General Type-2 Fuzzy Logic Systems Made Simple: A Tutorial

Jerry M. Mendel, *Life Fellow, IEEE*

***Abstract*—The purpose of this tutorial paper is to make general type-2 fuzzy logic systems (GT2 FLSs) more accessible to fuzzy logic researchers and practitioners, and to expedite their research, designs, and use. To accomplish this, the paper 1) explains four different mathematical representations for general type-2 fuzzy sets (GT2 FSs); 2) demonstrates that for the optimal design of a GT2 FLS, one should use the vertical-slice representation of its GT2 FSs because it is the only one of the four mathematical representations that is parsimonious; 3) shows how to obtain set theoretic and other operations for GT2 FSs using type-1 (T1) FS mathematics ($\alpha - cuts$ play a central role); 4) reviews Mamdani and TSK interval type-2 (IT2) FLSs so that their mathematical operations can be easily used in a GT2 FLS; 5) provides all of the formulas that describe both Mamdani and TSK GT2 FLSs; 6) explains why center-of sets type-reduction should be favored for a GT2 FLS over centroid type-reduction; 7) provides three simplified GT2 FLSs (two are for Mamdani GT2 FLSs and one is for a TSK GT2 FLS), all of which bypass type reduction and are generalizations from their IT2 FLS counterparts to GT2 FLSs; 8) explains why gradient-based optimization should not be used to optimally design a GT2 FLS; 9) explains how derivative-free optimization algorithms can be used to optimally design a GT2 FLS; and 10) provides a three-step approach for optimally designing FLSs in a progressive manner, from T1 to IT2 to GT2, each of which uses a quantum particle swarm optimization algorithm, by virtue of which the performance for the IT2 FLS cannot be worse than that of the T1 FLS, and the performance for the GT2 FLS cannot be worse than that of the IT2 FLS.**

***Index Terms*—Alpha-plane, general type-2 fuzzy logic system, horizontal slice, interval type-2 fuzzy logic system, vertical slice, zSlice.**

TABLE I
ABBREVIATIONS AND THEIR DEFINITIONS

Abbreviation	Definition
A2-C0	Antecedents are type-2 fuzzy sets and consequents are type-0 fuzzy sets (crisp)
COG	Center of gravity
COS	Center of sets (type-reduction)
EIASC	Enhanced iterative algorithm + stopping condition (algorithms)
EKM	Enhanced KM (algorithms)
FLS	Fuzzy logic system
FOU	Footprint of uncertainty
FS	Fuzzy set
FWA	Fuzzy weighted average
GA	Genetic algorithm
GT2	General type-2
IT2	Interval type-2
IV	Interval valued (fuzzy set)
IWA	Interval weighted average
KM	Karnik-Mendel (algorithms)
LMF	Lower membership function
LWA	Linguistic weighed average
MF	Membership function
NT	Nie-Tan (simplification)
QPSO	Quantum particle swarm optimization
TR	Type-reduction
TSK	Takagi-Sugeno-Kang
T1	Type-1
T2	Type-2
UMF	Upper membership function
WM UBs	Wu-Mendel uncertainty bounds

I. INTRODUCTION

DURING the past 15 years great progress has been made in transitioning from type-1 (T1[1]) fuzzy logic systems (FLSs) to interval-type-2 (IT2) FLSs and, most recently, to general T2 FLSs (GT2 FLSs) (e.g., [1]–[23], [27]–[58], [60]–[64], [67]–[70], [73]–[79]). Naturally, an IT2 FLS uses interval type-2 fuzzy sets (IT2 FSs), whereas a GT2 FLS uses general type-2 fuzzy sets[2] (GT2 FSs). Using IT2 FSs in an FLS has the potential to provide better (and certainly no worse[3]) performance for a FLS than using T1 FSs, and using GT2 FSs in an FLS has the potential to provide better (and certainly no worse) performance for a FLS than using IT2 FSs, which is why there has been so much interest in IT2 FLSs and GT2 FLSs.

Recall (e.g., [24] and [35]) that an IT2 FS can be thought of as a blurred T1 FS that lets one account for membership function (MF) uncertainties; however, an IT2 FS weights all such uncertainties uniformly and can therefore be thought of only as a first-order fuzzy set uncertainty model. On the other hand, a GT2 FS also lets us account for MF uncertainties, but it

Manuscript received May 11, 2013; revised July 22, 2013; accepted September 9, 2013. Date of publication November 6, 2013; date of current version October 2, 2014.

J. M. Mendel is with the Signal and Image Processing Institute, University of Southern California, Los Angles, CA 90089-2564 USA (e-mail: mendel@sipi.usc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2013.2286414

[1]Because many abbreviations are used in this paper, they are given in Table I for the convenience of the reader.

[2]In the early days of type-2 fuzzy sets and systems, the phrases "type-2 fuzzy set" or "type-2 fuzzy system" were used in an all-inclusive way, meaning any kind of type-2 fuzzy set or system. During the past 15 years, most of the attention has been given to interval type-2 fuzzy sets and systems. It is only within the past five years or so that there has been a return to more general type-2 fuzzy sets and systems, and to distinguish them from the more specialized interval type-2 fuzzy sets and systems, the term "general" is being used. In essence, type-2 fuzzy sets and systems now consist of the union of interval and general type-2 fuzzy sets and systems.

[3]Some authors maintain that an IT2 FLS may not perform as well as a T1 FLS. We will show in this paper that by optimizing an IT2 FLS in a certain way, this can never happen.

weights all such uncertainties nonuniformly and can therefore be thought of as a second-order fuzzy set uncertainty model.

Both IT2 and GT2 FSs are parametric models. An IT2 FS is described by more parameters than is a T1 FS, and a GT2 FS is described by even more parameters. Each FS parameter in an FLS can be thought of as one of its design degrees of freedom; hence, an IT2 FLS has more design degrees of freedom than a T1 FLS, and a GT2 FLS has even more design degrees of freedom. It is the increase in the number of design degrees of freedom that offers the possibility of an IT2 FLS or a GT2 FLS outperforming a T1 FLS.[4]

It is very important to understand that there is a natural progression from T1 to IT2 to GT2 FSs and FLSs. In fact, all the computations that are needed for an IT2 FLS can be derived and performed using T1 FS mathematics, as was demonstrated in [46]; this means that someone who is already familiar with T1 FSs and FLSs does not have to invest a lot of time to learn about IT2 FSs and FLSs. In this paper, we will show and explain how all computations that are needed for a GT2 FLS can be derived and performed again using only T1 FS (or IT2 FS) mathematics; this means that someone who is already familiar with IT2 FSs and FLSs does not have to invest a lot of time to learn about GT2 FSs and FLSs.

We are not content though only to explain the connections between GT2 FSs and FLSs to their IT2 and T1 counterparts. In order for practitioners to use a GT2 FLS, there are some major design issues that need to be exposed and clarified, or else, there will still be stumbling blocks to designing and using a GT2 FLS. Therefore, we shall also focus on the following design issues: 1) representation of a GT2 FS for optimal designs; 2) extending some already well-known IT2 FLS architectures to GT2 FLSs; 3) avoiding computing derivatives; and 4) one method for optimizing a GT2 FLS that is guaranteed to provide system performance that is at least as good as a T1 FLS or an IT2 FLS.

The *purpose of this tutorial paper* is to make GT2 FLSs more accessible to FL researchers and practitioners to expedite the research about and use of general type-2 fuzzy sets (GT2 FSs) and FLSs.

The rest of this tutorial paper is organized as follows. Section II provides background about T2 FSs. Section III describes the geometry of GT2 FSs and three important representations for them. Section IV examines which representation of a GT2 FS is useful for optimal design applications. Section V derives the operations that are needed by a GT2 FLS. Section VI provides a brief review of IT2 FLSs (Mamdani and TSK). Section VII is about GT2 FLSs (Mamdani and TSK). Section VIII proposes three simplified architectures for a GT2 FLS. Section IX discusses some important issues about optimizing a GT2 FLS, and Section X draws conclusions.

[4]Some authors feel that in order to fairly compare an IT2 FLS with a T1 FLS, both must have the same number of design degrees of freedom. This author does not subscribe to that notion because this can only be accomplished by using fewer membership functions for the IT2 FLS, (or more for the T1 FLS), which is not equitable from a design perspective.

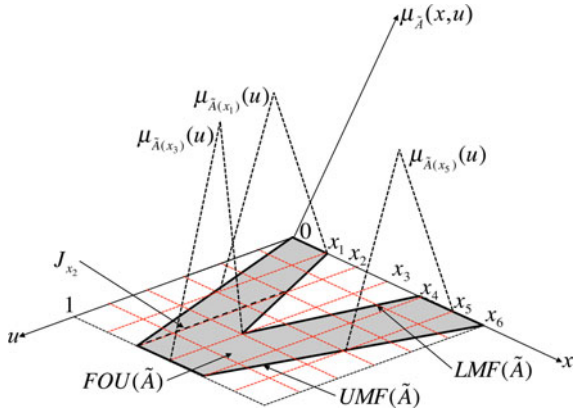

Fig. 1. Various elements of a GT2 FS.

II. Type-2 Fuzzy Sets

We assume that readers of this paper are already familiar with T1 FSs, including $\alpha-$ cuts, their associated MF decomposition theorem [26], and the function decomposition theorem [26]. The latter can be stated in words, as: *The MF for a function of T1 FSs equals the supremum (over all values of α)*[5] *of the MFs of the same function applied to the $\alpha-$ cuts of the FSs.*

To begin we define a GT2 FS and an IT2 FS[6,7]. The MF of a GT2 FS is 3-D[8] (see Fig. 1), with the *x-axis* called [35], [44] the *primary variable*, the *y-axis* called the *secondary variable* that is denoted u, and the *z-axis* called the *MF value* (or *secondary MF value* or *secondary grade*) that is denoted $\mu_{\tilde{A}}(x,u)$.

A T2 FS $\tilde{A}$ is [1] a bivariate function on the Cartesian product, $\mu_{\tilde{A}}: X\times[0,1]$ into $[0,1]$, where X is the universe for the primary variable, x, of $\tilde{A}$, i.e.,

$$\tilde{A}=\{(x,u),\mu_{\tilde{A}}(x,u)\,|\forall x\in X,\forall u\in[0,1]\} \tag{1}$$

which is often referred to as the *point-valued representation* of a T2 FS and is useful as a starting point for obtaining the other representations of a GT2 FS (described in Section III). $\mu_{\tilde{A}}(x,u)$ is also designated as $f_x(u)$.

The 2-D support of $\mu_{\tilde{A}}(x,u)$ is called the *footprint of uncertainty* (FOU) of $\tilde{A}$, i.e., [1]:

$$\text{FOU}(\tilde{A})=\{(x,u)\in X\times[0,1]|\mu_{\tilde{A}}(x,u)>0\} \tag{2}$$

$\text{FOU}(\tilde{A})$ is bounded by lower and upper MFs that are T1 FSs, denoted $\underline{\mu}_{\tilde{A}}(x)$ and $\bar{\mu}_{\tilde{A}}(x)$, [or $\text{LMF}(\tilde{A})$ and $\text{UMF}(\tilde{A})$], respectively (see Fig. 1), where [1]:

$$\text{LMF}(\tilde{A})=\underline{\mu}_{\tilde{A}}(x)=\inf\{u|u\in[0,1],\mu_{\tilde{A}}(x,u)>0\} \tag{3}$$

[5]This is also known as the *fuzzy union*.

[6]Different notations can be used to do this. Table I in [1] delineates the "fuzzy set notation" (which has been used in most articles) and the "standard mathematical notation" (which is more precise). In this paper, a mixture of the two notations is used.

[7]Some of the material in this section is taken from [40].

[8]Three-dimensional figures such as the ones in this paper are not so easy to draw, especially during face-to-face technical discussions about GT2 FSs. See [42] for a way to draw 2-1/2 D figures for GT2 FSs that are relatively easy to draw.

$$\text{UMF}(\tilde{A}) = \bar{\mu}_{\tilde{A}}(x) = \sup\{u | u \in [0,1], \mu_{\tilde{A}}(x,u) > 0\}. \quad (4)$$

The *primary membership* (or codomain) at each $x \in X$ of $\tilde{A}$ is denoted J_x and (see Fig. 1 when $x = x_2$) is the interval $[\underline{\mu}_{\tilde{A}}(x), \bar{\mu}_{\tilde{A}}(x)]$, i.e., [1]:

$$J_x = \{u \in [0,1] | \mu_{\tilde{A}}(x,u) > 0\} = [\underline{\mu}_{\tilde{A}}(x), \bar{\mu}_{\tilde{A}}(x)]. \quad (5)$$

The *secondary MF* of $\tilde{A}$ is usually denoted $\mu_{\tilde{A}}(x)$ and is [1] the restriction of function $\mu_{\tilde{A}} : X \times [0,1] \to [0,1]$ to $x \in X$; it is also called a *vertical slice* of $\mu_{\tilde{A}}(x,u)$. Note that the secondary MF is itself a T1 FS,[9] $\tilde{A}(x)$, that is a function of the secondary variable u; hence, in order to make the dependence of $\mu_{\tilde{A}}(x)$ on u explicit, we shall use the following (new) notation for the secondary MF, $\mu_{\tilde{A}}(x) \to \mu_{\tilde{A}(x)}(u)$ (see Fig. 1 for three triangle secondary MFs). This notation will be very helpful when $\alpha - \text{cuts}$ of the secondary MF are taken, as is done later in this paper.

An *embedded T2 FS* of $\tilde{A}$, $\tilde{A}_e$, is described by the MF $\mu_{\tilde{A}_e} : X \to [0,1]$, where $\mu_{\tilde{A}_e} = \mu_{\tilde{A}}(x, u(x)), u(x) \in J_x$; $\tilde{A}_e$ is also commonly expressed as [35]

$$\tilde{A}_e = \int_{x \in X} [f_x(u)/u]/x, \quad u \in J_x. \quad (6)$$

Note that $\tilde{A}_e$ is a GT2 FS with only one primary membership u' at each x', and only one secondary membership $f_{x'}(u')$ at that u'.

An *embedded T1 FS* of $\tilde{A}$, A_e, is described by the MF $\mu_{A_e} : X \to [0,1]$, where $\mu_{A_e} \in J_x$ [for which $\mu(x, u(x)) > 0, \forall x \in X$]; A_e is also commonly expressed as [35]

$$A_e = \int_{x \in X} u/x, \quad u \in J_x. \quad (7)$$

Note that the embedded T1 FS A_e that corresponds to an embedded GT2 FS $\tilde{A}_e$ contains the primary memberships of that $\tilde{A}_e$, and acts as the support of $\tilde{A}_e$. When both the primary and secondary variables are discretized, as is done during computations involving T2 FSs, there will be n_A embedded T1 FSs that are contained within FOU($\tilde{A}$). Note, finally, that A_e does not have to be either a normal or a convex T1 FS.

An *interval T2 FS* (IT2 FS), also known as an *interval-valued fuzzy set* (IVFS) [4], is a T2 FS all of whose secondary grades equal 1. It is [1] a function on X into $D[0,1]$, where $D[0,1]$ is the set of closed subintervals of $[0,1]$, i.e., $\mu_{\tilde{A}} : X \to D[0,1]$. Because the secondary grades are all the same, they convey no useful information for the IT2 FS; hence, an IT2 FS is completely described by its FOU, and consequently by its LMF and UMF.

An example of the FOU of an IT2 FS is depicted in Fig. 2. Also, shown on this figure are the lower and upper MFs for such an FOU, the primary membership at $x = x'$, and an example of an embedded T1 FS.

[9]It may be confusing to see the tilde symbol over a T1 FS; however, this is the commonly used notation for the secondary T1 FS, and so to preserve the connections between this article and all those that have preceded it, this notation will not be changed.

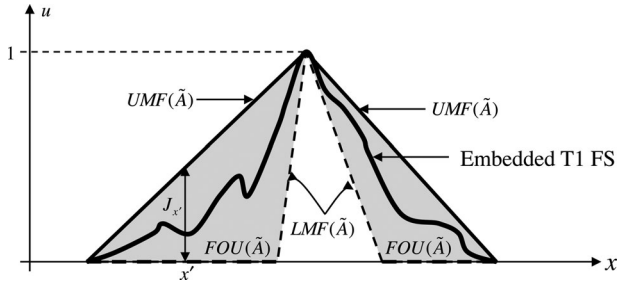

Fig. 2. IT2 FS and related quantities ([38], © 2007, IEEE).

III. Geometry of GT2 Fuzzy Sets and Different Representations for Them

As mentioned at the start of Section II, the MF of a GT2 FS is three-dimensional (3-D). Just as it is possible to cut the 2-D MF of a T1 FS using horizontal slices ($\alpha - \text{cuts}$), and to then represent that MF as the fuzzy union of such horizontal slices (each raised to level α), it is possible to decompose the 3-D MF of a GT2 FS by using different kinds of cuts. The most commonly used cuts are vertical slices, horizontal slices, and wavy slices. We depict examples of these three kinds of cuts in Figs. 3–5, respectively.

Just as it is possible to rigorously prove that a 2-D T1 MF can be represented as the fuzzy union of all of its $\alpha - \text{cuts}$ (each raised to level α), it is also possible to rigorously prove that a 3-D MF can be represented as the union of all of its vertical slices, or horizontal slices or wavy slices. In retrospect, all of these decompositions are visually obvious and (in the opinion of this author) no longer need rigorous proofs.

A. Vertical Slice Representation of a GT2 FS

The *vertical-slice representation* (see Fig. 3; see, also, Fig. 1 for some vertical slices) [35] of $\tilde{A}$ focuses on each value of the primary variable x, and expresses (1) as the union of all of its secondary T1 FSs:

$$\tilde{A} = \int_{x \in X} \tilde{A}(x)/x \quad (8)$$

where

$$\tilde{A}(x) = \int_{\forall u \in [0,1]} \mu_{\tilde{A}(x)}(u)/u \quad (9)$$

The vertical slice representation is extremely useful for computation and can also be used for theoretical studies. In fact, as shall be demonstrated in Section IV, it is the preferred representation for solving real-world GT2 FLS optimal design problems.Let $\tilde{A}_\alpha(x)$ denote the $\alpha - \text{cut}$ of the T1 FS $\tilde{A}(x)$, i.e., [26] ($\alpha \in [0,1]$)

$$\tilde{A}_\alpha(x) = \{u | \mu_{\tilde{A}(x)}(u) \geq \alpha\} = [a_\alpha(x), b_\alpha(x)]. \quad (10)$$

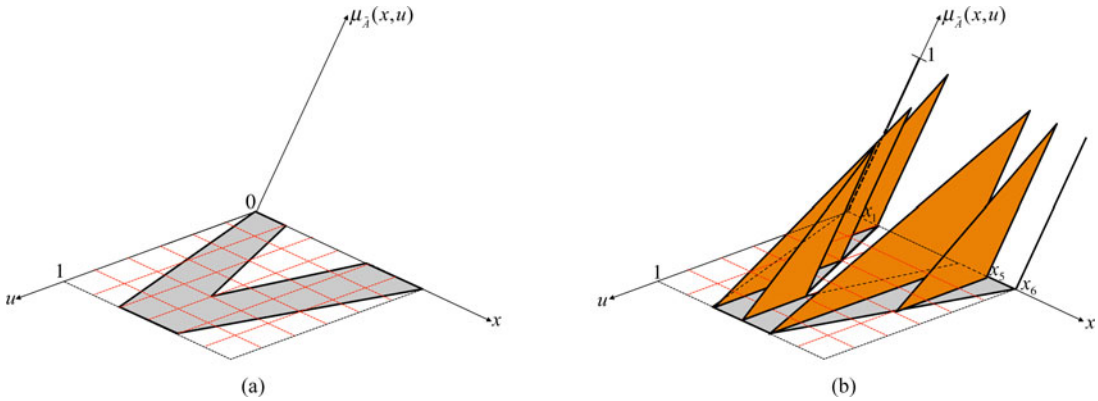

Fig. 3. (a) FOU($\tilde{A}$) and (b) vertical-slice representation of a GT2 FS. Shown are five vertical slices at $x = x_1, \ldots, x_5$ that sit on the FOU. Note that the two vertical lines at $x = 0$ and $x = x_6$ are also vertical slices. The filled-in (normal) triangles are only for artistic purposes.

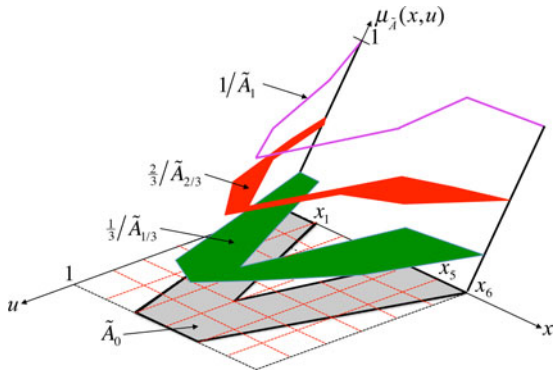

Fig. 4. Horizontal-slice representation of the Fig. 3 GT2 FS. Shown are $\tilde{A}_0$ (the FOU of $\tilde{A}$), $\frac{1}{3}/\tilde{A}_{1/3}$, $\frac{2}{3}/\tilde{A}_{2/3}$ and $1/\tilde{A}_1$. Because the secondary MFs are all normal T1 FSs, $1/\tilde{A}_1$ is a T1 FS.

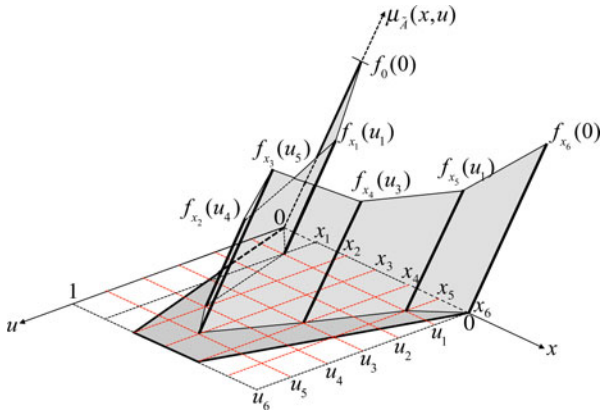

Fig. 5. Wavy-slice representation of the Fig. 3 GT2 FS. Shown is one wavy slice. The shading of the wavy slice is only for artistic purposes; however, for continuous X and U, the shaded regions become a continuous foil.

At each value of $x \in X$, the T1 FS $\tilde{A}(x)$ has the following $\alpha-$cut decomposition [26][10]:

$$\tilde{A}(x) = \bigcup_{\forall \alpha \in [0,1]} \left[\alpha/\tilde{A}_\alpha(x)\right] = \sup_{\forall \alpha \in [0,1]} \left[\alpha/\tilde{A}_\alpha(x)\right] = \sup_{\forall \alpha \in [0,1]} [\alpha/[a_\alpha(x), b_\alpha(x)]] \tag{11}$$

Note that, in (11), $\alpha/\tilde{A}_\alpha(x)$ is an *indicator function* whose amplitude is α for all $u \in [a_\alpha(x), b_\alpha(x)]$ and is zero for all other values of $u \in [0,1]$.

Substituting the first part of (11) into (8), we find that

$$\tilde{A} = \int_{\forall x \in X} \left[\bigcup_{\forall \alpha \in [0,1]} \left[\alpha/\tilde{A}_\alpha(x)\right] \right] \Big/ x. \tag{12}$$

This vertical-slice representation of $\tilde{A}$ (i.e., at each x there is a T1 secondary FS that is described by its $\alpha-$cut decomposition) strongly suggests that $\alpha-$cuts will also play a useful and central role for GT2 FSs.

[10]In (11) $\cup$ denotes the standard fuzzy union.

B. Horizontal Slice Representation of a GT2 FS

It is a simple matter to reexpress (12) as

$$\tilde{A} = \bigcup_{\forall \alpha \in [0,1]} \left[\int_{\forall x \in X} [\alpha/\tilde{A}_\alpha(x)]/x \right] = \bigcup_{\forall \alpha \in [0,1]} \left[\alpha \Big/ \left[\int_{\forall x \in X} \tilde{A}_\alpha(x)/x \right] \right]. \tag{13}$$

Note that $\alpha/[\int_{\forall x \in X} \tilde{A}_\alpha(x)/x] = \alpha/[\int_{\forall x \in X} [a_\alpha(x), b_\alpha(x)]/x]$ is a plane at level α. Each of the planes is obtained by connecting respective $\alpha-$cuts of all T1 vertical-slice secondary MFs for $\forall x \in X$.

More formally, an $\alpha-$plane [31], [48] for the GT2 FS $\tilde{A}$, denoted $\tilde{A}_\alpha$, is the union of all primary memberships of $\tilde{A}$ whose secondary grades are greater than or equal to $\alpha (0 \le \alpha \le 1)$, i.e.,

$$\tilde{A}_\alpha = \{(x,u), \mu_{\tilde{A}}(x,u) \ge \alpha | \forall x \in X, \forall u \in [0,1]\} = \int_{\forall x \in X} \int_{\forall u \in [0,1]} \{(x,u) | f_x(u) \ge \alpha\}. \tag{14}$$

TABLE II
COMPARISONS OF α- PLANE AND zSLICE DESCRIPTIONS

Item	α - plane description	zSlice description
Coordinates	(x,u,μ)	(x,y,z)
$\alpha-$plane	$\tilde{A}_\alpha = \int_{\forall x\in X}\int_{\forall u\in[0,1]} \{(x,u)\big\vert f_x(u)\geq\alpha\}$	$\tilde{Z}_\alpha$ projected onto $X\times Y$
zSlice	$\alpha / \tilde{A}_\alpha$	$\tilde{Z}_z = \int_{\forall x\in X}\int_{\forall y\in[0,1]} z/(x,y)$
$FOU(\tilde{A})$	$\tilde{A}_0$	$\tilde{Z}_0$
Vertical slice $(x_j\in X)$	$\int_{\forall\alpha\in[0,1]}\int_{\forall u\in[0,1]} \alpha/(x_j,u)$	$\int_{\forall z\in[0,1]}\int_{\forall y\in[0,1]} z/(x_j,y)$
Representation of $\tilde{A}$	$\tilde{A}=\bigcup_{\alpha\in[0,1]} R_{\tilde{A}_\alpha} = \sup_{\alpha\in[0,1]}\left[\alpha/\tilde{A}_\alpha\right]$	$\tilde{A}=\bigcup_{z\in[0,1]} \tilde{Z}_z = \sup_{z\in[0,1]}\left[\tilde{Z}_z\right]$

Alternatively, $\tilde{A}_\alpha$ can be expressed directly from the rightmost part of (13) and (10), as

$$\tilde{A}_\alpha = \int_{\forall x\in X} A_\alpha(x)/x = \int_{\forall x\in X}[a_\alpha(x), b_\alpha(x)]/x. \quad (15)$$

Therefore, an $\alpha-\text{plane}$ can be obtained directly from the $\alpha-\text{cuts}$ of the secondary MFs. Just as an $\alpha-\text{cut}$ of a T1 FS resides on its 1D domain X, $\tilde{A}_\alpha$ resides on its 2-D domain $X\times U$. Note, also, $\text{FOU}(\tilde{A})=\tilde{A}_0$. Recall that by raising an $\alpha-\text{cut}$ to level α one obtains an $\alpha-\text{Level}$. Analogously, by raising an $\alpha-\text{plane}$ to level α one obtains a *horizontal slice at level* α, $R_{\tilde{A}_\alpha}$ [39], i.e.,

$$R_{\tilde{A}_\alpha} = \alpha/\tilde{A}_\alpha. \quad (16)$$

This has been called an "$\alpha-\text{plane}$ raised to Level α" or a[11] "zSlice" [62]–[64]. Of course, one can also shorten the rather tedious expression "$\alpha-\text{plane}$ raised to Level α" to the much simpler expression "$\alpha-\text{Level plane}$." Note, also, $R_{\tilde{A}_\alpha}$ is an IT2 FS all of whose secondary MFs equal α (rather than 1 as would be the case for the usual IT2 FS).

Comment: The term $\alpha-plane$ was introduced first by Liu in 2008 [31], and the term *zSlice* was introduced first by Wagner and Hagras also in 2008 [62]. The connection between these two representations is summarized in Table II.

It is a bit confusing to have two different terminologies for the same things; therefore, in this paper I muddy the waters further by calling this kind of a representation a *horizontal slice representation* since "horizontal" compliments "vertical" rather nicely. The *horizontal slice representation* for a GT2 FS $\tilde{A}$ is (see Fig. 4):

$$\tilde{A} = \bigcup_{\alpha\in[0,1]} R_{\tilde{A}_\alpha} = \sup_{\alpha\in[0,1]}\left[\alpha/\tilde{A}_\alpha\right]. \quad (17)$$

The great value of the horizontal-slice representation is that it lets one apply everything that has already been developed for IT2 FSs to each of the horizontal slices, and therefore, all operations that have been developed for IT2 FSs (e.g., union, intersection, complement), as well as other computations (e.g., centroid, similarity) can be applied to GT2 FSs, but for each of their horizontal slices (e.g., [19], [31], [48], [62]–[64], [75]–[79]). it must be pointed out, however, that the actual calculations are still performed using vertical slices.

[11] In the zSlice literature, coordinates are called (*x, y, z*) instead of (*x, u, μ*); hence, z is also the same as α.

C. Wavy-Slice Representation of a GT2 FS

Mendel and John provided the following[12] (*Wavy Slice*) *Representation Theorem* [44] for $\tilde{A}$: $\tilde{A}$ is the set theoretic union of all of its embedded T2 FSs, i.e., (see Fig. 5)

$$\tilde{A} = \bigcup_{\forall j} \tilde{A}_e^j \quad (18)$$

Although impractical for most computations, because (18) requires the enumeration of the embedded T2 FSs, and their number can be extremely large, depending upon the discretization levels of the primary and secondary variables, this *wavy slice representation* of a GT2 FS has proved to be of great value for developing new theoretical results. Arguably, its most important use has been for IT2 FSs, namely:

The *Wavy-Slice Representation Theorem* for an *IT2 FS* [49, Ch. 2] is that its FOU is (covered by) the union of all of its embedded T1 FSs.

Because such embedded sets are T1 FSs, it is therefore possible to apply everything that has already been developed for T1 FSs to each of the embedded T1 FSs, after which, the union in (18) is performed. This has been done in [46].

IV. REPRESENTATION OF A GENERAL TYPE-2 FUZZY SET USEFUL FOR OPTIMAL DESIGN APPLICATIONS

We have just seen that a GT2 FS can be represented in four different ways such as points, wavy slices, horizontal slices, and vertical slices. While each of the latter three plays an important role for theoretical aspects of GT2 FSs, we are interested to know *which of these representations should be used*[13] *during an optimal design of a GT2 FLS*. What is meant by an "optimal design" is explained in Section IX.

The basic premise of this section is that *one should use a parsimonious parametric representation of a GT2 FS during an optimal design of a GT2 FLS*. Such a representation is one that is described by as few parameters as possible.

Clearly, the point-representation of a GT2 FS is not parsimonious, because it is not even a parametric representation. Similarly, the wavy-slice representation, in which the wavy slices must be enumerated from all of the points of the GT2 FS, is also a nonparametric representation. Neither of these representations is useful for an optimal design of a GT2 FLS.

[12] This representation theorem has also been referred to as the *Mendel-John Representation Theorem*.

[13] Many papers have been published about different kinds of computations for a GT2 FS (e.g., operations [9], [13]–[15], [17], [22], [37], [41], [44], [48], [51], [52], [55], [57], [60], [63]; centroid/type-reduction [23], [30], [31], [37], [48], [70], [73], [75], [77]; defuzzification [18], [29], [63]; solving fuzzy equations [61]; uncertainty measures [76]; and, similarity [19]). For such studies, the nature of the representation of a GT2 FS is usually not a vital issue; therefore, when we discard a particular representation in this paper, we are in no way disparaging those earlier works.

The horizontal-slice representation needs to be made parametric for it to be useful for an optimal design of a GT2 FLS. This representation requires choosing the number of horizontal-slices and then parameterizing each of them. If, e.g., each horizontal-slice needs n_h parameters to describe it (i.e., assume that all of the horizontal slices have the same shape, and a higher horizontal slice is a squished version of the one just below it), and there are k horizontal-slices, then this representation will require kn_h parameters, which is not parsimonious. Additionally, a change in the numerical value of k from k_1 to k_2, where $k_2 > k_1$ leads to $(k_2 - k_1)n_h$ additional parameters. Finally, there does not seem to be a simple way to map (squish) a horizontal-slice at level α_1 into a horizontal-slice at level $\alpha_2 (\alpha_2 > \alpha_1)$, e.g., a simple way to shrink the horizontal-slice at level α_1 into a horizontal-slice at level α_2 such that the latter horizontal-slice is contained (nested) within the former horizontal-slice, as would be the case if, e.g., all of the secondary MFs were triangles or trapezoids, is not known. Even if one could find such a mathematical transformation, it would have to be parsimonious, and then the quantification of the nesting of successive horizontal-slices would have to be included as constraints during the optimization of a performance objective function as part of the optimal design of a GT2 FLS. Such a constrained optimization problem (although it does not yet exist) would be very challenging to solve, to say the least. Based on these arguments I conclude (somewhat surprisingly) that *the horizontal-slice representation of a GT2 FS is not useful for an optimal design of a GT2 FLS*.

This leaves only the vertical-slice representation of a GT2 FS. We will now demonstrate that this is a very flexible and parsimonious representation of such a FS.

There can be different ways to parameterize vertical slices. The worst way to do this is to parameterize each of them separately; such a representation would suffer from the same nonparsimony that the horizontal-slice representation does.

Our suggestion for parameterizing the vertical-slice representation of a GT2 FS is to 1) parameterize its FOU exactly as one presently parameterizes the FOU of an IT2 FS and 2) parameterize the secondary MFs by choosing a fairly simple function that introduces only one new parameter.[14] Because the secondary MFs are vertical slices, they are always anchored on the already-parameterized FOU. Examples of such secondary MFSs are as follows.

1) *Triangle*: The base of each triangle equals $\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)$, and its apex location Apex (x) is parameterized [31], [48] as $(w \in [0,1])$

$$\text{Apex}(x) = \underline{\mu}_{\tilde{A}}(x) + w[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)]. \tag{19}$$

When $w = 0$, the secondary MF is a right triangle whose right angle is perpendicular to $\underline{\mu}_{\tilde{A}}(x)$; when $w = 1/2$, the secondary MF is an isosceles triangle; and when $w = 1$, the secondary MF is a right triangle whose right angle is perpendicular to $\bar{\mu}_{\tilde{A}}(x)$. w is treated as a design parameter, but it is the *same* for all of the vertical slices.

2) *Symmetrical trapezoid*: The base of each trapezoid equals $\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)$ and its top is defined by left and right endpoints, $\text{EP}_l(x)$ and $\text{EP}_r(x)$, that are parameterized [31], [48] as $(w \in [0,1])$

$$\text{EP}_l(x) = \underline{\mu}_{\tilde{A}}(x) + \tfrac{1}{2} w[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)] \tag{20}$$

$$\text{EP}_r(x) = \bar{\mu}_{\tilde{A}}(x) - \tfrac{1}{2} w[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)]. \tag{21}$$

When $w = 0$, the trapezoid reduces to a square well, and *the GT2 FS reduces to an IT2 FS*; and when $w = 1, \text{EP}_l(x) = \text{EP}_r(x)$, so that the trapezoid reduces to an isosceles triangle. w is treated as a design parameter, but it is the *same* for all of the vertical slices.

Another choice for a secondary MF is a nonsymmetrical trapezoid; however, it requires two parameters to define it, and so we leave its formulas to the reader (see [48] for an example).

Examples of the number of parameters that would have to be optimized during an optimal design of a GT2 FLS, for each of its GT2 FSs, are as follows.

1) FOU is Gaussian with uncertain mean $m \in [m_1, m_2]$ and/or standard deviation $\sigma \in [\sigma_1, \sigma_2]$, and triangle or symmetrical trapezoid secondary MFs: four or five parameters per GT2 FS, namely, $\theta = \{m_1, m_2, \sigma, w\}$, or $\theta = \{m, \sigma_1, \sigma_2, w\}$ or $\theta = \{m_1, m_2, \sigma_1, \sigma_2, w\}$.
2) FOU is Trapezoidal with a normal (not necessarily symmetrical) trapezoid UMF $\{a, b, c, d\}$ and a sub-normal (not necessarily symmetrical) triangle LMF $\{e, f, g, h\}$ [h is the height of the LMF; if the LMF is normal, then $h = 1$ and it is described by $\{e, f, g\}$], and triangle or symmetrical trapezoid secondary MFs: nine or eight parameters per GT2 FS, namely, $\theta = \{a, b, c, d, e, f, g, h, w\}$ or $\theta = \{a, b, c, d, e, f, g, w\}$.

Clearly, a Gaussian FOU is more parsimonious than a trapezoidal FOU.

Our conclusion is that for the optimal design of a GT2 FLS one should use the vertical-slice representation of its GT2 FSs.

During computations, one will need to use the $\alpha - \text{cuts}$ of either the triangle or symmetrical trapezoid secondary MFs. Formulas for these $\alpha - \text{cuts}$ are as follows.[15]

1) Formulas for the $\alpha - \text{cuts}$ of Triangle secondary MFs:

$$\begin{cases} \tilde{A}_\alpha(x) = [a_\alpha(x), b_\alpha(x)] \\ a_\alpha(x) = \underline{\mu}_{\tilde{A}}(x) + w[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)]\alpha \\ b_\alpha(x) = \bar{\mu}_{\tilde{A}}(x) - (1-w)[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)]\alpha. \end{cases} \tag{22}$$

2) Formulas for the $\alpha - \text{cuts}$ of symmetrical trapezoid secondary MFs:

$$\begin{cases} \tilde{A}_\alpha(x) = [a_\alpha(x), b_\alpha(x)] \\ a_\alpha(x) = \underline{\mu}_{\tilde{A}}(x) + \dfrac{1}{2} w[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)]\alpha \\ b_\alpha(x) = \bar{\mu}_{\tilde{A}}(x) - \dfrac{1}{2} w[\bar{\mu}_{\tilde{A}}(x) - \underline{\mu}_{\tilde{A}}(x)]\alpha. \end{cases} \tag{23}$$

[14] Begin with secondary MFs that can be described by using only one new parameter; if performance is not acceptable, then use secondary MFs that can be described by using two new parameters; etc.

[15] $a_\alpha(x)$ is found by equating the formula for the left-leg of the triangle (trapezoid) to α, and $b_\alpha(x)$ is found by equating the formula for the right-leg of the triangle (trapezoid) to α.

Finally, we are reminded that $\alpha-\text{plane}\,\tilde{A}_\alpha$, in (15), can be obtained from (22) or (23) by storing $\tilde{A}_\alpha(x)$ at sampled values of $x \in X$.

V. Operations

The operations that are used in a GT2 FLS are *intersection* (which models the word AND in each rule), *union* (which models the way in which fired-rules may be combined), and *type reduction* (which maps a GT2 FS into a T1 FS, after which, T1 FS is defuzzified). Because of the horizontal-slice representation of a GT2 FS, it is intuitive that all these operations can be performed first for each horizontal slice after which the resulting horizontal slices can be combined by computing their fuzzy union with respect to α. It is actually easier to develop the equations for intersection and union by using the vertical-slice representation for a GT2 FS, which is what we do next.

A. Intersection

Consider two GT2 FSs $\tilde{A}$ and $\tilde{B}$, where from (8) and (11)

$$\tilde{A} = \int_{\forall x\in X} \tilde{A}(x)/x = \int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/\tilde{A}_\alpha(x)\right] \Big/ x$$

$$= \int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/[a_\alpha(x), b_\alpha(x)]\right] \Big/ x \tag{24}$$

$$\tilde{B} = \int_{\forall x\in X} \tilde{B}(x) \Big/ x = \int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/\tilde{B}_\alpha(x)\right] \Big/ x$$

$$= \int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/[c_\alpha(x), d_\alpha(x)]\right] \Big/ x. \tag{25}$$

It follows, therefore, that $\tilde{A}\cap\tilde{B}$ can be expressed as

$$\tilde{A}\cap\tilde{B} = \left(\int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/[a_\alpha(x), b_\alpha(x)]\right] \Big/ x\right)$$
$$\cap \left(\int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/[c_\alpha(x), d_\alpha(x)]\right] \Big/ x\right) \tag{26}$$

which can be reorganized, as follows:[16]

$$\tilde{A}\cap\tilde{B} = \int_{\forall x\in X} \left[\sup_{\forall \alpha\in[0,1]} \alpha/[a_\alpha(x), b_\alpha(x)]\right.$$
$$\left. \cap \sup_{\forall \alpha\in[0,1]} \alpha/[c_\alpha(x), d_\alpha(x)]\right] \Big/ x \tag{27}$$

$$\tilde{A}\cap\tilde{B} = \int_{\forall x\in X} \sup_{\forall \alpha\in[0,1]} [\alpha/\{[a_\alpha(x), b_\alpha(x)]\cap[c_\alpha(x), d_\alpha(x)]\}]/x \tag{28}$$

$$\tilde{A}\cap\tilde{B} = \int_{\forall x\in X} \sup_{\forall \alpha\in[0,1]} [\alpha/\{[a_\alpha(x)\wedge c_\alpha(x), b_\alpha(x)\wedge d_\alpha(x)]\}]/x \tag{29}$$

[16] In going from (27) to (28), we use the well-known $\alpha-\text{cut}$ result for T1 FSs, that the intersection of $\alpha-\text{cuts}$ equals the $\alpha-\text{cut}$ of the intersection [26].

where interval arithmetic has been used in going from (28) to (29), and $\wedge$ is the minimum or product operation. Equation (29) is a vertical-slice representation of the intersection. Each vertical slice in (29) is known as the[17] *meet* of the vertical slices of $\tilde{A}$ and $\tilde{B}$ [52], [22], [35], [44]. To compute $\tilde{A}\cap\tilde{B}$ using (29), proceed as follows.

1) Choose the number of $\alpha-\text{cuts}, k$, for all of the vertical slices of $\tilde{A}$ and $\tilde{B}$, and denote the values of α as $\alpha_1,\ldots,\alpha_k$.
2) Discretize X over the supports of $\tilde{A}$ and $\tilde{B}$ calling those values of $x, x_1,\ldots,x_N$.
3) For each $x_i (i=1,\ldots,N)$: Compute
   a) $[a_{\alpha_j}(x_i), b_{\alpha_j}(x_i)], [c_{\alpha_j}(x_i), d_{\alpha_j}(x_i)], j=1,\ldots,k$;
   b) $a_{\alpha_j}(x_i)\wedge c_{\alpha_j}(x_i), b_{\alpha_j}(x_i)\wedge d_{\alpha_j}(x_i)$; $j=1,\ldots,k$;
   c) $\sup_{\forall\alpha\in[0,1]} [\alpha/\{[a_\alpha(x_i)\wedge c_\alpha(x_i), b_\alpha(x_i)\wedge d_\alpha(x_i)]\}]$.

Doing this provides the vertical slice of $\tilde{A}\cap\tilde{B}$ for each x_i.

To obtain the horizontal-slice representation of the intersection, simply interchange the sup and $\int$ operations in (29), i.e.,

$$\tilde{A}\cap\tilde{B} = \sup_{\forall\alpha\in[0,1]} \alpha \Big/ \left\{\int_{\forall x\in X} [a_\alpha(x)\wedge c_\alpha(x), b_\alpha(x)\wedge d_\alpha(x)]/x\right\}. \tag{30}$$

B. Union

Because the details of the derivations of the formulas for computing $\tilde{A}\cup\tilde{B}$ are so similar to those for $\tilde{A}\cap\tilde{B}$, we leave them to the readers. The vertical-slice results [analogous to [29]] are

$$\tilde{A}\cup\tilde{B} = \int_{\forall x\in X} \sup_{\forall\alpha\in[0,1]} [\alpha/\{[a_\alpha(x)\vee c_\alpha(x), b_\alpha(x)\vee d_\alpha(x)]\}]/x \tag{31}$$

where $\vee$ is the maximum operation. Each vertical slice in (31) is known as the *join* of the vertical slices of $\tilde{A}$ and $\tilde{B}$ [22], [35], [44], [52].

The procedure for computing $\tilde{A}\cup\tilde{B}$ is similar to the one just provided for $\tilde{A}\cap\tilde{B}$. Steps 1–3a are the same, but Steps 3b and 3c are changed the following: Compute (3b) as $a_{\alpha_j}(x_i)\vee c_{\alpha_j}(x_i)$ and $b_{\alpha_j}(x_i)\vee d_{\alpha_j}(x_i), j=1,\ldots,k$ and (3c) as

$$\sup_{\forall\alpha\in[0,1]} [\alpha/\{[a_\alpha(x_i)\vee c_\alpha(x_i), b_\alpha(x_i)\vee d_\alpha(x_i)]\}].$$

Doing this provides the vertical slice of $\tilde{A}\cup\tilde{B}$ for each x_i.

The horizontal-slice results [analogous to [30]] are

$$\tilde{A}\cup\tilde{B} = \sup_{\forall\alpha\in[0,1]} \alpha \Big/ \left\{\int_{\forall x\in X} [a_\alpha(x)\vee c(x), b_\alpha(x)\vee d_\alpha(x)]/x\right\}, \tag{32}$$

[17] The formulas for the meet (join) that we find in the literature (e.g., [22], [35], [52]) will not look like (29), [see (31)] because they are based on the Extension Principle [74] rather than on $\alpha-\text{cuts}$. The connection between these different looking formulas comes through the proof of the Function Decomposition Theorem [26] that is based on the Extension Principle.

C. Centroid Type Reduction

Type reduction (TR) [24], [35] is a mathematical operation that maps a GT2 FS into a T1 FS, and is used as a first step toward computing a single number at the output of a GT2 FLS (obtained through any kind of T1 defuzzification method that is applied to the type-reduced set). Of all T2 operations TR is the most novel because such an operation does not exist for a T1 FS.

A constraint for any TR method is that: if a GT2 FS reduces to a T1 FS then TR must reduce to its comparable T1 defuzzification method.

The two most popular forms of TR are *Centroid TR* (also called the *Centroid of a GT2 FS*) and *Center-of-Sets (COS) TR*. Centroid TR requires that a complete description of a GT2 FS be available (e.g., its vertical slices) and is explained below. Center-of-sets TR does not require that a complete description of a GT2 FS be available and is covered in Section VII, where the operations that are performed for a GT2 FLS are explained.

The main principle that is used to derive the centroid of a GT2 FS is a generalization of the function decomposition theorem for T1 FSs (stated in words at the beginning of Section II) to GT2 FSs. Because each secondary MF of a GT2 FS can be represented using its $\alpha-\text{cut}$ decomposition, and the aggregation of each $\alpha-\text{cut}$ for $\forall x \in X$ is a horizontal slice, the generalization of the function decomposition for T1 FSs to GT2 FSs is the following: *The MF for a function of GT2 FSs equals the fuzzy union (over all values of α) of the MFs for the same function applied to the horizontal slices of the GT2 FS*. A derivation of this somewhat intuitive result for measures of a GT2 FS (e.g., centroid, variance, skewness) is in [76].

Using this result, it is straightforward to obtain the following formula for centroid TR of GT2 FS $\tilde{B}$:

$$C_{\tilde{B}} = \sup_{\forall \alpha \in [0,1]} C_{\tilde{B}_\alpha} \tag{33}$$

where $C_{\tilde{B}_\alpha}$ is the centroid of $\alpha-\text{plane}\ \tilde{B}_\alpha$ raised to level α. Because $\tilde{B}_\alpha$ is analogous to an FOU, its centroid is analogous to the centroid of an IT2 FS but at level α. We are assumed to already be familiar with IT2 FSs, so we know that the centroid of an IT2 FS is an interval fuzzy set. For $\tilde{B}_\alpha$, this means that

$$C_{\tilde{B}_\alpha} = \alpha/[l_{\tilde{B}_\alpha}, r_{\tilde{B}_\alpha}]. \tag{34}$$

The interval $[l_{\tilde{B}_\alpha}, r_{\tilde{B}_\alpha}]$ lies on the primary-variable (x) axis; its two end-points can be computed by means of KM, EKM, or EIASC algorithms, as well as by other algorithms[18] (e.g., [23], [68], [70]). The details of these very-well documented algorithms are not needed for the rest of this paper.

VI. Brief Review of Interval Type-2 Fuzzy Logic Systems

A GT2 FLS builds upon an IT2 FLS; hence, this section provides a brief summary of an IT2 FLS (e.g., [16], [28], [35], [37], [38]). Such an FLS is depicted in Fig. 6; it is very similar to a T1 FLS, the major structural difference being that the defuzzifier block of a T1 FLS is replaced by the *Output Processing* block in the IT2 FLS. That block consists of type-reduction followed by defuzzification. As already explained in Section V, type-reduction maps a T2 FS into a T1 FS, after which defuzzification (as usual) maps that T1 FS into a crisp number.

As for a T1 FLS, there are two major architectures for an IT2 FLS, Mamdani and TSK. Both are covered in this section.

A. Interval Type-2 Mamdani Fuzzy Logic System[19]

An IT2 Mamdani FLS has p inputs $x_1 \in X_1, \ldots, x_p \in X_p$, and one output $y \in Y$ and is characterized by M rules, where the sth rule has the form

$$R^s: \text{IF } x_1 \text{ is } \tilde{F}_1^s \text{ and } \ldots \text{ and } x_p \text{ is } \tilde{F}_p^s \quad \text{THEN } y \text{ is } \tilde{G}^s \quad s = 1, \ldots, M. \tag{35}$$

The structure of rules does not change in going from T1 FSs to IT2 FSs; it is only the way in which the antecedent and consequent words are modeled that changes. Here, we assume that *all* the antecedent and consequent fuzzy sets in rules are IT2 FSs; however, this is not necessary in practice. All results remain valid as long as any one of a rule's antecedent or consequent (or input) FSs is T2.

For simplicity, in this paper we assume singleton fuzzification,[20] i.e., that the measured crisp inputs are modeled by type-0 sets. For each rule, one always first computes its *firing interval*, $F^s(\mathbf{x}')$, i.e., when $\mathbf{x} = \mathbf{x}'$ one computes[21]

$$F^s: \begin{cases} F^s(\mathbf{x}') \equiv [\underline{f}^s(\mathbf{x}'), \bar{f}^s(\mathbf{x}')] \\ \underline{f}^s(\mathbf{x}') \equiv T_{m=1}^p \underline{\mu}_{\tilde{F}_m^s}(x'_m) \\ \bar{f}^s(\mathbf{x}') \equiv T_{m=1}^p \bar{\mu}_{\tilde{F}_m^s}(x'_m) \end{cases} \tag{36}$$

in which T denotes a t-norm, either the minimum or product.

The next computation depends on the choice of type-reduction.

1) Centroid Type reduction: First, the firing interval for each rule is combined with its consequent IT2 FS to produce the *fired-rule output set* $\tilde{B}^s$ that is described by its FOU:

$$\tilde{B}^s: \begin{cases} \text{FOU}(\tilde{B}^s) = \left[\underline{\mu}_{\tilde{B}^s}(y|\mathbf{x}'), \bar{\mu}_{\tilde{B}^s}(y|\mathbf{x}')\right] \forall y \in Y \\ \underline{\mu}_{\tilde{B}^s}(y|\mathbf{x}') = \underline{f}^s(\mathbf{x}') \bigstar \underline{\mu}_{\tilde{G}^s}(y) \\ \bar{\mu}_{\tilde{B}^s}(y|\mathbf{x}') = \bar{f}^s(\mathbf{x}') \bigstar \bar{\mu}_{\tilde{G}^s}(y) \end{cases} \tag{37}$$

where $\bigstar$ is a t-norm, also either the minimum or product, and (usually) chosen the same as the t-norm in (36).

Next, the union of all of the fired-rule output sets is performed to obtain an *aggregated output IT2 FS* $\tilde{B}$ that is described by its

[18] See [43, Sec. VIII-A] for discussions about dramatic ways to speed up the computations of $C_{\tilde{B}_\alpha}$ that make use of connections (or other information) about the secondary MFs in going (flowing) from one $\alpha-$plane to the next.

[19] See [46] for derivations of the results that are in this section using T1 mathematics.

[20] See [35, ch. 11 and 12] for extensions to two kinds of non-singleton fuzzification: T1 and IT2.

[21] Because a FLS is a "system" whose inputs change with respect to time (as in a fuzzy logic controller or forecaster), or spatially (as in rule-based image processing), or as features change (as in rule-based classification), we show the computations that are performed in the FLS as a function of the input $\mathbf{x}$.

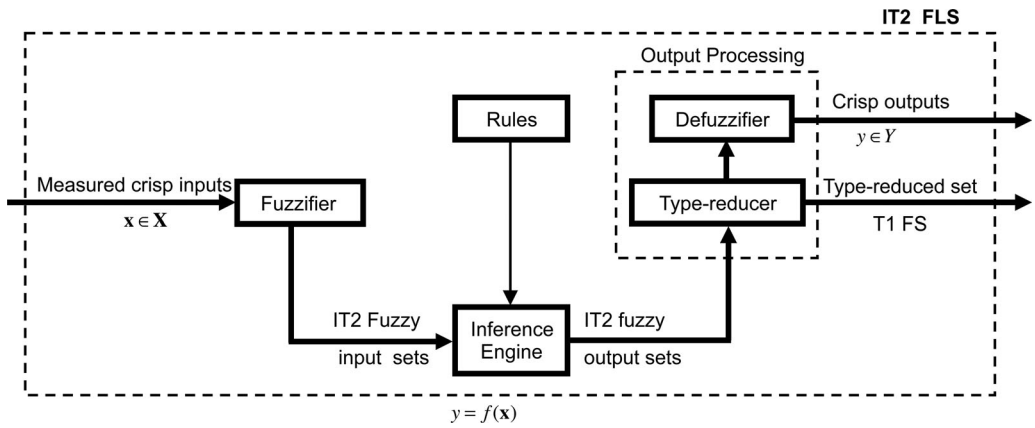

Fig. 6. IT2 FLS [46; © 2006, IEEE].

FOU:

$$\tilde{B}: \begin{cases} \text{FOU}(\tilde{B}) = \left[\underline{\mu}_{\tilde{B}}(y|\mathbf{x}'), \bar{\mu}_{\tilde{B}}(y|\mathbf{x}')\right] & \forall y \in Y \\ \underline{\mu}_{\tilde{B}}(y|\mathbf{x}') = \underline{\mu}_{\tilde{B}^1}(y|\mathbf{x}') \vee \underline{\mu}_{\tilde{B}^2}(y|\mathbf{x}') \vee \cdots \vee \underline{\mu}_{\tilde{B}^M}(y|\mathbf{x}') \\ \bar{\mu}_{\tilde{B}}(y|\mathbf{x}') = \bar{\mu}_{\tilde{B}^1}(y|\mathbf{x}') \vee \bar{\mu}_{\tilde{B}^2}(y|\mathbf{x}') \vee \cdots \vee \bar{\mu}_{\tilde{B}^M}(y|\mathbf{x}') \end{cases} \tag{38}$$

where $\vee$ is the maximum operation. Finally, the centroid of $\tilde{B}$ is computed to obtain the type-reduced set, $Y_C(\mathbf{x}')$:

$$Y_C(\mathbf{x}') = C_{\tilde{B}}(\mathbf{x}') = 1/[l_{\tilde{B}}(\mathbf{x}'), r_{\tilde{B}}(\mathbf{x}')] \tag{39}$$

where $l_{\tilde{B}}(\mathbf{x}')$ and $r_{\tilde{B}}(\mathbf{x}')$ are found by using algorithms such as the EKM algorithms, and $Y_C(\mathbf{x}')$ lies on the axis of the consequent's primary variable, y.

2) Center of Sets Type reduction: This TR method [23], [35] bypasses both (37) and (38), thereby saving computation time; however, it requires the computation of the centroids $C_{\tilde{G}^s}$ of the consequent IT2 FSs, $\tilde{G}^s$:

$$C_{\tilde{G}^s} = 1/[l_{\tilde{G}^s}, r_{\tilde{G}^s}], \quad s = 1, \ldots, M \tag{40}$$

where $l_{\tilde{G}^s}$ and $r_{\tilde{G}^s}$ are found by using algorithms such as the EKM algorithms, and $C_{\tilde{G}^s}$ lies on the axis of the consequent's primary variable, y. After the design of the IT2 FLS has been completed these centroids can be computed one last time and stored because they do not depend upon $\mathbf{x}'$.

The $MC_{\tilde{G}^s}$ are used, along with the firing intervals, $F^s(\mathbf{x}')$, to compute the following *interval weighted average* (IWA)[22]:

$$Y_{\text{COS}}(\mathbf{x}') = 1 \Bigg/ \underset{\substack{y_s \in [l_{\tilde{G}^s}, r_{\tilde{G}^s}] \\ w_s \in [\underline{f}^s(\mathbf{x}'), \bar{f}^s(\mathbf{x}')]}}{\forall} \sum_{s=1}^{M} y_s w_s \Bigg/ \sum_{s=1}^{M} w_s. \tag{41}$$

This is called an IWA because y_s and w_s in the scalar average have intervals of values associated with them. Note that $Y_{\text{COS}}(\mathbf{x}')$ is also an interval fuzzy set, i.e.,

$$Y_{\text{COS}}: \begin{cases} Y_{\text{COS}}(\mathbf{x}') = 1/[y_l(\mathbf{x}'), y_r(\mathbf{x}')] \\ y_l(\mathbf{x}') = \min\limits_{\forall w_s \in [\underline{f}^s(\mathbf{x}'), \bar{f}^s(\mathbf{x}')]} \sum\limits_{s=1}^{M} l_{\tilde{G}^s} w_s \Big/ \sum\limits_{s=1}^{M} w_s \\ y_r(\mathbf{x}') = \max\limits_{\forall w_s \in [\underline{f}^s(\mathbf{x}'), \bar{f}^s(\mathbf{x}')]} \sum\limits_{s=1}^{M} r_{\tilde{G}^s} w_s \Big/ \sum\limits_{s=1}^{M} w_s \end{cases} \tag{42}$$

where $y_l(\mathbf{x}')$ and $y_r(\mathbf{x}')$ are computed using EKM or similar algorithms. Observe that, unlike the computation of the centroid of an IT2 FS, where $l_{\tilde{G}^s}$ and $r_{\tilde{G}^s}$ are replaced by the same sampled and increasingly ordered independent variable (e.g., $y_1 < y_2 < \ldots < y_N$), in order to compute $y_l(\mathbf{x}')$ by an EKM algorithm, the $Ml_{\tilde{G}^s}$ must first be reordered in increasing order, and, in order to compute $y_r(\mathbf{x}')$ by an EKM algorithm, the $Mr_{\tilde{G}^s}$ must also first be reordered in increasing order. The order of the reordered $l_{\tilde{G}^s}$ is usually different from the order of the reordered $r_{\tilde{G}^s}$

After TR (Centroid or COS), defuzzification is very simple, i.e.,

$$y(\mathbf{x}') = \begin{cases} \dfrac{1}{2}[l_{\tilde{B}}(\mathbf{x}') + r_{\tilde{B}}(\mathbf{x}')], & \text{Centroid TR} \\ \dfrac{1}{2}[y_l(\mathbf{x}') + y_r(\mathbf{x}')], & \text{COS TR} \end{cases} \tag{43}$$

B. Interval Type-2 Takagi—Sugeno–Kang Fuzzy Logic System

An IT2 TSK FLS [27], [35] is also described by fuzzy IF–THEN rules that represent input–output relations of a system. In the mostly commonly used IT2 TSK FLS the sth rule can be expressed as

$$R^s_{A2-C0}: \text{IF } x_1 \text{ is } \tilde{F}^s_1 \text{ and } \ldots \text{ and } x_p \text{ is } \tilde{F}^s_p \quad \text{THEN } y^s(\mathbf{x}) = c^s_0 + c^s_1 x_1 + c^s_2 x_2 + \cdots + c^s_p x_p \tag{44}$$

where $s = 1, \ldots, M$. The *A2-C0* notation means that the antecedents are modeled by IT2 FSs, whereas the consequent is modeled as a type-0 FS, i.e., as a number. Although more general kinds of IT2 TSK rules have been formulated [27], [35], [50,

[22] This average was originally called a *Generalized Centroid* [23], [35]; however, it has since been recognized to be one kind of so-called *novel weighted averages* [49], the others being called a *fuzzy weighted average* (FWA) and a *linguistic weighted average* (LWA). The IWA is the most basic of these novel weighted averages and is used to compute both the FWA and LWA.

(see also other references therein)], they are beyond the scope of this paper; consequently, in the sequel, $R^s_{A2-C0} \equiv R^s_{\rm TSK}$.

The output, $y_{{\rm TSK},I2}(\mathbf{x}')$, of an IT2 TSK FLS is obtained according to the following steps.

1) Compute the value of each of the consequents of the A2-C0 rules, i.e. compute the numbers $\{y^s(\mathbf{x}')\}_{s=1}^M$. Then, rank order the $\{y^s(\mathbf{x}')\}_{s=1}^M$ in increasing order calling them $\{\gamma^r(\mathbf{x}')\}_{r=1}^M$.

2) Compute the firing interval $F^s(\mathbf{x}') = [\underline{f}^s(\mathbf{x}'), \bar{f}^s(\mathbf{x}')]$ for each of the M rules using (36). Relabel these firing intervals so that they conform to the $\{\gamma^r(\mathbf{x}')\}_{r=1}^M$.

3) Compute the following IWA[23]:

$$Y_{\rm TSK}(\mathbf{x}') = 1 \Bigg/ \left[\sum_{s=1}^{M} F^s(\mathbf{x}')\gamma^s(\mathbf{x}') \Bigg/ \sum_{s=1}^{M} F^s(\mathbf{x}')\right] = 1/[y_l(\mathbf{x}'), y_r(\mathbf{x}')]. \tag{45}$$

The actual computations of $y_l(\mathbf{x}')$ and $y_r(\mathbf{x}')$ are performed by using EKM algorithms that are applied to

$$y_l(\mathbf{x}') = \min_{\substack{f^s(\mathbf{x}')\in[\underline{f}^s(\mathbf{x}'),\bar{f}^s(\mathbf{x}')]\\ s=1,\ldots,M}} \frac{\sum_{s=1}^M f^s(\mathbf{x}')\gamma^s(\mathbf{x}')}{\sum_{s=1}^M f^s(\mathbf{x}')} \tag{46}$$

$$y_r(\mathbf{x}') = \max_{\substack{f^s(\mathbf{x}')\in[\underline{f}^s(\mathbf{x}'),\bar{f}^s(\mathbf{x}')]\\ s=1,\ldots,M}} \frac{\sum_{s=1}^M f^s(\mathbf{x}')\gamma^s(\mathbf{x}')}{\sum_{s=1}^M f^s(\mathbf{x}')}. \tag{47}$$

4) The interval $[y_l(\mathbf{x}'), y_r(\mathbf{x}')]$ is defuzzified to provide $y_{{\rm TSK},I2}(\mathbf{x}')$, *where*

$$y_{{\rm TSK},I2}(\mathbf{x}') = \tfrac{1}{2}[y_l(\mathbf{x}') + y_r(\mathbf{x}')] \tag{48}$$

Although there is a strong similarity between these computations and the ones for an IT2 Mamdani FLS that uses COS type-reduction, there is also the following big difference: In the IT2 Mamdani FLS the centroids of its rule-consequent IT2 FSs are computed just once, and they are not a function of the input $\mathbf{x} = \mathbf{x}'$; but, in the IT2 TSK FLS, the point-value of each rule's consequent has to be recomputed for each value of $\mathbf{x} = \mathbf{x}'$.

VII. General Type-2 Fuzzy Logic Systems

Figure 6 is applicable for any kind of T2 FLS, interval, or general. For clarity in this paper, we repeat it as Fig. 7, but for a GT2 FLS. Based on the horizontal-slice representation of a GT2 FS, we have the following:

1) A *horizontal-slice Mamdani (TSK) FLS* is analogous to an IT2 Mamdani (TSK) FLS where all of the IT2 FS computations that are given in Section VI occur for each of the horizontal slices.
2) A *GT2 Mamdani (TSK) FLS* can be created as the fuzzy union over α of the horizontal-slice Mamdani (TSK) FLSs.

For clarity and completeness, we provide the equations for these GT2 FLSs next.

[23]The middle term of (45) is an expressive formula; actual computations are performed according to (46) and (47).

A. General Type-2 Mamdani Fuzzy Logic System

As in Section VI, we assume singleton fuzzification. For each rule, when $x_m = x'_m$ only the T1 vertical slice of the rule-antecedent GT2 FS $\tilde{F}^s_m, \tilde{F}^s_m(x'_m)$ is activated (see Fig. 1 for examples of activated vertical slices at x_1, x_3, and x_5; of course, in an FLS, only one of them is activated at a time), and it has the following $\alpha-$cut decomposition[24] [see (11)]:

$$\tilde{F}^s_m(x'_m) = \sup_{\forall\alpha\in[0,1]} \alpha/[a^s_{m,\alpha}(x'_m), b^s_{m,\alpha}(x'_m)]. \tag{49}$$

One then computes the *firing interval* $F^s_\alpha(\mathbf{x}')$ *for level* α, as [see (36)]:

$$F_\alpha: \begin{cases} F^s_\alpha(\mathbf{x}') \equiv [\underline{f}^s_\alpha(\mathbf{x}'), \bar{f}^s_\alpha(\mathbf{x}')] \\ \underline{f}^s_\alpha(\mathbf{x}') \equiv T^p_{m=1} a^s_{m,\alpha}(x'_m) \\ \bar{f}^s_\alpha(\mathbf{x}') \equiv T^p_{m=1} b^s_{m,\alpha}(x'_m). \end{cases} \tag{50}$$

1) Centroid Type Reduction: A horizontal slice (for level$-\alpha$), $\tilde{G}^s_\alpha$, of a rule's consequent GT2 FS, $\tilde{G}^s$, is [see (15)] $(s = 1, \ldots, M)$:

$$\tilde{G}^s_\alpha = \int_{\forall y\in Y} G^s_\alpha(y)/y = \int_{\forall y\in Y} [g^s_{L,\alpha}(y), g^s_{R,\alpha}(y)]/y. \tag{51}$$

Each rule's firing interval for level α is combined with its respective $\tilde{G}^s_\alpha$, to provide a *fired-rule horizontal slice* $\tilde{B}^s_\alpha$ as [see (36)]:

$$\tilde{B}^s_\alpha: \begin{cases} \text{FOU}(\tilde{B}^s_\alpha) = \left[\underline{\mu}_{\tilde{B}^s_\alpha}(y|\mathbf{x}'), \bar{\mu}_{\tilde{B}^s_\alpha}(y|\mathbf{x}')\right] \forall y \in Y \\ \underline{\mu}_{\tilde{B}^s_\alpha}(y|\mathbf{x}') = \underline{f}^s_\alpha(\mathbf{x}') \bigstar g^s_{L,\alpha}(y) \\ \bar{\mu}_{\tilde{B}^s_\alpha}(y|\mathbf{x}') = \bar{f}^s_\alpha(\mathbf{x}') \bigstar g^s_{R,\alpha}(y). \end{cases} \tag{52}$$

Next, the union of all of the $\tilde{B}^s_\alpha$ is performed, so as to obtain a new *aggregated output horizontal slice*, namely $\tilde{B}_\alpha$, where [see (38)]:

$$\tilde{B}_\alpha: \begin{cases} \text{FOU}(\tilde{B}_\alpha) = \left[\underline{\mu}_{\tilde{B}_\alpha}(y|\mathbf{x}'), \bar{\mu}_{\tilde{B}_\alpha}(y|\mathbf{x}')\right] \quad \forall y \in Y \\ \underline{\mu}_{\tilde{B}_\alpha}(y|\mathbf{x}') = \underline{\mu}_{\tilde{B}^1_\alpha}(y|\mathbf{x}') \vee \underline{\mu}_{\tilde{B}^2_\alpha}(y|\mathbf{x}') \vee \cdots \vee \underline{\mu}_{\tilde{B}^M_\alpha}(y|\mathbf{x}') \\ \bar{\mu}_{\tilde{B}_\alpha}(y|\mathbf{x}') = \bar{\mu}_{\tilde{B}^1_\alpha}(y|\mathbf{x}') \vee \bar{\mu}_{\tilde{B}^2_\alpha}(y|\mathbf{x}') \vee \cdots \vee \bar{\mu}_{\tilde{B}^M_\alpha}(y|\mathbf{x}'). \end{cases} \tag{53}$$

Finally, the centroid of $\tilde{B}_\alpha$ is computed to obtain the *type-reduced set at level-*α, $Y_{C,\alpha}(\mathbf{x}')$[see (39)]

$$Y_{C,\alpha}(\mathbf{x}') = C_{\tilde{B}_\alpha}(\mathbf{x}') = \alpha/[l_{\tilde{B}_\alpha}(\mathbf{x}'), r_{\tilde{B}_\alpha}(\mathbf{x}')] \tag{54}$$

where $l_{\tilde{B}_\alpha}(\mathbf{x}')$ and $r_{\tilde{B}_\alpha}(\mathbf{x}')$ are found by using algorithms such as the EKM algorithms that are applied to $\tilde{B}_\alpha$.

2) Center of sets Type-reduction: To begin, one must compute the centroids of the M consequent GT2 FSs, as explained in Section V-C, i.e., [see (33) and (34)]:

$$C_{\tilde{G}^s} = \sup_{\forall\alpha\in[0,1]} C_{\tilde{G}^s_\alpha} \tag{55}$$

[24]Apologies for the very heavy subscript and superscript notation, but in a GT2 FLS, we need to keep track of which antecedent is referred to (the first subscript, m), which rule is referred to (the superscript s), and which $\alpha-$cut is referred to (the second subscript, α).

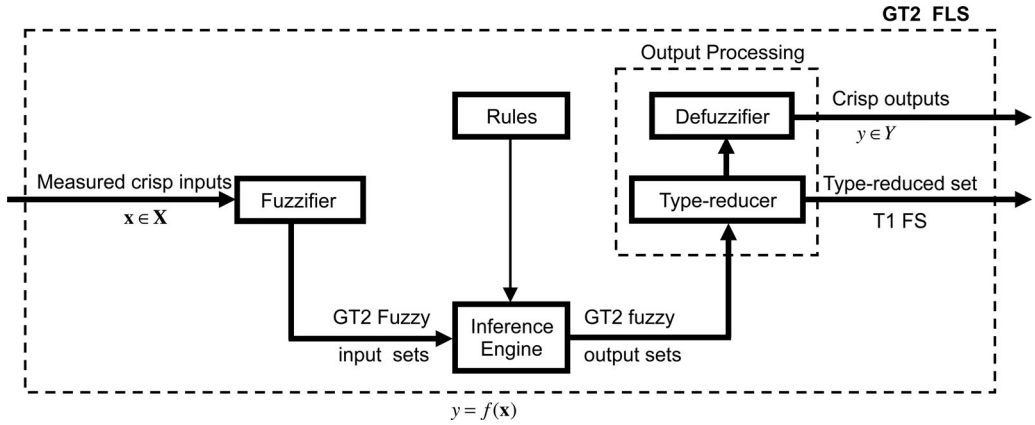

Fig. 7. GT2 FLS [46; © 2006, IEEE].

$$C_{\tilde{G}^s_\alpha} = \alpha/[l_{\tilde{G}^s_\alpha}, r_{\tilde{G}^s_\alpha}] \tag{56}$$

where $C_{\tilde{G}^s_\alpha}$ is the centroid of $\alpha-$plane $\tilde{G}^s_\alpha$ raised to level-α. After the design of the GT2 FLS has been completed, these centroids can be computed one last time and then stored because they do not depend upon $\mathbf{x}'$.

The $MC_{\tilde{G}^s_\alpha}$ are used, along with the firing interval $F^s_\alpha(\mathbf{x}')$ for level-α to compute the following IWA for level-α [see (42)]:

$$Y_{\mathrm{COS},\alpha}:$$
$$\begin{cases} Y_{\mathrm{COS},\alpha}(\mathbf{x}') = 1/[y_{l,\alpha}(\mathbf{x}'), y_{r,\alpha}(\mathbf{x}')] \\ y_{l,\alpha}(\mathbf{x}') = \min\limits_{\forall w_{s,\alpha} \in [\underline{f}^s_\alpha(\mathbf{x}'), \bar{f}^s_\alpha(\mathbf{x}')]} \sum\limits_{s=1}^{M} l_{\tilde{G}^s_\alpha} w_{s,\alpha} \Big/ \sum\limits_{s=1}^{M} w_{s,\alpha} \\ y_{r,\alpha}(\mathbf{x}') = \max\limits_{\forall w_{s,\alpha} \in [\underline{f}^s_\alpha(\mathbf{x}'), \bar{f}^s_\alpha(\mathbf{x}')]} \sum\limits_{s=1}^{M} r_{\tilde{G}^s_\alpha} w_{s,\alpha} \Big/ \sum\limits_{s=1}^{M} w_{s,\alpha}. \end{cases} \tag{57}$$

After type-reduction (Centroid or COS) defuzzification is performed. Three approaches for doing this are possible. We explain them for COS TR; however, exactly the same explanations apply to Centroid TR.

All three begin by aggregating all of the horizontal slices, constructing the T1 FS Y_{COS} as (see Fig. 8(a)):

$$Y_{\mathrm{COS}} = \sup_{\forall \alpha \in (0,1]} \alpha / Y_{\mathrm{COS},\alpha}(\mathbf{x}'). \tag{58}$$

It is important to observe that the $\alpha = 0$ slice is excluded because it does not contribute anything to the defuzzified result.[25]

1) *Approximation and Defuzzification*: Observe (see Fig. 8(a)) that when Y_{COS} is viewed as a function of y, it is only the two end-points of each $\alpha/Y_{\mathrm{COS},\alpha}(\mathbf{x}')$ that are associated with it, and, when those end-points are projected down onto the y-axis, the result is a nonuniformly sampled set of points[26], $y_1, y_2, \ldots, y_{2k}$. In order to achieve uniform sampling along the y-axis, the $2k$ end-point values of $\alpha/Y_{\mathrm{COS},\alpha}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ can be approximated using a spline approximation, the result being $\hat{Y}_{\mathrm{COS}}(y)$ (see Fig. 8(b)), which is then sampled uniformly (i.e., $y'_{j+1} - y'_j = \Delta$ for $\forall j$), after which, the defuzzified value of $\hat{Y}_{\mathrm{COS}}(y)$, $y_1(\mathbf{x}')$ is computed as its COG, i.e.,

$$y_1(\mathbf{x}') = \sum_{j=1}^{J} y'_j \hat{Y}_{\mathrm{COS}}(y'_j) \Big/ \sum_{j=1}^{J} \hat{Y}_{\mathrm{COS}}(y'_j) \tag{59}$$

Clearly, there are a lot of computations needed in order to obtain $y_1(\mathbf{x}')$, which makes it not very practical for real-time applications of a GT2 FLS.

2) *End-Point Defuzzification*: In this approach a set of *2k spikes* is created, two spikes located at the end-points of each $\alpha/Y_{\mathrm{COS},\alpha}(\mathbf{x}')$, both with amplitude α (see Fig. 8(c)). These spikes are then defuzzified as

$$y_2(\mathbf{x}') = \sum_{i=1}^{k} [\alpha_i y_{l,\alpha_i}(\mathbf{x}') + \alpha_i y_{r,\alpha_i}(\mathbf{x}')] \Big/ \sum_{i=1}^{k} 2\alpha_i. \tag{60}$$

Note that, in Fig. 8(c), $y_1 = y_{l,\alpha_1}(\mathbf{x}'), y_2 = y_{l,\alpha_2}(\mathbf{x}'),\ldots, y_k = y_{l,\alpha_k}(\mathbf{x}')$, and $y_{2k} = y_{r,\alpha_1}(\mathbf{x}'), y_{2k-1} = y_{r,\alpha_2}(\mathbf{x}'),\ldots, y_{k+1} = y_{r,\alpha_k}(\mathbf{x}')$.

3) *Average of End-Points Defuzzification*: In this approach one first computes the average value of each $\alpha/Y_{\mathrm{COS},\alpha}(\mathbf{x}')$ and locates a spike at that value with amplitude α (see Fig. 8(d)) after which the k spikes are defuzzified according to

$$y_3(\mathbf{x}') = \sum_{i=1}^{k} \alpha_i \left[(y_{l,\alpha_i}(\mathbf{x}') + y_{r,\alpha_i}(\mathbf{x}'))/2\right] \Big/ \sum_{i=1}^{k} \alpha_i. \tag{61}$$

This approach was first suggested by Wagner and Hagras [63], and is a much simpler way to obtain a defuzzified value than by approximation and defuzzification; however, it still requires a lot of computation because EKM algorithms have to be used in order to compute $Y_{\mathrm{COS},\alpha}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$. Of course, if $2k$ parallel processors are available then there will be a huge speedup in computation time, but this comes at the price of needing a lot of processors. Many in the IT2 FLS community are unhappy about even needing two such processors for real-time applications, which has led to some very clever ways for going directly to a defuzzified value without having to perform

[25] Wagner and Hagras [63] were the first to make this interesting observation.

[26] Of course, when α is sampled very finely, then $y_1, y_2, \ldots, y_{2k}$ will approach a uniformly sampled set of samples; however, in a GT2 FLS, one does not want to use many horizontal slices, or else, computational complexity greatly increases, as does computation time.

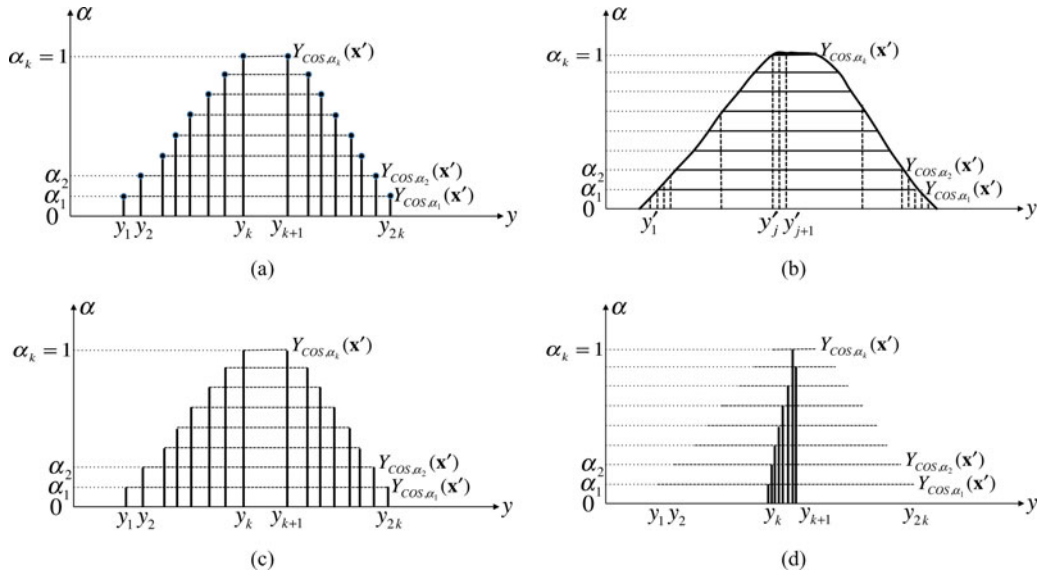

Fig. 8. (a) $Y_{\rm COS}$ as constructed from its $\alpha-$cuts [the filled-in dots represent the right-hand side of (58)], (b) spline approximation to $Y_{\rm COS}, \hat{Y}_{\rm COS}(y)$, after which, the continuous function $\hat{Y}_{\rm COS}(y)$ is uniformly sampled, (c) spikes of height α are located at the two end-points of each $\alpha-$cut of $Y_{\rm COS}$, and (d) spikes of height α are located at the average of the two end-points of each $\alpha-$cut of $Y_{\rm COS}$.

TR. We describe some of those methods in Section VIII, as well as their proposed extensions to a GT2 FLS.

Comparing the formulas for $y_3(\mathbf{x}')$ and $y_2(\mathbf{x}')$, it is clear that they are the same; hence, *there is no difference between End-Point Defuzzification and Average of End-Points Defuzzification*, something that was pointed out in [77].

Summary: In order to see the forest from the trees so that one can distinguish all of the computations for centroid TR versus COS TR, Fig. 9 has been created. It should be clear that COS TR requires many fewer computations than Centroid TR; hence, if COS TR was favored for an IT2 FLS, as it has been by many practitioners, it will be greatly favored for a GT2 FLS.

B. General Type-2 Takagi–Sugeno–Kang Fuzzy Logic System

Although there can be different kinds of GT2 TSK FLSs, we again focus only on the one that presently is practical, A2-C0. Now, however, the antecedents are modeled as GT2 FSs. The output $y_{{\rm TSK},G2}(\mathbf{x}')$ of a GT2 TSK FLS is obtained according to the following steps.

1) Compute the value of each of the consequents of the A2-C0 rules, i.e., compute the numbers $\{y^s(\mathbf{x}')\}_{s=1}^M$. Then, rank order the $\{y^s(\mathbf{x}')\}_{s=1}^M$ in an increasing order calling them $\{\gamma^r(\mathbf{x}')\}_{r=1}^M$. This is exactly the same as Step 1 in a TSK IT2 FLS, because in both cases the consequents are not fuzzy.

2) For each of the M rules, compute its *firing interval* $F_\alpha^s(\mathbf{x}')$ *for level-*α, exactly as in (50). Relabel these firing intervals so that they conform to the $\{\gamma^r(\mathbf{x}')\}_{r=1}^M$.

3) Compute the following ${\rm level}-\alpha$ *IWA* [see (45)]:

$$Y_{{\rm TSK},\alpha}(\mathbf{x}') = 1 \Bigg/ \left[\sum_{s=1}^M F_\alpha^s(\mathbf{x}')\gamma^s(\mathbf{x}') \Bigg/ \sum_{s=1}^M F_\alpha^s(\mathbf{x}')\right] = 1/[y_{l,\alpha}(\mathbf{x}'), y_{r,\alpha}(\mathbf{x}')]. \quad (62)$$

The actual computations of $y_{l,\alpha}(\mathbf{x}')$ and $y_{r,\alpha}(\mathbf{x}')$ are performed by using EKM algorithms that are applied to [see (46) and (47)]:

$$y_{l,\alpha}(\mathbf{x}') = \min_{\substack{f_\alpha^s(\mathbf{x}')\in[\underline{f}_\alpha^s(\mathbf{x}'),\bar{f}_\alpha^s(\mathbf{x}')] \\ s=1,\ldots,M}} \frac{\sum_{s=1}^M f_\alpha^s(\mathbf{x}')\gamma^s(\mathbf{x}')}{\sum_{s=1}^M f_\alpha^s(\mathbf{x}')} \quad (63)$$

$$y_{r,\alpha}(\mathbf{x}') = \max_{\substack{f_\alpha^s(\mathbf{x}')\in[\underline{f}_\alpha^s(\mathbf{x}'),\bar{f}_\alpha^s(\mathbf{x}')] \\ s=1,\ldots,M}} \frac{\sum_{s=1}^M f_\alpha^s(\mathbf{x}')\gamma^s(\mathbf{x}')}{\sum_{s=1}^M f_\alpha^s(\mathbf{x}')}. \quad (64)$$

4) Compute the defuzzified value, $y_{\rm TSK}(\mathbf{x}')$, using either *Approximation and Defuzzification* or *Average of End-Points Defuzzification*, as explained in Section A-3.

This TSK GT2 FLS suffers from many of the same computational issues that were explained below (61), which has motivated one method of bypassing Step 3 that is described in Section VIII-C.

VIII. Simplified General Type-2 Fuzzy Logic Systems

There have been a number of simplified IT2 FLSs proposed that bypass TR.[27] In this section, we explain how three of those simplified and very practical IT2 FLSs can be extended to simplify GT2 FLSs. None of the simplifications that we are about to describe have been applied yet to any application (to the best knowledge of this author); however, since their IT2 FLS counterparts have been, there is no reason to believe that they will not be. Consequently, the descriptions of these extensions that are given next are provided to expedite this.

In the following sections, we first provide the IT2 FLS results, after which we explain how they can be extended to a GT2 FLS.

[27]See [43, Sec. VI] for discussions about them, as well as the references for them.

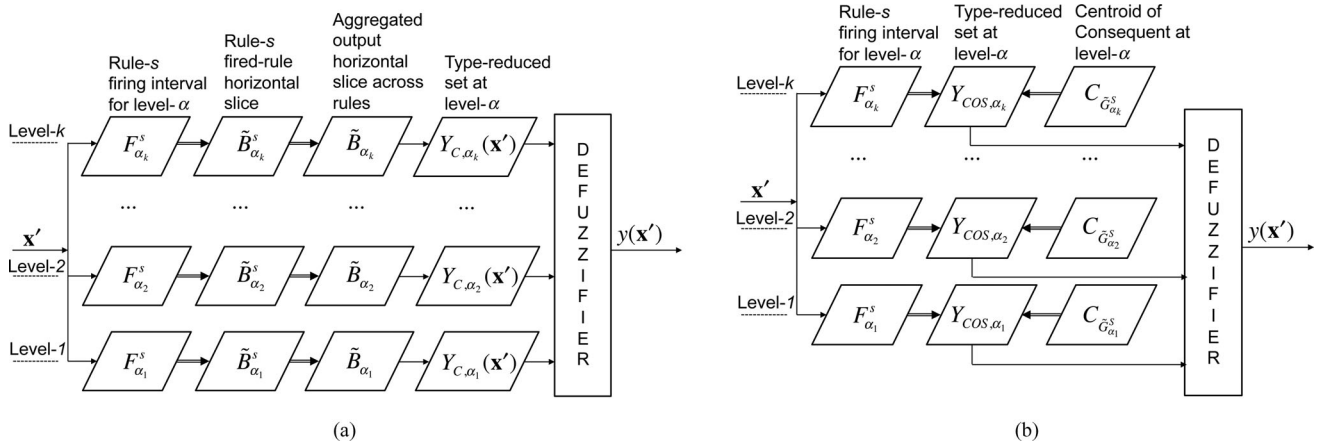

Fig. 9. GT2 Mamdani FLS computations. (a) Centroid TR and (b) COS TR.

A. Wu–Mendel Uncertainty Bounds (WM UBs) for a General Type-2 Mamdani Fuzzy Logic System

1) Interval Type-2 Fuzzy Logic System Results: Wu and Mendel [69] replace type reduction with lower and upper bounds—*uncertainty bounds*—for the end-points of the type-reduced set, and those bounds, which are optimal in a mini–max sense, are computed without having to perform type reduction. To begin, four centroids (also called *boundary T1 FLSs*) are defined, all of which can be computed once the left and right end-points of the firing interval, $\underline{f}^s(\mathbf{x}')$ and $\overline{f}^s(\mathbf{x}')(s=1,\ldots,M)$, have been computed. In these centroids, a^s and b^s are the left- and right-end points of the centroid of the sth consequent IT2 FS, which only have to be computed (and stored) one last time after the IT2 FLS has been designed, since they do not depend upon the input to the FLS, and $a^1<\ldots<a^s, b^1<\ldots<b^s$. The boundary T1 FLS centroids are[28]

$$\{\text{LMFs, left}\}:\ \mathrm{y}_l^{(0)}(\mathbf{x}')=\sum_{s=1}^{M}\underline{f}^s a^s\Bigg/\sum_{s=1}^{M}\underline{f}^s \tag{65}$$

$$\{\text{LMFs, right}\}:\ \mathrm{y}_r^{(M)}(\mathbf{x}')=\sum_{s=1}^{M}\underline{f}^s b^s\Bigg/\sum_{s=1}^{M}\underline{f}^s \tag{66}$$

$$\{\text{UMFs, left}\}:\ \mathrm{y}_l^{(M)}(\mathbf{x}')=\sum_{s=1}^{M}\overline{f}^s a^s\Bigg/\sum_{s=1}^{M}\overline{f}^s \tag{67}$$

$$\{\text{UMFs, right}\}:\ \mathrm{y}_r^{(0)}(\mathbf{x}')=\sum_{s=1}^{M}\overline{f}^s b^s\Bigg/\sum_{s=1}^{M}\overline{f}^s \tag{68}$$

Theorem 1 (WM UBs)[29] [69]: The end-points $y_l(\mathbf{x}')$ and $y_r(\mathbf{x}')$ of the TR set of an IT2 FLS for the input $\mathbf{x}=\mathbf{x}'$ are bounded from below and above as $\underline{y}_l(\mathbf{x}')\le y_l(\mathbf{x}')\le\bar{y}_l(\mathbf{x}')$ and $\underline{y}_r(\mathbf{x}')\le y_r(\mathbf{x}')\le\bar{y}_r(\mathbf{x}')$, where

$$\bar{y}_l(\mathbf{x}')=\min\left\{y_l^{(0)}(\mathbf{x}'),y_l^{(M)}(\mathbf{x}')\right\} \tag{69}$$

$$\underline{y}_r(\mathbf{x}')=\max\left\{y_r^{(0)}(\mathbf{x}'),y_r^{(M)}(\mathbf{x}')\right\} \tag{70}$$

$$\underline{y}_l(\mathbf{x}')=\bar{y}_l(\mathbf{x}')-\left[\frac{\sum_{s=1}^{M}\left(\bar{f}^s-\underline{f}^s\right)}{\sum_{s=1}^{M}\bar{f}^s\sum_{s=1}^{M}\underline{f}^s}\times\frac{\sum_{s=1}^{M}\underline{f}^s\left(a^s-a^1\right)\sum_{s=1}^{M}\bar{f}^s\left(a^M-a^s\right)}{\sum_{s=1}^{M}\underline{f}^s\left(a^s-a^1\right)+\sum_{s=1}^{M}\bar{f}^s\left(a^M-a^s\right)}\right] \tag{71}$$

$$\bar{y}_r(\mathbf{x}')=\underline{y}_r(\mathbf{x}')+\left[\frac{\sum_{s=1}^{M}\left(\bar{f}^s-\underline{f}^s\right)}{\sum_{s=1}^{M}\bar{f}^s\sum_{s=1}^{M}\underline{f}^s}\times\frac{\sum_{s=1}^{M}\bar{f}^s\left(b^s-b^1\right)\sum_{s=1}^{M}\underline{f}^s\left(b^M-b^s\right)}{\sum_{s=1}^{M}\bar{f}^s\left(b^s-b^1\right)+\sum_{s=1}^{M}\underline{f}^s\left(b^M-b^s\right)}\right]. \tag{72}$$

Observe that the four bounds in (69)–(72) can be computed without having to perform TR. Wu and Mendel then approximate the type-reduced set, as

$$[y_l(\mathbf{x}'),y_r(\mathbf{x}')]\approx[\hat{y}_l(\mathbf{x}'),\hat{y}_r(\mathbf{x}')]=\left[\frac{\underline{y}_l(\mathbf{x}')+\bar{y}_l(\mathbf{x}')}{2},\frac{\underline{y}_r(\mathbf{x}')+\bar{y}_r(\mathbf{x}')}{2}\right] \tag{73}$$

and approximate the output of the IT2 FLS as

$$y(\mathbf{x}')\approx\hat{y}(\mathbf{x}')=\frac{1}{2}\left[\frac{\underline{y}_l(\mathbf{x}')+\bar{y}_l(\mathbf{x}')}{2}+\frac{\underline{y}_r(\mathbf{x}')+\bar{y}_r(\mathbf{x}')}{2}\right] \tag{74}$$

[instead of as $(y_l(\mathbf{x}')+y_r(\mathbf{x}'))/2$]. Therefore, *by using the WM UBs, one obtains both an approximate type-reduced set as well as a defuzzified output*. In a real-time application, only $\hat{y}(\mathbf{x}')$ is computed.

Lynch *et al.* [33] replace all of the IT2 Mamdani FLS computations with those in (69)–(74), i.e., (69)–(74) are their IT2 FLS.

[28]Note, e.g., that in (65) {LMF, left} refers to the fact that this centroid only uses lower MFs of the firing interval and left-end point values of the consequent set centroid. In addition, in (65)-(68) and (71) and (72), $\underline{f}^s(\mathbf{x}')$ and $\overline{f}^s(\mathbf{x}')$ have been shortened to $\underline{f}^s$ and $\overline{f}^s$, respectively.

[29]The derivations of (65)–(73) in [69] are for COS TR; however, [69, Tab. V] explains how these equations can be used for other kinds of TR, one of which, of course, is Centroid TR. Appendix II in that paper gives the simpler forms of (65)-(73) for the centroid. This means, of course, that the WM UBs can be used for both COS and Centroid TR, which makes them unique among the simplifications that are described in this section.

2) Proposed General Type-2 Fuzzy Logic System Results: Because this paper is already getting long, I will only give the gist of how the WM UBs can be used for a GT2 FLS:

1) For each of the M rules, compute its firing interval $F_\alpha^s(\mathbf{x}')$ for level α, exactly as in (50).
2) Compute $\alpha-$level WM UB outputs, $y_{WM,\alpha}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ by using (69)-(72) and (74) for the k horizontal slices.
3) Defuzzify the $y_{WM,\alpha}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ by creating a spike of amplitude α_j for $y_{WM,\alpha_j}(\mathbf{x}')$, and computing $y_{WM}(\mathbf{x}')$, as:

$$y_{\mathrm{WM}}(\mathbf{x}') = \sum_{j=1}^{k} \alpha_j y_{\mathrm{WM},\alpha_j}(\mathbf{x}') \Big/ \sum_{j=1}^{k} \alpha_j \tag{75}$$

B. Nie–Tan Simplification for a General Type-2 Mamdani Fuzzy Logic System

1) Interval Type-2 Fuzzy Logic System Results: Nie and Tan [54] defuzzify an IT2 FS $\tilde{B}$ by computing the COG of the average of its lower and upper MFs, i.e.,

$$\begin{aligned} y_{\mathrm{NT}}(\mathbf{x}') &= \mathrm{COG}\left\{\frac{1}{2}\left[\underline{\mu}_{\tilde{B}}(y|\mathbf{x}') + \bar{\mu}_{\tilde{B}}(y|\mathbf{x}')\right]\right\} \\ &= \frac{\sum_{i=1}^{N} y_i \left[\underline{\mu}_{\tilde{B}}(y_i|\mathbf{x}') + \bar{\mu}_{\tilde{B}}(y_i|\mathbf{x}')\right]}{\sum_{i=1}^{N} \left[\underline{\mu}_{\tilde{B}}(y_i|\mathbf{x}') + \bar{\mu}_{\tilde{B}}(y_i|\mathbf{x}')\right]} \end{aligned} \tag{76}$$

Recently, Mendel and Liu [45] have proven that $y_{\mathrm{NT}}(\mathbf{x}')$ is a first-order approximation to the actual defuzzified value of $\tilde{B}$, $m_{\tilde{B}}(\mathbf{x}')$, where

$$m_{\tilde{B}}(\mathbf{x}') = [c_l(\tilde{B}|\mathbf{x}') + c_r(\tilde{B}|\mathbf{x}')]/2 \tag{77}$$

and $c_l(\tilde{B}|\mathbf{x}')$ and $c_r(\tilde{B}|\mathbf{x}')$ are the left and right end-points of the actual centroid. More specifically, they prove

$$y_{\mathrm{NT}}(\mathbf{x}') \approx m_{\tilde{B}}(\mathbf{x}') + \delta(\tilde{B}|\mathbf{x}') \tag{78}$$

where a formula for $\delta(\tilde{B}|\mathbf{x}')$ is given in their paper but is not needed for this paper. Examples given in [45] show that $\delta(\tilde{B}|\mathbf{x}')$ is quite small; therefore, a very reasonable approximation to $m_{\tilde{B}}(\mathbf{x}')$ is $y_{NT}(\mathbf{x}')$.

In order to apply (76) in an IT2 FLS, one must first aggregate all fired-rule output sets, as in (38), so as to obtain the IT2 FS $\tilde{B}$. Such an IT2 FS is not computed for COS TR.

2) Proposed General Type-2 Fuzzy Logic System Results: The gist of how the Nie–Tan (NT) simplification can be used for a GT2 FLS is as follows:

1) For each of the M rules, compute its firing interval $F_\alpha^s(\mathbf{x}')$ for level α, exactly as in (50).
2) For each of the M rules compute a fired-rule horizontal slice $\tilde{B}_\alpha^s$, exactly as in (52).
3) Compute the aggregated output horizontal slice $\tilde{B}_\alpha$, exactly as in (53).
4) Compute $\alpha-$level NT outputs, $y_{NT,\alpha}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ by using (76) at each $\alpha-$level.
5) Defuzzify the $y_{NT,\alpha}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ by creating a spike of amplitude α_j for $y_{NT,\alpha_j}(\mathbf{x}')$, and computing $y_{\mathrm{NT}}(\mathbf{x}')$, as

$$y_{\mathrm{NT}}(\mathbf{x}') = \sum_{j=1}^{k} \alpha_j y_{\mathrm{NT},\alpha_j}(\mathbf{x}') \Big/ \sum_{j=1}^{k} \alpha_j \tag{79}$$

Referring to Fig. 9(a), Step 4 replaces the blocks which compute the type-reduced sets at level-α that appear just before the Defuzzifier block, resulting in a big savings in computation, and (79) is used in the Defuzzifier block.

C. General Type-2 m–n Takagi—Sugeno–Kang Fuzzy Logic System

1) Interval Type-2 Fuzzy Logic System Results: Biglarbegian *et al*. [2], [3] have proposed the following "IT2 *m-n* TSK FLS," whose output, $y_{mn}(\mathbf{x}')$, is defined as[30]

$$y_{mn}(\mathbf{x}') \equiv m\frac{\sum_{s=1}^{M} y_s \underline{f}^s(\mathbf{x}')}{\sum_{s=1}^{M} \underline{f}^s(\mathbf{x}')} + n\frac{\sum_{s=1}^{M} y_s \bar{f}^s(\mathbf{x}')}{\sum_{s=1}^{M} \bar{f}^s(\mathbf{x}')}. \tag{80}$$

In (80), $\underline{f}^s(\mathbf{x}')$ and $\bar{f}^s(\mathbf{x}')$ are computed using (36), y_s is a consequent parameter, and M denotes the number of rules. Additionally, there is no requirement that $m + n = 1$, and m and n are optimized during a tuning procedure during the design of the FLS.

Equation (80) can only be used when one does not begin with an FOU, as in COS TR. To-date, it is the only IT2 FLS architecture that has been used for theoretical studies of stability and robustness in a FLC application, because (74) is still too complicated to do this, and (76) has yet to be tried for this.

2) Proposed General Type-2 Takagi—Sugeno–Kang Fuzzy Logic System Results: The gist of how the *m–n* TSK architecture can be used for a GT2 TSK FLS is as follows:

1) For each of the M rules, compute its firing interval $F_\alpha^s(\mathbf{x}')$ for level-α, exactly as in (50).
2) Decide if the same or different values of m, n and y_s will be used for each α.
3) Compute $\alpha-$level *m-n* outputs, $y_{mn,\alpha_j}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ by using (80) at each $\alpha-$level.
4) Defuzzify the $y_{mn,\alpha_j}(\mathbf{x}')(\alpha = \alpha_1, \ldots, \alpha_k)$ by creating a spike of amplitude α_j for $y_{mn,\alpha_j}(\mathbf{x}')$, and computing $y_{mn}(\mathbf{x}')$, as

$$y_{mn}(\mathbf{x}') = \sum_{j=1}^{k} \alpha_j y_{mn,\alpha_j}(\mathbf{x}') \Big/ \sum_{j=1}^{k} \alpha_j \tag{81}$$

D. Comment

In all of the proposed simplified GT2 FLSs, one must still choose k, the number of horizontal slices. On the one hand, it may be argued that greater accuracy in approximating a GT2 FS

[30] It is explained in [2] how (80) can be interpreted as a truncated version of (74); hence, one can say that (80) represents an approximation of an approximation. Observe, also, that the first term can be interpreted as the sum of (65) and (66), in which $y_s \equiv a^s + b^s$, and the second term can be interpreted as the sum of (67) and (68), in which y_s is also $a^s + b^s$.

TABLE III
EXAMPLE OF NUMBER OF DESIGN PARAMETERS IN A GT2 FLS; THERE ARE p ANTECEDENTS FOR EACH OF THE M RULES AND k HORIZONTAL SLICES; ANTECEDENT AND CONSEQUENT FOUS ARE GAUSSIAN WITH UNCERTAIN MEAN $m \in [m_1, m_2]$ AND HAVE SYMMETRICAL TRAPEZOID SECONDARY MFS; FOUR PARAMETERS PER GT2 FS, NAMELY $\theta = \{m_1, m_2, \sigma, w\}$

GT2 FLS	Antecedent parameters	Consequent parameters	Subtotal	Approximation+ Defuzzification	Weighted Average of End Points Defuzzification[d]
1. Mamdani + Centroid TR	$4pM$	$4M$	$4M(p+1)$	—	k
2. Mamdani + COS TR	$4pM$	$4M$[a]	$4M(p+1)$	—	k
3. TSK	$4pM$	pM	$5pM$	—	k
4. WM UB (Mamdani)	$4pM$	$2Mk$[b]	$2M(2p+k)$	—	k
5. NT (Mamdani)	$4pM$	$4M$	$4M(p+1)$	—	k
6. *m-n* (TSK)	$4pM$	$(M+2)k$[c]	$M(4p+k)+2k$	—	k

[a] For an IT2 FLS it is very common to replace the consequent's FOU by its two centroid end-points (e.g., [35]); doing this for a GT2 FLS is not parsimonious (for $k > 2$) because it would have to be done for each of the k horizontal slices.

[b] There are two a^s and b^s parameters for each of the k horizontal slices.

[c] This assumes m, n and the M y_s are different for each of the k horizontal slices.

[d] We are assuming, optionally, that α_j is replaced by weight ω_j $(j = 1,...,k)$ in each of the average end-points defuzzification formulas.

will be achieved by choosing a large value for k, because doing this discretizes $\alpha \in [0, 1]$ more finely. On the other hand, it may be argued that greater accuracy in approximating a GT2 FS is not the main issue in an FLS. For a GT2 FLS, the main issue is achieving better application-related performance than can be obtained by using an IT2 FLS (e.g., in FLC, that might be faster rise time and less overshoot, or smaller integral time absolute error).

An interesting possibility for $y_{WM}(\mathbf{x}'), y_{NT}(\mathbf{x}')$, and $y_{mn}(\mathbf{x}')$ is to replace α_j in each of their formulas by new design parameters, ω_j $(j = 1, \ldots, k)$, that are optimized during the design of the GT2 FLS. One could begin with a small value for k (e.g., 2) and increase it by 1 to see how much improved performance is obtained as k increases. Finding the smallest number of horizontal slices that lead to acceptable performance should be one of the design goals when designing (optimizing) a GT2 FLS, a topic that we turn to next.

IX. OPTIMIZATION OF A GENERAL TYPE-2 FUZZY LOGIC SYSTEM

A. Design Parameters

It is quite common for the parameters of an IT2 FLS to be optimized (tuned, learned) during its design phase.[31] The optimized parameters are then fixed during its operational phase, unless continued adaptation is required, in which case online changes to parameters take place. Consequently, we propose that the parameters of a GT2 FLS also be optimized during its design phase.

The parameters of a GT2 FLS are in the antecedents, consequent, and (possibly) defuzzifier. One example of how many parameters there can be for all six of the GT2 FLSs that have been described above is given in Table III. We have chosen symmetrical trapezoid secondary MFs because, as mentioned in Section IV, when $w = 0$, the trapezoid reduces to a square well and *the GT2 FS reduces to an IT2 FS*. Why this fact is so important will be made clear shortly.

Observe, from Table III, that: 1) the number of antecedent parameters is the same for all six GT2 FLSs; 2) it is the number of consequent parameters that varies from one GT2 FLS to another; 3) Approximation + Defuzzification does not introduce any new design parameter; and 4) Weighted Average + End Points Defuzzification introduces the same number of parameters as the number of horizontal slices—k—for all six GT2 FLSs.

Our objective in showing the number of design parameters is not to recommend one GT2 FLS over another on the basis of these numbers; it is only to make the reader focus on where the parameters are located.

B. Design by Optimization

During the design of a GT2 FLS, one frequently sets up a mathematical objective function, $J(\boldsymbol{\phi})$, that depends upon the design parameters, $\boldsymbol{\phi}$. The elements of $\boldsymbol{\phi}$ include all of the antecedent and consequent MF parameters as well as any defuzzification parameters (if there are any). A specific illustration of $\boldsymbol{\phi}$ is provided in Example 2. $J(\boldsymbol{\phi})$ is a nonlinear function of $\boldsymbol{\phi}$, and therefore, some sort of mathematical programming approach has to be used to optimize it. There are many different kinds of optimization algorithms that can be used to do this. For the rest of this section, we assume that $\boldsymbol{\phi}$ has $N_{\boldsymbol{\phi}}$ elements.

C. Why Gradient-Based Optimization Algorithms Should Not Be Used

Until recently, most designs of IT2 FLSs used gradient-based optimization algorithms that require mathematical formulas for derivatives of $J(\boldsymbol{\phi})$ with respect to each of the design parameters. As stated in[32] [36], we have the following:

"Generally, it is much more complicated to compute such derivatives for an IT2 FLS than it is for a T1 FLS, because:

[31] In this paper we are focusing only on the optimization of MF parameters. Choosing the number of rules (M), MF shapes and t-norms are other important design issues but are not covered herein.

[32] It was only when I was preparing a solution manual to 2001 textbook [35], and I was working out the solution to Problem 10–18, that I realized how complicated it was to compute these derivatives. This led to [36].

1) in an IT2 FLS, the design parameters appear in *upper* and *lower* MFs, whereas in a T1 FLS, they appear in a single MF; and 2) in an IT2 FLS, type-reduction establishes the two parameters $L(\mathbf{x}')$and $R(\mathbf{x}')$ (the switch points) which in turn establish the upper and lower firing-interval MFs that are used to compute the left and right end points of the type-reduced set. There is no type-reduction in a T1 FLS."

When EKM algorithms (or any of the other comparable algorithms) are used to compute the COS type-reduced set, they require (without exception) that the upper and lower firing intervals must first be reordered in an increasing order. Usually, the orderings are different for the upper and lower firing intervals. Derivative computations require that the upper and lower firing intervals must then be put back into their original rule-ordering so that one knows exactly where the MFs are, because the parameters in $\boldsymbol{\phi}$ occur in specific MFs. Keeping track of where the parameters are requires using permutation matrices, and detailed equations are provided in [36]. It is very easy to make mistakes in computing the required derivatives. Not only that, but one must also include tests on the primary variables, because lower and upper MFs can have different formulas for different subranges of the primary variables.

Therefore, if computing derivatives was very difficult for IT2 FLSs, it is considerably more difficult for GT2 FLSs. This is because not only must such derivatives be computed for each horizontal slice, where the same secondary parameter (w) appears in all of the horizontal slices [e.g., see (22) and (23)], but they must also be computed for the defuzzification method, in which the horizontal slices become coupled.

Note, also, that gradient-based optimization algorithms are not globally convergent, and therefore, the solution one obtains by using them to optimize $J(\boldsymbol{\phi})$ will only be a local extremum. Of course, there are strategies for repeating the optimization by randomly rechoosing initial values for the parameters, but there is no guarantee that even doing this will provide the global extremum.

In conclusion, my recommendation is to not use gradient-based optimization algorithms for the designs of GT2 FLSs.

D. Derivative-Free Optimization Algorithms

Fortunately, there now are globally convergent iterative search algorithms that do not require any derivatives, e.g., simulated annealing, GA, PSO, QPSO, ant colony optimization, etc. I strongly advocate using any of these instead of gradient-based algorithms. Many of these have already been applied to the designs of IT2 FLSs (e.g., [7], [20], [34], and [53]). The one that we are especially fond of is QPSO (see, e.g., [59], [66], [71], and [72]).

QPSO is a globally convergent [66] iterative search algorithm that does not use derivatives, generally outperforms the original PSO [25], [65], and has fewer parameters to control. It is a population-based optimization technique where a population is called a *swarm* that contains a set of different particles. Each *particle* represents a possible solution to an optimization problem. The position of each particle is updated (in each QPSO iteration) by using its most recent own best solution, mean of the personal best positions of all particles, and the global best solution found by all particles so far.

QPSO finds optimized $\boldsymbol{\phi}$ based on the following criterion: $\min_{\boldsymbol{\phi}_m} J(\boldsymbol{\phi}_m)$. The *current position* (vector) of the mth particle is defined, as $(m = 1, \ldots, N_m)$:

$$\boldsymbol{\phi}_m = col(\boldsymbol{\phi}_{m,1}, \boldsymbol{\phi}_{m,2}, \ldots, \boldsymbol{\phi}_{m,N_{\boldsymbol{\phi}}}). \tag{82}$$

A *particle best position* (i.e., the position that produces the minimal value of $J(\phi)$ over the entire history of that particle) $\mathbf{p}_m = col(p_{m,1}, p_{m,2}, \ldots, p_{m,N_{\boldsymbol{\phi}}})$ is computed as $(t = 1, \ldots, N_{\boldsymbol{\phi}})$:

$$p_{m,t}(g+1) = \eta \times p_{m,t}(g) + (1-\eta) \times p_{g\text{best},t}(g) \tag{83}$$

where $g = 1, \ldots, G-1$ is the *index of a generation* (iteration), $p_{m,t}(1)$ is initialized by $\boldsymbol{\phi}_{m,t}(1)$, η is a random variable uniformly distributed in (0,1], and $\mathbf{p}_{g\text{best}}(g)$ [whose components are $p_{g\text{best},t}(g)$] denotes the *global best* (*gbest*) *position* found in the history of the entire swarm, i.e. $(m = 1, \ldots, N_m)$:

$$\mathbf{p}_{g\text{best}}(g) = \arg \min_{\mathbf{p}_m(g), \forall m = 1 \ldots, N_m} J(\mathbf{p}_m(g)). \tag{84}$$

A global point, the *mean best position* of the population is introduced into QPSO; it is denoted as $\mathbf{m}(g)$ and is defined as the sample mean of the $\mathbf{p}_m(g)$ positions of all N_m particles, i.e.,

$$\mathbf{m}(g) = \frac{1}{N_m} \sum_{m=1}^{N_m} \mathbf{p}_m(g). \tag{85}$$

At the end of each generation, a new position of a particle is obtained as $(t = 1, \ldots, N_{\boldsymbol{\phi}})$:

$$\boldsymbol{\phi}_{m,t}(g+1) = p_{m,t}(g+1) \pm \beta |m_t(g) - \boldsymbol{\phi}_{m,t}(g)| \ln(1/\rho) \tag{86}$$

where parameter β, called the *contraction–expansion coefficient*, can be tuned to control the convergence speed of the algorithm, and ρ is also a random variable uniformly distributed in (0,1]. In (86), the plus and minus signs are randomly selected to generate the new position of a particle.

Pseudocode for QPSO is given in Table IV. Appendix A walks the reader through this pseudocode.

E. Guaranteed Performance Improvement

In Section I, I made the following bold statement: "Using IT2 FSs in an FLS has the potential to provide better (and certainly no worse) performance for an FLS than using T1 FSs, and using GT2 FSs in an FLS has the potential to provide better (and certainly no worse) performance for an FLS than using IT2 FSs, which is why there has been so much interest in IT2 FLSs and GT2 FLSs." Using QPSO lets us do this by using the following *design procedure*: 1) Design a T1 FLS by optimizing its parameters using QPSO; 2) design an IT2 FLS by optimizing its parameters using QPSO in which one particle is associated with the just designed T1 FLS; and 3) design a GT2 FLS by optimizing its parameters using QPSO in which one particle is associated with the just designed IT2 FLS.

Theorem 2: (a) By virtue of the QPSO algorithm, the performance of the optimized IT2 FLS cannot be worse than that of

TABLE IV
PSEUDOCODE FOR QPSO AS USED IN OPTIMAL DESIGNS OF AN IT2 FLS OR A GT2 FLS

Initialize $\boldsymbol{\phi}_1(1)$ as either a T1 FLS or an IT2 FLS particle, and all other $\boldsymbol{\phi}_m(1)$ randomly $(m = 2,...,N_m)$
Set $\mathbf{p}_m(1) = \boldsymbol{\phi}_m(1)$ $(m = 1,...,N_m)$
For g = 1 to G-1
  Calculate $\mathbf{m}(g) = \frac{1}{N_m}\sum_{m=1}^{N_m}\mathbf{p}_m(g)$
  Calculate $J(\mathbf{p}_m(g))$ $(m = 1,...,N_m)$
  $\mathbf{p}_{gbest}(g) = arg \min_{\mathbf{p}_m(g),\,\forall m=1,...,N_m} J(\mathbf{p}_m(g))$
  for m = 1 to N_m (number of particles)
  Calculate $J(\boldsymbol{\phi}_m(g))$
    If $J(\boldsymbol{\phi}_m(g)) < J(\mathbf{p}_m(g))$
      $\mathbf{p}_m(g) = \boldsymbol{\phi}_m(g)$
    end if
    for t = 1 to N_ϕ (number of components in each particle)
      $\eta = rand(0,1)$
      $p_{mJ}(g+1) = \eta \times p_{mJ}(g) + (1-\eta) \times p_{gbestJ}(g)$
      $\rho = rand(0,1)$
      if $rand(0,1) > 0.5$ **then**
        $\phi_{mJ}(g+1) = p_{mJ}(g+1) - \beta\,|m_t(g) - \phi_{mJ}(g)|\ln(1/\rho)$
      else
        $\phi_{mJ}(g+1) = p_{mJ}(g+1) + \beta\,|m_t(g) - \phi_{mJ}(g)|\ln(1/\rho)$
      end if
    end for
  end for
end for

the optimized T1 FLS. (b) By virtue of the QPSO algorithm, the performance of the optimized GT2 FLS cannot be worse than that of the optimized IT2 FLS.

Proof: See Appendix A. Note also that [59] proves that QPSO is a form of a contraction mapping and that it converges in probability.

By this three-step systematic design approach, it is not possible for the performance of an optimized IT2 FLS to be worse than that of an optimized T1 FLS, nor is it possible for the performance of an optimized GT2 FLS to be worse than that of an optimized IT2 FLS.[33] This does not mean that the performance of either the optimized IT2 FLS or optimized GT2 FLS will be *significantly better* than that of either the optimized T1 FLS or optimized IT2 FLS. There is no analysis that is available to-date that focuses on such relative performance improvements. Of course, relative improvements are very application dependent because objective function $J(\phi)$ is application dependent.

By the way, this kind of guaranteed performance improvement can also be achieved with a gradient-based algorithm if after each iteration the performance of the IT2 FLS (or GT2 FLS) is compared with that obtained from a T1 FLS (or IT2 FLS). A new set of parameters is accepted only if the performance from the IT2 FLS (or GT2 FLS) is better than that of the T1 FLS (or IT2 FLS).

Example 1: In Step 2 of the above design procedure, one particle is associated with a just designed T1 FLS. Suppose, for example, that the IT2 FLS is Mamdani + COS TR, and the antecedent and consequent FOUs are the ones stated in the title of Table III. The structure of a particle for such an IT2 FLS is[34] given in (87). The T1 particle, which must be of the same length as this IT2 particle, begins with (87) and can be expressed as in (88). Observe, in (88), that 1) all of the MF parameters are taken from the optimized T1 FLS design; (2) by setting the values for the endpoints of the means for an antecedent FOU to be the same, and the end points of the centroids for a consequent to be the same, the uncertainty in each rule about the means for all p antecedents and its consequent has disappeared; and (3) in this way, it is straightforward to embed a T1 particle into an IT2 particle.

$$\phi_{\mathrm{IT2}} = \mathrm{col}\left(\underbrace{\overbrace{m_{11}^1, m_{12}^1, \sigma_1^1}^{\text{Antecedent 1}}, \ldots, \overbrace{m_{p1}^1, m_{p2}^1, \sigma_p^1}^{\text{Antecedent } p}, \overbrace{l_{\tilde{G}^1}, r_{\tilde{G}^1}}^{\text{Consequent}}}_{\text{Rule 1}}; \ldots; \underbrace{\overbrace{m_{11}^M, m_{12}^M, \sigma_1^M}^{\text{Antecedent 1}}, \ldots, \overbrace{m_{p1}^M, m_{p2}^M, \sigma_p^M}^{\text{Antecedent } p}, \overbrace{l_{\tilde{G}^M}, r_{\tilde{G}^M}}^{\text{Consequent}}}_{\text{Rule } M}\right) \tag{87}$$

$$\phi_{T1} = \mathrm{col}\left(\underbrace{\overbrace{m_1^1, m_1^1, \sigma_1^1}^{\text{Antecedent 1}}, \ldots, \overbrace{m_p^1, m_p^1, \sigma_p^1}^{\text{Antecedent } p}, \overbrace{y_{G^1}, y_{G^1}}^{\text{Consequent}}}_{\text{Rule 1}}; \ldots; \underbrace{\overbrace{m_1^M, m_1^M, \sigma_1^M}^{\text{Antecedent 1}}, \ldots, \overbrace{m_p^M, m_p^M, \sigma_p^M}^{\text{Antecedent } p}, \overbrace{y_{G^M}, y_{G^M}}^{\text{Consequent}}}_{\text{Rule } M}\right) \tag{88}$$

Example 2: In Step 3 of the above design procedure, one particle is associated with a just designed IT2 FLS. Suppose, for example, (see Table III) that the GT2 FLS is Mamdani + COS TR, weighted average of end points defuzzification is used, and the antecedent and consequent FOUs are the ones stated in the title of Table III. The structure of a particle for such a GT2 FLS is given in (89). The IT2 particle, which must be of the same length as this GT2 particle, begins with (89), and can be expressed as in (90). Observe in (90) that 1) all of the MF parameters are taken from the optimized IT2 FLS design; 2) by setting all of the secondary MF w-parameters equal to 0, an IT2 FS is embedded into a GT2 FS [see the discussion below (20) and (21)]; 3) because an IT2 FS is a GT2 FS all of whose secondary MFs equal 1, all k of the horizontal slices of a GT2 FS are the same, which is why all k of the defuzzifier parameters

[33] If a FLS cannot be designed by means of optimization but can only be designed by trial and error, then it is possible that the performance of the IT2 FLS may be worse than that of a T1 FLS, and the performance of a GT2 FLS may be worse than that of an IT2 FLS. This occurs because it may not be humanly possible to try all possible combinations of the design parameters, and is therefore the result of an incomplete design procedure. One must be very cautious about drawing conclusions from such designs, since they are analogous to a poorly designed experiment in statistics.

[34] In (87), we have replaced each consequent's FOU by its two centroid endpoints (see footnote a to Table III), and in (88), we have replaced each consequent's T1 FS by its COG location, as is done when COS defuzzification is used.

have been set equal to the same value of 1; and 4) in this way, it is straightforward to embed an IT2 particle into a GT2 particle.

$$\begin{aligned}\boldsymbol{\phi}_m = \; & \mathrm{col}(\overbrace{\underbrace{m_{11}^1, m_{12}^1, \sigma_1^1, w_1^1}_{\text{Antecedent 1}}, \ldots, \underbrace{m_{p1}^1, m_{p2}^1, \sigma_p^1, w_p^1}_{\text{Antecedent } p}, \underbrace{m_{1c}^1, m_{2c}^1, \sigma_c^1, w_c^1}_{\text{Consequent}}; \ldots}^{\text{Rule1}} \\ & \overbrace{\underbrace{m_{11}^M, m_{12}^M, \sigma_1^M, w_1^M}_{\text{Antecedent 1}}, \ldots, \underbrace{m_{p1}^M, m_{p2}^M, \sigma_p^M, w_p^M}_{\text{Antecedent } p},}^{\text{Rule}M} \\ & \underbrace{\underbrace{m_{1c}^M, m_{2c}^M, \sigma_c^M, w_c^M}_{\text{Consequent}}; \underbrace{\omega_1, \ldots, \omega_k}_{\text{Defuzzifier}}}_{\text{RuleM}}) \end{aligned} \tag{89}$$

$$\begin{aligned}\boldsymbol{\phi}_{\mathrm{IT2}} = \; & \mathrm{col}(\overbrace{\underbrace{m_{11}^1, m_{12}^1, \sigma_1^1, 0}_{\text{Antecedent 1}}, \ldots, \underbrace{m_{p1}^1, m_{p2}^1, \sigma_p^1, 0}_{\text{Antecedent } p}, \underbrace{m_{1c}^1, m_{2c}^1, \sigma_c^1, 0}_{\text{Consequent}}; \ldots}^{\text{Rule1}} \\ & \overbrace{\underbrace{m_{11}^M, m_{12}^M, \sigma_1^M, 0}_{\text{Antecedent 1}}, \ldots, \underbrace{m_{p1}^M, m_{p2}^M, \sigma_p^M, 0}_{\text{Antecedent } p}, \underbrace{m_{1c}^M, m_{2c}^M, \sigma_c^M, 0}_{\text{Consequent}};}^{\text{Rule}M} \\ & \underbrace{1, \ldots, 1}_{\text{Defuzzifier}}) \end{aligned} \tag{90}$$

X. Conclusion

This paper has 1) explained four different mathematical representations for GT2 FSs; 2) demonstrated that for the optimal design of a GT2 FLS, one should use the vertical-slice representation of its GT2 FSs, because it is the only one of the four mathematical representations of a GT2 FS that is parsimonious; 3) shown how to obtain set theoretic and other operations for GT2 FSs using T1 FS mathematics ($\alpha-$cuts play a central role); 4) reviewed IT2 Mamdani and TSK FLSs so that their mathematical operations can be easily used in a GT2 FLS; 5) provided all of the formulas that describe both GT2 Mamdani and TSK FLSs (these are implemented as a collection of horizontal slices, each of which acts like an IT2 FLS); 6) explained why center-of sets type-reduction should be favored for a GT2 FLS (over centroid type- reduction); 7) provided three simplified GT2 FLSs (two are for GT2 Mamdani FLSs and one is for a GT2 TSK FLS) all of which bypass type- reduction and are generalizations of their IT2 FLS counterparts to GT2 FLSs; 8) explained why gradient-based optimization should not be used to optimally design a GT2 FLS; 9) explained why and how derivative-free optimization algorithms should be used to optimally design a GT2 FLS; and 10) provided a three-step approach for designing FLSs, from T1 to IT2 to GT2 (it is now incumbent upon authors of papers about GT2 FLSs to compare their results with those for optimally-designed T1 and IT2 FLSs), each of which uses a QPSO algorithm for which (by virtue of the QPSO algorithm) the performance for the IT2 FLS cannot be worse than that of the T1 FLS, and the performance for the GT2 FLS cannot be worse than that of the IT2 FLS.

GT2 FLSs and IT2 FLSs were introduced by this author and his students more than 15 years ago [21], [24], [28], and during this time, much has been learned about them, especially about IT2 FLSs and different ways to represent a GT2 FS. It is hoped that this tutorial paper will make GT2 FLSs more accessible to FL researchers and practitioners and that it will also expedite the research about and use of GT2 FSs and FLSs.

Appendix A

Proof of Theorem 2

Because the proof of Part (b) is the same as the proof of Part (a), modulo some simple changes in notation, we only provide the proof of Part (a).

We want to show that, because the QPSO algorithm retains the best particle, this particle must have a performance that is at least as good as that of the optimized T1 FLS. To demonstrate that this is true we use the pseudocode in Table IV in which the first particle for the optimized design of an IT2 FLS is the optimized T1 FLS [i.e., $\mathbf{p}_1(1) = \boldsymbol{\phi}_1(1) = \boldsymbol{\phi}_{TI}$] and the remaining $N_m - 1$ particles are chosen randomly for the IT2 FLS. Our approach is to go through the entire pseudo-code two times in order to clearly understand how or if the T1 FLS particle can survive the QPSO iterations.

A.1 Quantum Particle Swarm Optimization Iteration 1 ($g = 1$)

$\mathbf{m}\,(1)$ is computed as

$$\mathbf{m}(1) = \frac{1}{N_m}\left[\boldsymbol{\phi}_{TI} + \sum_{m=2}^{N_m} \mathbf{p}_m(1)\right] \tag{A-1}$$

then, $J(\mathbf{p}_m(1))$ is computed for $m = 1, \ldots, N_m$, after which, $\mathbf{p}_{g\mathrm{best}}(1)$ is computed as

$$\mathbf{p}_{g\mathrm{best}}(1) = arg \min_{\mathbf{p}_m(1), \forall m = 1, \ldots, N_m} J(\mathbf{p}_m(1)) \tag{A-2}$$

Two cases are now possible: $\mathbf{p}_{g\mathrm{best}}(1) = \mathbf{p}_1(1) = \boldsymbol{\phi}_{TI}$ or $\mathbf{p}_{g\mathrm{best}}(1) \neq \mathbf{p}_1(1)$.

$$\mathbf{p}_{g\mathrm{best}}(1) = \mathbf{p}_1(1) = \boldsymbol{\phi}_{TI}$$

For $g = 1$ it is always true, for *all particles*, that $J(\boldsymbol{\phi}_m(1)) = J(\mathbf{p}_m(1))$ because $\mathbf{p}_m(1) = \boldsymbol{\phi}_m(1)$; hence, no changes are made as a result of the test "If $J(\boldsymbol{\phi}_m(1)) < J(\mathbf{p}_m(1))$, then $\mathbf{p}_m(1) = \boldsymbol{\phi}_m(1)$." For each component of *Particle 1* ($m = 1$), regardless of the value chosen for η, we therefore find that ($t = 1, \ldots, N_{\boldsymbol{\phi}}$):

$$\begin{aligned} p_{1,t}(2) &= \eta \times p_{1,t}(1) + (1-\eta) \times p_{g\mathrm{best},t}(1) \\ &= \eta \times p_{1,t}(1) + (1-\eta) \times p_{1,t}(1) = p_{1,t}(1) = \phi_{TI,t}. \end{aligned} \tag{A-3}$$

This means that $\mathbf{p}_1(2) = \boldsymbol{\phi}_{TI}$.

Note, also that 1) the $\mathbf{p}$-vectors for particles $m = 2, \ldots, N_m$ change from $\mathbf{p}_m(1)$ to $\mathbf{p}_m(2)$ by blending $\mathbf{p}_m(1)$ and $\mathbf{p}_{g\mathrm{best}}(1)$ according to $p_{m,t}(2) = \eta p_{m,t}(1) + (1-\eta)\phi_{T1,t} \neq \phi_{T1,t}$, since $p_{m,t}(1) \neq \phi_{T1,t}$; and 2) another N_m $\boldsymbol{\phi}$-particles are created by making use of both $\mathbf{m}(1)$ and the just-computed $\mathbf{p}_m(2)$ (this is also done for Particle 1).

Therefore, when $\mathbf{p}_{g\text{best}}(1) = \mathbf{p}_1(1) = \boldsymbol{\phi}_{TI}$ the T1 particle survives to be used during the next iteration of QPSO.

$$\mathbf{p}_{g\text{best}}(1) \neq \mathbf{p}_1(1).$$

For $g = 1$, it is still always true, for all particles, that $J(\boldsymbol{\phi}_m(1)) = J(\mathbf{p}_m(1))$ because $\mathbf{p}_m(1) = \boldsymbol{\phi}_m(1)$; hence, again no changes are made as a result of the test "If $J(\boldsymbol{\phi}_m(1)) < J(\mathbf{p}_m(1))$, then $\mathbf{p}_m(1) = \boldsymbol{\phi}_m(1)$." Now, for *Particle 1* $(m = 1)$

$$p_{1,t}(2) = \eta \times p_{1,t}(1) + (1-\eta) \times p_{g\text{best},t}(1) \neq p_{1,t}(1) = \phi_{\text{TI},t}. \tag{A-4}$$

This means the T1 particle does not survive to be used during the next iteration of QPSO, i.e., it is the end of the line for the T1 FLS, and the performance of the IT2 FLS is already better than that of the T1 FLS.

A.2 Quantum Particle Swarm Optimization Iteration 2 $(g = 2)$

$\mathbf{m}(2)$ is computed as

$$\mathbf{m}(2) = \begin{cases} \dfrac{1}{N_m}\left[\boldsymbol{\phi}_{TI} + \displaystyle\sum_{m=2}^{N_m} \mathbf{p}_m(2)\right], & \text{if } \mathbf{p}_{g\text{best}}(1) = \mathbf{p}_1(1) = \boldsymbol{\phi}_{TI} \\ \dfrac{1}{N_m}\displaystyle\sum_{m=1}^{N_m} \mathbf{p}_m(2), & \text{if } \mathbf{p}_{g\text{best}}(1) \neq \mathbf{p}_1(1). \end{cases} \tag{A-5}$$

It is only necessary to focus on the results from iteration 1 of QPSO for which the T1 particle survived to see how or if it can survive this second iteration of QPSO. Again, two cases are possible: (1) $\mathbf{p}_{g\text{best}}(2) = \mathbf{p}_1(2) = \boldsymbol{\phi}_{TI}$ or (2) $\mathbf{p}_{g\text{best}}(2) \neq \mathbf{p}_1(2)$.

$$\mathbf{p}_{g\text{best}}(2) = \mathbf{p}_1(2) = \boldsymbol{\phi}_{TI}$$

For $g = 2$ the T1 particle survives only if, in addition to $\mathbf{p}_{g\text{best}}(2) = \mathbf{p}_1(2) = \boldsymbol{\phi}_{TI}$, it is also true that $J(\boldsymbol{\phi}_1(2)) \not< J(\mathbf{p}_1(2)) = J(\boldsymbol{\phi}_{TI})$. It is conceivable that this can occur, in which case

$$\begin{aligned} p_{1,t}(3) &= \eta \times p_{1,t}(2) + (1-\eta) \times p_{g\text{best},t}(2) \\ &= \eta \times p_{1,t}(2) + (1-\eta) \times p_{1,t}(2) = p_{1,t}(2) = \phi_{TI,t.} \end{aligned} \tag{A-6}$$

This means that $\mathbf{p}_1(3) = \boldsymbol{\phi}_{TI}$. Therefore, when $\mathbf{p}_{g\text{best}}(2) = \mathbf{p}_1(2) = \boldsymbol{\phi}_{TI}$, it is indeed possible for the T1 Particle 1 to survive to be used during the next iteration of QPSO, although it is more difficult for it to do so in this second iteration of QPSO because for it to happen it must survive two tests.

$$\mathbf{p}_{g\text{best}}(2) \neq \mathbf{p}_1(2).$$

In this case, regardless of whether or not the test "If $J(\boldsymbol{\phi}_m(2)) < J(\mathbf{p}_m(2))$" is passed, none of the $\mathbf{p}$- or $\boldsymbol{\phi}$-particles will be the same as the T1 particle; hence, the T1 particle again does not survive to be used during the next iteration of QPSO, i.e. it is the end of the line for the T1 FLS, and the performance of the IT2 FLS is now better than that of the T1 FLS.

A.3 Further Quantum Particle Swarm Optimization Iterations

It should be clear, from our details of iterations 1 and 2 of QPSO that it is extremely difficult for the T1 particle to continue to survive. It may happen, in which case the performance of the IT2 FLS is that of the T1 FLS; however, it is highly unlikely to happen especially if the number of iterations of QPSO is made large enough.

It appears, from our brief analysis of the QPSO algorithm that it minimizes $J(\boldsymbol{\phi})$ in a monotonically nonincreasing manner.

ACKNOWLEDGMENT

The author would like to thank the following people who have (unknowingly) contributed to this paper through their past collaborations with him over the past 17 years: N. Karnik, Q. Liang, H. Wu, F. Liu, D. Wu, D. Zhai, M. Biglarbegian, M. Hao, S. Coupland, X. Liu, and R. John. Although he has not worked directly with H. Hagras and C. Wagner, their works on zSlice GT2 FLSs have been very important to the GT2 field and has had a great influence on the author's thinking about GT2 FLSs.

Also, he would like to thank the reviewers of this paper who have made some very useful suggestions that have been incorporated into the final version of this paper.

REFERENCES

[1] J. Aisbett, J. T. Rickard, and D. G. Morgenthaler, "Type-2 fuzzy sets as functions on spaces," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 4, pp. 841–844, Aug. 2010.

[2] M. Biglarbegian, W. W. Melek, and J. M. Mendel, "On the stability of interval type-2 TSK fuzzy logic control systems," *IEEE Trans. Syst., Man, Cybern., B Cybern.*, vol. 40, no. 3, pp. 798–818, Jun. 2010.

[3] M. Biglarbegian, W. W. Melek, and J. M. Mendel, "On the robustness of type-1 and interval type-2 fuzzy logic systems in modeling," *Inf. Sci.*, vol. 181, pp. 1325–1347, 2011.

[4] H. Bustince, "Indicator of inclusion grade for interval-valued fuzzy sets. Application to approximate reasoning based on interval-valued fuzzy sets," *Int. J. Approx. Reason.*, vol. 23, pp. 137–209, 2000.

[5] O. Castillo and P. Melin, *Type-2 Fuzzy Logic Theory and Applications*. Berlin, Germany: Springer-Verlag, 2008.

[6] O. Castillo and P. Melin, *Recent Advances in Interval Type-2 Fuzzy Systems*, (Springer Briefs in Applied Sciences Computational Intelligence). Heidelberg, Germany: Springer, 2012.

[7] O. Castillo and P. Melin, "A review on the design and optimization of interval type-2 fuzzy controllers," *Appl. Soft. Comput.*, vol. 12, no. 4, pp. 1267–1278, 2012.

[8] S. Coupland, "Type-2 fuzzy sets: Geometric defuzzification and type-reduction," in *Proc. IEEE Symp. Found. Comput. Intell.*, Honolulu, HI, USA, Apr. 2007, pp. 622–629.

[9] S. Coupland and R. I. John, "A new and efficient method for the type-2 meet operation," in *Proc. IEEE Conf. Fuzzy Syst.*, Budapest, Hungary, Jul. 2004, pp. 959–964.

[10] S. Coupland and R. I. John, "Towards more efficient type-2 fuzzy logic systems," in *Proc. IEEE Conf. Fuzzy Syst.*, Reno, NV, USA, May 2005, pp. 236–241.

[11] S. Coupland and R. I. John, "Geometric type-1 and type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 1, pp. 3–15, Feb. 2007.

[12] S. Coupland and R. I. John, "Geometric type-2 fuzzy sets," in *Advances in Type-2 Fuzzy Sets and Systems: Theory and Applications*, A. Sadeghian, J. M. Mendel, and H. Tahayori, Eds. New York, NY, USA: Springer, 2013.

[13] D. Dubois and H. Prade, "Operations on fuzzy numbers," *Int. J. Syst. Sci.*, vol. 9, pp. 613–626, 1978.

[14] D. Dubois and H. Prade, "Operations in a fuzzy-valued logic," *Inf. Control*, vol. 43, pp. 224–240, 1979.

[15] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*. New York, NY, USA: Academic, 1980.

[16] M. B. Gorzalczany, "A method of inference in approximate reasoning based on interval-valued fuzzy sets," *Fuzzy Sets Syst.*, vol. 21, pp. 1–17, 1987.

[17] S. Greenfield and R. I. John, "Optimized generalized type-2 join and meet operations," in *Proc. Int. Fuzzy Syst. Conf.*, London, U.K., 2007, pp. 141–146.

[18] S. Greenfield, R. I. John, and S. Coupland, "A novel sampling method for type-2 defuzzification," presented at the Annual Workshop UK, Comput. Intell., London, U.K., Sep. 2005.

[19] M. Hao and J. M. Mendel, "Similarity measures for general type-2 fuzzy sets based on the α-plane representation," *Inf. Sci.*, 2013, to be published.

[20] D. Hidalgo, P. Melin, and O. Castillo, "An optimization method for designing type-2 fuzzy inference systems based on the footprint of uncertainty using genetic algorithms," *Expert Syst. Appl.*, vol. 39, no. 4, pp. 4590–4598, 2012.

[21] N. N. Karnik and J. M. Mendel, "Introduction to type-2 fuzzy logic systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, USA, May 1998, pp. 915–920.

[22] N. N. Karnik and J. M. Mendel, "Operations on type-2 fuzzy sets," *Fuzzy Sets Syst.*, vol. 122, pp. 327–348, 2001.

[23] N. N. Karnik and J. M. Mendel, "Centroid of a type-2 fuzzy set," *Inf. Sci.*, vol. 132, pp. 195–220, 2001.

[24] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 6, pp. 643–658, Dec. 1999.

[25] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, 1995, pp. 1942–1948.

[26] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, NJ, USA: Prentice–Hall, 1995.

[27] Q. Liang and J. M. Mendel, "An introduction to type-2 TSK fuzzy logic systems," presented at IEEE Int. Conf. Fuzzy Syst., Seoul, Korea, Aug. 1999.

[28] Q. Liang and J. M. Mendel, "Interval type-2 fuzzy logic systems: Theory and design," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 4, pp. 535–550, Oct. 2000.

[29] O. Linda and M. Manic, "Importance sampling based defuzzification for general type-2 fuzzy sets," in *Proc. FUZZ-IEEE, WCCI IEEE World Congr. Comput. Intell.*, Barcelona, Spain, Jul. 2010, pp. 1943–1949.

[30] O. Linda and M. Manic, "Monotone centroid flow algorithm for type-reduction of general type-2 fuzzy sets," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 5, pp. 805–819, Oct. 2012.

[31] F. Liu, "An efficient centroid type reduction strategy for general type-2 fuzzy logic system," *Inf. Sci.*, vol. 178, pp. 2224–2236, 2008.

[32] L. A. Lucas, T. M. Centeno, and R. M. Delgado, "General type-2 inference systems: Analysis, design, and computational aspects," in *Proc. Int. Fuzzy Syst. Conf.*, London, U.K., 2007, pp. 1107–1112.

[33] C. Lynch, H. Hagras, and V. Callaghan, "Using uncertainty bounds in the design of an embedded real-time type-2 neuro-fuzzy speed controller for marine diesel engines," in *Proc. Int. Conf. Fuzzy Syst.*, Vancouver, BC, Canada, 2006, pp. 7217–7224.

[34] P. Melin, L. Astudillo, O. Castillo, F. Valdez, and M. Garcia, "Optimal design of type-2 and type-1 fuzzy tracking controllers for autonomous mobile robots under perturbed torques using a new chemical optimization paradigm," *Expert Syst. Appl.*, vol. 40, no. 8, pp. 3185–3195, 2013.

[35] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*. Upper-Saddle River, NJ, USA: Prentice–Hall, 2001.

[36] J. M. Mendel, "Computing derivatives in interval type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 12, no. 1, pp. 84–98, Feb. 2004.

[37] J. M. Mendel, "Advances in type-2 fuzzy sets and systems," *Inf. Sci.*, vol. 177, pp. 84–110, 2007.

[38] J. M. Mendel, "Type-2 fuzzy sets and systems: An overview," *IEEE Comput. Intell. Mag.*, vol. 2, no. 2, pp. 20–29, Feb. 2007.

[39] J. M. Mendel, "Comments on $\propto$-plane representation for type-2 fuzzy sets: Theory and applications'," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 1, pp. 229–230, Feb. 2010.

[40] J. M. Mendel, "On centroid calculations for type-2 fuzzy sets," *Appl. Comput. Math.*, vol. 10, no. 1, pp. 88–96, 2011.

[41] J. M. Mendel, "On the geometry of join and meet calculations for general type-2 fuzzy sets," in *Proc. Int. Conf. Fuzzy Syst.*, Taipei, Taiwan, Jun. 2011, pp. 2407–2413.

[42] J. M. Mendel, "Plotting 2-1/2-D figures for general type-2 fuzzy sets by hand or by powerpoint," in *Proc. Int. Conf. Fuzzy Syst.*, Brisbane, Australia, Jun. 2012, pp. 1490–1497.

[43] J. M. Mendel, "On KM algorithms for solving type-2 fuzzy set problems," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 13, pp. 426–446, Jun. 2013.

[44] J. M. Mendel and R. I. John, "Type-2 fuzzy sets made simple," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 117–127, Apr. 2002.

[45] J. M. Mendel and X. Liu, "Simplified interval type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, 2013, to be published.

[46] J. M. Mendel, R. I. John, and F. Liu, "Interval type-2 fuzzy logic systems made simple," *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 6, pp. 808–821, Dec. 2006.

[47] J. M. Mendel and F. Liu, "On new quasi-type-2 fuzzy logic systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Hong Kong, China, Jun. 2008, pp. 354–360.

[48] J. M. Mendel, F. Liu, and D. Zhai, "Alpha-plane representation for type-2 fuzzy sets: Theory and applications," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 5, pp. 1189–1207, Oct. 2009.

[49] J. M. Mendel and D. Wu, *Perceptual Computing: Aiding People in Making Subjective Judgments*. Hoboken, NJ, USA: Wiley, 2010.

[50] G. M. Mendez and M. H. Hernandez, "Hybrid-learning mechanism for interval A2-C1 type-2 non-singleton type-2 Takagi-Sugano-Kang fuzzy logic systems," *Inf. Sci.*, vol. 220, pp. 149–169, 2013.

[51] M. Mizumoto and K. Tanaka, "Some properties of fuzzy sets of type-2," *Inf. Contr.*, vol. 31, pp. 312–340, 1976.

[52] M. Mizumoto and K. Tanaka, "Fuzzy sets of type-2 under algebraic product and algebraic sum," *Fuzzy Sets Syst.*, vol. 5, pp. 277–290, 1981.

[53] Y. Moldonado, O. Castillo, and P. Melin, "Particle swarm optimization of interval type-2 fuzzy systems for FPGA applications," *Appl. Soft. Comput.*, vol. 13, no. 1, pp. 496–508, 2013.

[54] M. Nie and W. W. Tan, "Towards an efficient type-reduction method for interval type-2 fuzzy logic systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Hong Kong, China, Jun. 2008, pp. 1425–1432.

[55] J. Nieminen, "On the algebraic structure of fuzzy sets of type-2," *Kybernetica*, vol. 13, no. 4, 1977.

[56] A. Sadeghian, J. M. Mendel, and H. Tahayori, Eds., *Advances in Type-2 Fuzzy Sets and Systems: Theory and Applications*. New York, NY, USA: Springer, 2013.

[57] J. T. Starczewski, "Extended triangular norms on Gaussian fuzzy sets," in *Proc. 7th Conf. Eur. Society Fuzzy Logic Technol.*, Barcelona, Spain, Sep. 2005, pp. 872–877.

[58] J. T. Starczewski, "A triangular type-2 fuzzy logic system," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Vancouver, BC, Canada, Jul. 2006, pp. 7231–7238.

[59] J. Sun, X. Wu, V. Palade, W. Fang, C. H. Lai, and W. Xu, "Convergence analysis and improvements of quantum-behaved particle swarm optimization," *Inf. Sci.*, vol. 193, pp. 81–103, 2012.

[60] H. Tahayori, A. G. B. Tettamanzi, and G. D. Antoni, "Approximated type-2 fuzzy set operations," in *Proc. Int. Conf. Fuzzy Syst.*, Vancouver, BC, Canada, Jul. 2006, pp. 9042–9049.

[61] 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.

[62] C. Wagner and H. Hagras, "z Slices—towards bridging the gap between interval and general type-2 fuzzy logic," presented at the IEEE FUZZ Conf., Hong Kong, China, Jun. 2008.

[63] C. Wagner and H. Hagras, "Towards general type-2 fuzzy logic systems based on zSlices," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 4, pp. 637–660, Aug. 2010.

[64] C. Wagner and H. Hagras, "zSlices based general type-2 fuzzy sets and systems," in *Advances in Type-2 Fuzzy Sets and Systems: Theory and Applications*, A. Sadeghian, J. M. Mendel, and H. Tahayori, Eds. New York, NY, USA: Springer, 2013.

[65] X. Wang, Y. He, L. Dong, and H. Zhao, "Particle swarm optimization for determining fuzzy measures from data," *Inf. Sci.*, vol. 181, pp. 4230–4252, Oct. 2011.

[66] F. Wei, S. Jun, X. Z. -Ping, and W.-B. Xu, "Convergence analysis of quantum-behaved particle swarm optimization algorithm and study on its control parameter," *Acta Phys. Sin.*, vol. 59, no. 6, pp. 3686–3694, 2010.

[67] D. Wu, "Approaches for reducing the computational costs of interval type-2 fuzzy logic systems: Overview and comparisons," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 1, p. 80–93, Feb. 2013.

[68] D. Wu and J. M. Mendel, "Enhanced Karnik-Mendel algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 4, pp. 923–934, Aug. 2009.

[69] H. Wu and J. M. Mendel, "Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 5, pp. 622–639, Oct. 2002.

[70] D. Wu and M. Nie, "Comparison and practical implementations of type-reduction algorithms for type-2 fuzzy sets and systems," in *Proc. Int. Conf. Fuzzy Syst.*, Taipei, Taiwan, Jun. 2011, pp. 2131–2138.
[71] M. Xi, J. Sun, and W. Xu, "An improved quantum-behaved particle swarm optimization algorithm with weighted mean best position," *Appl. Math. Comput.*, vol. 205, p. 751–759, 2008.
[72] S. Yang, M. Wang, and L. Jiao, "A quantum particle swarm optimization," *Congr. Evol. Comput.*, vol. 1, p. 320–324, 2004.
[73] C.-Y. Yeh, W.-H. Roger Jeng, and S.-J. Lee, "An enhanced type-reduction algorithm for type-2 fuzzy sets," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 2, pp. 227–240, Apr. 2011.
[74] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning–1," *Inf. Sci.*, vol. 8, pp. 199–249, 1975.
[75] D. Zhai and J. M. Mendel, "Computing the centroid of a general type-2 fuzzy set by means of the centroid flow algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 3, pp. 401–422, Jun. 2011.
[76] D. Zhai and J. M. Mendel, "Uncertainty measures for general type-2 fuzzy sets," *Inf. Sci.*, vol. 181, pp. 503–518, 2011.
[77] D. Zhai and J. M. Mendel, "Enhanced centroid-flow algorithm for computing the centroid of general type-2 fuzzy sets," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 5, pp. 939–956, Oct. 2012.
[78] D. Zhai and J. M. Mendel, "Comment on 'Toward general type-2 fuzzy logic systems based on zslices," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 5, pp. 996–997, Oct. 2012.
[79] H. Zhou and H. Ying, "A method for deriving the analytical structure of a broad class of typical interval type-2 Mamdani fuzzy controllers," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 3, pp. 447–458, Jun. 2013.

Jerry M. Mendel (S'59–M'61–SM'72–F'78–LF'04) received the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, NY, USA.

He is currently a Professor of electrical engineering and systems architecting engineering with the University of Southern California, Los Angeles, CA, USA, where he has been since 1974. He has contributed to more than 550 technical papers and is author, coauthor, or editor of ten books, including *Advances in Type-2 Fuzzy Sets and Systems* (A. Sadeghian and H. Tahayori, Coeds., Springer, 2013). His current research interests include type-2 fuzzy logic systems and their applications to a wide range of problems, including smart oil field technology, computing with words, and fuzzy set qualitative comparative analysis.

He is a Distinguished Member of the IEEE Control Systems Society and a Fellow of the International Fuzzy Systems Association. He was the President of the IEEE Control Systems Society in 1986. He is a member of the Administrative Committee of the IEEE Computational Intelligence Society and was Chairman of its Fuzzy Systems Technical Committee (TC). He was also the Chairman of the Computing With Words Task Force of that TC. 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, the 2002 and 2014 TRANSACTIONS ON FUZZY SYSTEMS Outstanding Paper Awards, a 1984 IEEE Centennial Medal, an IEEE Third Millenium Medal, and a Fuzzy Systems Pioneer Award (2008) from the IEEE Computational Intelligence Society.