ELSEVIER

Signal Processing 80 (2000) 913–933

SIGNAL PROCESSING

www.elsevier.nl/locate/sigpro

Uncertainty, fuzzy logic, and signal processing

Jerry M. Mendel*

Signal and Image Processing Institute, Department of Electrical Engineering Systems, University of Southern California, Los Angeles, CA 90089-2564, USA

Received 30 April 1999; received in revised form 15 November 1999

Abstract

In this paper we focus on model-based statistical signal processing and how some problems that are associated with it can be solved using fuzzy logic. We explain how uncertainty (which is prevalent in statistical signal processing applications) can be handled within the framework of fuzzy logic. Type-1 singleton and non-singleton fuzzy logic systems (FLSs) are reviewed. Type-2 FLSs, which are relatively new, and are very appropriate for signal processing problems, because they can handle linguistic and numerical uncertainties, are overviewed in some detail. The output of a type-2 FLS is a type-2 fuzzy set. Using a new operation called *type-reduction*, the type-2 set can be reduced to a type-1 set – the *type-reduced set* – which plays the role of a confidence interval for linguistic uncertainties. No such result can be obtained for a type-1 FLS. We demonstrate, by means of examples, that a type-2 FLS can outperform a type-1 FLS for one-step prediction of a Mackey–Glass chaotic time series whose measurements are corrupted by additive noise, and equalization of a nonlinear time-varying channel. © 2000 Elsevier Science B.V. All rights reserved.

Zusammenfassung

In diesem Artikel setzen wir den Schwerpunkt auf modellbasierte statistische Signalverarbeitung und zeigen Möglichkeiten zur Lösung von Problemen dieses Umfeldes mit Hilfe der Fuzzy Logik auf. Wir erläutern, wie Entscheidungsunsicherheit (die in sämtlichen Anwendunger der statistischen Signalverarbeitung vorherrscht) im Rahmen der Fuzzy Logik behandelt werden kann. Zunächst werden Typ-1 Singleton und Nicht-Singleton Fuzzy Logik Systeme (FLS) besprochen. Typ-2 FLS, die relativ neu sind, werden etwas detaillierter behandelt. Sie eignen sich sehr zur Lösung von Signalverarbeitungsproblemen, da sie linguistische und numerische Unsicherheit handhaben können. Die Ausgabe eines Typ-2 FLS stellt die Typ-2 Fuzzy-Menge dar. Mit Hilfe einer neuen, als *Typ-Reduktion bezeichneten* Operation, kann die Typ-2 Menge auf eine Typ-1 Menge – die *Typ-reduzierte* Menge – überführt werden. Sie entspricht einem Konfidenzintervall für linguistische Unsicherheit. Kein derartiges Resultat kann für eine Typ-1 FLS abgeleitet werden. Anhand von Beispielen zeigen wir, daß eine Typ-2 FLS eiher Typ-1 FLS zum einen als Einschritt-Prädiktion überlegen sein kann, wenn, sie auf eine chaotische Mackey–Glass Zeitreihe angwandt wird, deren Meßwerte durch additives Rauschen gestört sind oder wenn sie zum anderen bei der Entzerrung eines nichtlinearen zeitvarianten Kanals angewandt wird. © 2000 Elsevier Science B.V. All rights reserved.

Résumé

Dans cet article, nous nous concentrons sur le traitement statisque de signaux à base de modèles et sur la façon dont certains problèmes qui lui sont associés peuvent être résolus en utilisant de la logique floue. Nous expliquons comment

* Tel.: (213) 740-4445; fax: (213) 740-4651.
E-mail address: mendel@sipi.usc.edu (J.M. Mendel)

l'incertitude (qui prévaut en traitement statistique des signaux) peut être manipuléoe dans le cadre de la logique floue. Nous passons en revue des systémes à logique floue (SLF) à singletones de type 1 et sans singletons. Nous présentons avec quelques détails les SLF de type 2, qui sont relativement nouveaux et trés appropriés pour des problémes de traitement de signaux, car ils peuvent traiter des incertitudes linguistiques et numériques. La sortie d'un SLF de type 2 est un ensemble flou de type 2. En utilisant une nouvelle opération appelée réduction de type, l'ensemble de type 2 peut être réduit á un ensemble de type l, l'ensemble de type réduit, qui joue le rôle d'un intervalle de confiance pour les incertitudes linguistiques. Un tel résultat ne peut être atteint pour un SLF de type 1, Nous démontrons, au moyen d'exemples, qu'une SLF de type 2 peut dépasser un SLF de type 1 pour une prédiction d'un pas d'une série temporelle chaotique de Mackey–Glass dont les mesures sont corrompues par un bruit additif, ainsi que pour l'égalisation d'un canal non linéaire variant dans le temps. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Uncertainty; Fuzzy logic; Type-2 fuzzy logic; Signal processing; Forecasting of time series; Equalization

1. Introduction

In this paper we focus on model-based statistical signal processing and how some problems that are associated with it can be solved using fuzzy logic. To some readers this may sound like a contradiction because model-based statistical signal processing is formulated within the framework of probability and random processes whereas fuzzy logic is not based on a probability model.

Is there a shortcoming to model-based statistical signal processing that calls its use into question? There is – the assumed probability model, for which model-based statistical signal processing results will be good if the data agrees with the model, but may not be so good if the data does not. Let me be specific about this by way of an example. So as not to pick on any one else's work, I will pick on my own.

Starting in 1975, my students and I developed a novel and comprehensive model-based approach to seismic deconvolution. It was based on a widely used convolutional model for a seismogram. One of our contributions was to depart from the then commonly used white-Gaussian model for the seismic reflectivity sequence. Instead we modeled the reflectivity sequence as a white Bernoulli–Gaussian (product model) random sequence. This sequence could provide strong isolated spikes indicative of a strong reflector, as well as lots of smaller amplitude spikes, indicative of multiple reflections; so, it was physically motivated. The non-Gaussian nature of this sequence meant that it should be possible to improve upon deconvolution results that only used second-order statistics. We then developed maximum-likelihood deconvolution (MLD), which involved detection and estimation. In more realistic situations, where the seismic source signature was not measured, and had to be estimated, the entire MLD procedure alternated between optimization (to estimate the parameters in an ARMA model for the source signature and some of the statistical parameters), detection (to determine the time points at which significant events occurred) and estimation (to restore the amplitudes at the significant time points). This work led to two books [30,31] and to many journal and conference articles (see the references in the books). Then, in 1988 one of my students (Li-Xin Wang) got the idea of using Hopfield neural networks to do what we were doing in MLD – without a probability-based model [50]. A convolutional model was still used, and the product structure of the Bernoulli–Gaussian product model was also used. We developed three Hopfield neural networks, one each for estimating the coefficients in the source signature's IR, detecting where a significant event was located, and then restoring the amplitude of such events. When the neural network and MLD deconvolution procedures were tested on real data, the former significantly outperformed the latter. The reason for this was that, for the data we used, the convolutional Bernoulli–Gaussian model did not fit it as well as did a convolutional model in which we did not use an a priori probability model for the reflectivity sequence. This work opened my eyes toward model-free signal processing.

Today, the two major approaches to model-free signal processing are artificial neural networks and fuzzy logic. In this paper, I focus on the latter, and how uncertainty can be handled within the framework of fuzzy logic. Note though that I will not abandon our traditional ideas about noisy measurements, i.e., measurement = signal + noise. What I will abandon (at least in this paper) is the frequently made assumption of a priori knowledge of a probability model (i.e., a probability density function) for either the signal or the noise.

Four applications to which we have applied fuzzy logic are identification of non-linear non-dynamic systems, forecasting of time series, digital modulation classification, and equalization of non-linear time-varying channels. Uncertainties can occur in the: identification application if only noisy training data is available; time-series forecasting application if the training data is perfect but the later measurements are noisy, or if both the training data and the later measurements are noisy; modulation classification application if the modulation constellations drift or if measurement noise is present; and, equalization application due to unknown time variations of the channels, especially in a mobile communication environment.

Type-1 fuzzy logic systems, which are widely prevalent in the fuzzy logic literature, are reviewed in Section 2, with emphasis on what sources of uncertainty can be accommodated by them. Type-2 fuzzy logic systems, which are not widely known, are described in Section 3, because they are potentially very applicable in signal processing and digital communications applications of fuzzy logic. Two examples are given in Section 4. Conclusions are drawn in Section 5.

2. Type-1 fuzzy logic systems

Fig. 1 depicts a type-1 fuzzy logic system (FLS) (e.g., [32]) that is widely used in fuzzy logic controllers and signal processing applications (because later we describe type-2 FLSs, we must distinguish between them and existing FLSs; hence, we refer to the latter as *type*-1 *FLSs*). According to Jang [10] a type-1 FLS is also known as a *fuzzy inference*

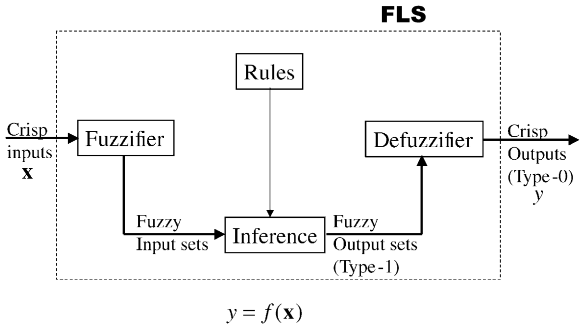

Fig. 1. Type-1 fuzzy logic system.

system, *fuzzy rule-based system*, *fuzzy model*, *fuzzy associative memory*, or a *fuzzy controller* when it is used as a controller. A type-1 FLS maps crisp inputs into crisp outputs. It contains four components: rules, fuzzifier, inference engine, and defuzzifier. Once the rules have been established, the type-1 FLS can be viewed as a mapping from inputs to outputs (the solid path in Fig. 1, from Crisp Inputs to Crisp Outputs), and this mapping can be expressed quantitatively as $y = f(\mathbf{x})$.

Fuzzy sets can be interpreted as membership functions μ_X that associate with each element x of the universe of discourse, U, a number $\mu_X(x)$ in the interval [0,1]:

$$\mu_X : U \to [0,1]. \tag{1}$$

The fuzzifier maps a crisp point $x \in U$ into a fuzzy set $X \in U$. In the case of a *singleton* fuzzifier (which is useful when input measurements are perfect), the crisp point $x \in U$ is mapped into a fuzzy set X with support x_i, where $\mu_X(x_i) = 1$ for $x = x_i$ and $\mu_X(x_i) = 0$ for $x \neq x_i$, i.e., the *single* point in the support of X with non-zero membership function value is $x = x_i$. In the case of a *non-singleton* fuzzifier (which is useful when input measurements are corrupted by noise), the point $X \in U$ is mapped into a fuzzy set X with support x_i, where μ_X achieves maximum value at $x = x_i$ and decreases while moving away from $x = x_i$. A non-singleton fuzzifier treats the point $X \in U$ as a fuzzy number. We assume that fuzzy set X is normalized so that $\mu_X(x_i) = 1$.

The most widely used fuzzifier is the singleton fuzzifier, mainly because of its simplicity and lower computational requirements; however, this kind of fuzzifier may not always be adequate, especially in cases where noise is present in the measurements that are later processed by the system. A different approach is necessary in order to account for uncertainty in the data, one which we describe in Section 2.2.

2.1. Singleton type-1 FLS

Here we consider a type-1 FLS with a rule-base of M rules; let the lth rule be denoted by

R^l: IF u_1 is F_1^l and u_2 is F_2^l and…
u_p is F_p^l THEN v is G,

where u_k, $k = 1,2,\ldots,p$, and v are the input and output linguistic variables, respectively, $F_k^l \subset U_k$, and $G^l \subset V$. Each rule can be viewed as a fuzzy relation R^l [55] from a set U to a set V, where U is the Cartesian product space $U = U_1 \times U_2 \times \cdots \times U_p$. R^l itself is a subset of the Cartesian product $U \times V = \{(\boldsymbol{x}, y): \boldsymbol{x} \in U, y \in V\}$, where $\boldsymbol{x} \equiv (x_1, x_2, \ldots, x_p)$, and x_k and y are the points in the universes of discourse, U_k and V, of u_k and v. Given the input $A \subset U$ to R^l, we can obtain the lth output fuzzy set $Y^l \subset V$ using the Compositional Rule of Inference (also known as the *sup-star composition*), as $Y^l = A \circ R^l = \sup_{x \in U}(A \bigstar R^l)$, where $\bigstar$ denotes a t-norm.

Using all of this as the starting point, it can be shown [32,40] that a very widely used FLS, one with singleton fuzzification, product implication, product t-norm, and modified height defuzzification, can be written as a fuzzy basis function (FBF) expansion:

$$f(\boldsymbol{x}(t)) = \frac{\sum_{l=1}^{M} \bar{y}^l T_{k=1}^{p} \mu_{F_k^l}(x_k(t))/(\delta^l)^2}{\sum_{l=1}^{M} T_{k=1}^{p} \mu_{F_k^l}(x_k(t))/(\delta^l)^2} = \sum_{l=1}^{M} \bar{y}^l \phi^l(\boldsymbol{x}(t)) \tag{2}$$

where the fuzzy basis functions are given by

$$\phi^l(\boldsymbol{x}(t)) = \frac{T_{k=1}^{p} \mu_{F_k^l}(x_k(t))/(\delta^l)^2}{\sum_{l=1}^{M} T_{k=1}^{p} \mu_{F_k^l}(x_k(t))/(\delta^l)^2}. \tag{3}$$

In these equations M denotes the number of rules; p the number of antecedents; $\bar{y}^l$ the point of maximum membership of the lth consequent fuzzy set; $\mu_{F_k^l}$ the kth antecedent membership function for the lth rule; $\bigstar$ is a t-norm (e.g., min or product); $T_{k=1}^{p}$ is a sequence of $p-1$ t-norm operations, i.e., $T_{k=1}^{p}\alpha_k = \alpha_1 \bigstar \alpha_2 \bigstar \cdots \bigstar \alpha_p$; δ^l is proportional to the uncertainty in the consequent fuzzy sets (e.g., if the consequent membership function is triangular, then δ^l could be chosen as the length of its base; the smaller δ^l is then the more certain we are that the consequent of the lth rule is G^l); and, $\boldsymbol{x}(t) = \mathrm{col}[x_1(t), \ldots, x_p(t)]$ is the p-dimensional input vector to the type-1 FLS.

This way for handling consequent uncertainty (i.e., using δ^l) is a bit artificial, but appears to be the only way we can do it within the framework of type-1 FLSs. Eq. (2) also can be derived in a different way by requiring that the output of the FLS be a weighted linear combination of each fired rule's fuzzy set followed by centroid defuzzification. This result is called the *standard additive model*, and was developed, for example, in [23], but just for a product t-norm.

Note that (2) has the appearance of a regression model, where the regressors, the $\phi^l(\boldsymbol{x}(t))$, are *nonlinear* functions of the system's inputs $\boldsymbol{x}(t)$, and the weights (regression coefficients) are the $\bar{y}^l$'s. Although (2) looks like a regression equation, the $\bar{y}^l$'s are *not* determined by regression analysis. The structure of (2) is a direct result of the mathematics of fuzzy logic. There is yet another major difference between (2) and a regression equation. In a standard regression model, *all M* of its terms are activated for each $\boldsymbol{x}(t)$. Although it is not obvious by the way in which (2) is written, (2) is a *variable-structure model*, in the sense that all of its M terms are not activated by each $\boldsymbol{x}(t)$. The activated terms depend upon $\boldsymbol{x}(t)$. Each fired rule corresponds to a fuzzy basis function in (1), and, in general only a very small number of rules are fired for each $\boldsymbol{x}(t)$. So, the FLS in (2) can be viewed as a collection of localized sub-systems.

2.2. Non-singleton type-1 FLS

Non-singleton fuzzification is especially useful in cases where the input data to the FLS contains uncertainty, such as additive noise. Conceptually, the non-singleton fuzzifier implies that the given input value x is the most likely value to be the correct one from all the values in its immediate neighborhood; however, because of the presence of uncertainty, neighboring points are also likely to be the correct value, but to a lesser degree. The shape of the membership function μ_x can be determined by the designer, based on an estimate of the kind and quantity of uncertainty present. It would be the logical choice, though, for the membership function to be symmetric about x, since the effect of noise is most likely to be equivalent on all points. Gaussian or triangular membership functions are popular for μ_X. In this way the measured value of x is treated as a fuzzy number, which is different from the way it would be treated in the framework of statistical signal processing.

A popular non-singleton FLS (NSFLS), one with product implication, product t-norm, and modified height defuzzification, can be written as a non-singleton fuzzy basis function expansion [39,40], i.e.,

$$f_{\mathrm{ns}}(\boldsymbol{x}(t)) = \frac{\sum_{l=1}^{M} \bar{y}^l T_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l(t))/(\delta^l)^2}{\sum_{l=1}^{M} T_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l(t))/(\delta^l)^2} = \sum_{l=1}^{M} \bar{y}^l \phi_{\mathrm{ns}}^l(\boldsymbol{x}(t)) \tag{4}$$

where the non-singleton fuzzy basis functions are given by

$$\phi_{\mathrm{ns}}^L(\boldsymbol{x}(t)) = \frac{T_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l(t))/(\delta^l)^2}{\sum_{l=1}^{M} T_{k=1}^{p} \mu_{Q_k^l}(x_{k,\max}^l(t))/(\delta^l)^2}. \tag{5}$$

In (4) and (5) the symbols that are common to (2) and (3) have been defined previously, and

$$\mu_{Q_k^l} \equiv \mu_{F_k^l} \bigstar \mu_{X_k} \tag{6}$$

where $\mu_{F_k^l}$ denotes the kth antecedent membership function for the lth rule, and μ_{X_k} denotes the kth input membership function. In (5) and (6) $x_{k,\max}^l$ is the point that maximizes $\mu_{Q_k^l}$. Note that the maximization of $\mu_{Q_k^l}$ is a direct result of the sup-star composition and non-singleton fuzzification. For singleton fuzzification no maximization is necessary, which is why, perhaps, that singleton FLSs are so popular.

When the t-norm is the product, and all membership functions are Gaussian, then it is straightforward to carry out the maximization computations in (6). In this case, the kth input fuzzy set and the corresponding rule antecedent fuzzy sets are assumed to have the following forms: $\mu_{X_k}(x_k) = \exp\{-1/2[(x_k - m_{X_k})/\sigma_{X_k}]^2\}$ and $\mu_{F_k^l}(x_k) = \exp\{-1/2[(x_k - m_{F_k^l})/\sigma_{F_k^l}]^2\}$. By maximizing the function

$$\mu_{Q_k^l}(x_k) = \mu_{F_k^l}(x_k)\mu_{X_k}(x_k) \tag{7}$$

we find that it is maximum at

$$x_{k,\max} = (\sigma_{X_k}^2 m_{F_k^l} + \sigma_{F_k^l}^2 m_{X_k})/(\sigma_{X_k}^2 + \sigma_{F_k^l}^2). \tag{8}$$

In the special but important case when all input points for each input variable have the same level of uncertainty (as in the application of stationary time-series forecasting, when each input is a time-delayed version of the measured time series), the spreads of the input sets will be the same, in which case $\sigma_{X_k}^2$ in (8) is a constant. Usually, we choose the mean of the fuzzy input sets, m_{X_k}, to be the noisy measured input, x_k'; hence, under these conditions, (8) simplifies to

$$x_{k,\max} = (\sigma_X^2 m_{F_k^l} + \sigma_{F_k^l}^2 x_k')/(\sigma_X^2 + \sigma_{F_k^l}^2). \tag{9}$$

This formula can be interpreted as a pre-filtering of the noisy data x_k'. A FLS, therefore, has a built-in front-end mechanism for such pre-filtering, namely the fuzzifier. Interestingly, a neural network does not appear to have this capability.

When the uncertainty of the input becomes zero (i.e., $\sigma_X^2 = 0$), then (9) reduces to the singleton case, i.e., $x_{k,\max} = x_k'$, for which

$$\mu_{X_k}(x_{k,\max} = x') = \exp\{-1/2[(x_k' - x_k')/\sigma_X]^2\} = 1 \quad (k = 1, \ldots, p),$$

so that, from (7), $\mu_{Q_k^l}(x_{k,\max}) = \mu_{F_k^l}(x_{k,\max}) = \mu_{F_k^l}(x_k')$.

2.3. Designs of type-1 FLSs

Type-1 FLSs contain parameters that can either be pre-specified or can be tuned during a training process. The parameters for a singleton FLS are: antecedent membership function parameters (e.g., the mean and the variance of a Gaussian membership function; the apex point and spread of a triangular membership function), the point of maximum membership of the lth consequent fuzzy set ($\bar{y}^l$), and the uncertainty spread of the lth consequent fuzzy set (δ^l). For a p-antecedent, M-rule system that uses Gaussian or symmetrical triangular membership functions, there can be a maximum of $M(2p+2)$ parameters. For a NS FLS, there is at most one additional parameter for each antecedent (the variance of the measurement's fuzzy number representation if a Gaussian membership function is used, or the spread of that number if a triangular membership function is used). So, there will be at most $M(2p+2)+p$ parameters for a NS FLS. If the variances or spread of the antecedent memberships are equal, then there will be at most $M(2p+2)+1$ parameters for a NS FLS.

A multitude of design methods exist for tuning some or all of the FLS's parameters. A discussion about this is outside of the scope of the present paper (see, e.g., [10,11,34,28]).

3. Type-2 fuzzy logic systems

3.1. Introduction

Quite often, the knowledge that is used to construct the rules in a FLS is *uncertain*. Three ways in which such rule uncertainty can occur are: (1) the words that are used in antecedents and consequents of rules can mean different things to different people [33]; (2) consequents obtained by polling a group of experts will often be different for the same rule, because the experts will not necessarily be in agreement; and, (3) only noisy training data is available. Antecedent or consequent uncertainties translate into uncertain antecedent or consequent membership functions. Type-1 FLSs, whose membership functions are type-1 fuzzy sets, are unable to directly handle rule uncertainties. This section, therefore, addresses a new class of FLSs – *type-2 FLSs* – in which antecedent or consequent membership functions are type-2 fuzzy sets whose membership grades are themselves type-1 fuzzy sets. Type-2 FLSs 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 rule uncertainties, and, as we shall demonstrate, they let us propagate such uncertainties through them, so that we can establish their effects at the output of the FLS.

We pause to distinguish between two types of uncertainties, *random* and *linguistic*. Probability theory is associated with the former, and, as we shall demonstrate below, FL can be associated with the latter. Within probability theory we begin with a pdf which embodies total information about random uncertainties. In most practical applications it is impossible to know or determine the pdf; so, we fall back on using the fact that a pdf is completely characterized by all of its moments. For most pdf's, an infinite number of moments are required. Of course, it is not possible, in practice, to determine an infinite number of moments; so, instead, we compute as many moments as are necessary to extract as much information as possible from the data. At the very least, we use two moments – the mean and variance; and, in some cases, we use even moments higher than second order. 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. So, our accepted probabilistic modeling of random uncertainty focuses to a large extent on methods that use at least the first two moments of a pdf. This is, for example, why designs based on minimizing mean-squared errors are so popular.

Should we expect any less of a FLS for rule uncertainties? The defuzzified output of a type-1 FLS is analogous (in spirit) to computing the mean of a pdf. Just as variance provides a measure of dispersion about the mean, and is used to capture more about probabilistic uncertainty in practical statistical-based designs, FLSs also need some measure of dispersion to capture more about rule uncertainties than just a single number – the traditional defuzzified output. Type-2 FL provides this measure of dispersion.

In a FLS, rule uncertainties occur due to linguistic or numerical uncertainties about the knowledge used to construct the rules. These uncertainties can be handled by using type-2 fuzzy sets. Because knowledge about type-2 fuzzy sets and FLSs is not widespread, we provide a much more extensive discussion about them in this section than we did for type-1 FLSs.

The concept of a *type-2 fuzzy set* was introduced by Zadeh [56] as an extension of the concept of an ordinary fuzzy set (henceforth called a *type-1 fuzzy set*). A type-2 fuzzy set is characterized by a fuzzy membership function, i.e., the membership value (or membership grade) for each element of this set is a fuzzy set in [0,1], unlike a type-1 set where the membership grade is a crisp number in [0,1]. Such sets are useful in circumstances where it is difficult to determine the exact membership function for a fuzzy set, and have been studied by only a relatively small number of people, including: Mizumoto and Tanaka [35,36], Nieminen [41], Dubois and Prade [5,6], Turksen [48], Gorzalczany [9], Wagenknecht and Hartmann [49], and, more recently, by Karnik and Mendel [13,15,18].

Fig. 2 shows a type-2 set, where the membership grade for every point is a crisp set, the domain of which is an interval contained in [0,1]. We call such type-2 sets "interval type-2 sets" and their membership grades "interval type-1 sets". Since all the memberships in an interval type-1 set are unity, in

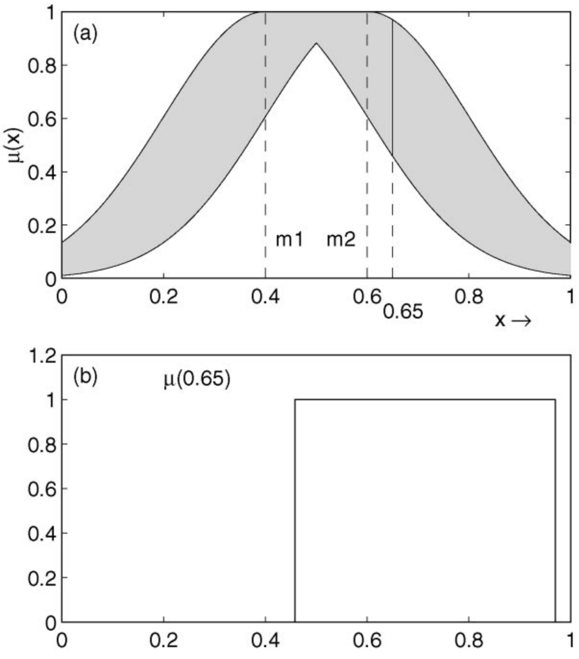

Fig. 2. (a) An interval type-2 set. Since all the secondary membership functions are unity, the shading is uniform all over. The domain of the membership grade at $x = 0.65$ is also shown. The secondary memberships in this type-1 set are shown in (b), and are all equal to 1, i.e., the membership grade is an interval type-1 set.

the sequel we represent such a set just by its domain interval, which can be represented in terms of its left and right end points as $[l, r]$, or in terms of its center and spread as $[m - s, m + s]$, where $m = (l + r)/2$ and $s = (r - l)/2$. Interval type-2 sets are the simplest kind of type-2 sets, and we have developed fast algorithms to compute the output of an "interval" type-2 FLS, i.e., a type-2 FLS which uses interval type-2 sets. Gaussian and triangular type-2 sets are described in [13,19]. Note, however, that *although our discussions here focus on interval type-2 sets, type-2 FLSs are in no way limited to such sets.*

The shaded area in Fig. 2a is a *footprint of uncertainty* for a Gaussian membership function whose mean value is uncertain but is known to lie in the interval $[m_1, m_2]$. The footprint of uncertainty represents the domain for the secondary membership functions. Regardless of the origin of the footprint of uncertainty (e.g., a Gaussian membership function whose standard deviation is uncertain but is known to lie in an interval; or, a triangular membership function whose vertices are uncertain but are known to lie in intervals), when the secondaries are intervals (Gaussians, triangles, etc.) the type-2 fuzzy set is called an interval (Gaussian, triangle, etc.) type-2 fuzzy set.

3.2. Notation and terminology

We must now introduce notation that lets us distinguish between type-1 and type-2 sets and their associated membership functions. We will use the earlier well-established and widely used notation for type-1 sets, i.e., a type-1 fuzzy set A in X is written as A. The membership grade (a synonym for the degree of membership) of $x \in X$ in A is $\mu_A(x)$, which is a crisp number in [0,1]. If X is a continuum, we represent A as $A = \int_{x \in X} \mu_A(x)/x$, where the integral denotes logical union. If X is discrete, we replace the integral by a summation. A type-2 fuzzy set in X is $\tilde{A}$, and the membership grade of $x \in X$ in $\tilde{A}$ is $\mu_{\tilde{A}}(x)$, which is a type-1 fuzzy set in [0,1]. The elements of the domain of $\mu_{\tilde{A}}(x)$ (e.g., the vertical axis in Fig. 2a and the horizontal axis in Fig. 2b) are called *primary memberships* of x in $\tilde{A}$ and the memberships of the primary memberships in $\mu_{\tilde{A}}(x)$ (e.g., the vertical axis in Fig. 2b) are called *secondary memberships* of x in $\tilde{A}$. If all the membership grades of $\tilde{A}$ are such that, for every $x \in X$, only one primary membership has a secondary membership equal to 1 (e.g., triangles), then we call the set of all such primary memberships the *principal* membership function of $\tilde{A}$. The membership grade of any $x \in X$ in $\tilde{A}$ can be represented as $\mu_{\tilde{A}}(x) = \int_{u \in [0,1]} f_x(u)/u$. The membership function of a type-2 fuzzy set can be thought of as a fuzzy-valued function which assigns to every $x \in X$ a type-1 fuzzy membership grade. It is in this sense that we call X the domain of the type-2 fuzzy set.

3.3. Operations on type-2 sets

We have just seen that the membership grades of type-2 sets are type-1 sets; therefore, in order to perform operations like union and intersection on type-2 sets (which are needed to implement a type-2 FLS), we need to be able to perform t-conorm and t-norm operations between type-1 sets. This is done using Zadeh's Extension Principle (e.g., [56,7,13]). As is well known [22], a binary operation $*$ between two crisp numbers can be extended to two type-1 sets $F = \int_v f(v)/v$ and $G = \int_w g(w)/w$, as (as above, the integrals denote logical unions)

$$F * G = \int_v \int_w [f(v) \bigstar g(w)]/(v * w) \tag{10}$$

where $\bigstar$ denotes the chosen t-norm. We will generally use product or minimum t-norm and maximum t-conorm. For example, the extension of the t-conorm operation to type-1 sets is

$$F \sqcup G \int_v \int_w [f(v) \bigstar g(w)]/(v \vee w). \tag{11}$$

This is called the *join* operation and $\sqcup$ denotes the *join* operation [35]. Similarly, the extension of the t-norm operation to type-1 sets, which is known as the *meet* operation [35], is

$$F \sqcap G \int_v \int_w [f(v) \bigstar g(w)]/(v \wedge w) \tag{12}$$

where $\sqcap$ denotes the *meet* operation.

Example 1 (Karnik and Mendel [13]). Let F and G be two interval type-1 sets with domains $[l_f, r_f]$ and $[l_g, r_g]$, respectively (we drop the tilde, since the sets are crisp). Using (12), the *meet* between F and G, under product t-norm, can be obtained as $F \sqcap G \int_{v \in F} \int_{w \in G} (1 \times 1)/vw$. Observe that: (a) each term in $F \sqcap G$ is equal to the product vw for some $v \in F$ and $w \in G$, the smallest term being $l_f l_g$ and the largest $r_f r_g$; and, (b) since both F and G have continuous domains, $F \sqcap G$ also has a continuous domain; consequently, $F \sqcap G$ is an interval type-1 set with domain $[l_f l_g, r_f r_g]$, i.e., $F \sqcap G = \int_{u \in [l_f l_g, r_f r_g]} 1/u$. In a similar manner, the *meet* $\sqcap_{i=1}^n F_i$ of n interval type-1 sets $F_1, \ldots, F_n$, having domains $[l_1, r_1], \ldots, [l_n, r_n]$, respectively, under product t-norm, is an interval set with domain $[\sqcap_{i=1}^n l_i, \sqcap_{i=1}^n r_i]$. Note that Kaufman and Gupta [20] give a similar result for the multiplication of fuzzy numbers.

We have developed fast algorithms for computing *join* and *meet* of type-1 fuzzy sets for some cases where the sets involved are not interval type-1 sets (see [13,15] for details; see, also [5]).

Algebraic operations between type-1 sets are also defined using (10), e.g., the algebraic sum of F and G can be defined as

$$F + G = \int_v \int_w [f(v) \bigstar g(w)]/(v + w). \tag{13}$$

Example 2 (Karnik and Mendel [13]). Using the same reasoning as in Example 1, it can be shown that when F and G are interval type-1 sets with domains $[l_f, r_f]$ and $[l_g, r_g]$, respectively, their algebraic sum is also an interval type-1 set with domain $[l_f + l_g, r_f + r_g]$ (see [20] for a similar result). More generally, we have proved [13,15] the following result for interval type-1 sets: Given n interval type-1 sets $F_1, F_2, \ldots, F_n$ with means $m_1, m_2, \ldots, m_n$ and spreads $s_1, s_2, \ldots, s_n$, their affine combination $\sum_{i=1}^n \alpha_i F_i + \beta$, where α_i $(i = 1, \ldots, n)$ and β are crisp constants, is also an interval type-1 set with mean $\sum_{i=1}^n \alpha_i m_i + \beta$ and spread $\sum_{i=1}^n |\alpha_i| s_i$.

Observe, from (10) and (12), that, when using product t-norm, the product of F and G is the same as the *meet* of F and G; hence, all our earlier discussions about the *meet* operation under product t-norm apply to the multiplication of type-1 sets under product t-norm.

Using the Extension Principle, an n-ary operation $f(\theta_1,\dots,\theta_n)$ on crisp numbers can be extended to n type-1 fuzzy sets $F_1,\dots,F_n$ as [35]

$$f(F_1,\dots,F_n) = \int_{\theta_1}\cdots\int_{\theta_n}\mu_{F_1}(\theta_1)\bigstar\cdots\bigstar\mu_{F_n}(\theta_n)/f(\theta_1,\dots,\theta_n) \tag{14}$$

where all the integrals denote logical union, and $\theta_i \in F_i$ for $i = 1,\dots,n$.

Next, we define the concept of the "centroid" of a type-2 set using (14) [13,18]. This concept is required in a type-2 FLS, as we explain below in Section 3.4. Recall that the centroid of a type-1 set A, whose domain is discretized into N points, is given as

$$C_A = \sum_{i=1}^{N} x_i\mu_A(x_i)\bigg/\sum_{i=1}^{N}\mu_A(x_i). \tag{15}$$

Similarly, the centroid of a type-2 set $\tilde{A}$, whose domain is discretized into N points, can be defined using (14) as follows. If we let $D_i = \mu_{\tilde{A}}(x_i)$, then

$$C_{\tilde{A}} = \int_{\theta_1}\cdots\int_{\theta_N}\mu_{D_1}(\theta_1)\bigstar\cdots\bigstar\mu_{D_N}(\theta_N)\bigg/\left[\sum_{i=1}^{N}x_i\theta_i\bigg/\sum_{i=1}^{N}\theta_i\right] \tag{16}$$

where $\theta_i \in D_i$, and all the integrals denote logical union. Because of the importance of (16) to a type-2 FLS, we shall elaborate on its meaning and computation.

Every combination, $\{\theta_1,\dots,\theta_N\}(\theta_i \in D_i)$, considered when computing $C_{\tilde{A}}$, can be thought to form the membership function of some type-1 set A' which has the same domain as $\tilde{A}$. We call A' an *embedded type-1 set* in $\tilde{A}$ (see Fig. 3a for two examples of embedded type-1 sets). Every embedded type-1 set also has a weight associated with it, which is calculated as the t-norm of the secondary memberships corresponding to $\{\theta_1,\dots,\theta_N\}$ that make up that embedded set. The type-2 set $\tilde{A}$ can, therefore, be thought of as a large collection of embedded type-1 sets, each having a weight associated with it, and, its centroid $C_{\tilde{A}}$ can be thought of as a type-1 set whose elements are the centroids

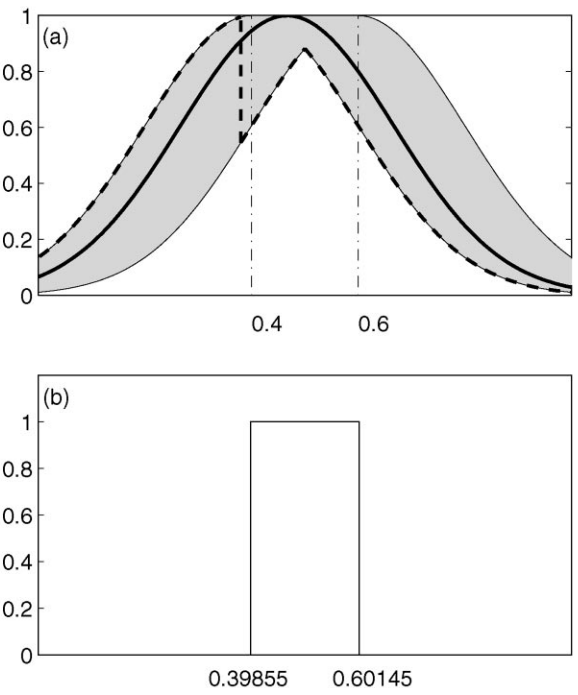

Fig. 3. (a) The interval type-2 set shown in Fig. 2a. Two embedded type-1 sets are also shown, one with a thick dashed line and the other with a thick solid line. (b) Centroid of the type-2 set in Fig. 3a.

[computed via (15) by using θ_i in place of $\mu_{\tilde{A}}(x_i)$] of all the embedded type-1 sets in $\tilde{A}$, and their memberships are the weights associated with the corresponding embedded sets. If the domain of $\tilde{A}$ and/or $\mu_{\tilde{A}}(x_i)$ $(x \in \tilde{A})$ is continuous, the domain of $C_{\tilde{A}}$ is also continuous. The number of all the embedded type-1 sets in $\tilde{A}$, in this case, is uncountable; therefore the domains of $\tilde{A}$ and each $\mu_{\tilde{A}}(x_i)$ $(x \in \tilde{A})$ have to be discretized for the calculation of $C_{\tilde{A}}$. We assume discretization.

Example 3 (Karnik and Mendel [13,18]). If $\tilde{A}$ is an interval type-2 set, (16) simplifies to $C_{\tilde{A}} = \int_{\theta_1}\cdots\int_{\theta_N} 1\ /[\sum_{i=1}^{N}x_i\theta_i/\sum_{i=1}^{N}\theta_i]$ where each θ_i belongs to some interval in [0,1]. Consider the interval type-2 set in Fig. 3a. Using the simple computational procedure described in [13,18], we find that $C_{\tilde{A}}$ is an interval type-1 set with domain [0.39855,0.60145]. As explained in [13,18], only two sets of computations are needed to obtain $C_{\tilde{A}}$, one each for its left and right end-points, and, each of these computations requires at most N iterations, where N is the number of points that the domain of $\tilde{A}$ has been discretized into.

See [13,18] for more discussions about the centroid of a type-2 set, including how it can be computed for general type-2 sets.

3.4. Type-2 fuzzy logic systems

Fig. 4 depicts the structure of a type-2FLS [12,13,19]; it is quite similar to a type-1 FLS, the only difference being that the antecedent and/or consequent sets in a type-2 FLS are type-2, so that each rule output set is type-2. *Extended* versions of type-1 defuzzification methods [obtained using (14)] yield a type-1 set from the type-2 rule output sets. We call this process *type reduction* [14] rather than defuzzification, and the resulting type-1 set, the *type-reduced set*. The type-reduced set can then be defuzzified to obtain a crisp output. We call the combination of type reduction and defuzzification *output processing*. An example of output processing is depicted in Fig. 5. The type-reduced set of a type-2 FLS shows the possible variation in the crisp output of the FLS due to uncertain natures of the antecedents and/or consequents. It establishes a band of values around a crisp output value in much the same way that a confidence interval establishes a band about a point estimate when stochastic uncertainty is present; but, it does this for linguistic uncertainties.

The fundamental design requirement in our work is that a type-2 FLS must reduce to a type-1 FLS when all rule uncertainties disappear, in which case, output processing reduces just to defuzzification.

Just as a type-2 set can be thought of as a collection of a large number of embedded type-1 sets, a type-2 FLS can be thought of as a collection of a large number of embedded type-1 FLSs (see Fig. 6), each of which has a certain weight associated with it. Consequently, as shown in Fig. 6, each point and its membership function in the type-reduced set of a type-2 FLS can be thought of as, respectively, the output and the weight associated with an embedded type-1 FLS. *When interval type-2 sets are used, as in this paper, then a type-2 FLS is a combination of just two type-1 FLSs, one for the left-end of the type-reduced set* (y_L) *and the other for the right-end of the type-reduced set* (y_R). This represents a huge savings in computation for an interval type-2 FLS over other type-2 FLSs.

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

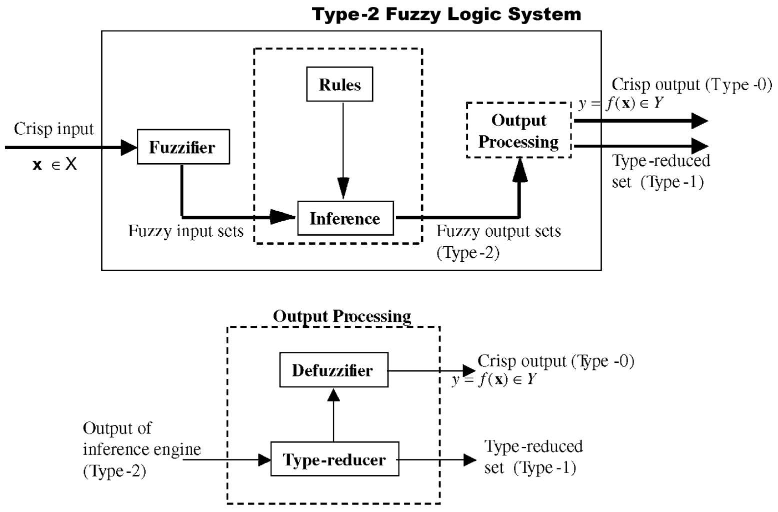

Fig. 4. The structure of a type-2 FLS. Note that output processing consists of type reduction followed by defuzzification.

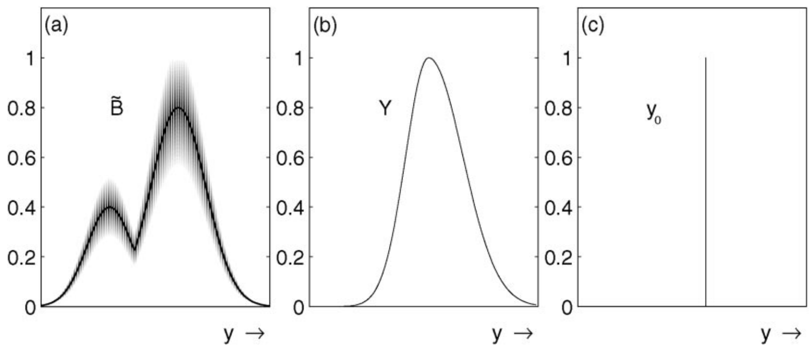

Fig. 5. Pictorial representation of output processing. First, the individual rule output sets are combined in some manner to obtain type-2 fuzzy set $\tilde{B}$ (Fig. a); then, type reduction is applied to $\tilde{B}$ to obtain the type-1 set Y (Fig. b); finally, Y is defuzzified to produce the crisp output y_0 (Fig. c).

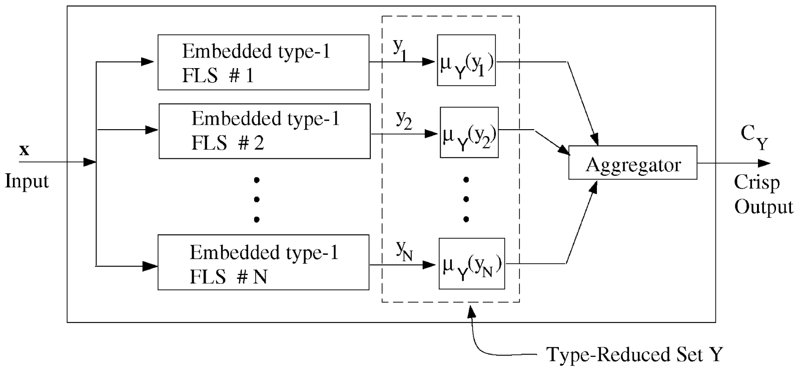

Fig. 6. Interpretation of a type-2 FLS as a collection of type-1 FLSs. This figure is drawn assuming that membership grades have been discretized, as would be done in practice, so that there are a finite number of type-1 FLSs.

the form:

R^l: IF x_1 is $\tilde{F}^l_1$ and x_2 is $\tilde{F}^l_2$ and $\cdots$ and x_p is $\tilde{F}^l_p$, THEN y is $\tilde{G}^l$,

$l = 1,\dots,M$. This rule represents a fuzzy relation between the input space $X_1 \times \cdots \times X_p$ and the output space Y of the FLS. We denote the membership function of this type-2 relation as $\mu_{\tilde{F}^l_1 \times \cdots \times \tilde{F}^l_p \to \tilde{G}^l}(\boldsymbol{x}, y)$ where $\tilde{F}^l_1 \times \cdots \times \tilde{F}^l_p$ denotes the Cartesian product of $\tilde{F}^l_1,\dots,\tilde{F}^l_p$ [9], and $\boldsymbol{x} = \mathrm{col}(x_1,\dots,x_p)$. When an input $\boldsymbol{x}'$ is applied, the composition of the fuzzy set $\tilde{X}'$, to which $\boldsymbol{x}'$ belongs and the rule R^l is found by using the *extended sup-star composition* that is derived in [13,15],

$$\mu_{\tilde{X}' \circ \tilde{F}^l_1 \times \cdots \times \tilde{F}^l_p \to \tilde{G}^l}(y) = \bigsqcup_{\boldsymbol{x} \in \tilde{X}'} [\mu_{\tilde{X}'}(\boldsymbol{x}) \sqcap \mu_{\tilde{F}^l_1 \times \cdots \times \tilde{F}^l_p \to \tilde{G}^l}(\boldsymbol{x}, y)]. \tag{17}$$

This extended sup-star composition uses the join and meet operations instead of the sup and t-norm operations that are used in the sup-star composition of type-1 FL (which helps explain why we spent time describing how to compute the join and meet operations).

When we use singleton fuzzification, the fuzzy set $\tilde{X}'$ is such that it has a membership grade of unity corresponding to $\boldsymbol{x} = \boldsymbol{x}'$ and zero membership grades for all other inputs (as a type-2 set, this means that its secondary membership function at $\boldsymbol{x} = \boldsymbol{x}'$ is 1/1); therefore, (17) reduces to

$$\mu_{\tilde{X}' \circ \tilde{F}_1^l \times \cdots \times \tilde{F}_p^l \to \tilde{G}^l}(y) = \mu_{\tilde{F}_1^l \times \cdots \times \tilde{F}_p^l \to \tilde{G}^l}(\boldsymbol{x}', y). \tag{18}$$

We denote $\tilde{X}' \circ \tilde{F}_1^l \times \cdots \times \tilde{F}_p^l \to \tilde{G}^l$ as $\tilde{B}^l$, the output set corresponding to the lth rule. The right-hand side of (18) is computed using an implication membership function (for an interesting discussion about *rule-based* versus *implication-based* membership functions, see [54]). Since we generally use product or minimum implication (corresponding to the *meet* operation with product or minimum t-norm in the type-2 case), (18) can be written as

$$\mu_{\tilde{B}^l}(y) = \mu_{\tilde{F}_1^l \times \cdots \times \tilde{F}_p^l}(\boldsymbol{x}') \sqcap \mu_{\tilde{G}^l}(y). \tag{19}$$

The membership function for the Cartesian product of type-2 fuzzy sets is given by the *meet* of the associated type-1 membership functions at each $\boldsymbol{x}'$ [13,15]; hence, (19) can be rewritten as

$$\begin{aligned}\mu_{\tilde{B}^l}(y) &= \mu_{\tilde{F}_1^l}(x_1) \sqcap \cdots \sqcap \mu_{\tilde{F}_p^l}(x_p) \sqcap \mu_{\tilde{G}^l}(y) \\ &= \mu_{\tilde{G}^l}(y) \sqcap \left[\prod_{i=1}^{p} \mu_{\tilde{F}_i^l}(x_i) \right]\end{aligned} \tag{20}$$

where we have used the fact that type-2 membership functions commute for minimum or product t-norms [13,15]. Note that the *degree of firing* corresponding to the lth rule is $\prod_{i=1}^{p} \mu_{\tilde{F}_i^l}(x_i)$.

As mentioned above, type-reduction methods in a type-2 FLS are extended versions of type-1 defuzzification methods. We have developed the following type reducers [13,14,19]: centroid, center-of-sums, height, modified height, and center-of sets. Here, by way of illustration, we only describe *center-of-sets type reduction*. We introduced this method to overcome some shortcomings of height type reduction. In the type-1 case, the two defuzzification methods are quite similar, and, when all consequent sets are convex, normal and symmetrical, they give the same answer. In the type-2 case, however, the two methods differ. For example, when only one rule is fired, the height type reducer (surprisingly) gives a single point value instead of an interval; so, it loses its ability to account for the rule's uncertainties. A center-of-sets type reducer, however, gives an interval; so, it is able to account for the rule's uncertainties (although it only accounts for the uncertainty in the consequent).

The center-of-sets type reducer requires the centroid of each rule consequent $C_{\tilde{G}^l}$ (e.g., see Example 3). Once all the consequent centroids are computed (which can be done ahead of time), the center-of-sets type-reduced set is computed by using [13,14,19]:

$$Y_{\cos}(\boldsymbol{x}') = \int_{c_1} \cdots \int_{c_M} \int_{e_1} \cdots \int_{e_M} T_{l=1}^{M}\, \mu_{C^l}(c_l) \bigstar T_{l=1}^{M}\, \mu_{E^l}(e_l) \Bigg/ \left[\sum_{l=1}^{M} c_l e_l \Bigg/ \sum_{l=1}^{M} e_l \right] \tag{21}$$

where T and $\bigstar$ indicate the chosen t-norm; $c_l \in C^l = C_{\tilde{G}^l}$; and, $e_l \in E^l = \prod_{i=1}^{p} \mu_{\tilde{F}_i^l}(x_i')$. A crisp output for the FLS is obtained by finding the centroid of $Y_{\cos}(\boldsymbol{x}')$. This is the defuzzifier step shown in Fig. 4.

For an interval type-2 FLS, i.e., when all the antecedents and consequents in a FLS are interval type-2 sets, (21) reduces to

$$Y_{\cos}(\boldsymbol{x}') = \int_{c_1} \cdots \int_{c_M} \int_{e_1} \cdots \int_{e_M} 1 \Bigg/ \left[\sum_{l=1}^{M} c_l e_l \Bigg/ \sum_{l=1}^{M} e_l \right] \tag{22}$$

where each c_l and e_l belongs to some interval in [0,1]. Eq. (22) can be computed using the computational procedure presented in [13,18]. The defuzzified ouput of an interval type-2 FLS is simply the average value of the left- and right-end points of its type-reduced set, i.e., $y = (y_{\mathrm{L}} + y_{\mathrm{R}})/2$.

Example 4 (Karnik and Mendel [13]). Here we present a function approximation example of a type-2 FLS which uses center-of-sets-type reduction. For simplicity, only the consequent sets are assumed to be type-2. The function to be approximated is $y = 100 - x^2$ for $x \in [-10,10]$. Given are 10 realizations, each with 9 (x, y) pairs. Each of these pairs include values of y corrupted by additive noise which is uniformly distributed in $[-10,10]$. For each applied input x^l ($l = 1,\ldots,9$) we find the minimum ($y_{\min}^l$) and the maximum ($y_{\max}^l$) of the 10 y values. For example, 3 of the 9 $(x^l,[y_{\min}^l, y_{\max}^l])$

pairs are:

$$(x^1, [y^1_{\min}, y^1_{\max}]) = (-10, [-7.79, 6.49]),$$

$$(x^5, [y^5_{\min}, y^5_{\max}]) = (0, [93.09, 109.95]),$$

$$(x^8, [y^8_{\min}, y^8_{\max}]) = (7.5, [34.14, 50.85]).$$

The FLS forms one rule from each pair. The rules are of the form IF x is A THEN y is $\tilde{B}$. Since only the y_i are uncertain in the given input–output pairs, we choose the antecedents as type-1 sets and the consequents as type-2 sets. In this example, the antecedent sets are chosen to be type-1 Gaussian and the type-2 consequent sets are described as type-1 Gaussian fuzzy sets with uncertain means (as in Fig. 2). For the lth rule, the mean of the consequent Gaussian is perturbed in the range $[y^l_{\min}, y^l_{\max}]$. The FLS uses singleton fuzzification, max t-conorm, product t-norm and product inference. The details of this example can be found in [13]. Results are depicted in Fig. 7, which compares the true function values with the crisp output of the type-2 FLS and also shows upper and lower bounds which are the results of type reduction. They give a measure of the uncertainty in the approximation caused by the noisy training values, something that cannot be obtained using just a type-1 FLS approximation.

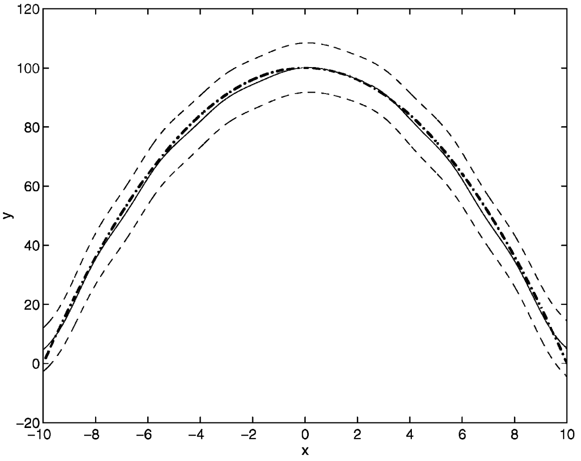

Fig. 7. The solid line shows the crisp output of the type-2 FLS and the dashed lines indicate the upper and lower bounds obtained from type reduction. The true function value is shown by the thick dash-dotted line.

One may argue that, in a type-1 FLS, the output set before defuzzification can be used in place of the type-reduced set, so we do not need type-2 FLSs; but to do so is incorrect. The output of a type-1 FLS just represents a combination of all the rule outputs of a single type-1 FLS, whereas the type-reduced set for a type-2 FLS represents a collection of outputs of a large number of type-1 FLSs, each having a level of uncertainty associated with it equal to the membership in the type-reduced set. Rule uncertainties flow through a type-2 FLS in a very natural manner, whereas they do not flow through a type-1 FLS at all.

Other applications of type-2 FLSs can be found in [16,17,19].

3.5. *Fuzzy basis functions in a type-2 FLS*

We have seen, in Sections 2.1 and 2.2, that the defuzzified output of a type-1 FLS can be described as a *fuzzy basis function expansion*. Unfortunately, this does not carry over to general type-2 FLSs. Karnik and Mendel [13] and Karnik et al. [19] explain that even though each term to the right of the slash in (21) can be interpreted as a valid FBF expansion, the defuzzifed value that is obtained as the center of gravity of this center of sets type-reduced set is not a FBF expansion. Interestingly enough, the terms to the right of the slash in the other type reducers that have been considered in [13,14] and [19] cannot even be interpreted as a valid FBF expansion. So, the concept of a FBF expansion does not seem to be of utility in general type-2 FLSs; however, *it is still useful for a type*-2 *interval-set FLS*, because, as we have stated above, each output of such a FLS can be represented as $y = (y_L + y_R)/2$, and y_L and y_R each can be represented as a fuzzy basis function expansion.

4. Examples

4.1. *Signal processing example*

In this example we demonstrate how available information about the training data can be incorporated into a type-2 FLS to obtain more information about its output than can be obtained using a type-1 FLS, and also a lower mean-squared error (MSE) than for a comparably designed type-1 FLS. We consider the following problem. A FLS is trained with noisy data; if we know the noise strength, how can we obtain bounds on the output of the FLS within which the true value of the output is likely to lie? We demonstrate this for a 3 input–1 output one-step predictor of the Mackey–Glass chaotic time series [29]. If $x(k)$ $(k = 1,2,\dots)$ is the time series, given $x(k-2), x(k-1)$ and $x(k)$, we predict $x(k+1)$. Prediction for this time series has been very widely studied in the neural network and fuzzy logic literatures (e.g., [24,37,38,4,45,51,10,11]); but, for all prior studies (to the best knowledge of this author) perfect training and testing data were used. The present study, in which only noisy measured values are available both for training and testing, is more in the spirit of the types of problems that signal processors deal with.

The Mackey–Glass chaotic time series is the solution to the following non-linear time-delay differential equation:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t). \tag{23}$$

It is known that when $\tau > 17$ this time series is chaotic. We chose $\tau = 30$. To simulate (23) we used Euler's method [44]. Letting

$$h(x,k) = \frac{0.2x(k-\tau)}{1+x^{10}(k-\tau)} - 0.1x(k),$$

we computed $x(k+1) = x(k) + gh(x,k)$ where g is a small number (we chose $g = 1$), and the initial values of $x(k)$, $k \leqslant \tau$ were set randomly.

To keep things really simple, we only used three fuzzy sets (F_1, F_2, and F_3) to describe $x(k)$; so, there are a total of $3^3 = 27$ three-antecedent rules. Each antecedent fuzzy set was initially characterized by a type-1 Gaussian membership function, with mean m and standard deviation σ, where initially: m_{F_2} = average value of the training data, $\sigma_{F_2} = \sigma$ (training data), $m_{F_1} = m_{F_2} - 2\sigma_{F_2}, m_{F_3} = m_{F_2} + 2\sigma_{F_2}$, and $\sigma_{F_1} = \sigma_{F_3} = 2\sigma_{F_2}$. The centers of the consequent fuzzy sets were initially chosen as random. Both our type-1 and type-2 FLSs used singleton fuzzification, product t-norm and product inference. Each of the type-1 27 rules has 7 parameters associated with it: the means and standard deviations of the three fuzzy sets and the center of the consequent fuzzy set. So, the entire type-1 rule base is described by 189 parameters.

Our *design approach* was to first design (i.e., optimize) a type-1 FLS using available training data. Then we created a type-2 FLS from this type-1 FLS by incorporating information that is available about the measurement noise. The design of the type-2 FLS did not change the optimized parameters of the type-1 FLS that also appeared in the type-2 FLS; it only chose the new parameters of the type-2 FLS. Because these new parameters accounted for the uncertainty of the measurement noise in the training data, if there is no noise, then these new parameters disappeared, and the type-2 FLS reduced to an optimized type-1 FLS. This is certainly not the only way to design a type-2 FLS (we are presently developing other design methods and will report on them in a future publication), but it is consistent with our basic design requirement that, if all uncertainty disappears, a type-2 FLS must reduce to a type-1 FLS. Both the type-1 and type-2 FLSs were designed using a single available training realization, because if we are given a different noisy realization of the same data set, we would normally choose a different set of parameters to obtain the best predictions.

Our training data consisted of the 500 data points $z(1), z(2), \dots, z(500)$, where $z(k) = x(k) + n(k)$ and $n(k)$ is uniformly distributed noise. We considered two SNRs – 0 and 10 dB. Our testing data consisted of the 500 data points $z(501), z(502), \dots, z(1{,}000)$.

4.1.1. Designing the type-1 FLS

We used center-of-sets defuzzification (which, in this case, is equivalent to height defuzzification because all the consequent fuzzy sets are symmetric about their mean values) and optimized the 189 parameters using a steepest descent (i.e., back propagation) algorithm to minimize the average sum of the squared errors between the output of the FLS and the training data. We do not provide any of the details for doing this here because they are very well known (e.g., [28]).

4.1.2. Designing the type-2 FLS

We used center-of-sets-type reduction and interval type-1 sets. Note that, if we had many different realizations of the training data and if we followed the same design procedure for the type-1 FLS, as just described, the means of the three sets would not remain the same from realization to realization. The means would vary in some range depending on the strength of the noise. Because the noise is uniform, $n(k) \in [-\delta, \delta]$ and $\delta = \sqrt{3}\sigma_{n(k)}$; so, we modeled each fuzzy set in the type-2 FLS as a type-2 set that is associated with a Gaussian type-1 set, where the latter has an uncertain mean, as in Fig. 2. For the type-2 antecedent membership functions, we centered the footprint of uncertainty about the optimized mean, as determined from the type-1 design, and arbitrarily chose the interval $[m_1, m_2]$ to equal $\sigma_{n(k)}$. We also assumed that the footprint of uncertainty for the consequent set was centered about its optimized value, as determined from the type-1 design, and also chose its uncertainty interval $[m_1, m_2]$ equal to $\sigma_{n_{(k)}}$. No attempt was made to optimize the footprints of uncertainty (i.e., optimize the values of m_1 and m_2) for the antecedents or consequents (although normally this would be done), because the goal of this simple example is merely to demonstrate that a type-2 FLS can outperform a type-1 FLS.

Results for SNR = 0 and 10 dB are depicted in Figs. 8 and 9, respectively. Because it is difficult to tell the difference between the type-1 output and the defuzzified output of the type-2 FLS, we summarize RMSEs for the two designs in Table 1. Observe that even without any optimization of the type-2 parameters, the RMSEs of the type-2 FLSs are always lower than those of the type-1 FLSs. Observe, also, from part (b) of the figures, that at low SNRs, the noise-free time series almost always lies within the upper and lower curves of the type-reduced set; but, that at higher values of SNR the noise-free time series does not always lie within those curves. We conjecture that these results can

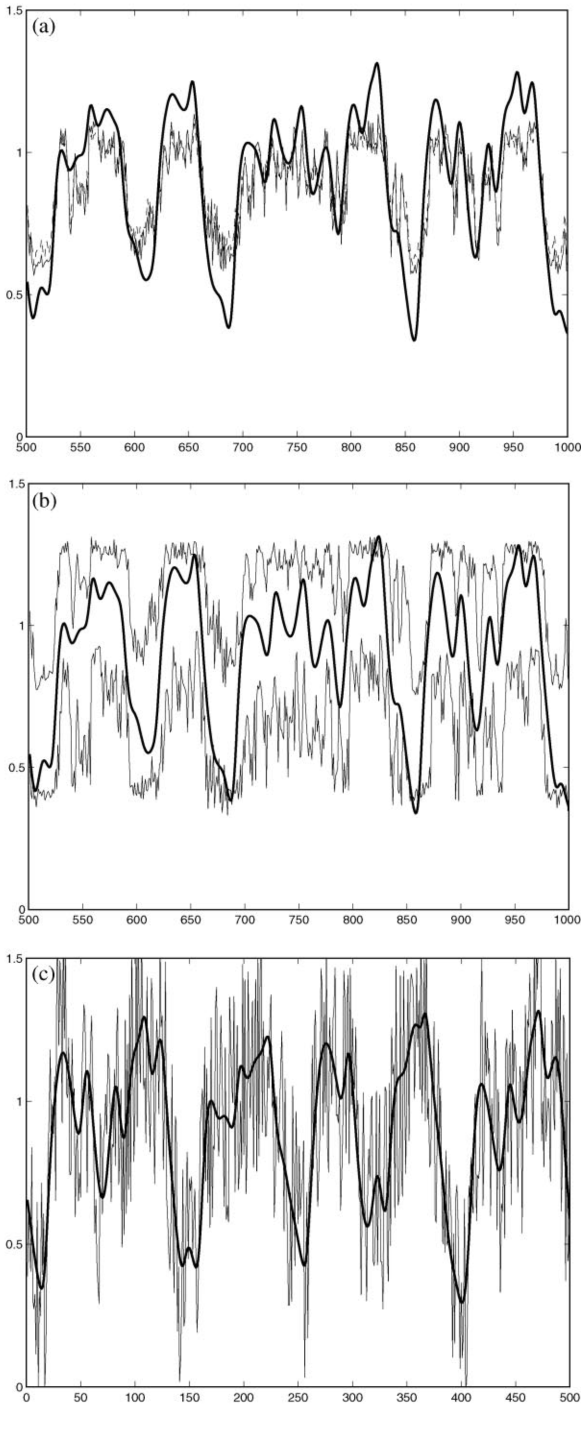

Fig. 8. Prediction of Mackey–Glass chaotic time-series in the presence of uniform noise, when SNR = 0 dB. The thick solid line in (a) and (b) indicates the true time series. In (a) the dash-dotted line indicates the type-1 output and the dashed line indicates the type-2 crisp output. Fig. (b) shows the upper and lower bands of the type-reduced set for the type-2 FLS. In (c), the heavy solid line shows the noisy data used for training and the thick solid line shows the noise-free data.

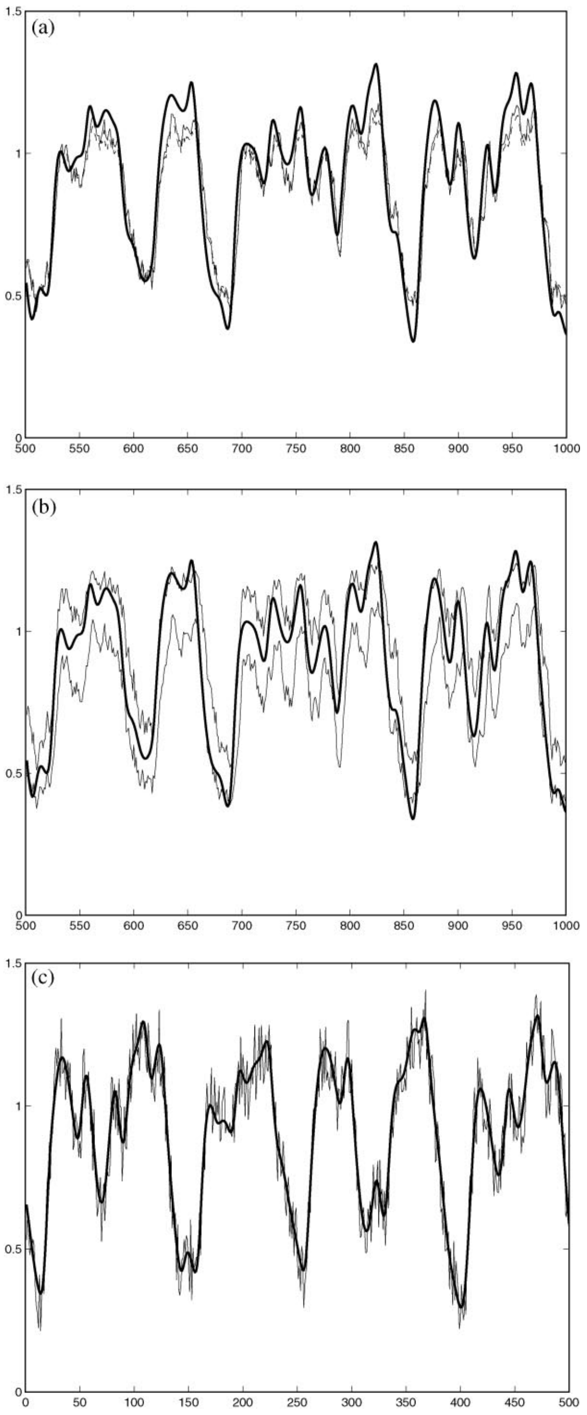

Fig. 9. As in Fig. 8, except for SNR = 10 dB.

be improved by optimizing the parameters of the type-2 FLSs to further minimize MSE.

It is important to again point out that the results that we have presented are for a single realization. A more complete study would repeat the type-1 and type-2 designs for a Monte-Carlo of say 50 realizations, and then provide a table like Table 1, but for the 50 realizations. We are presently doing this type of Monte Carlo simulation study for designs in which parameters of the type-1 and type-2 FLSs are all optimized, and will report on them in a future publication.

These results demonstrate that one can indeed obtain better forecasts using a type-2 FLS – a FLS that accounts for measurement noise uncertainties.

4.2. Digital communication example

Wang and Mendel [52] applied a type-1 FLS to equalization of a time-invariant non-linear channel, and demonstrated that the bit error rates (BER) of their fuzzy equalizer were close to that of the optimal equalizer. Sarwal and Srinath [46] observed that a transversal filter requires a much larger training set to achieve the same BERs as compared to a FL equalizer. Lee [25] proposed a complex fuzzy adaptive filter for QAM constellation channel equalization. Patra and Mulgrew [42] used a fuzzy adaptive filter to implement a Bayesian equalizer, and also used it to eliminate co-channel interference [43]. All of these fuzzy approaches to adaptive equalization focus on time-invariant channels. In today's communication world, such as in mobile communication, channels are time varying and non-linear. Observing this, we apply our type-2 FLS to such channels.

In a baseband communication system subject to inter-symbol interference (ISI) and additive Gaussian noise (AGN), the measured channel output,

Table 1
RMSEs for type-1 and type-2 FLS forecasters

SNR (dB)	Type-1 FLS	Type-2 FLS
0	0.1517	0.1429
10	0.0953	0.0838

$r(k)$, can be represented as:

$$r(k) = \hat{r}(k) + e(k) = \sum_{i=0}^{n} a_i(k)s(k-i) + e(k) \tag{24}$$

where $\hat{r}(k)$ is the noise-free signal, $e(k)$ is AGN, $s(k)$ is the symbol to be transmitted, $a_i(k)$ $(i = 0,1,\dots,n)$ are the time-varying channel coefficients (taps), and n is the order of the channel (there are $n+1$ taps). 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 $s(k)$ $(k = 1,2,\dots)$ based on a sequence of $r(k)$ values without knowing or estimating the channel coefficients.

In a FL-based equalizer (such as in [52]) the rule antecedents are $r(k), r(k-1), \dots, r(k-p+1)$, where p is the equalizer order (number of taps in the equalizer). $\hat{\boldsymbol{r}}(k) = [\hat{r}(k), \dots, \hat{r}(k-p+1)]^{\mathrm{T}}$ is called the *channel state* [1], and there are $n_s = 2^{n+p}$ channel states [1].

Patra and Mulgrew [42] developed a normalized Bayesian equalizer whose output is given by $\mathrm{sgn}[f(\boldsymbol{r}(k))]$, where $\boldsymbol{r}(k) = [r(k), \dots, r(k-p+1)]^{\mathrm{T}}$, and

$$f(\boldsymbol{r}(k)) = \frac{\sum_{i=1}^{n_s} \bar{y}^i \prod_{l=0}^{p-1} \exp\left[-\frac{1}{2}\left(\frac{r(k-l)-\hat{r}_i(k-l)}{\sigma_e}\right)^2\right]}{\sum_{i=1}^{n_s} \prod_{l=0}^{p-1} \exp\left[-\frac{1}{2}\left(\frac{r(k-l)-\hat{r}_i(k-l)}{\sigma_e}\right)^2\right]}. \tag{25}$$

In (25) $\bar{y}^i$ equals $+1$ or -1, σ_e is the standard deviation (std) of $e(k)$, and $\hat{r}_i(k-l)$ is the lth element $(l = 1,2,\dots,p)$ of the ith channel state $(i = 1,\dots,2^{n+p})$. They also showed that a type-1 FLS whose antecedent MFs are Gaussian, and uses product inference and height defuzzification can implement (25) perfectly.

Eq. (25) was derived in [42] for time-invariant channels. For time-varying channels, the channel coefficients, $a_i(k)$ $(i = 0,1,\dots,n)$, are uncertain, which means that each channel state element, $\hat{r}_i(k-l)$, is also uncertain. We represent this uncertainty in a type-2 FL equalizer using type-2 Gaussian antecedent MFs with uncertain means, as described below.

In the rest of this section we focus on the following channel model that was used in [21]:

$$\begin{aligned} r(k) = {} & a_1 s(k) + a_2 s(k-1) + [a_1 s(k) + a_2 s(k-1)]^2 \\ & + 0.7[a_1 s(k) + a_2 s(k-1)]^3 + 0.5[a_1 s(k) \\ & + a_2 s(k-1)]^4 + e(k) \end{aligned} \tag{26}$$

where nominal values for the channel's coefficients are $a_1 = 1$, and $a_2 = 0.7$, and we assume that we know the order of the channel, n, i.e., that it has 2 taps. We choose the number of equalizer taps, p, to be equal to $n+1$, i.e., $p = 2$; hence, there are $2^{n+p} = 8$ channel states. When a_1 and a_2 are constants, the channel states of (26) are 8 individual points on the $r(k) - r(k-1)$ plane.

We focus on the case when the channel is time varying, i.e., when a_1 and a_2 are time varying. The time variations of each of these coefficients was simulated, as in [3], by using a second-order Markov model in which a white noise source drove a second-order Butterworth low-pass filter. The amplitude of the random noise was controlled by a scalar parameter β; when $\beta = 0$ then each coefficient was a constant, whereas for $\beta > 0$ they were both time varying, and, the larger β was, the larger were the ranges of the time-varying coefficients. For $\beta > 0$ the channel states now are 8 smeared out clusters, because $\hat{\boldsymbol{r}}_i$ $(i = 1,\dots,8)$ are uncertain.

The 8 clusters establish 8 rules in a type-2 FLS, where the lth rule is:

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

In this rule: $\tilde{F}_1^l$ and $\tilde{F}_2^l$ are type-2 fuzzy sets characterized by Gaussian MFs with uncertain means, and G^l is a type-1 fuzzy set characterized by a Gaussian MF with mean $+1$ or -1 as determined by the channel state category (which is crisp, either $+1$ or -1). 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), i.e.,

$$Y_h(\boldsymbol{x}) = \int_{\theta_1}\int_{\theta_2}\cdots\int_{\theta_M} 1 \Big/ \frac{\sum_{l=1}^{M} \bar{y}^l \theta_l}{\sum_{l=1}^{M} \theta_l} \tag{27}$$

where $\bar{y}^l$ equals $+1$ or -1 (and is the center of consequent set G^l), and $\theta_1 \in D^l = \sqcap_{k=1}^{2} \mu_{\tilde{F}_k^l}(x_k)$

in which

$$\mu_{\tilde{F}_k^l}(x_k) = \exp\left[-\frac{1}{2}\left(\frac{x_k - m_k^l}{\sigma_e}\right)^2\right],$$
$$m_k^l \in [m_{k1}^l, m_{k2}^l],\ k = 1,2. \tag{28}$$

In order to complete the description of the type-2 FLS, we must specify numerical values for $[m_{k1}^l, m_{k2}^l]$ $(k = 1,2)$ and σ_e. We used a clustering method [1,8,27] applied to a training sample of noisy measurements to determine the former parameters. In [2] it is shown that equalizer performance is not very sensitive to the value used for σ_e. In our simulations we assumed that the value of σ_e was known.

We compared our type-2 FL equalizer with the Patra and Mulgrew's type-1 FL equalizer [42] and Savazzi, et al.'s K-nearest neighbor (NN) equalizer [47]. In our simulations, we used a training sequence of length 121, a testing sequence of length 879, and additive Gaussian noise. Here we show only a small part of our simulation results, those for a moderate amount of channel-coefficient time variability, for which $\beta = 0.15$. Simulations were run for 5 different SNRs ranging from 26 to 34dB (26:2:34). Note that we ran our simulations for high SNRs, because the channel in (26) is very nonlinear, and because the simulations in [21] were presented for SNR ranging from 22 to 40 dB. We ran 50 Monte Carlo simulations for each value of SNR.

In Fig. 10 we show the mean and std of bit error rate (BER) for the 50 MC realizations. Observe that (1) in terms of the mean values of BER, the type-2 FL equalizer performs better than either the NN or type-1 fuzzy equalizers; and, (2) in terms of the std of BER, the type-2 FL equalizer is more robust to the additive Gaussian noise than either the NN or type-1 fuzzy equalizers. More detailed results are given in [27], where it is also shown that the type-2 FL equalizer outperforms the NN and type-1 fuzzy equalizers over a wide range of β values.

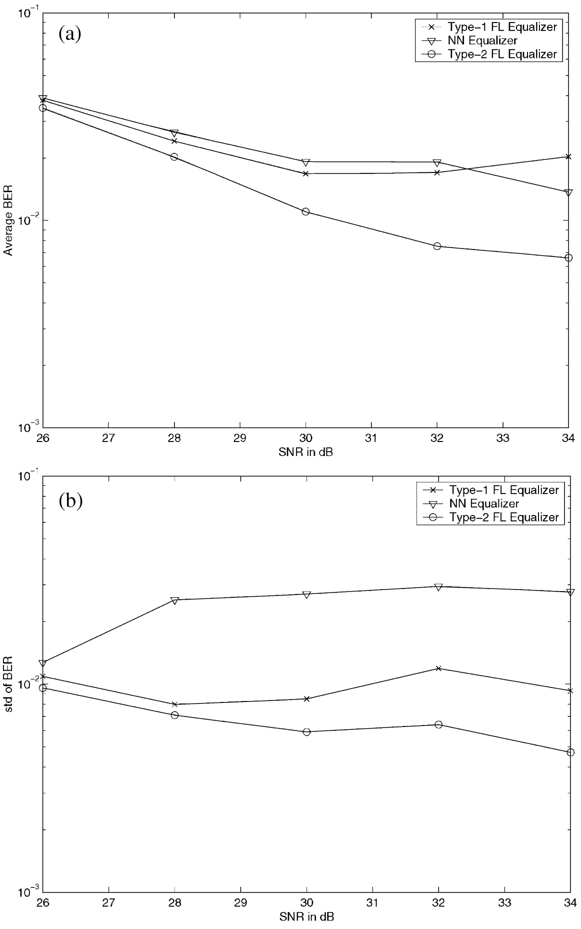

Fig. 10. Performance of type-1 FL equalizer, nearest neighbor equalizer, and type-2 FL equalizer, versus SNR when $\beta = 0.15$ and the number of training prototypes is 121. (a) average BER, and (b) std of BER for 50 Monte-Carlo realizations.

5. Conclusions

Fuzzy logic systems are comprised of rules, and quite often, the knowledge that is used to construct these rules is *uncertain*. Antecedent or consequent uncertainties translate into uncertain antecedent or consequent membership functions. Type-1 FLSs, whose membership functions are type-1 fuzzy sets, are unable to directly handle rule uncertainties. They can only handle measurement uncertainties. Type-2 FLSs, in which antecedent or consequent membership functions are type-2 fuzzy sets whose membership grades are themselves type-1 fuzzy sets, are very useful in circumstances where it is difficult to determine an exact membership function for a fuzzy set; hence, they are useful for

Table 2
Comparisons of type-1 and type-2 singleton and non-singleton FLSs

	Type-1 FLS	Type-2 FLS
Singleton fuzzification	1. No uncertainties about antecedents or consequents 2. No uncertainties on measurements that activate the FLS 3. Only a point output is obtained	1. Uncertainties about antecedents or consequents are accounted for (the footprint of uncertainty) 2. No uncertainties on measurements that activate the FLS 3. Both a type-reduced set and a point output are obtained
Non-singleton fuzzification	1. No uncertainties about antecedents or consequents 2. There are uncertainties on measurements that activate the FLS; and they are handled by treating the measurements as fuzzy numbers 3. Only a point output is obtained	1. Uncertainties about antecedents or consequents are accounted for (the footprint of uncertainty) 2. There are uncertainties on measurements that activate the FLS; and they are handled by treating the measurements as fuzzy numbers (type-1 or type-2) 3. Both a type-reduced set and a point output are obtained

incorporating rule uncertainties, and, they let us propagate such uncertainties through them, so that we can establish their effects at the output of the FLS.

Table 2 summarizes the similarities and differences between type-1 and type-2 singleton and non-singleton FLSs. Much work needs to be done on non-singleton type-2 FLSs, since they are the ones that are most appropriate for signal processing applications. We will report on such work in a future publication.

Zadeh, the father of fuzzy logic, lately has been advocating computing with words (CW) and using fuzzy logic to do this. Since *words mean different things to different people*, there is *uncertainty* associated with words, which means that FL must somehow use this uncertainty when it computes with words. Type 1 FL cannot do this, but type 2 FL can. So, we believe that type-2 FLSs will also have a significant impact on the growing movement to compute with words.

There are innumerable directions for extending and applying type-2 FLSs. We have, for example, recently developed type-2 TSK FLSs [26]. We are also applying type-2 FLSs to classification problems [53], forecasting of noisy time series, and equalization of non-linear time-varying channels [27]. We are also developing methods for the tuning and design of type-2 FLSs and for the design of non-singleton type-2 FLSs.

Now that measurement and rule uncertainties can be handled in the framework of FL, this author believes it is time for signal processors to become more interested in FLSs. They let us do nonlinear model-free robust signal processing. In order to expedite this, we have created a collection of type-2 FLS M-files (to be used in conjunction with MATLAB) that are on-line at: http://sipi.usc.edu/~mendel/software.

Acknowledgements

The author would like to thank his student Qilian Liang for generating the results in Section 4, and acknowledges the significant contributions of his former Ph. D. students, Li-Xin Wang, George Mouzouris and Nilesh Karnik to the development of many of the things that are discussed in this paper.

References

[1] S. Chen, B. Mulgrew, S. McLaughlin, A clustering technique for digital communications channel equalization using radial basis function network, IEEE Trans. Neural Networks 4 (July 1993) 570.

[2] S. Chen, S. McLaughlin, B. Mulgrew, P.M. Grant, Adaptive Bayesian decision feedback equalizer for dispersive mobile radio channels, IEEE Trans. Commun. (May 1995) 1937–1956.

[3] C.F.N. Cowan, S. Semnani, Time-variant equalization using a novel non-linear adaptive structure, Int. J. Adaptive Control Signal Process. 12 (2) (1998) 195–206.

[4] R.S. Crowder, Predicting the Mackey–Glass time series with cascade-correlation learning, in: D. Touretsky, G. Hinton, T. Sejnowski (Eds.), Proceedings of the 1990 Connectionist Models Summer School, Carnegie Mellon University, 1990, pp. 117–123.

[5] D. Dubois, H. Prade, Operations on fuzzy numbers, Int. J. Systems Sci. 9 (1978) 613–626.

[6] D. Dubois, H. Prade, Operations in a fuzzy-valued logic, Inform. Control 43 (1979) 224–240.

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

[8] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, Wiley, NewYork, USA, 1973.

[9] M.B. Gorzalczany, A method of inference in approximate reasoning based on interval-valued fuzzy sets, Fuzzy Sets and Systems 21 (1987) 1–17.

[10] J-S.R. Jang, ANFIS: Adaptive-network-based fuzzy inference system, IEEE Trans. Systems, Man Cybernetics 23 (May/June 1993) 665–684.

[11] J-S.R. Jang, C-T. Sun, Neuro-fuzzy modeling and control, IEEE Proc. 83 (March 1995) 378–406.

[12] N.N. Karnik, J.M. Mendel, Introduction to type-2 fuzzy logic systems, Presented at 1998 IEEE FUZZ Conference, Anchorage, AK, May 1998.

[13] N.N. Karnik, J. M. Mendel, Introduction to Type-2 Fuzzy Logic Systems, University of Southern California, Los Angeles, CA, June 1998; see http://sipi.usc.edu/~mendel/report.

[14] N.N. Karnik, J.M. Mendel, Type-2 fuzzy logic systems: type-reduction, Presented at IEEE Conference on Systems, Man and Cybernetics, La Jolla, CA, Oct. 11–14, 1998.

[15] N.N. Karnik, J.M. Mendel, Operations on type-2 fuzzy sets, 1999, submitted for publication.

[16] N.N. Karnik, J.M. Mendel, Applications of type-2 fuzzy logic systems: handling the uncertainty associated with surveys, presented at FUZZ-IEEE'99, Seoul, Korea, August 1999.

[17] N.N. Karnik, J.M. Mendel, Applications of type-2 fuzzy logic systems to forecasting of time-series, Inform. Sci. 120 (1999) 89–111.

[18] N.N. Karnik, J.M. Mendel, Centroid of a type-2 fuzzy set, 1999, submitted for publication.

[19] N.N. Karnik, J.M. Mendel, Q. Liang, Type-2 fuzzy logic systems, IEEE Trans. Fuzzy Systems 7 (1999) 643–658.

[20] A. Kaufman, M.M. Gupta, Introduction to Fuzzy Arithmetic: Theory and Applications, Van Nostrand Reinhold, New York, 1991.

[21] G. Kechriotis, E. Zervas, E. Manolakis, Using recurrent neural networks for adaptive communication channel equalization, IEEE Trans. Neural Networks 5 (May 1994) 2670–2678.

[22] G.J. Klir, B. Yuan, Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice-Hall, Upper Saddle River, NJ, 1995.

[23] B. Kosko, Fuzzy Engineering, Prentice-Hall, Englewood Cliffs, NJ, 1997.

[24] A.S. Lapedes, R. Farber, Nonlinear signal processing using neural networks: prediction and system modeling, Tech. Report LA-UR-87-2662, Los Alamos National Lab., Los Alamos, NM (1987).

[25] K.Y. Lee, Complex fuzzy adaptive filters with LMS algorithm, IEEE Trans. Signal Process. 44 (1996) 424–429.

[26] Q. Liang, J.M. Mendel, An introduction to type-2 TSK fuzzy logic systems, 1999, submitted for publication.

[27] Q. Liang, J.M. Mendel, Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters, 1999, submitted for publication.

[28] C-T. Lin, C.S.G. Lee, Neural Fuzzy Systems, Prentice-Hall PTR, Upper Saddle River, NJ, 1996.

[29] M.C. Mackey, L. Glass, Oscillation and chaos in physiological control systems, Sci. 197 (1977) 287–289.

[30] J.M. Mendel, Optimal Seismic Deconvolution: An Estimation Based Approach, Academic Press, New York, 1983.

[31] J.M. Mendel, Maximum-Likelihood Deconvolution: a Journey into Model-Based Signal Processing, Springer, Berlin, 1990.

[32] J.M. Mendel, Fuzzy logic systems for engineering: a tutorial, IEEE Proc. 83 (March 1995) 345–377.

[33] J.M. Mendel, Computing with words, when words can mean different things to different people, presented at Third International ICSC Symposium on Fuzzy Logic and Applications, Rochester University, Rochester, NY, June 1999.

[34] J.M. Mendel, G.C. Mouzouris, Designing fuzzy logic systems, IEEE Trans. Circuits Systems -II: Analog Digital Signal Process. 44 (November 1997) 885–895.

[35] M. Mizumoto, K. Tanaka, Some properties of fuzzy sets of type-2, Inform. and Control 31 (1976) 312–340.

[36] M. Mizumoto, K. Tanaka, Fuzzy sets of type-2 under algebraic product and algebraic sum, Fuzzy Sets and Systems 5 (1981) 277–290.

[37] J. Moody, Fast learning in multi-resolution hierarchies, in: Morgan Kaufman, D.S. Touretzky (Ed.), Advances in Neural Information Processing Systems I, San Mateo, CA, 1989, pp. 29–39 (Chapter 1).

[38] J. Moody, Fast learning in networks of locally-tuned processing units, Neural Comput. 1 (1989) 281–294.

[39] G.C. Mouzouris, J.M. Mendel, Non-singleton fuzzy logic systems, in: Proceedings of the IEEE World Congress on Computational Intelligence, Orlando, FL, 1994, pp. 456–461.

[40] G.C. Mouzouris, J.M. Mendel, Non-singleton fuzzy logic systems: theory and applications, IEEE Trans. Fuzzy Systems 5 (February 1997) 56–71.
[41] J. Nieminen, On the algebraic structure of fuzzy sets of type-2, Kybernetica 13 (4) (1977).
[42] S.K. Patra, B. Mulgrew, Efficient architecture for Bayesian equalization using fuzzy filters, IEEE Trans. Circuits Systems – II: Analog Digital Signal Process. 45 (July 1998) 812–820.
[43] S.K. Patra, B. Mulgrew, Fuzzy implementation of Bayesian equalizer in the presence of intersymbol and co-channel interference, IEE Proc. Commun. 145 (October 1998) 323–330.
[44] D. Quinney, An Introduction to the Numerical Solution of Differential Equations, Research Studies Press, England, 1985.
[45] T.D. Sanger, A tree-structured adaptive network for function approximation in high-dimensional spaces, IEEE Trans. Neural Networks 2 (March 1991) 285–293.
[46] P. Sarwal, M.D. Srinath, A fuzzy logic system for channel equalization, IEEE Trans. Fuzzy Systems 3 (2) (May 1995) 246–249.
[47] P. Savazzi, L. Favalli, E, Costamagna, A. Mecocci, A suboptimal approach to channel equalization based on the nearest neighbor rule, IEEE J. Selected Areas Commun. 16 (9) (December 1998) 1640–1648.
[48] I. Turksen, Interval valued fuzzy sets based on normal forms, Fuzzy Sets and Systems 20 (1986) 191–210.
[49] M. Wagenknecht, K. Hartmann, Application of fuzzy sets of type-2 to the solution of fuzzy equation systems, Fuzzy Sets and Systems 25 (1988) 183–190.
[50] L-X. Wang, J.M. Mendel, Adaptive minimum prediction-error deconvolution and source wavelet estimation using Hopfield neural networks, Geophysics 57 (May 1992) 670–679.
[51] L-X. Wang, J.M. Mendel, Generating fuzzy rules by learning from examples, IEEE Trans. Systems, Man Cybernetics 22 (6) (November/December 1992) 1414–1427.
[52] L.-X. Wang, J.M. Mendel, Fuzzy adaptive filters, with application to nonlinear channel equalization, IEEE Trans. Fuzzy Systems 1 (3) (August 1993) 161–170.
[53] W. Wei, J.M. Mendel, A fuzzy logic method for modulation classification in non-ideal environments, IEEE Trans. on Fuzzy Systems 7 (1999) 333–344.
[54] J. Yen, R. Langari, Fuzzy Logic: Intelligence, Control, and Information, Prentice-Hall, Upper Saddle River, NJ, 1999.
[55] L.A. Zadeh, Outline of a new approach to the analysis of complex systems and decision processes, IEEE Trans. Systems, Man, and Cybernetics SMC-3 (1) (1973) 28–44.
[56] L.A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning – 1, Inform. Sci. 8 (1975) 199–249.