., uniform (interval type-1 sets), Gaussian, triangular), we can obtain the exact or approximate centroid without having to do all the calculations [16], [20]. For interval type-1 sets, the actual centroid can be obtained relatively easily without performing computations for all the combinations by using the computational procedure described in Section VI.

The type-reduced set of a type-2 FLS is the centroid of a type-2 output set for the FLS; consequently, each element of the type-reduced set is the centroid of some type-1 set embedded in the output set of the type-2 FLS. Each of these embedded sets can be thought of as an output set of some type-1 FLS and, correspondingly, the type-2 FLS can be thought of as a collection of many different type-1 FLS's. Each of these type-1 FLS's is *embedded* in the type-2 FLS, so *the type-reduced set is a collection of the outputs of all the type-1 FLS's embedded in the type-2 FLS* (see Fig. 3) and it lets us represent the output of the type-2 FLS as a fuzzy set rather than as a crisp number.

It can be shown from (5) that if all the antecedent and consequent membership grades of the type-2 FLS are normal and have only one point having unity secondary membership, then the output set membership grade of every $y \in Y$ will also be normal and will have only one point having unity secondary membership. Consequently, the type-reduced set will also be normal and will have only one point having unity membership. This point will correspond to the centroid of the principal membership function of the type-2 output set. If all the type-2 uncertainties in this FLS were to collapse to type-1 uncertainties, i.e., if all the type-2 membership functions in the FLS were to collapse to their principal membership functions, the antecedent and consequent membership grades of each point would collapse to their unity membership points. This would cause the type-reduced set to collapse to its unity height point. This shows that all our results are valid when all the type-2 uncertainties collapse to type-1 uncertainties, in which case our type-2 FLS collapses to a type-1 FLS.

If we think of a type-2 FLS as a "perturbed" version of a type-1 FLS, due to uncertainties in the membership functions, the type-reduced set of the type-2 FLS can then be thought of as representing the uncertainty in the crisp output due to uncertainties in the membership functions. Some measure of the spread of the type-reduced set may then be taken to indicate the possible variation in the crisp output due to variations in the membership function parameters. This is analogous to using confidence intervals in a stochastic-uncertainty situation; however, what we have developed is for *linguistic uncertainties*.

In the following subsections, we present details for centroid, height, and center-of-sets type reduction. Details for modified height and center-of-sums type reduction can be found in [16].

A. Centroid Type Reduction

The *centroid* defuzzifier [21] combines the output type-1 sets using a t-conorm and then finds the centroid of this set. If we denote the composite output fuzzy set as B, the centroid defuzzifier is given as

$$y_c(\mathbf{x}) = \frac{\sum_{i=1}^{N} y_i \mu_{\mathrm{B}}(y_i)}{\sum_{i=1}^{N} \mu_{\mathrm{B}}(y_i)} \tag{9}$$

where the output set B is discretized into N points.

The centroid type-reducer combines all the output type-2 sets by finding their union. The membership grade of $y \in Y$ is given as

$$\mu_{\tilde{\mathrm{B}}}(y) = \sqcup_{l=1}^{M} \mu_{\tilde{\mathrm{B}}^l}(y) \tag{10}$$

where $\mu_{\tilde{\mathrm{B}}^l}(y)$ is as defined in (5). The centroid type-reducer then calculates the centroid of $\tilde{\mathrm{B}}$. The expression for the centroid type-reduced set is an extended version of (9), i.e.,

$$\mathrm{Y}_c(\mathbf{x}) = \int_{\theta_1} \cdots \int_{\theta_N} [\mu_{\mathrm{D}_1}(\theta_1) \star \cdots \star \mu_{\mathrm{D}_N}(\theta_N)] \Big/ \frac{\sum_{i=1}^{N} y_i \theta_i}{\sum_{i=1}^{N} \theta_i} \tag{11}$$

where $\mathrm{D}_i = \mu_{\tilde{\mathrm{B}}}(y_i)$ and $\theta_i \in \mu_{\tilde{\mathrm{B}}}(y_i)$ $(i = 1, \cdots, N)$. Let

$$a \triangleq \frac{\sum_{i=1}^{N} y_i \theta_i}{\sum_{i=1}^{N} \theta_i} \tag{12}$$

$$b \triangleq \mu_{\mathrm{D}_1}(\theta_1) \star \cdots \star \mu_{\mathrm{D}_N}(\theta_N). \tag{13}$$

The computation of $\mathrm{Y}_c(\mathbf{x})$ involves computing the tuple (a, b) many times. Suppose, for example, (a, b) is computed α times; then we can view the computation of $\mathrm{Y}_c(\mathbf{x})$ as the computation of the α tuples (a_1, b_1), (a_2, b_2), $\cdots$, (a_α, b_α). Important information from type reduction is the domain of the a_i's, i.e., a_L and a_R, where $a_L = \min a_i$ and $a_R = \max a_i$, $(i = 1, \cdots, \alpha)$, because it is the bound $[a_L, a_R]$ that we associate with a linguistic confidence interval. Only storing a_L and a_R, instead of all α a_i reduces the storage requirements a lot. How to efficiently use all α a_i and b_i to perform defuzzification is discussed in Section V.

The sequence of computations needed to obtain $\mathrm{Y}_c(\mathbf{x})$ is as follows.

1) Compute the combined output set using (10). This is possible, because we can first compute $\mu_{\tilde{\mathrm{B}}^l}(y)$ $(l = 1, \cdots, M)$ for all $y \in Y$ using (5). Theorem 1 (or Theorem 2) in [19] is used to compute (10).
2) Discretize the output space Y into N points, $y_1, \cdots, y_N$.
3) Discretize the domain of each $\mu_{\tilde{\mathrm{B}}}(y_i)$ $(i = 1, \cdots, N)$ into a suitable number of points.
4) Enumerate all the embedded sets. For example, if each $\mu_{\tilde{\mathrm{B}}}(y_i)$ is discretized into M_i points, there will be $\Pi_{j=1}^{N} M_j$ embedded sets.
5) Compute the centroid type-reduced set using (11), i.e., compute the tuples (a_i, b_i) $i = 1, 2, \cdots, \Pi_{j=1}^{N} M_j$, where a_i and b_i are given in (12) and (13), respectively. We must use the minimum t-norm here, as explained in Appendix A.

In step 5, the centroid and membership computation has to be repeated $\Pi_{j=1}^{N} M_j$ times and so, in general, will involve an enormous amount of computation. Computational complexity and ways to reduce it are discussed in Section IV-E for all three of the type-reduction methods described herein.

B. Height Type Reduction

The height defuzzifier [5] replaces each rule output set by a singleton at the point having maximum membership in that output set and then calculates the centroid of the type-1 set comprised of these singletons. The output of a height defuzzifier is given as

$$y_h(\mathbf{x}) = \frac{\sum_{l=1}^{M} \overline{y}^l \mu_{\mathrm{B}^l}(\overline{y}^l)}{\sum_{l=1}^{M} \mu_{\mathrm{B}^l}(\overline{y}^l)} \tag{14}$$

where $\overline{y}^l$ is the point having maximum membership in the lth output set (if there is more than one such point, their average can be taken as $\overline{y}^l$); and its membership grade in the lth output set $\mu_{\mathrm{B}^l}(\overline{y}^l)$ is given as [26]

$$\mu_{\mathrm{B}^l}(\overline{y}^l) = \mu_{\mathrm{G}^l}(\overline{y}^l) \star \mathcal{T}_{i=1}^{p} \mu_{\mathrm{F}_i^l}(x_i) \tag{15}$$

where $\star$ and $\mathcal{T}$ indicate the chosen t-norm (assuming the inference uses the same operation as the t-norm $\cdots$ product or minimum).

The height type reducer replaces each output set by a type-2 singleton, i.e., by a fuzzy set whose domain consists of a single point, the membership of which is a type-1 set in $[0, 1]$. The lth output set is replaced by a singleton situated at $\overline{y}^l$, where $\overline{y}^l$ can be chosen to be the point having the highest membership in the principal membership function of output set $\tilde{\mathrm{B}}^l$. If $\tilde{\mathrm{B}}^l$ is such that a principal membership function cannot be defined, one may choose $\overline{y}^l$ as the point having the highest primary membership with a secondary membership equal to one or as a point satisfying some similar criterion. The membership grade of $\overline{y}^l$ in the type-2 case is obtained from (5) as follows:

$$\mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l) = \mu_{\tilde{\mathrm{G}}^l}(\overline{y}^l) \sqcap [\sqcap_{i=1}^{p} \mu_{\tilde{\mathrm{F}}_i^l}(x_i)]. \tag{16}$$

If we let $\mathrm{D}^l = \mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$, the expression for the type-reduced set is obtained as an extension of (14) as

$$\mathrm{Y}_h(\mathbf{x}) = \int_{\theta_1} \int_{\theta_2} \cdots \int_{\theta_M} [\mu_{\mathrm{D}^1}(\theta_1) \star \cdots \star \mu_{\mathrm{D}^M}(\theta_M)] \Big/ \frac{\sum_{l=1}^{M} \overline{y}^l \theta_l}{\sum_{l=1}^{M} \theta_l} \tag{17}$$

where $\theta_l \in \mathrm{D}^l$ for $l = 1, \cdots, M$. In this case, $a = (\Sigma_{l=1}^{M} \overline{y}^l \theta_l / \Sigma_{l=1}^{M} \theta_l)$ and b is still given by (13).

The sequence of computations needed to obtain $Y_h(\mathbf{x})$ is as follows.

1) Choose $\overline{y}^l$ for each output set $l = 1, 2, \cdots, M$ and compute $\mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$ in (16).

2) Discretize the domain of each $\mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$ into a suitable number of points. The discretization is carried out in a manner similar to that for centroid type reduction, the only difference is that the number of points on the horizontal axis is now M instead of N.
3) Enumerate all the possible combinations $\{\theta_1, \cdots, \theta_M\}$ such that $\theta_l \in \mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$. For example, if $\mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$ is discretized into N_l points, there will be $\Pi_{j=1}^{M} N_j$ combinations.
4) Compute the height type-reduced set using (17). Since the domain of the combined output set is discrete, we can now use product or minimum t-norm in (17), as explained in Appendix A.

In step 4, the weighted sum and membership computations in (17) have to be repeated $\Pi_{j=1}^{M} N_j$ times; but, generally, $\Pi_{j=1}^{M} N_j \ll \Pi_{j=1}^{N} N_j$ (where N is the number of discrete y points in case of centroid type reduction), so computing the height type-reduced set involves much fewer computations than computing the centroid type-reduced set. Our discussion about storing a_L and a_R in Section IV-A applies here as well.

C. Center-of-Sets Type Reduction

For reasons that will be explained in Section IV-D, we introduce a new center-of-sets defuzzification method in which we replace each rule consequent set by a singleton situated at its *centroid* and then find the centroid of the type-1 set comprised of these singletons. The expression for the output is given as

$$y_{cos}(\mathbf{x}) = \frac{\sum_{l=1}^{M} c^l \mathcal{T}_{i=1}^{p} \mu_{\mathrm{F}_i^l}(x_i)}{\sum_{l=1}^{M} \mathcal{T}_{i=1}^{p} \mu_{\mathrm{F}_i^l}(x_i)} \tag{18}$$

where $\mathcal{T}$ indicates the chosen t-norm and c^l is the centroid of the lth consequent set. Observe that if each consequent set is symmetric, normal and convex, $c^l = \overline{y}^l$ and $\mu_{\mathrm{G}^l}(\overline{y}^l) = 1$ for $l = 1, \cdots, M$; hence, in this (special but important) case, $y_{cos}(\mathbf{x}) = y_h(\mathbf{x})$.

The center-of-sets type reducer replaces each consequent set by its centroid (which itself is a type-1 set if the consequent set is type-2) and finds a weighted average of these centroids, where the weight associated with the lth centroid is the degree of firing corresponding to the lth rule, namely $\sqcap_{i=1}^{p} \mu_{\tilde{\mathrm{F}}_i^l}(x_i)$. The expression for the type-reduced set is the following extension of (18):

$$\mathrm{Y}_{cos}(\mathbf{x}) = \int_{d_1} \cdots \int_{d_M} \int_{e_1} \cdots \int_{e_M} \mathcal{T}_{l=1}^{M} \mu_{\mathrm{C}_l}(d_l) \star \mathcal{T}_{l=1}^{M} \mu_{\mathrm{E}_l}(e_l) \Big/ \frac{\sum_{l=1}^{M} d_l e_l}{\sum_{l=1}^{M} e_l} \tag{19}$$

where $\mathcal{T}$ and $\star$ indicate the chosen t-norm; $d_l \in \mathrm{C}_l = C_{\tilde{\mathrm{G}}^l}$ the centroid of the lth consequent set and $e_l \in \mathrm{E}_l = \sqcap_{i=1}^{p} \mu_{\tilde{\mathrm{F}}_i^l}(x_i)$, the degree of firing associated with the lth consequent set for $l = 1, \cdots, M$. In this case, $a = (\Sigma_{l=1}^{M}\ d_l e_l / \Sigma_{l=1}^{M}\ e_l)$ and $b = \mathcal{T}_{l=1}^{M} \mu_{\mathrm{C}_l}(d_l) \star \mathcal{T}_{l=1}^{M} \mu_{\mathrm{E}_l}(e_l)$.

The sequence of computations needed to obtain $\mathrm{Y}_{cos}(\mathbf{x})$ is as follows.

1) Discretize the output space Y into a suitable number of points and compute the centroid $C_{\tilde{\mathrm{G}}^l}$ of each consequent set on the discretized output space using (7). This is possible because we know $\mu_{\tilde{\mathrm{G}}^l}(y)$ $(l = 1, 2, \cdots, M)$ for all $y \in Y$. *These consequent centroid sets can be computed ahead of time and stored for future use.*
2) Compute the degree of firing $\mathrm{E}_l = \sqcap_{i=1}^{p} \mu_{\tilde{\mathrm{F}}_i^l}(x_i)$ associated with the lth consequent set using the results in [16], [19] $l = 1, 2, \cdots, M$.
3) Discretize the domain of each $C_{\tilde{\mathrm{G}}^l}$ into a suitable number of points, say M_l $(l = 1, 2, \cdots, M)$.
4) Discretize the domain of each E_l into a suitable number of points, say N_l $(l = 1, 2, \cdots, M)$.
5) Enumerate all the possible combinations $\{c_1, \cdots, c_M, e_1, \cdots, e_M\}$ such that $d_l \in C_{\tilde{\mathrm{G}}^l}$ and $e_l \in \mathrm{E}_l$. The total number of combinations will be $\Pi_{j=1}^{M} M_j N_j$.
6) Compute the type-reduced set using (19).

In step 6, the weighted sum and t-norm operations in (19) have to be repeated $\Pi_{j=1}^{M} M_j N_j$ times. Observe, from Sections IV-A and IV-B, that this number is, in general, larger than that required for a height type reducer, but is less than that required for a centroid type reducer. Our discussion about a_L and a_R in Section IV-A applies here as well.

D. Comparison of Height and Center-of-Sets Type Reducers

In a type-1 FLS, height defuzzification is computationally inexpensive and gives satisfactory results (see, for example, [5] and [39]). In a type-2 FLS, however, height type reduction may not perform so well. Center-of-sets type reduction does a better job. Here we explain why this is so.

For expressions for the height and center-of-sets type-reduced sets, see (17) and (19), respectively. When only one rule is fired corresponding to $l = l'$, the height type-reduced set is (assuming maximum t-conorm and normal membership grades)

$$\mathrm{Y}_h(\mathbf{x}) = \int_{\theta_{l'}} \mu_{\mathrm{D}^{l'}}(\theta_{l'}) \Big/ \frac{\overline{y}^{l'} \theta_{l'}}{\theta_{l'}} = [\sup_{l'} \mu_{\mathrm{D}^{l'}}(\theta_{l'})] / \overline{y}^{l'} = 1/\overline{y}^{l'} = \overline{y}^{l'}. \tag{20}$$

Equation (20) shows that when a single rule is fired, the height type-reduced set collapses to a single point! This is certainly undesirable because it means that when the input is such that only one rule is fired, no uncertainty is associated with the output, which is generally not true.

We invented the center-of-sets type-reducer to avoid this problem. It is easy to see that even when only a single rule is fired, $\mathrm{Y}_{cos}(\mathbf{x})$ is a type-1 fuzzy set that equals the centroid of

TABLE I
COMPUTATION COMPLEXITY FOR EACH PARALLEL PROCESSOR OF DIFFERENT TYPE REDUCERS. THE NUMBER OF FIRED RULES IS M. THE NUMBER OF ANTECEDENTS IS p. THE FIRED PRIMARY MEMBERSHIP DEGREE $\mu_{\tilde{\mathrm{F}}_j^i}(x)$ $(i = 1, \cdots, M,\ j = 1, \cdots, p)$ IS SAMPLED TO N_{ij} POINTS. THE CONSEQUENT MEMBERSHIP DEGREE $\mu_{\tilde{\mathrm{B}}}(y_k)$ $(k = 1, \cdots, N)$ IS SAMPLED TO M_k POINTS. THE CENTER OF THE iTH CONSEQUENT SET CONTAINS K_i POINTS. THE TOTAL OUTPUT DOMAIN IS SAMPLED TO N POINTS

Type-reducer	t-norm	multiplications	additions	divisions	parallel processors
Centroid	$N-1$	N	$2(N-1)$	1	$\prod_{k=1}^{N} M_k$
Height	$M-1$	M	$2(M-1)$	1	$\prod_{i=1}^{M}\prod_{j=1}^{p} N_{ij}$
Center-of-sets	$2M-1$	$M-1$	$2(M-1)$	1	$\prod_{i=1}^{M}\prod_{j=1}^{p} K_i N_{ij}$

the type-2 output set corresponding to the fired rule, i.e.,

$$\begin{aligned} \mathrm{Y}_{cos}(\mathbf{x}) &= \int_{d_{l'}}\int_{e_{l'}} \mu_{\mathrm{C}_{l'}}(d_{l'})\mu_{\mathrm{E}_{l'}}(e_{l'}) \Big/ \frac{d_{l'}e_{l'}}{e_{l'}} \\ &= \int_{d_{l'}} \mu_{\mathrm{C}_{l'}}(d_{l'}) \int_{e_{l'}} \mu_{\mathrm{E}_{l'}}(e_{l'}) \Big/ d_{l'} \\ &= \int_{d_{l'}} \mu_{\mathrm{C}_{l'}}(d_{l'})[\sup_{e_{l'}} \mu_{\mathrm{E}_{l'}}(e_{l'})] \Big/ d_{l'} \\ &= \int_{d_{l'}} \mu_{\mathrm{C}_{l'}}(d_{l'}) \Big/ d_{l'} = \mathrm{C}_{l'} \end{aligned} \tag{21}$$

where we have again assumed that the degree of firing associated with the l'th consequent is normal. (If each $\mu_{\tilde{\mathrm{F}}_i^l}(x_i)$ is normal, $\mathrm{E}_l = \sqcap_{i=1}^{p}\mu_{\tilde{\mathrm{F}}_i^l}(x_i)$ will also be normal (see [16] and [19] for a detailed discussion of the *meet* operation.)

Finally, note that in order to measure the effects of *both* antecedent and consequent measurements when only one rule is fired, one needs to use centroid type reduction.

E. Computational Complexity

We have introduced three type-reduction methods. Unfortunately, they have high-computation complexity. Fortunately, however, a type-2 FLS can be thought of as a collection of a large number of type-1 FLS's. As described in the beginning of Section IV, the type-reduced set Y is a collection of the outputs of all these embedded type-1 FLS's, as shown in Fig. 3. The operations for each type-1 FLS can be processed in parallel; hence, the computational complexity for each type-1 FLS is the same as its corresponding defuzzification operation. The number of parallel processors depends on how many type-1 FLS's are needed for a specific type-reduction method and this depends on the sampling rates in the primary membership and output domains. The computations for a in (12) show the computational complexity in terms of multiplications, additions, and divisions; and the computations for b in (13) give the number of operations for t-norm. The number of parallel processors equals the number of tuples (a_i, b_i). In Table I, we summarize the computation complexity for the three type reducers described above. For example, in centroid type reduction, there are $N-1$ t-norm operations, N multiplications, $2(N-1)$ additions and one division, and $\Pi_{k=1}^{N}\ M_k$ parallel processors are required. In actual computation, not all M rules are fired, so the computation will be simplified a lot.

V. DEFUZZIFICATION

We *defuzzify* the type-reduced set to get a crisp output from the type-2 FLS. The most natural way of doing this seems to be by finding the centroid of the type-reduced set. Finding the centroid is equivalent to finding a weighted average of the outputs of all the type-1 FLS's embedded in the type-2 FLS, where the weights correspond to the memberships in the type-reduced set (see Fig. 3). If the type-reduced set Y for an input $\mathbf{x}$ is discrete and consists of N points, the expression for its centroid is

$$C_{\mathrm{Y}}(\mathbf{x}) = \frac{\sum_{k=1}^{N} y_k \mu_{\mathrm{Y}}(y_k)}{\sum_{k=1}^{N} \mu_{\mathrm{Y}}(y_k)}. \tag{22}$$

If the type-reduced set has only one point having unity membership and if we wish to reduce computational complexity, we may think that a more straightforward choice for the defuzzified value is the unity membership point in the type-reduced set. Choosing the unity membership point, however, means that we are doing away with all the type-2 analysis and are choosing the output corresponding to only the principal membership function type-1 FLS that is embedded in the type-2 FLS. Since it conveys no information about membership function uncertainties, it does not make sense to use the unity height point as the crisp output; unless, of course, the type-reduced set is convex and symmetric in which case the unity height point is the same as the centroid. In general, for arbitrary membership functions, the type-reduced set is not symmetrical and the centroid location is different from the location of the unity height point.

Since parallel processing is possible for type reduction, the output of each processor can be aggregated using the centroid defuzzifier in (22) to obtain a crisp value; therefore, data storage will not be a problem. Currently, however, most researchers still depend on software for simulations and cannot make use of parallel processing. We can, however, use a recursive method to vastly reduce the memory needed for storing the data needed to compute the defuzzified output. Examining (22), we compute

$$A(i) = A(i-1) + y_i\mu_{\mathrm{Y}}(y_i), \quad A(0) \triangleq 0 \tag{23}$$

$$B(i) = B(i-1) + \mu_{\mathrm{Y}}(y_i), \quad B(0) \triangleq 0 \tag{24}$$

for $i = 1, 2, \cdots, N$. After the Nth iteration, the defuzzified output is $C_Y(\mathbf{x}) = A(N)/B(N)$. By this means, we just need

to store A and B in each iteration. Note that $\mu_Y(y_i)$ in (23) and (24) is the same as b in (13).

VI. Type Reduction for Interval Type-2 FLS

A general type-2 FLS is very complicated because of type reduction. The most general form to compute the type-reduced set is given by

$$\begin{aligned}&\mathrm{Y}(\mathrm{Z}_1,\cdots,\mathrm{Z}_M,\mathrm{W}_1,\cdots,\mathrm{W}_M)\\&\quad=\int_{z_1}\cdots\int_{z_M}\int_{w_1}\cdots\int_{w_M}\mathcal{T}_{l=1}^{M}\mu_{\mathrm{Z}_l}(z_l)\\&\qquad\star\,\mathcal{T}_{l=1}^{M}\mu_{\mathrm{W}_l}(w_l)\Big/\frac{\sum_{l=1}^{M}w_lz_l}{\sum_{l=1}^{M}w_l}\end{aligned}\tag{25}$$

where $\mathcal{T}$ and $\star$ both indicate the t-norm used $\cdots$ product or minimum, $w_l\in\mathrm{W}_l$, and $z_l\in\mathrm{Z}_l$ for $l=1,\cdots,M$.

Things simplify a lot when the secondary MF's are interval sets in which case we call the type-2 FLS an *interval type-2 FLS*. In this section, we focus on type reduction for interval type-2 FLS's. We present a computational procedure that lets us compute the type-reduced set for an interval type-2 FLS without actually having to consider all the combinations of the z_l's and w_l's.

For an interval type-2 FLS, each Z_l and W_l $(l=1,\cdots,M)$ in (25) is an interval type-1 set; then using the fact that $\mu_{\mathrm{Z}_l}(z_l)=\mu_{\mathrm{W}_l}(w_l)=1$, (25) can be rewritten as

$$\begin{aligned}&Y(Z_1,\cdots,Z_M,W_1,\cdots,W_M)\\&\quad=\int_{z_1}\cdots\int_{z_M}\int_{w_1}\cdots\int_{w_M}1\Big/\frac{\sum_{l=1}^{M}w_lz_l}{\sum_{l=1}^{M}w_l}\end{aligned}\tag{26}$$

so, in this case, type reduction means only computing a in (12) since b in (13) is unity.

Since all the memberships in an interval type-1 set are crisp, in the sequel, we represent an interval set just by its domain interval, which can be represented by its left and right end points as $[l,r]$ or by its center and spread as $[c-s,c+s]$, where $c=(l+r)/2$ and $s=(r-l)/2$.

In a general interval type-2 FLS, each Z_l in (26) is an interval type-1 set having center c_l and spread s_l $(s_l\geq 0)$ and each W_l is also an interval type-1 set with center h_l and spread Δ_l, $(\Delta_l\geq 0)$ (we assume that $h_l\geq\Delta_l$ so that $w_l\geq 0$ for $l=1,\cdots,M$). Y is also an interval type-1 set; therefore, we only need to compute its two endpoints y_l and y_r. As shown in [16] and [20], y_l only depends on c_l-s_l and on one of the two endpoints of W_i and y_r only depends on c_l+s_l and on one of the two endpoints of W_i. Next, we state an iterative procedure to compute y_l and y_r for an interval type-2 FLS. The derivation of this procedure is given in [16] and [20].

For convenience, we let

$$S(w_1,\cdots,w_M)\triangleq\frac{\sum_{l=1}^{M}z_lw_l}{\sum_{l=1}^{M}w_l}\tag{27}$$

where $w_l\in[h_l-\Delta_l,\,h_l+\Delta_l]$ and $h_l\geq\Delta_l$ for $l=1,\cdots,M$ and $z_l\in[c_l-s_l,\,c_l+s_l]$. The maximum of S, y_r, is obtained as follows. We set $z_l=c_l+s_l$ $(l=1,\cdots,M)$ and, without loss of generality, assume that the z_l's are arranged in ascending order, i.e., $z_1\leq z_2\leq\cdots\leq z_M$. Then

1) set $w_l=h_l$ for $l=1,\cdots,M$ and compute $S'=S(h_1,\cdots,h_M)$ using (27);
2) find k $(1\leq k\leq M-1)$ such that $z_k\leq S'\leq z_{k+1}$;
3) set $w_l=h_l-\Delta_l$ for $l\leq k$ and $w_l=h_l+\Delta_l$ for $l\geq k+1$ and compute $S''=S(h_1-\Delta_1,\cdots,h_k-\Delta_k,h_{k+1}+\Delta_{k+1},\cdots,h_M+\Delta_M)$ using (27);
4) check if $S''=S'$; if yes, stop. S'' is the maximum value of $S(w_1,\cdots,w_M)$; if no, go to step 5;
5) set S' equal to S''; go to step 2.

It can easily be shown that *this iterative procedure converges in at most M iterations* [16], [20], where one iteration consists of one pass through steps 2) to 5).

The minimum of $S(w_1,\cdots,w_M)$ can be obtained by using a procedure similar to the one just described. Only two changes need to be made: 1) we must set $z_l=c_l-s_l$ for $l=1,\cdots,M$ and 2) in Step 3) we must set $w_l=h_l+\Delta_l$ for $l\leq k$ and $w_l=h_l-\Delta_l$ for $l\geq k+1$ to compute the weighted average $S''=S(h_1+\Delta_1,\cdots,h_k+\Delta_k,h_{k+1}-\Delta_{k+1},\cdots,h_M-\Delta_M)$.

This computational procedure can be used to compute the type-reduced set for all of the type reducers, with a vast reduction in computational complexity. y_l and y_r can be computed in parallel and regardless of the kind of type reduction used, only two processors are needed. This is quite different from the general results in Table I.

Next, we explain exactly how to apply this four-step procedure to centroid, height, and center-of-sets type reduction.

1) *Centroid type reduction*: For an interval type-2 FLS, the degree of firing corresponding to every rule as well as the output set for every rule are interval type-1 sets, so that (11) reduces to

$$\mathrm{Y}_c(\mathbf{x})=\int_{\theta_1}\cdots\int_{\theta_N}1\Big/\frac{\sum_{i=1}^{N}y_i\theta_i}{\sum_{i=1}^{N}\theta_i}\tag{28}$$

where all the symbols have the same meaning as in (11). Let $[L_i,R_i]$ be the domain of $\mu_{\tilde{\mathrm{B}}}(y_i)$. To use the computational procedure described in this section, note that the sum in (27) now goes from 1 to N instead of from 1 to M; y_i plays the role of c_l; $s_l=0$ for all l since y_i is crisp; $(L_i+R_i)/2=h_l$; and $(R_i-L_i)/2=\Delta_l$.

2) *Height type reduction*: For an interval type-2 FLS, (17) reduces to

$$\mathrm{Y}_h(\mathbf{x}) = \int_{\theta_1}\int_{\theta_2}\cdots\int_{\theta_M} 1 \Big/ \frac{\sum_{l=1}^{M} \overline{y}^l \theta_l}{\sum_{l=1}^{M} \theta_l} \qquad (29)$$

where all the symbols have the same meaning as in (17). Here, $\overline{y}^l$ plays the role of c_l; $s_l = 0$ for all l since $\overline{y}^l$ is crisp; and the output set membership of $\overline{y}^l$, $\mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$ plays the role of W^l. If the domain of each $\mu_{\tilde{\mathrm{B}}^l}(\overline{y}^l)$ is represented as $[L_l, R_l]$, then $h_l = (L_l + R_l)/2$ and $\Delta_l = (R_l - L_l)/2$.

3) *Center-of-sets type reduction*: For an interval type-2 FLS, (19) reduces to

$$\mathrm{Y}_{cos}(\mathbf{x}) = \int_{d_1}\cdots\int_{d_M}\int_{e_1}\cdots\int_{e_M} 1 \Big/ \frac{\sum_{l=1}^{M} d_l e_l}{\sum_{l=1}^{M} e_l} \qquad (30)$$

where all the symbols have the same meaning as in (19). In this case, our computational procedure needs to be applied in two stages. In the first stage, we compute the centroid C_l of each interval type-2 consequent set. The centroid of an interval type-2 set $\tilde{\mathrm{A}}$ is given as [see (7)]

$$C_{\tilde{\mathrm{A}}} = \int_{\theta_1}\cdots\int_{\theta_N} 1 \Big/ \frac{\sum_{i=1}^{N} x_i \theta_i}{\sum_{i=1}^{N} \theta_i} \qquad (31)$$

where each θ_i belongs to some interval in $[0,1]$. To use the computational procedure described above to compute $C_{\tilde{\mathrm{A}}}$, observe that x_i and θ_i in (31) play the role of z_l and w_l in (27), respectively. In this case, each s_l is zero because each x_i is crisp.

In the second stage, we compute the type-reduced set using (30). When computing the type-reduced set, C_l plays the role of Z_l in (26). If the domain of C_l is the interval $[L_l^c, R_l^c]$, then $c_l = (L_l^c + R_l^c)/2$ and $s_l = (R_l^c - L_l^c)/2$. The degree of firing E_l plays the role of W_l. If the domain of E_l is the interval $[L_l, R_l]$, then $h_l = (L_l + R_l)/2$ and $\Delta_l = (R_l - L_l)/2$.

In each of the above cases, a crisp output for the type-2 FLS can be found by computing the centroid of the type-reduced set. Since the type-reduced set is an interval, the centroid is the midpoint of its domain.

Example 6.1: In this example, we illustrate the use of the just-described type-reduction methods for an interval type-2 FLS. We consider a single-input single-output type-2 FLS using product t-norm and product inference, which has rules of the form: R^l: IF x is $\tilde{\mathrm{F}}^l$, THEN y is $\tilde{\mathrm{G}}^l$, where $x, y \in [0, 10]$.

Fig. 4(a) and (b) depicts the antecedent and consequent sets. The footprints of uncertainty are uniformly shaded, because the secondary MF's are interval sets. Each of these sets can be described by two Gaussians, which have the same mean and standard deviation. The two Gaussians are scaled to different heights. The maximum height reached by the taller Gaussian is unity, whereas that reached by the shorter Gaussian is s. If the mean and standard deviation of the Gaussians is m and σ, respectively, the membership grade of a domain point x' is an interval $[s\exp\{-0.5(\frac{x'-m}{\sigma})^2\}, \exp\{-0.5(\frac{x'-m}{\sigma})^2\}]$. The m values for each of the antecedent sets $\tilde{\mathrm{F}}^1$, $\tilde{\mathrm{F}}^2$, and $\tilde{\mathrm{F}}^3$ are 2, 5, and 8, respectively; the σ values are 1, 1, and 1, respectively; and the s values are 0.8, 0.6, and 0.9, respectively. For the three consequent sets, $\tilde{\mathrm{G}}^1$, $\tilde{\mathrm{G}}^2$, and $\tilde{\mathrm{G}}^3$, the m values are 6, 2, and 9, respectively; the σ values are 1, 1.2, and 1, respectively; and, the s values are 0.75, 0.75, and 0.8, repectively.

The applied input is $x = 4$ [shown in Fig. 4(a)]. It has nonzero memberships in two antecedents $\tilde{\mathrm{F}}_1$ and $\tilde{\mathrm{F}}_2$ so that two rules are fired.

The type-reduction results for this example are collected in Table II, where each interval type-reduced set is represented using its center and spread and the center is the defuzzified output of for each type-reduction method. For comparison, in Table II, we also give the defuzzified output of a type-1 FLS where $s = 1$ for all the antecedent and consequent MF's. The difference between the output of the type-1 FLS and the defuzzified output of the comparable type-2 FLS reflects the flow of antecedent and consequent uncertainties through the type-2 FLS.

Just as different defuzzification methods provide different results, different type-reduction methods also provide different results. Which type-reduction method to choose is an open issue, as is which defuzzification method to choose. □

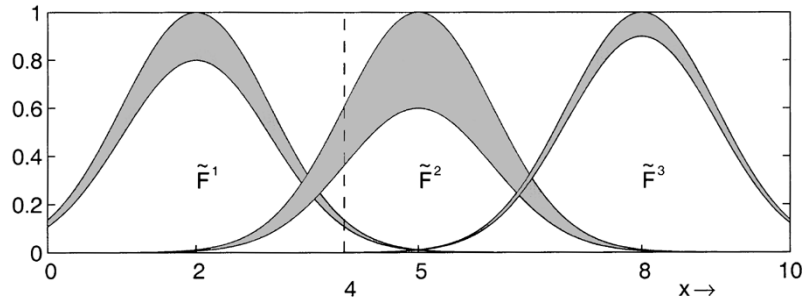
TABLE II
RESULTS FOR THE EXAMPLE 6.1

Type-reduced set	Center	Spread	Type-1 Output
Centroid	2.8179	0.4918	2.7079
Height	2.9026	0.4280	2.8341
Center-of-sets	2.9462	0.3563	2.8341

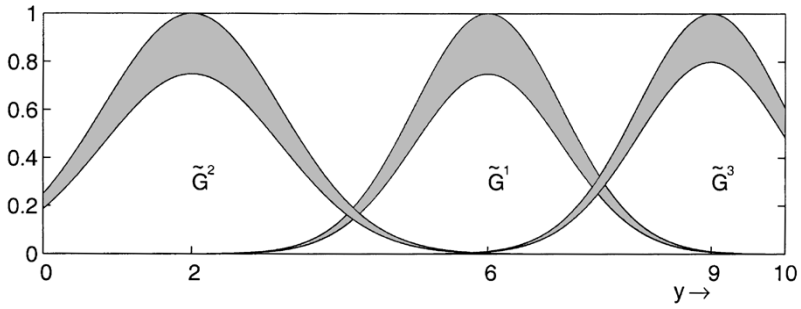
VII. APPLICATION TO TIME-VARYING CHANNEL EQUALIZATION

Wang and Mendel [40] applied a type-1 FLS to time-invariant nonlinear channel equalization and demonstrated that the bit error rates (BER) of the fuzzy equalizers are close to that of the optimal equalizer [40]. Sarwal and Srinath [35] observed that a transversal filter requires a much larger training set to achieve the same error rate as compared to a fuzzy logic equalizer. Lee [22] proposed a complex fuzzy adaptive filter for QAM constellation channel equalization. Patra and Mulgrew used a fuzzy adaptive filter to implement a Bayesian equalizer [33] and also used it to eliminate cochannel interference [34].

All the above fuzzy approaches in the area of adaptive equalization are focused on time-invariant channels. In today's communication world (such as mobile communications) the channels are time-varying and nonlinear. Observing this, we apply our type-2 FLS to this challenging area and compare it



(a)



(b)

Fig. 4. (a) Antecedent sets (the vertical axis shows the primary memberships of x in the antecedent sets) and (b) consequent sets (the vertical axis shows the primary memberships of y in the consequent sets) for the interval type-2 FLS in Example 6.1. The applied input $(x = 4)$ is shown in (a).

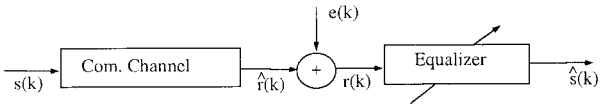


Fig. 5. Block diagram of a base-band communication system subject to ISI and AGN.

with the methods based on a type-1 FLS [33] and a nearest neighbor classifier (NNC) [36]. The NNC approach [36] has been successfully used in a global system for mobile communications (GSM) receiver and no complicated optimization method [such as recursive least squares (RLS) or least means squares (LMS)] has been used in it. We also design our type-2 FLS based on a clustering method, and do not use the RLS or LMS methods.

A block diagram of a base-band communication system subject to intersymbol interference (ISI) and additive Gaussian noise (AGN) is shown in Fig. 5, where $s(k)$ is the symbol to be transmitted, $\hat{r}(k)$ is the noise-free signal, $e(k)$ is the noise, the channel order is n $(n+1$ taps), and time-varying tap coefficients are $a_i(k)$ $(i = 0, 1, \cdots, n)$; so $r(k)$ can be represented as (we assume the channel delay $d = 0$)

$$r(k) = \hat{r}(k) + e(k) = \sum_{i=0}^{n} a_i(k) s(k-i) + e(k) \tag{32}$$

where k denotes time index. Here we assume that $s(k)$ is binary, either $+1$ or -1 with equal probability. Our channel equalization goal is to recover the input sequence $s(k)$ based on a sequence of $r(k)$ values without knowing (estimating) the channel coefficients—the $a_i(k)$.

A. Why a Type-2 FLS Is Needed for Time-Varying Channel Equalization

In a FLS-based equalizer (such as in [40]), the antecedents are $r(k)$, $r(k-1)$, $\cdots$, $r(k-p+1)$, where p is the equalizer order (number of taps in the equalizer). We let

$$\mathbf{r}(k) = [r(k), r(k-1), \cdots, r(k-p+1)]^T \tag{33}$$

and

$$\hat{\mathbf{r}}(k) = [\hat{r}(k), \cdots, \hat{r}(k-p+1)]^T \tag{34}$$

where $\hat{\mathbf{r}}(k)$ is called the *channel state* [2] and there are $n_s = 2^{n+p}$ channel states [2].

A normalized Bayesian equalizer $f(\mathbf{r}(k))$ has been developed in [33] as

$$f(\mathbf{r}(k)) = \frac{\sum_{i=1}^{n_s} \overline{y}^i \prod_{l=0}^{p-1} \exp\left[-\frac{1}{2}\left(\frac{r(k-l)-\hat{r}_i(k-l)}{\sigma_e}\right)^2\right]}{\sum_{i=1}^{n_s} \prod_{l=0}^{p-1} \exp\left[-\frac{1}{2}\left(\frac{r(k-l)-\hat{r}_i(k-l)}{\sigma_e}\right)^2\right]} \tag{35}$$

where σ_e is the standard deviation (std) of the additive noise, and $\hat{r}_{il}$ is the lth $(l = 1, \cdots, p)$ element of the ith $(i = 1, \cdots, 2^{n+p})$ channel state. The decision output of this equalizer is based on the sign of $f(\mathbf{r}(k))$. A type-1 FLS with Gaussian antecedent MF's, product inference, and height defuzzifier [as in (14)] can be used to implement the normalized Bayesian equalizer in (35) perfectly [33] because it has the same structure as (14).

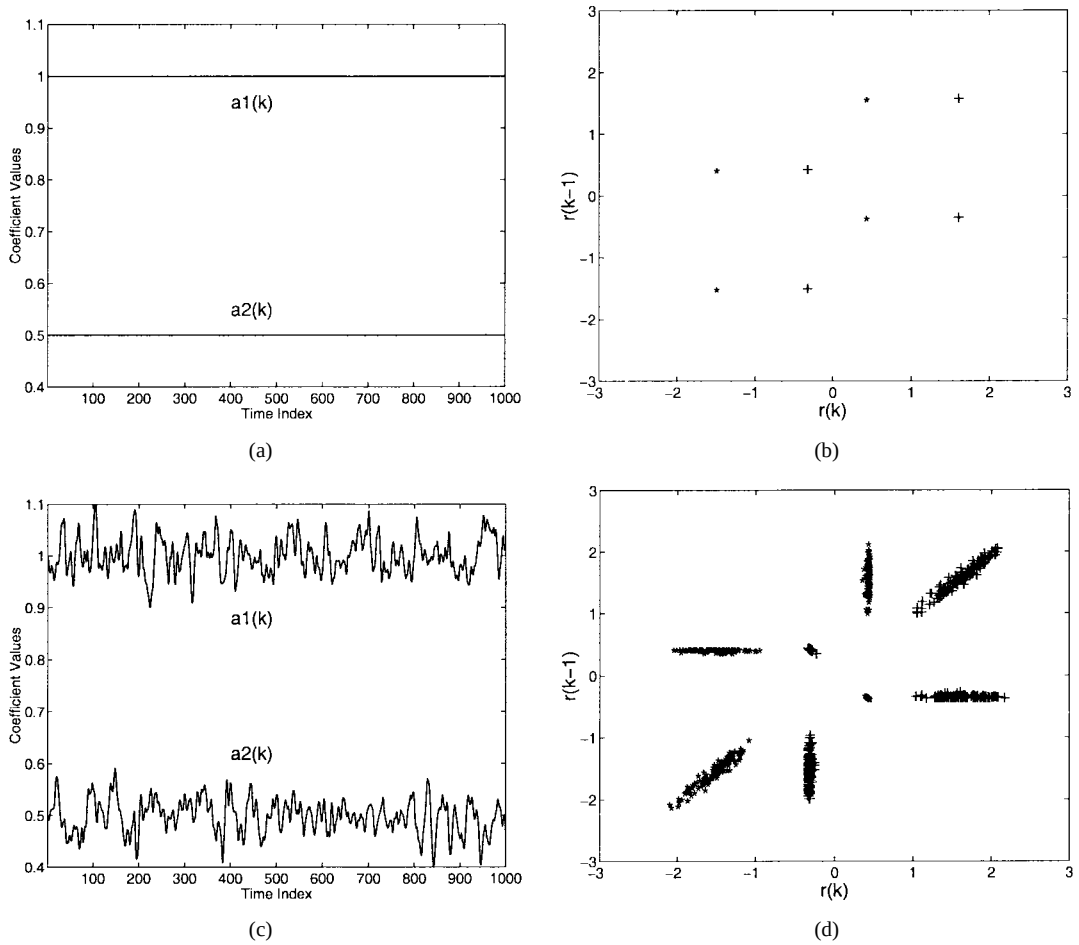


Fig. 6. For the channel in (36): (a) time-invariant channel $a_1 = 1$ and $a_2 = 0.5$. (b) Channel states (noise free) of time invariant channel, $*$ denotes the category $\hat{\mathbf{r}}(k)$ is $+1$, and $+$ denotes the category $\hat{\mathbf{r}}(k)$ is -1. (c) Example of time-varying channel with $\beta = 0.1$. (d) Channel states (noise free) of the time-varying channel in (c).

TABLE III
NOMINAL CHANNEL STATES FOR CHANNEL MODEL (36) WITH BINARY SYMBOLS. THE CHANNEL DELAY $d = 0$ AND $p = 2$. THE CHANNEL STATE CATEGORY IS DETERMINED BY $s(k)$

$s(k)$	$s(k-1)$	$s(k-2)$	$\hat{r}(k)$	$\hat{r}(k-1)$
1	1	1	$a_1 + a_2 - 0.9(a_1 + a_2)^3$	$a_1 + a_2 - 0.9(a_1 + a_2)^3$
1	1	-1	$a_1 + a_2 - 0.9(a_1 + a_2)^3$	$a_1 - a_2 - 0.9(a_1 - a_2)^3$
1	-1	1	$a_1 - a_2 - 0.9(a_1 - a_2)^3$	$-a_1 + a_2 - 0.9(-a_1 + a_2)^3$
1	-1	-1	$a_1 - a_2 - 0.9(a_1 - a_2)^3$	$-a_1 - a_2 - 0.9(-a_1 - a_2)^3$
-1	1	1	$-a_1 + a_2 - 0.9(-a_1 + a_2)^3$	$a_1 + a_2 - 0.9(a_1 + a_2)^3$
-1	1	-1	$-a_1 + a_2 - 0.9(-a_1 + a_2)^3$	$a_1 - a_2 - 0.9(a_1 - a_2)^3$
-1	-1	1	$-a_1 - a_2 - 0.9(-a_1 - a_2)^3$	$-a_1 + a_2 - 0.9(-a_1 + a_2)^3$
-1	-1	-1	$-a_1 - a_2 - 0.9(-a_1 - a_2)^3$	$-a_1 - a_2 - 0.9(-a_1 - a_2)^3$

In this paper, we chose the following channel model that was used in [40]:

$$r(k) = a_1 s(k) + a_2 s(k-1) - 0.9[a_1 s(k) + a_2 s(k-1)]^3 + e(k) \qquad (36)$$

where nominal values for the channel's coefficients are $a_1 = 1$ and $a_2 = 0.5$, shown in Fig. 6(a). In this paper, we also assume that we know that the channel has two taps, i.e., $n = 1$. We also chose the number of equalizer taps to be equal to $n + 1$, i.e., $p = 2$. The nominal channel states are given in Table III and are plotted in Fig. 6(b). Observe that the nominal channel states are eight single points.

We focus on the case when the channel is time-varying, i.e., when a_1 and a_2 in (36) are time-varying coefficients. The time variations of the two coefficients were simulated, as in [4], by using a second-order Markov model in which two independent white Gaussian noise sources drive a second-order Butterworth low-pass filter (LPF). We used the function *butter* provided by the Matlab Signal Processing Toolbox to generate a second-order low-pass digital Butterworth filter with cutoff frequency 0.1; then the function *filter* was used to generate a colored Gaussian sequence, which we then used as the time-varying channel coefficients. Note that we centered $a_1(k)$ about 1 and $a_2(k)$ about 0.5. The input to the Butterworth filter is a white

Gaussian sequence with standard deviation (std) β. The source code for the time-varying coefficients is

```
[B,A]=butter(2,0.1)
% B (numerator) and A (denominator) of LPF
a1=1+filter(B,A,beta*randn(1,1000))
a2=0.5+filter(B,A,beta*randn(1,1000)).
```

Realizations of the time-varying coefficients and channel states are plotted in Fig. 6(c) and (d), respectively, for $\beta = 0.1$. Observe that the channel states are now eight clusters instead of eight single points. These clusters illustrate that $\hat{\mathbf{r}}_i$ is uncertain for all $i = 1, \cdots, 8$. Each cluster has category $+1$ or -1, as determined by $s(k)$ (see Table III); this establishes the value of $\overline{y}^i$ as 1 or -1 in (35), as determined by the Bayesian decision boundary [2], [33].

B. Designing the Type-2 FLS Equalizer

In our type-2 FLS design, there are eight rules (each rule describes one channel state corrupted by additive noise) and the lth rule R^l is expressed as

$$R^l: \quad \text{IF } r(k) \text{ is } \tilde{\mathrm{F}}_1^l \text{ and } r(k-1) \text{ is } \tilde{\mathrm{F}}_2^l \text{ THEN } \hat{s}(k) \text{ is } \mathrm{G}^l$$

where $\tilde{\mathrm{F}}_1^l$ and $\tilde{\mathrm{F}}_2^l$ are type-2 Gaussian MF's with uncertain means and G^l is a type-1 Gaussian MF with mean $+1$ or -1 as determined by the channel-state category. For rule l, the range of the mean of antecedent $\tilde{\mathrm{F}}_1^l$ ($\tilde{\mathrm{F}}_2^l$) corresponds to the horizontal (vertical) projection of the lth cluster in Fig. 6(d). We used height type reduction (which, in this case, is the same as center-of-sets type reduction because the rule consequent is type-1) as described in Section IV-B, so (29) defines the structure of our type-2 FLS equalizer where $\overline{y}^l$ equals 1 or -1 (center of G^l), $\theta_l \in D^l = \sqcap_{k=1}^2 \mu_{\tilde{\mathrm{F}}_k^l}(x_k)$

$$\mu_{\tilde{\mathrm{F}}_k^l}(x_k) = \exp\left[-\frac{1}{2}\left(\frac{x_k - m_k^l}{\sigma_e}\right)^2\right] \qquad m_k^l \in [m_{k1}^l, m_{k2}^l] \tag{37}$$

(see Fig. 7) and $k = 1, 2$. In order to specify these MF's, we need to specify their parameters, namely, $[m_{k1}^l, m_{k2}^l]$, and σ_e. Below, we let $\mathbf{m_1}^l = [m_{11}^l, m_{21}^l]^T$ and $\mathbf{m_2}^l = [m_{12}^l, m_{22}^l]^T$.

We used a clustering approach to estimate $\mathbf{m_1^l}$ and $\mathbf{m_2^l}$, because it is computationally simple. Here we briefly summarize this approach. Suppose the number of training prototypes $(\mathbf{s}(k), \mathbf{r}(k))$ is N where from Table III, we see that $\mathbf{s}(k) = [s(k), s(k-1), s(k-2)]^T$ determines to which cluster $\mathbf{r}(k) = [r(k), r(k-1)]^T$ belongs. So $\mathbf{r}(1), \mathbf{r}(2), \cdots, \mathbf{r}(N)$ are classified into 2^{p+n} clusters, where in this example, $2^{p+n} = 2^{2+1} = 8$. Suppose N_l training prototypes belong to the lth cluster, $l = 1, \cdots, 8$. The mean and std of these $\mathbf{r}(k)$ are $\mathbf{m}_\mathbf{r}^l$ and $\sigma_\mathbf{r}^l$, respectively. We let

$$\mathbf{m_1}^l \triangleq \mathbf{m}_\mathbf{r}^l - \sigma_\mathbf{r}^l \tag{38}$$

$$\mathbf{m_2}^l \triangleq \mathbf{m}_\mathbf{r}^l + \sigma_\mathbf{r}^l. \tag{39}$$

Consequently, $[m_{11}^l, m_{12}^l]$ is the range of the mean of the type-2 antecedent Gaussian MF $\mu_{\tilde{\mathrm{F}}_1^l}$; similarly, $[m_{21}^l, m_{22}^l]$ is the range of the mean of $\mu_{\tilde{\mathrm{F}}_2^l}$. For the type-1 FLS design, we used $\mathbf{m}_\mathbf{r}^l$ (i.e., $[m_{r1}^l, m_{r2}^l]^T$) as the centers of type-1 Gaussian antecedent MF's.

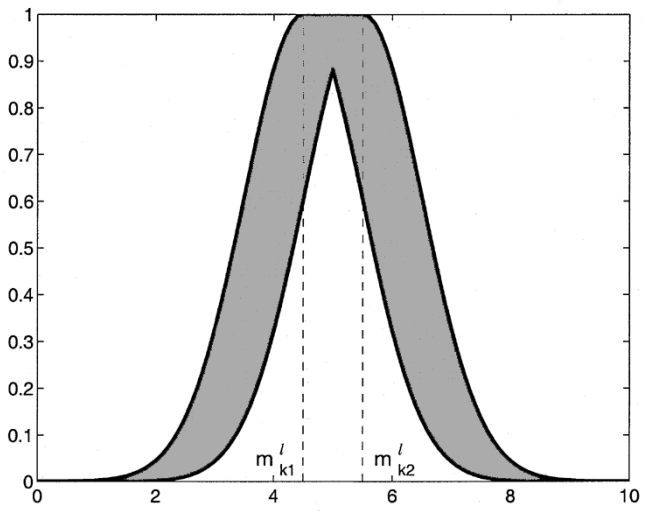


Fig. 7. The type-2 MF's for time-varying channel equalization. The shaded regions are the footprints of uncertainty for interval secondaries. For illustrative purpose, we show the centers of Gaussian MF's varying from 4.5 to 5.5.

To complete the specification of the MF's in (37), we also need to estimate the std of the noise, σ_e. In [3], it is shown that equalizer performance is not very sensitive to the value σ_e. In our simulations, we assumed that the value of σ_e is known exactly. How to handle the situation when this is not true will be reported on in a future publication.

Because we used an interval type-2 FLS, computation was greatly simplified, i.e., we used the type reduction computation procedure described in Section VI.

C. Simulations

We compared our type-2 FLS with the type-1 FLS in [33] and the K-nearest neighbor (NN) classifier in [36]. If the number of training prototypes is N, then $K = \sqrt{N}$ is the optimal choice of K [9]. The parameters of the type-1 and type-2 FLS's were set using the training sequence and we then determined the bit error rate (BER) of the three equalizers using the testing sequence. We performed our simulations for two cases, a small number of training prototypes and a large number of training prototypes.

1) Small Number of Training Prototypes: In this simulation, the training sequence we used is of length 81 and the testing sequence is of length 919. In our experiment, we fixed the $SNR = 20$ and ran simulations for five different β ranging from $\beta = 0.04$ to $\beta = 0.20$ ($0.04 : 0.04 : 0.20$). We ran 100 Monte Carlo (MC) simulations for each β value. In each realization, the channel coefficients and additive noise were uncertain. In Fig. 8, we plot the average BER for the 100 MC realizations.

In our second experiment, we fixed $\beta = 0.1$ and ran simulations for seven different signal-to-noise ratios (SNR's) ranging from SNR $= 15$ dB to SNR $= 30$ dB ($15 : 2.5 : 30$). We ran 100 MC simulations for each SNR value. In Fig. 9, we plot the average BER for 100 MC realizations.

2) Large Number of Training Prototypes: In this simulation, the training sequence we used is of length 169 and the testing sequence is of length 831. In our first experiment,

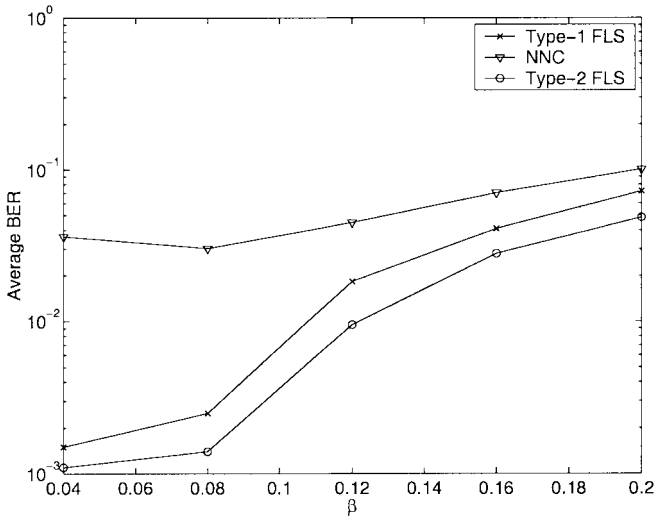


Fig. 8. The average BER of type-1 FLS, nearest neighbor classifier (NNC), and type-2 FLS versus β when $SNR = 20$ dB and the number of training prototypes is 81. Each point corresponds to an average over 100 realizations.

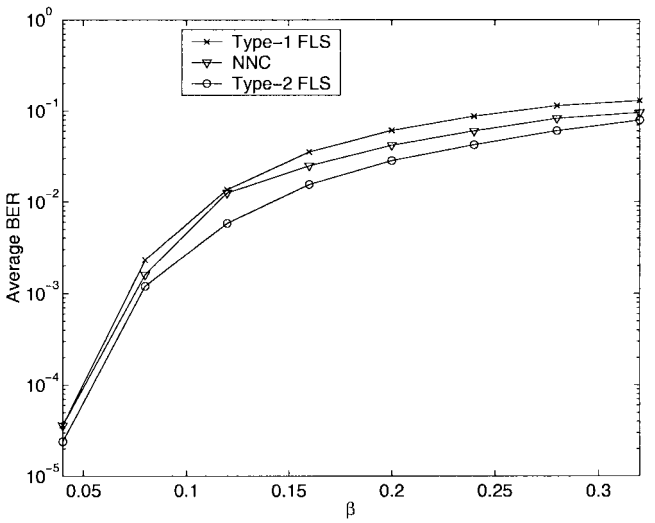


Fig. 10. The average BER of type-1 FLS, nearest neighbor classifier (NNC), and type-2 FLS versus β when SNR $= 20$ dB and the number of training prototypes is 169. Each point corresponds to an average over 100 realizations.

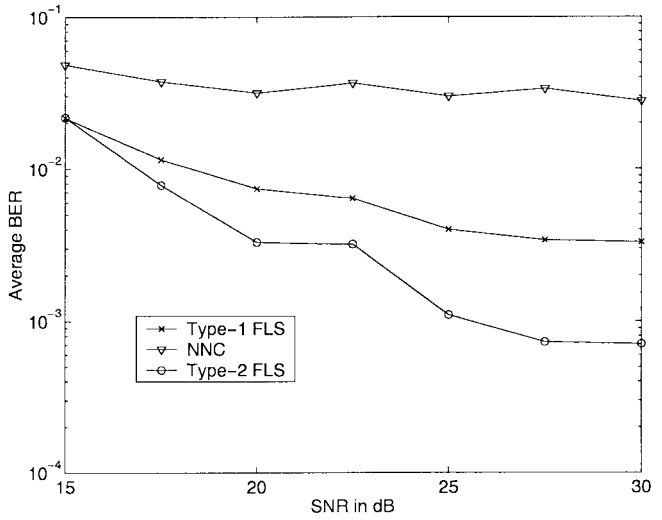


Fig. 9. The average BER of type-1 FLS, nearest neighbor classifier (NNC), and type-2 FLS versus SNR when $\beta = 0.1$ and the number of training prototypes is 81. Each point corresponds to an average over 100 realizations.

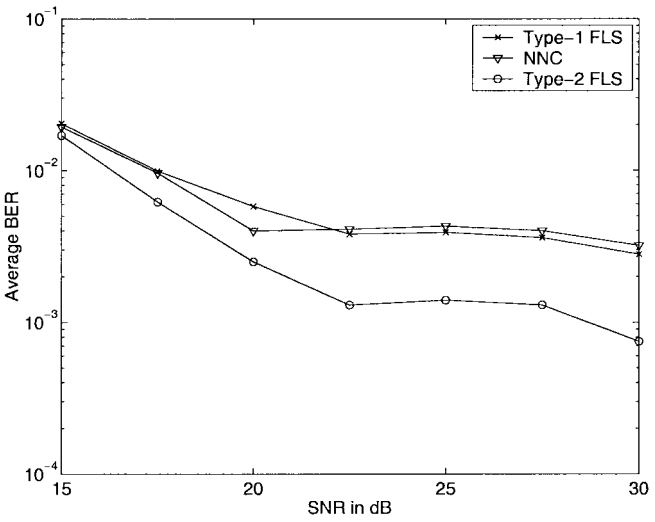


Fig. 11. The average BER of type-1 FLS, nearest neighbor classifier (NNC), and type-2 FLS versus SNR when $\beta = 0.1$ and the number of training prototypes is 169. Each point corresponds to an average over 100 realizations.

we again fixed the SNR $= 20$ and ran simulations for eight different β ranging from $\beta = 0.04$ to $\beta = 0.32$ ($0.04 : 0.04 : 0.32$). We ran 100 Monte Carlo (MC) simulations for each β value. In Fig. 10, we plot the average bit error rate (BER) for the 100 MC realizations. In our second experiment, we again fixed $\beta = 0.1$ and ran the simulations for seven different SNR's ranging from SNR $= 15$ dB to SNR $= 30$ dB ($15 : 2.5 : 30$). We ran 100 MC simulations for each SNR value. In Fig. 11, we plot the average BER for 100 MC realizations.

3) Remarks:

1) From Figs. 8–11, we see that our type-2 FLS performs better than the NNC and type-1 FLS in all experiments.
2) The NNC does not obtain good performance when the number of training prototypes is small, but it can perform better than a type-1 FLS when the number of training prototypes is large.
3) A type-1 FLS is able to handle the additive noise (Figs. 8 and 10), but it does not handle channel coefficient uncertainties as well. In Figs. 9 and 11, we see that for higher SNR's, (which means the channel uncertainties are due primarily to uncertain channel coefficients), the performance of the type-2 FLS is much better than that of the type-1 FLS.

VIII. Conclusions and Future Work

In this paper, we have developed a type-2 FLS as a natural extension of a type-1 FLS and have focused on the operations of inference, type reduction and defuzzification for a type-2 FLS. Type reduction is a new operation for a FLS; it reduces to defuzzification when all uncertainties disappear; and it is computationally intensive, except for interval type-2 FLS's. In that important case, we have provided a simple five-step iterative procedure for computing the type-reduced set, and

have shown that the complexity of an interval type-2 FLS has been tremendously reduced.

We have applied type-2 FLS's to time-varying channel equalization, for which channel uncertainty cannot be captured with a type-1 FLS, whereas it can be captured by a type-2 FLS. Simulation results show that our type-2 FLS outperforms a type-1 FLS and a NNC. This application suggests that type-2 FLS's are very promising for engineering applications where uncertainties are present.

The interval type-2 FLS we discussed in this paper uses a singleton fuzzifier. Mouzouris and Mendel [31] presented the theory of a type-1 FLS with nonsingleton fuzzification and applied it to Mackey–Glass chaotic time series forecasting when the time-series is corrupted by measurement noise. In actual time series such as the price curve for the United States dollar versus the German mark, market volatility can change noticeably over the course of time, so that the variance of the noise component, which is related to volatility, need not be constant [25]. In this case, we cannot fuzzify the crisp input as a type-1 fuzzy set, because type-1 MF's cannot fully represent the uncertainty associated with this linguistic knowledge. We believe that in this important case, *the input should be fuzzified into a type-2 fuzzy set*, i.e., a nonsingleton type-2 FLS should be used. The theory and design for an interval type-2 nonsingleton FLS is reported on in [24].

The two most popular fuzzy logic systems (models) used by engineers today are the Mamdani and TSK systems. The type-2 FLS's we presented in this paper are Mamdani systems. We have developed a type-2 TSK FLS [23], and are exploring the design and applications of interval type-2 TSK FLS's.

Finally, MATLAB files for performing type-2 computations discussed in this paper are available at URL: http://sipi.usc.edu/~mendel/software.

Appendix A
Centroid Computation Using Product t-norm

When a product t-norm is used to calculate the centroid of a type-2 set that has a continuous domain and whose secondary memberships are not all unity, we obtain an unexpected result. Here we discuss this problem and suggest a remedy.

We concentrate on type-2 sets that have a continuous domain and whose secondary membership functions are such that for any domain point, only one primary membership has a secondary membership equal to one, e.g., Gaussian or triangular type-2 sets. Let $\tilde{\mathrm{A}}$ be such a set. In the discussion associated with (7), we assumed that the domain of $\tilde{\mathrm{A}}$ is discretized into N points. The true centroid of $\tilde{\mathrm{A}}$ (assuming $\tilde{\mathrm{A}}$ has a continuous domain) is the limit of $C_{\tilde{\mathrm{A}}}$ in (7) as $N \to \infty$. When we use the product t-norm $\lim_{N\to\infty}\ \mathcal{T}_{i=1}^{N}\mu_{\mathrm{D}_i}(\theta_i) = \lim_{N\to\infty}\ \Pi_{i=1}^{N}\ \mu_{\mathrm{D}_i}(\theta_i)$.

Let B be an embedded type-1 set in $\tilde{\mathrm{A}}$. The centroid of B is computed as

$$C_{\mathrm{B}} = \frac{\sum_{i=1}^{N} x_i \mu_{\mathrm{B}}(x_i)}{\sum_{i=1}^{N} \mu_{\mathrm{B}}(x_i)} \tag{40}$$

and the membership of C_{B} in $C_{\tilde{\mathrm{A}}}$ [denoted as $\mu_C(C_{\mathrm{B}})$] is

$$\mu_C(C_{\mathrm{B}}) = \prod_{i=1}^{N} \mu_{\mathrm{D}_i}(\theta_i) \tag{41}$$

where $\{\theta_1, \cdots, \theta_N\}$ are the primary memberships that make up the type-1 set B. Also, let A denote the principal membership function of $\tilde{\mathrm{A}}$. Obviously, $\mu_C(C_{\mathrm{A}}) = 1$. Observe the following.

1) $\lim_{N\to\infty}\ \mu_C(C_{\mathrm{B}})$ is nonzero only if B differs from A in *at most a finite number of points*. For all other embedded sets B, the product of an infinite number of quantities less than one will cause $\mu_C(C_{\mathrm{B}})$ to go to zero as $N \to \infty$.
2) For any embedded set B whose membership function differs from that of A in only a finite number of points [i.e., when $\mu_{\mathrm{B}}(x) \neq \mu_{\mathrm{A}}(x)$ for only a finite number of points x] $C_{\mathrm{B}} = C_{\mathrm{A}}$. This can be explained as follows. The (true) centroid of B is the limit of (40) as $N \to \infty$, i.e., $C_{\mathrm{B}} = \int_x x\mu_{\mathrm{B}}(x)\,dx / \int_x \mu_{\mathrm{B}}(x)\,dx$, where $x \in \mathrm{B}$. Since A and B share the same domain (both are embedded sets in $\tilde{\mathrm{A}}$), $x \in \mathrm{A} \Leftrightarrow x \in \mathrm{B}$; and since $\mu_{\mathrm{A}}(x)$ and $\mu_{\mathrm{B}}(x)$ differ only in a finite number of points $\int_x x\mu_{\mathrm{B}}(x)\,dx = \int_x x\mu_{\mathrm{A}}(x)\,dx$ and $\int_x \mu_{\mathrm{B}}(x)\,dx = \int_x \mu_{\mathrm{A}}(x)\,dx$; therefore, $C_{\mathrm{B}} = C_{\mathrm{A}}$.

From these two observations, we can see that the only point having nonzero membership in $C_{\tilde{\mathrm{A}}}$ is equal to C_{A}; and its membership grade is equal to the supremum of the membership grades of all the embedded type-1 sets which have the same centroid, which is equal to one [since $\mu_C(C_{\mathrm{A}}) = 1$]. In other words, $C_{\tilde{\mathrm{A}}} = 1/C_{\mathrm{A}} = C_{\mathrm{A}}$, i.e., the centroid of $\tilde{\mathrm{A}}$, will be equal to a crisp number $\cdots$ the centroid of its principal membership function!

This problem occurs because under the product t-norm, $\lim_{N\to\infty}\ \mathcal{T}_{i=1}^{N}\mu_{\mathrm{D}_i}(\theta_i) = \lim_{N\to\infty}\ \Pi_{i=1}^{N}\ \mu_{\mathrm{D}_i}(\theta_i) = 0$, unless only a finite number of $\mu_{\mathrm{D}_i}(\theta_i)$'s are less than one. The minimum t-norm does not cause such a problem. To avoid this problem, *we will always use the minimum t-norm to calculate the centroid of a type-2 set having a continuous domain.*

Finally, note that while performing algebraic operations on interval sets, the choice of t-norm does not matter, because all the memberships are equal to one; therefore, the problem described above does not appear in the case of interval type-2 sets [16], [19].

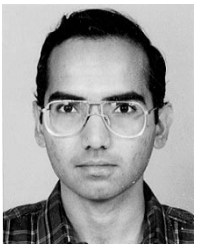
Acknowledgment

The authors would like to thank Dr. J. Keller for suggesting that our original more complicated notations for type-2 fuzzy sets in [14]–[16] could be simplified.

References

[1] J. L. Chaneau, M. Gunaratne, and A. G. Altschaeffl, "An application of type-2 sets to decision making in engineering," *Analysis of Fuzzy Information—Vol. II: Artificial Intelligence and Decision Systems*, J. C. Bezdek, Ed. Boca Raton, FL: CRC, 1987.
[2] S. Chen, B. Mulgrew, and S. McLaughlin, "A clustering technique for digital communications channel equalization using radial basis function network," *IEEE Trans. Neural Networks*, vol. 4, pp. 570–579, July 1993.

[3] S. Chen, S. McLaughlin, B. Mulgrew, and P. M. Grant, "Adaptive Bayesian decision feedback equalizer for dispersive mobile radio channels," *IEEE Trans. Communicat.*, vol. 43, pp. 1937–1956, May 1995.
[4] C. F. N. Cowan, and S. Semnani, "Time-variant equalization using a novel nonlinear adaptive structure," *Int. J. Adaptive Contr. Signal Processing*, vol. 12, no. 2, pp. 195–206, 1998.
[5] D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control*, 2nd ed. New York: Springer-Verlag, 1996.
[6] D. Dubois and H. Prade, "Operations on fuzzy numbers," *Int. J. Syst. Sci.*, vol. 9, pp. 613–626, 1978.
[7] ——, "Operations in a fuzzy-valued logic," *Informat. Contr.*, vol. 43, pp. 224–240, 1979.
[8] ——, *Fuzzy Sets and Systems: Theory and Applications*. New York: Academic, 1980.
[9] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.
[10] S. Ghosh, Q. Razouqi, H. J. Schumacher, and A. Celmins, "A survey of recent advances in fuzzy logic in telecommunications networks and new challenges," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 443–447, Aug. 1998.
[11] E. Hisdal, "The IF THEN ELSE statement and interval-valued fuzzy sets of higher type," *Int. J. Man-Machine Studies*, vol. 15, pp. 385–455, 1981.
[12] R. I. John, "Type 2 fuzzy sets: An appraisal of theory and applications," *Int. J. Uncertainty, Fuzziness, Knowledge-Based Syst.*, vol. 6, no. 6, pp. 563–576, Dec. 1998.
[13] R. I. John, P. R. Innocent, and M. R. Barnes, "Type 2 fuzzy sets and neuro-fuzzy clustering of radiographic tibia images," *IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, May 1998, pp. 1373–1376.
[14] N. N. Karnik and J. M. Mendel, "Introduction to Type-2 Fuzzy Logic Systems," presented at *IEEE FUZZ Conf.*, Anchorage, AK, May 1998.
[15] ——, "Type-2 Fuzzy Logic Systems: Type-Reduction," presented at *IEEE Syst., Man, Cybern. Conf.*, San Diego, CA, Oct. 1998.
[16] ——, "An introduction to type-2 fuzzy logic systems," Univ. Southern California, Rep., Oct. 1998, http://sipi.usc.edu/~mendel/report.
[17] ——, "Applications of type-2 fuzzy logic systems: Handling the uncertainty associated with surveys," presented at *FUZZ-IEEE Conf.*, Seoul, Korea, Aug. 1999.
[18] ——, "Applications of type-2 fuzzy logic systems to forecasting of time-series," *Inform. Sci.*, to be published.
[19] ——, "Operations on type-2 fuzzy sets," *Fuzzy Sets Syst.*, to be published.
[20] ——, "Centroid of a type-2 fuzzy set," *Inform. Sci.*, to be published.
[21] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
[22] K. Y. Lee, "Complex fuzzy adaptive filters with LMS algorithm," *IEEE Trans. Signal Processing*, vol. 44, pp. 424–429, Feb. 1996.
[23] Q. Liang and J. M. Mendel, "An introduction to type-2 TSK fuzzy logic systems," presented at *FUZZ-IEEE Conf.*. Seoul, Korea, Aug. 1999.
[24] ——, "Interval type-2 fuzzy logic systems: Theory and design," *IEEE Trans. Fuzzy Systems*, to be published.
[25] M. Magdon-Ismail, A. Nicholson, and Y. Abu-Mostafa, "Financial markets: Very noisy information processing," *Proc. IEEE*, vol. 86, pp. 2184–2195, Nov. 1998.
[26] J. M. Mendel, "Fuzzy logic systems for engineering: A tutorial," *Proc. IEEE*, vol. 83, pp. 345–377, Mar. 1995.
[27] ——, "Computing with words when words can mean different things to different people," in *Int. ICSC Congress Computat. Intell.: Methods Applicat., 3rd Annu. Symp. Fuzzy Logic Applicat.*, Rochester, NY, June 1999.
[28] ——, "Uncertainty, fuzzy logic, and signal processing," *Signal Processing*, to be published.
[29] M. Mizumoto and K. Tanaka, "Some properties of fuzzy sets of type-2," *Inform. Contr.*, vol. 31, pp. 312–340, 1976.
[30] ——, "Fuzzy sets of type 2 under algebraic product and algebraic sum," *Fuzzy Sets Syst.*, vol. 5, pp. 277–290, 1981.
[31] G. C. Mouzouris and J. M. Mendel, "Nonsingleton fuzzy logic systems: Theory and application," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 56–71, Feb. 1997.
[32] J. Nieminen, "On the algebraic structure of fuzzy sets of type-2," *Kybernetica*, vol. 13, no. 4, pp. 261–273, Feb. 1977.
[33] S. K. Patra and B. Mulgrew, "Efficient architecture for Bayesian equalization using fuzzy filters," *IEEE Trans. Circuits Syst.—II: Analog Digital Signal Processing*, vol. 45, pp. 812–820, July 1998.
[34] ——, "Fuzzy implementation of Bayesian equalizer in the presence of intersymbol and cochannel interference," *Proc. Inst. Elect. Eng.—Communicat.*, vol. 145, pp. 323–330, Oct 1998.
[35] P. Sarwal and M. D. Srinath, "A fuzzy logic system for channel equalization," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 246–249, May 1995.
[36] P. Savazzi, L. Favalli, E. Costamagna, and A. Mecocci, "A suboptimal approach to channel equalization based on the nearest neighbor rule," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 1640–1648, Dec. 1998.
[37] Y. Tanaka and S. Hosaka, "Fuzzy control of telecommunications networks using learning technique," *Electron. Communicat Japan*, vol. 76, pt. I, no. 12, pp. 41–51, Dec. 1993.
[38] M. Wagenknecht and K. Hartmann, "Application of fuzzy sets of type 2 to the solution of fuzzy equation systems," *Fuzzy Sets Syst.*, vol. 25, pp. 183–190, 1988.
[39] L. X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1994.
[40] L.-X. Wang, and J. M. Mendel, "Fuzzy adaptive filters, with application to nonlinear channel equalization," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 161–170, Aug. 1993.
[41] W. Wei and J. M. Mendel, "A fuzzy logic method for modulation classification in nonideal environments," *IEEE Trans. Fuzzy Syst.*, vol. 7, pp. 333–344, June 1999.
[42] K. C. Wu, "Fuzzy interval control of mobile robots," *Comput. Elect. Eng.*, vol. 22, no. 3, pp. 211–229, 1996.
[43] R. R. Yager, "Fuzzy subsets of type II in decisions," *J. Cybern.*, vol. 10, pp. 137–159, 1980.
[44] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning—1," *Inform. Sci.*, vol. 8, pp. 199–249. 1975.



Nilesh N. Karnik was born in Thane, India, in 1971. He received the B.Eng. degree in electronics engineering from Victoria Jubilee Technical Institute, University of Mumbai-India, in 1993, and the M.S. and Ph.D. degrees, both in electrical engineering, from the University of Southern California, Los Angeles, in 1997 and 1998, respectively.

Currently, he is with the Applied Technology Group, Tata Infotech Ltd., Mumbai, India. His interests include fuzzy systems and knowledge-based systems and their applications to scheduling and optimization problems, modeling, data mining, etc.

Jerry M. Mendel (S'59–M'61–SM'72–F'78) received the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, NY, in 1963.

Currently, he is a Professor of electrical engineering and the Associate Director for Education of the Integrated Media Systems Center, University of Southern California, Los Angeles, where he has been since 1974. He has published over 370 technical papers and is author and/or editor of seven books. His present research interests include type-2 fuzzy logic systems, higher order statistics, and neural networks and their applications to a wide range of signal processing problems.

Dr. Mendel is a distinguished member of the IEEE Control Systems Society. He was President of the IEEE Control Systems Society in 1986. Among his awards are the 1983 Best Transactions Paper Award of the IEEE Geoscience and Remote Sensing Society, the 1992 Signal Processing Society Paper Award, and a 1984 IEEE Centennial Medal.

Qilian Liang received the B.S. degree from Wuhan University, China, in 1993, and the M.S. degree from Beijing Unversity of Posts and Telecommunications, China, in 1996, both in electrical engineering. He is currently working toward the Ph.D. degree from the Department of Electrical Engineering Systems, University of Southern California, Los Angeles.

He is currently a Research Assistant in the Department of Electrical Engineering Systems, University of Southern California, Los Angeles. His main research interests include fuzzy logic systems, channel equalization, and video traffic classification.