International Journal of Innovative
Computing, Information and Control
Volume **11**, Number **6**, December **2015**

 ISSN 1349-4198
pp. **1987–2000**

STUDIES ON CENTROID TYPE-REDUCTION ALGORITHMS FOR GENERAL TYPE-2 FUZZY LOGIC SYSTEMS

Yang Chen[1,2], Dazhi Wang[1] and Wu Ning[1,3]

[1]Institute of Power System and Power Drives
Northeastern University
No. 3-11, Wenhua Road, Heping District, Shenyang 110819, P. R. China
chenyanghanyun@163.com; wangdazhi@ise.neu.edu.cn

[2]College of Science
[3]College of Electronic and Information Engineering
Liaoning University of Technology
No. 169, Shiying Street, Guta District, Jinzhou 121001, P. R. China
jsknw@vip.sina.com

Received May 2015; revised September 2015

Abstract. *Generally speaking, Karnik-Mendel algorithm is a standard way to calculate the centroid and perform type-reduction (TR) for interval type-2 fuzzy sets and systems. In this paper, an efficient centroid type-reduction strategy for general type-2 fuzzy sets is introduced based on Karnik-Mendel (KM) algorithm, enhanced Karnik-Mendel (EKM) algorithm, enhanced iterative algorithm+stopping condition (EIASC). The strategy uses the result of α-plane representation, performs the centroid type-reduction on each α-plane, and expands type-reduction algorithms for general type-2 fuzzy logic systems. Simulations performed and compared by each of three types of algorithms show that they usually need only several resolution of α values such that the defuzzified values converge to real values. Compared with the exhaustive computation method, the method can tremendously decrease the computation complexity from exponential into linear. So it provides the potential application value for general type-2 fuzzy logic systems.*

Keywords: General type-2 fuzzy logic systems, α-plane, KM algorithm, EKM algorithm, EIASC

1. **Introduction.** It is difficult to deploy general type-2 (GT2) fuzzy logic systems (FLSs) into practical applications, because the computation complexity of them is quite high. Hence, only an interval type-2 (IT2) FLS [32-34] is the most widely used T2 FLS. Until recently, GT2 FLSs [1,2] based on α-plane (or z-Slices) theory are used in some fields.

An IT2 FS can account for membership function (MF) uncertainties. However, the secondary membership grades of an IT2 fuzzy set (FS) all equal 1. The secondary membership grades of a GT2 FS locate between 0 and 1. So a GT2 FS can be thought of as a higher order FS uncertainties model than its IT2 counterpart. Both IT2 FSs and GT2 FSs are parametric models. Each FS parameter in an FLS can be considered as one of its design degrees of freedom. As the number of design degrees of freedom increases, an IT2 FLS or a GT2 FLS may outperform a T1 FLS.

The typical structure of a T2 FLS is shown in Figure 1 [4]. In a T2 FLS, the output processing is composed by blocks of type-reducer and defuzzifier. However, the output processing of a T1 FLS is just the defuzzifier block. In this paper, only the type-reducer is addressed.

Karnik and Mendel [5] develop type-reduction methods, and they are elaborated upon in [4]. The conventional and efficient Karnik-Mendel (KM) algorithm has been developed for centroid type-reduction for IT2 FLSs. Although KM algorithm [26] converges

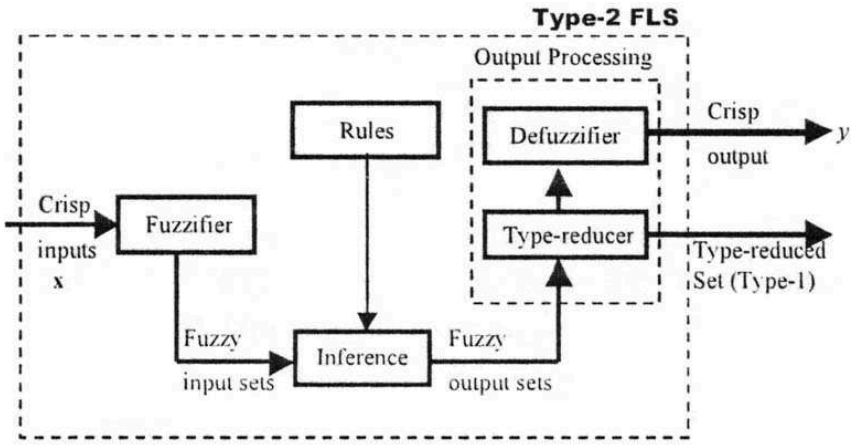


FIGURE 1. A T2 FLS

monotonically and super-exponentially fast, it usually achieves the final results from two to six iterations. In order to reduce computation time, EKM algorithm is developed by Wu and Mendel [6-8]. Compared with KM algorithm, EKM algorithm can save about two iterations. Another algorithm called EIASC was proposed in [9], which reduced the computation time more at the sacrifice of neglecting the uncertainty contained in FOU. According to [9], "The EIASC outperformed the KM algorithm, especially when N is small ($N \leq 100$, N is the number of discretization points of primary variable x)". However, no efficient strategies exist for centroid type-reduction of GT2 FLSs because: (1) in the wavy-slice representation theory, a T2 FS is expressed as the union of its T2 embedded sets. For a general T2 fuzzy system, centroid type-reduction was just derived from that theory in the previous idea; (2) the generalized centroid of these T2 embedded sets was first calculated and then combined into the type-reduced T1 FS; (3) an astronomical number of T2 embedded sets exists for the union operation, which makes the computation complexity of centroid type-reduction for GT2 FSs very high.

F. Liu [10] proposed an efficient strategy to perform centroid type-reduction for GT2 FLSs based on fuzzy weighted average [11,12]. The fuzzy weighted average is calculated on α-cuts. This paper aims to develop the efficient centroid type-reduction strategy for GT2 FSs based on KM algorithm, EKM algorithm, and EIASC. As shown in Figure 2, based on an α-plane Representation, a GT2 FS can be decomposed into several α-planes. For each α-plane, centroid type-reduction is achieved by KM algorithm, EKM algorithm, EIASC individually to get the α-cut of the type-reduced T1 FS. By combining these intervals or α-cuts, the type-reduced T1 FS is obtained.

The rest of this paper is organized as follows. Section 2 provides background material about T2 FSs, centroid of an IT2 FS, KM algorithm, EKM algorithm, EIASC. Section 3 introduces the definition of α-plane, α-plane representation, and the strategy to perform centroid type-reduction for GT2 FS. Section 4 gives the numerical simulations, compares and summarizes the test results, and analyzes the computation complexity of the proposed strategy. Finally, Section 5 gives the conclusions and expectations.

2. Backgrounds.

This section provides background material about T2 FSs, the centroid of such FSs and KM algorithm, EKM algorithm, EIASC.

2.1. Reviews on T2 FSs.

A T2 FS $\tilde{A}$ is characterized by a T2 MF $\mu_{\tilde{A}}(x,u)$, i.e., $\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) | \forall x \in X, \forall u \in J_x \subseteq [0,1]\}$ or $\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u)/(x,u)$, in

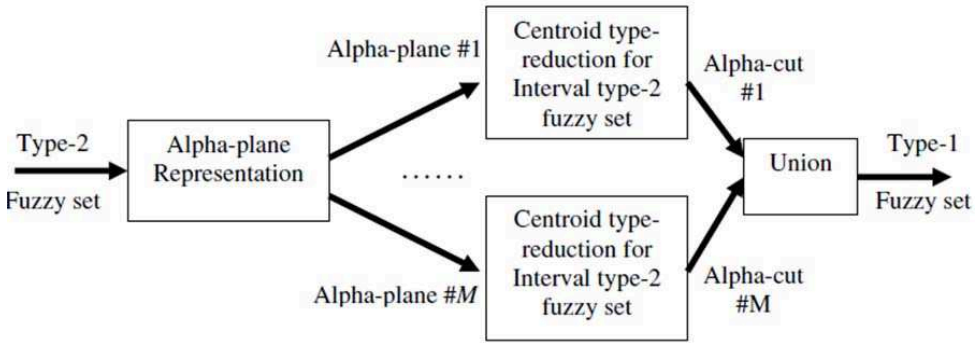


FIGURE 2. The key idea of centroid type-reduction for a GT2 FLS

which $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$, J_x is the primary membership, and $\int\int$ denotes all admissible x and u.

A secondary membership function (MF) is a vertical slice of $\mu_{\tilde{A}}(x,u)$. It is $\mu_{\tilde{A}}(x=x',u)$ for $x' \in X$ and $\forall u \in J_{x'} \subseteq [0,1]$, i.e., $\mu_{\tilde{A}}(x=x',u) \equiv \mu_{\tilde{A}}(x') = \int_{u \in J_{x'}} f_{x'}(u)/u$, $J_x \subseteq [0,1]$.

Uncertainty in the primary memberships of a T2 FS, $\tilde{A}$, consists of a bounded region that we call the footprint of uncertainty (FOU). It is the union of all primary memberships, i.e.,

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

which represents the uncertainty in the primary memberships of a T2 FS $\tilde{A}$. $FOU(\tilde{A})$ is bounded by lower membership function (LMF) and upper membership function (UMF), which are denoted as $\underline{\mu}_{\tilde{A}}(x)$ and $\overline{\mu}_{\tilde{A}}(x)$ [or LMF($\tilde{A}$) and UMF($\tilde{A}$)], respectively, where [16]

$$\text{LMF}(\tilde{A}) = \underline{\mu}_{\tilde{A}}(x) = \inf\{u | u \in [0,1], \mu_{\tilde{A}}(x,u) > 0\} \tag{2}$$

$$\text{UMF}(\tilde{A}) = \overline{\mu}_{\tilde{A}}(x) = \sup\{u | u \in [0,1], \mu_{\tilde{A}}(x,u) > 0\}. \tag{3}$$

An embedded T1 FS A_e is determined by $\mu_{\tilde{A}}(x,u)$, i.e.,

$$A_e = \{(x, u(x)) | \forall x \in X, u \in J_x\}. \tag{4}$$

A T2 FS $\tilde{A}$ can be expressed as a collection of its vertical slices, i.e.,

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u)/(x,u) = \int_{x' \in X} \left[\int_{u \in J_x} \mu_{\tilde{A}}(x=x',u)/u \right] \Big/ x' = \int_{x' \in X} \mu_{\tilde{A}}(x')/x' \tag{5}$$

for continuous, and

$$\tilde{A} = \sum_{x \in X} \left[\sum_{u \in J_x} \mu_{\tilde{A}}(x,u)/u \right] \Big/ x = \sum_{i=1}^{N} \left[\sum_{u \in J_{x_i}} \mu_{\tilde{A}}(x_i,u)/u \right] \Big/ x_i \tag{6}$$

for discrete case. This representation is called the vertical slice representation. A T2 FS $\tilde{A}$ can also be represented as the union of its embedded T2 FSs, i.e.,

$$\tilde{A} = \sum_{j=1}^{n} \tilde{A}_e^j. \tag{7}$$

The above representation is called the wavy-slice representation. It is useful for theoretical analysis [13,14].

Because the secondary grades of an IT2 FS [15,16] are all the same, an IT2 FS is completely described by its FOU [3], and consequently by its LMF and UMF. An example

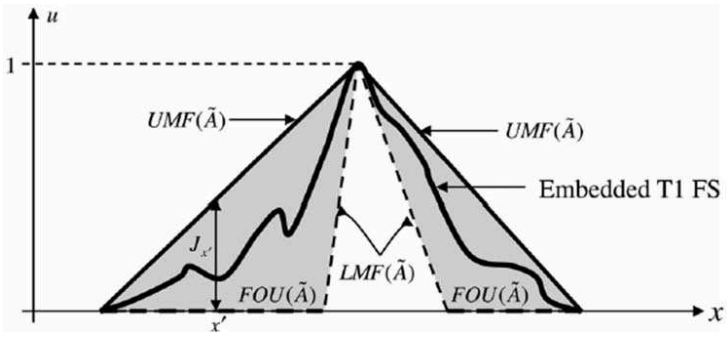


FIGURE 3. IT2 FS and related quantities [17]

of the FOU of an IT2 FS is depicted in Figure 3. Also, the lower and upper MFs for such an FOU [18,19], the primary membership at $x = x'$, and an example of an embedded T1 FS are shown in this figure.

2.2. **Centroid of an IT2 FS.** The centroid $C_{\tilde{A}}(x)$ of an IT2 FS $\tilde{A}$ is the union of the centroids, $C_{\tilde{A}}(A_e)$, of all its n_A embedded T1 FSs A_e. This means [4,5] that,

$$C_{\tilde{A}}(x) = 1 \Big/ \bigcup_{\forall A_e} c_{\tilde{A}}(A_e) = 1 \Big/ \bigcup_{\forall A_e} \frac{\sum_{i=1}^{N} x_i \mu_{A_e}(x_i)}{\sum_{i=1}^{N} \mu_{A_e}(x_i)} = 1 \Big/ \Big[c_l(\tilde{A}), c_r(\tilde{A})\Big] \tag{8}$$

where

$$\begin{aligned} c_l(\tilde{A}) &= \min_{\forall A_e} c_{\tilde{A}}(A_e) = \min_{\forall \theta_i \in \left[\underline{\mu}_{\tilde{A}}(x_i), \overline{\mu}_{\tilde{A}}(x_i)\right]} \left(\sum_{i=1}^{N} x_i \theta_i \Big/ \sum_{i=1}^{N} \theta_i \right) \\ &\approx c_l(k) = \frac{\sum_{i=1}^{k} x_i \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i \underline{\mu}_{\tilde{A}}(x_i)}{\sum_{i=1}^{k} \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \underline{\mu}_{\tilde{A}}(x_i)} \end{aligned} \tag{9}$$

$$\begin{aligned} c_r(\tilde{A}) &= \max_{\forall A_e} c_{\tilde{A}}(A_e) = \max_{\forall \theta_i \in \left[\underline{\mu}_{\tilde{A}}(x_i), \overline{\mu}_{\tilde{A}}(x_i)\right]} \left(\sum_{i=1}^{N} x_i \theta_i \Big/ \sum_{i=1}^{N} \theta_i \right) \\ &\approx c_r(k) = \frac{\sum_{i=1}^{k} x_i \underline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i \overline{\mu}_{\tilde{A}}(x_i)}{\sum_{i=1}^{k} \underline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \overline{\mu}_{\tilde{A}}(x_i)}. \end{aligned} \tag{10}$$

x_i ($x_1 < x_2 < \cdots < x_N$) are the sampled values of the primary variable. KM algorithm, EKM algorithm, or EIASC can be used to solve the counterparts of c_l and c_r.

2.3. **Three types of type-reduction algorithms.** KM algorithm was stated in [21,25], and given the tabular form in [22, Chapter 2]. Table 1 shows the details of this algorithm.

KM algorithm [20] is one of optimization algorithms that can be used to solve the two end points c_l and c_r. A good optimization algorithm requires: 1) a good way to initialize it, 2) a good way to move from one step to the next, and 3) a good way to stop.

KM algorithm supplies neither a good initialization nor a good stop. However, it provides a good way to move from one step to the next. So there exists improved space for the algorithm.

TABLE 1. KM algorithm to compute the centroid endpoints of an IT2 FS [22, Chapter 2]

Step	KM Algorithm for c_l
	$c_l = \min_{\forall\theta_i \in [\underline{\mu}_{\tilde{A}}(x_i), \overline{\mu}_{\tilde{A}}(x_i)]} \left(\sum_{i=1}^{N} x_i\theta_i \Big/ \sum_{i=1}^{N} \theta_i \right)$
1	Initialize θ_i by setting $\theta_i = \left[\underline{\mu}_{\tilde{A}}(x_i) + \bar{\mu}_{\tilde{A}}(x_i)\right]/2$, $i = 1, \ldots, N$, and then compute $c' = c(\theta_1, \ldots, \theta_N) = \sum_{i=1}^{N} x_i\theta_i \Big/ \sum_{i=1}^{N} \theta_i$
2	Find k $(1 \le k \le N-1)$ such that $x_k \le c' \le x_{k+1}$
3	Set $\theta_i = \overline{\mu}_{\tilde{A}}(x_i)$ when $i \le k$, and $\theta_i = \underline{\mu}_{\tilde{A}}(x_i)$ when $i \ge k+1$, and then compute $c_l(k) \equiv \frac{\sum_{i=1}^{k} x_i\overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i\underline{\mu}_{\tilde{A}}(x_i)}{\sum_{i=1}^{k} \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \underline{\mu}_{\tilde{A}}(x_i)}$
4	Check if $c_l(k) = c'$. If yes, stop and set $c_l(k) = c_l$ and call k L. If no, go to Step 5.
5	Set $c' = c_l(k)$ and go to Step 2
Step	**KM Algorithm for c_r**
	$c_r = \max_{\forall\theta_i \in [\underline{\mu}_{\tilde{A}}(x_i), \bar{\mu}_{\tilde{A}}(x_i)]} \left(\sum_{i=1}^{N} x_i\theta_i \Big/ \sum_{i=1}^{N} \theta_i \right)$
1	Same as Step 1 above
2	Same as Step 2 above
3	Set $\theta_i = \underline{\mu}_{\tilde{A}}(x_i)$ when $i \le k$, and $\theta_i = \overline{\mu}_{\tilde{A}}(x_i)$ when $i \ge k+1$, and then compute $c_r(k) = \frac{\sum_{i=1}^{k} x_i\underline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i\overline{\mu}_{\tilde{A}}(x_i)}{\sum_{i=1}^{k} \underline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \overline{\mu}_{\tilde{A}}(x_i)}$
4	Check if $c_r(k) = c'$. If yes, stop and set $c_r(k) = c_r$ and call k R. If no, go to Step 5.
5	Set $c' = c_r(k)$ and go to Step 2

Paper [5] proved that each KM algorithm needs at most N iterations. In fact, it is quite conservative. Paper [23] proposed (without proof) that the number of iterations for each KM algorithm is less than or equal to $(N+2)/4$ on average. This is also very conservative.

Extensive simulation studies show that KM algorithms obtain their final results from two to six iterations no matter what is the number of N is.

Mendel and Wu [6-8] developed the EKM algorithm, and it is given in Table 2. The EKM algorithm modifies the traditional KM algorithm in three ways: 1) in order to reduce the number of iterations, a better initialization is adopted; 2) an unnecessary iteration is removed by changing the termination condition of the iterations; 3) the computational cost of each algorithm's iterations is reduced by using a subtle calculating technique.

The EIASC algorithm is a non-KM algorithm, and it does not satisfy the three design requirements of a good optimization algorithm. The algorithm is based on the monotonic properties: $c_l(k)$ in (9) first monotonically decreases and then monotonically increases with increase of k; $c_r(k)$ in (10) first monotonically increases and then monotonically decreases with increase of k. The beautifully simple EIASC is given in Table 3. The EIASC is the simplest to understand. When one starts with or without an FOU, it can be used in both two cases. The fastest way to complete may be the EIASC.

TABLE 2. EKM algorithm to compute the centroid endpoints of an IT2 FS

Step	EKM Algorithm for c_l
1	Set $k = [N/2.4]$ (the nearest integer to $N/2.4$) and compute: $$a = \sum_{i=1}^{k} x_i \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i \underline{\mu}_{\tilde{A}}(x_i)$$ $$b = \sum_{i=1}^{k} \overline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \underline{\mu}_{\tilde{A}}(x_i)$$ Compute $c' = a/b$
2	Find $k' \in [1, N-1]$ such that $x_{k'} \le c' \le x_{k'+1}$
3	Check if $k' = k$. If yes, stop and set $c' = c_l$, and $k = L$. If no, go to Step 4.
4	Compute $s = sign(k' - k)$ and: $$a' = a + s \sum_{i=\min(k,k')+1}^{\max(k,k')} x_i \left[\overline{\mu}_{\tilde{A}}(x_i) - \underline{\mu}_{\tilde{A}}(x_i)\right]$$ $$b' = b + s \sum_{i=\min(k,k')+1}^{\max(k,k')} \left[\overline{\mu}_{\tilde{A}}(x_i) - \underline{\mu}_{\tilde{A}}(x_i)\right]$$ Compute $c''(k') = a'/b'$
5	Set $c' = c''(k')$, $a = a'$, $b = b'$ and $k = k'$ and go to Step 2
Step	**EKM Algorithm for c_r**
1	Set $k = [N/1.7]$ (the nearest integer to $N/1.7$) and compute: $$a = \sum_{i=1}^{k} x_i \underline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} x_i \overline{\mu}_{\tilde{A}}(x_i)$$ $$b = \sum_{i=1}^{k} \underline{\mu}_{\tilde{A}}(x_i) + \sum_{i=k+1}^{N} \overline{\mu}_{\tilde{A}}(x_i)$$ Compute $c' = a/b$
2	Same as Step 2 above
3	Check if $k' = k$. If yes, stop and set $c' = c_r$, and $k = R$. If no, go to Step 4.
4	Compute $s = sign(k' - k)$ and: $$a' = a - s \sum_{i=\min(k,k')+1}^{\max(k,k')} x_i \left[\overline{\mu}_{\tilde{A}}(x_i) - \underline{\mu}_{\tilde{A}}(x_i)\right]$$ $$b' = b - s \sum_{i=\min(k,k')+1}^{\max(k,k')} \left[\overline{\mu}_{\tilde{A}}(x_i) - \underline{\mu}_{\tilde{A}}(x_i)\right]$$ Compute $c''(k') = a'/b'$
5	Same as Step 5 above

3. The α-Plane Representation and Its Application to Centroid Type-Reduction.

3.1. α-plane representation.

Definition 3.1. *The α-plane $\tilde{A}_\alpha$ is the union of all primary membership whose secondary grades are greater than or equal to the value α, i.e.,*

$$\tilde{A}_\alpha = \bigcup_{x \in X} \{(x,u) | \mu_{\tilde{A}}(x,u) \ge \alpha\} = \bigcup_{x \in X} (\mu_{\tilde{A}}(x))_\alpha = \bigcup_{x \in X} [a_\alpha(x), b_\alpha(x)] \tag{11}$$

where $(\mu_{\tilde{A}}(x))_\alpha$ is the α-cut of the vertical slice $\mu_{\tilde{A}}(x)$. The FOU is obviously a specific α-plane with $\alpha = 0$.

Some properties for the α-plane $\tilde{A}_\alpha$ can be summarized in the following.

TABLE 3. EIASC [9]

Step	EIASC for c_l	EIASC for c_r
1	Initialize $a = \sum_{i=1}^{N} x_i \underline{\mu}_{\tilde{A}}(x_i)$ $b = \sum_{i=1}^{N} \underline{\mu}_{\tilde{A}}(x_i)$ $L = 0$	Initialize $a = \sum_{i=1}^{N} x_i \overline{\mu}_{\tilde{A}}(x_i)$ $b = \sum_{i=1}^{N} \overline{\mu}_{\tilde{A}}(x_i)$ $R = N$
2	Compute $L = L + 1$ $a = a + x_L \left[\overline{\mu}_{\tilde{A}}(x_L) - \underline{\mu}_{\tilde{A}}(x_L)\right]$ $b = b + \left[\overline{\mu}_{\tilde{A}}(x_L) - \underline{\mu}_{\tilde{A}}(x_L)\right]$ $c_l = a/b$	Compute $a = a + x_R \left[\overline{\mu}_{\tilde{A}}(x_R) - \underline{\mu}_{\tilde{A}}(x_R)\right]$ $b = b + \left[\overline{\mu}_{\tilde{A}}(x_R) - \underline{\mu}_{\tilde{A}}(x_R)\right]$ $c_r = a/b$ $R = R - 1$
3	If $c_l \le x_{L+1}$, stop, otherwise, go to Step 2	If $c_r \ge x_R$, stop, otherwise, go to Step 2

Property 3.1. *$\tilde{A}_{\alpha_1} \subseteq \tilde{A}_{\alpha_2}$, if $\alpha_1 \ge \alpha_2$.*

Property 3.2. $\left(\tilde{A} \cup \tilde{B}\right)_{\alpha} = \tilde{A}_{\alpha} \cup \tilde{B}_{\alpha}$.

Property 3.3. $\left(\tilde{A} \cap \tilde{B}\right)_{\alpha} = \tilde{A}_{\alpha} \cap \tilde{B}_{\alpha}$.

Property 3.1 shows that, for the same T2 FS $\tilde{A}$, the α-plane with greater α must be included in the α-plane with smaller α. Property 3.2 shows that, the α-plane of union of two T2 FS $\tilde{A}$ and $\tilde{B}$ is the same as the union of the two α-plane for the T2 FS $\tilde{A}$ and $\tilde{B}$. Property 3.3 shows that, the α-plane of intersection of two T2 FS $\tilde{A}$ and $\tilde{B}$ is the same as the intersection of the two α-plane for the T2 FS $\tilde{A}$ and $\tilde{B}$.

Definition 3.2. *The associated T2 FS of the α-plane $\tilde{A}_{\alpha}$ is defined as*

$$\tilde{A}(\alpha) = \left\{(x, u), \alpha | \forall (x, u) \in \tilde{A}_{\alpha}\right\}. \tag{12}$$

Next, the α-plane representation theorem is explained.

Theorem 3.1. *A GT2 $\tilde{A}$ can be represented as the union of its associated T2 FSs $\tilde{A}(\alpha)$, i.e.,*

$$\tilde{A} = \bigcup_{\alpha \in [0,1]} \tilde{A}(\alpha). \tag{13}$$

In Theorem 3.1, the α-plane representation theory for a T2 FS is proposed. It is different from the vertical slice representation and embedded set representation.

3.2. **Centroid type-reduction.** The centroid type-reduction for a T2 FS $\tilde{A}$ can be formulated as

$$Y_c = \int_{u_1 \in J_{x_1}} \cdots \int_{u_N \in J_{x_N}} \mu_{\tilde{A}}(x_1, u_1) \times \cdots \times \mu_{\tilde{A}}(x_N, u_N) \Big/ \frac{\sum_{i=1}^{N} x_i u_i}{\sum_{i=1}^{N} u_i} \tag{14}$$

where "$\times$" usually denotes the product or minimum t-norm. The minimum t-norm is used for the centroid type-reduction in the paper. The centroid type-reduced output $Y_c(x)$ is a T1 FS. x_i $(i = 1, 2, \ldots, N)$ are ordered in centroid type-reduction, for which

X_i, $i = 1, 2, \ldots, N$ have only a single value, and $\mu_{\tilde{A}}(x_i, u_i)$ is viewed as the vertical slices of a GT2 FS.

Theorem 3.2. *The centroid type-reduction for a T2 FS $\tilde{A}$ is the union of its related T2 FSs $\tilde{A}(\alpha)$, with $\alpha \in [0, 1]$, i.e.,*

$$Y_c = \int_{\alpha\in[0,1]} \text{Centroid}\left(\tilde{A}(\alpha)\right) = \int_{\alpha\in[0,1]} \alpha/\text{domain}\left(\text{Centroid}\left(\tilde{A}(\alpha)\right)\right) \tag{15}$$

$$(Y_c)_\alpha = \text{domain}\left(\text{Centroid}\left(\tilde{A}(\alpha)\right)\right)$$

$$Y_c(\alpha) = \text{Centroid}\left(\tilde{A}(\alpha)\right). \tag{16}$$

Then the centroid type-reduction for a T2 FS can be decomposed into individual centroid type-reduction calculation for its related T2 FS $\tilde{A}(\alpha)$, and they can be performed in parallel. We can also formulate (16) as follows:

$$Y_c(\alpha) = \text{Centroid}\left(\tilde{A}(\alpha)\right) = \int_{u_1\in{}^\alpha J_{x_1}} \cdots \int_{u_N\in{}^\alpha J_{x_N}} \alpha \Big/ \frac{\sum_{i=1}^{N} x_i u_i}{\sum_{i=1}^{N} u_i} = \alpha/\left[{}^\alpha y_l, {}^\alpha y_r\right] \tag{17}$$

where $[{}^\alpha y_l, {}^\alpha y_r]$ is the domain of the centroid. Therefore, we can use algorithms like KM, EKM or EIASC to calculate ${}^\alpha y_l$ and ${}^\alpha y_r$.

The strategy to calculate centroid type-reduction for a GT2 FS is:

Step 1: Break the α into Δ values of 0, $1/\Delta$, $2/\Delta$, $\ldots$, $(\Delta - 1)/\Delta$, 1. Decompose the GT2 FS into multiple α-planes $\tilde{A}_\alpha$ with above α values. Unite all these α-cuts with same α for the vertical slices.

Step 2: Calculate the centroid $\alpha/[{}^\alpha y_l, {}^\alpha y_r]$ for each related T2 FS $\tilde{A}(\alpha)$.

Step 3: Calculate the union of all these centroids.

4. **Simulations.** In this paper, two cases are presented, for which (1) the FOU is derived from two fired rules, and the FOU is bounded by Gaussian functions or triangular functions, and (2) non-symmetric or symmetric triangular MFs are adopted for the vertical slices. In practical applications, the T2 FS is usually continuous. In order to perform centroid type-reduction and defuzzification, the T2 MF needs to be discretized. Attention that, for any situation, we assumed that $x \in [0, 10]$, and x is uniformly sampled. The number of total samples for x is 200, so that $x_{i+1} - x_i = 0.05$. The range of Δ was from 1 to 100. For each α-plane, the KM, EKM or EIASC algorithms were used to compute the lower and upper bound for the centroid type-reduced interval. By the weighted average defuzzification, a crisp value was obtained.

Case A: Gaussian function with randomly generated triangular vertical slice.

As shown in Figure 4(a), the UMF of the FOU is composed of the maximum of the two Gaussian functions:

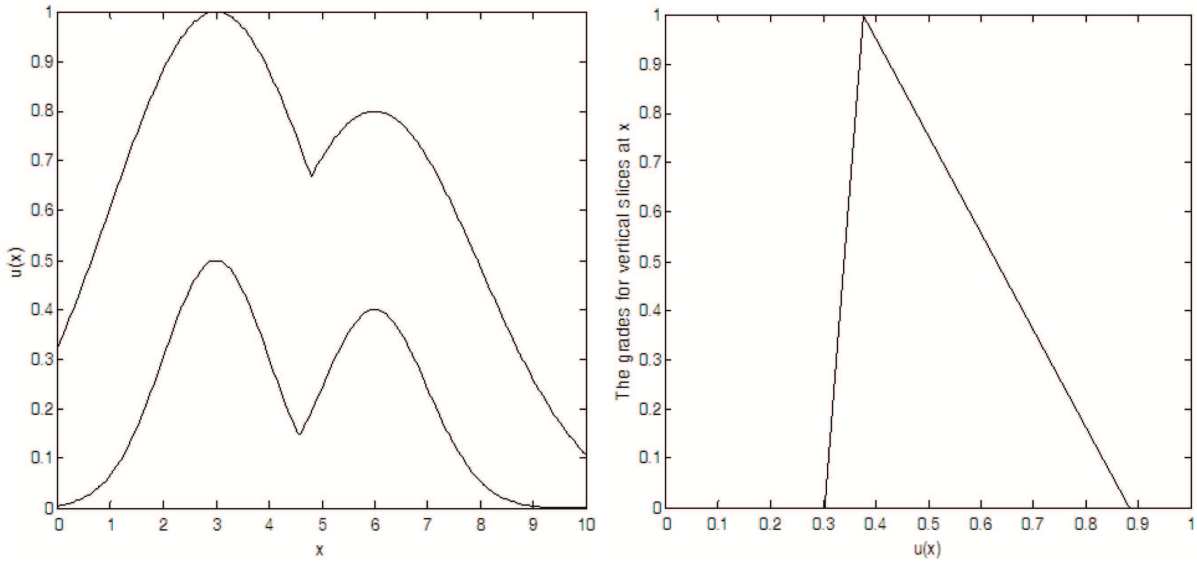
$$u_1(x) = \exp\left(-\frac{(x-3)^2}{8}\right) \tag{18}$$

and

$$u_2(x) = \exp\left(-\frac{(x-6)^2}{8}\right). \tag{19}$$

The LMF of the FOU is composed of the maximum of the two Gaussian functions:

$$u_3(x) = 0.5\exp\left(-\frac{(x-3)^2}{2}\right) \tag{20}$$



(a) (b)

FIGURE 4. (a) The FOU for Case A and (b) its corresponding vertical slice when $x = 4$

and

$$u_4(x) = 0.4 \exp\left(-\frac{(x-6)^2}{2}\right). \tag{21}$$

For any value of x, the related vertical slice is triangular MF, whose apex is decided by a randomly generated value $w(x) \in [0, 1]$ as

$$\text{Apex} = \underline{u}(x) + w(x)\left(\overline{u}(x) - \underline{u}(x)\right) \tag{22}$$

where the lower and upper bounds of the primary membership J_x are $\underline{u}(x)$ and $\overline{u}(x)$. The simulation graphs of three types of type-reduction algorithms of type-reduced MFs for $\Delta = 100$ and the defuzzified values for Δ ranging from 1 to 100 are shown in Figures 5 and 6.

Case B: Triangular function with non-symmetric trapezoid vertical slice

This case uses triangular function to bound the FOU. As shown in Figure 7, the upper UMF of the FOU is composed of the maximum of two triangular functions, i.e.,

$$u_1(x) = \begin{cases} \frac{x-1}{2}, & 1 \le x \le 3 \\ \frac{7-x}{4}, & 3 < x \le 7 \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

and

$$u_2(x) = \begin{cases} \frac{x-2}{5}, & 2 \le x \le 6 \\ \frac{16-2x}{5}, & 6 < x \le 8 \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

The upper bound of the FOU is composed of the maximum of the two triangular functions

$$u_3(x) = \begin{cases} \frac{x-1}{6}, & 1 \le x \le 4 \\ \frac{7-x}{6}, & 4 < x \le 7 \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

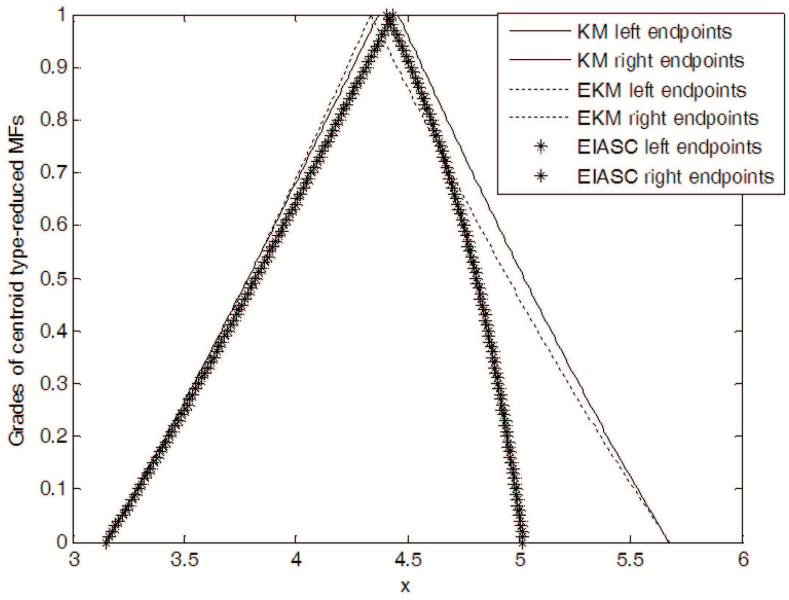


FIGURE 5. Three types of type-reduction algorithms of type-reduced MFs when $\Delta = 100$ for Case A

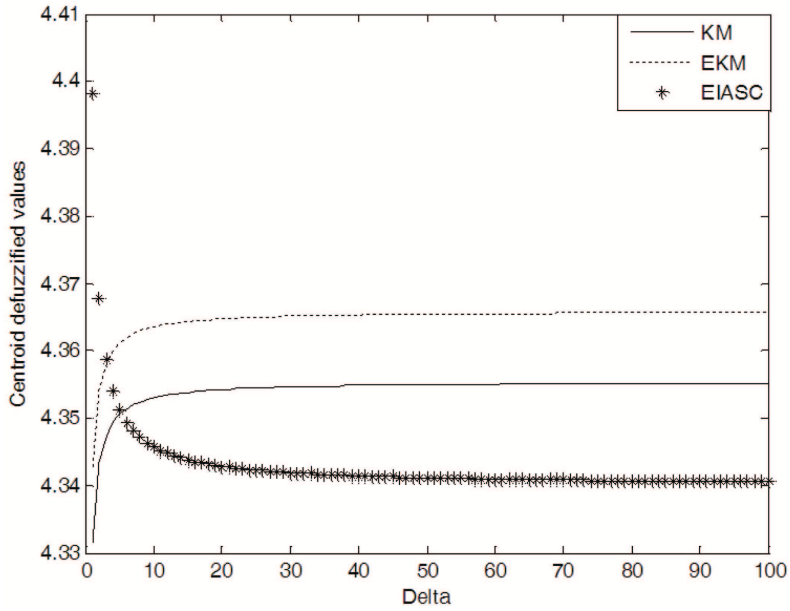


FIGURE 6. Three types of type-reduction algorithms of centroid defuzzified values of Δ ranging from 1 to 100 for Case A

and

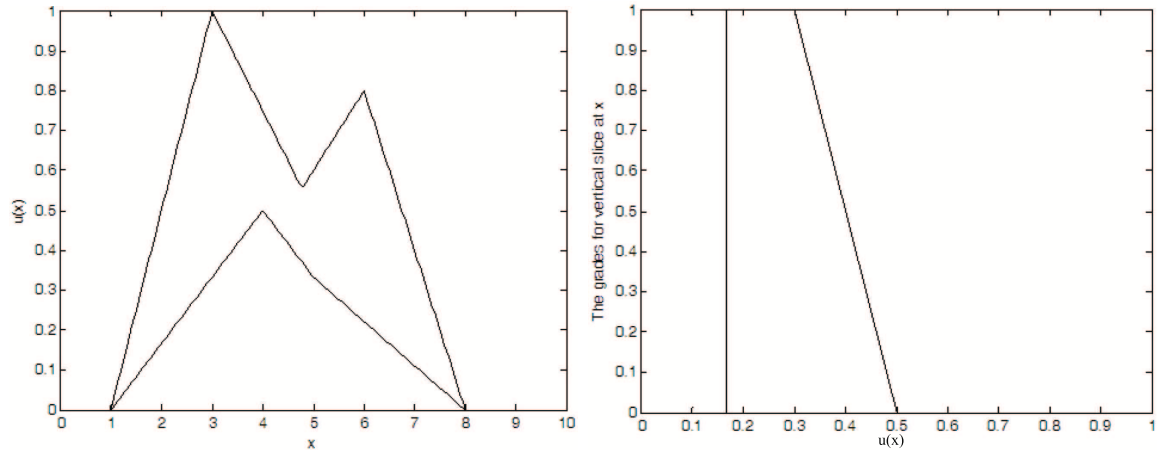
$$u_4(x) = \begin{cases} \frac{x-3}{6}, & 3 \le x \le 5 \\ \frac{8-x}{9}, & 5 < x \le 8 \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

For any value of x, the related vertical slice is trapezoid MF, whose top left and right end-points are decided by

$$\begin{aligned} L(x) &= \underline{u}(x) + 0.6w\left(\overline{u}(x) - \underline{u}(x)\right) \\ R(x) &= \overline{u}(x) - 0.6(1-w)\left(\overline{u}(x) - \underline{u}(x)\right) \end{aligned} \tag{27}$$

where the lower and upper bounds of the primary membership J_x are $\underline{u}(x)$ and $\overline{u}(x)$, and $w = 0$ is chosen for this test. The simulation graphs of three types of type-reduction algorithms of type-reduced MFs for $\Delta = 100$ and the defuzzified values for Δ ranging from 1 to 100 are shown in Figures 8 and 9.

Observing from these experiments, note that, regardless of the nature of the secondary MFs: (1) for three types of type-reduction algorithms, the α-cut with greater α is always included in the α-cut with small α. From Property 3.1, we can also see that, the α-plane with greater α is always included in the α-plane with small α; (2) for three types of type-reduction algorithms, the shape of the type-reduced T1 FS depends on the shape of the secondary MF, that is, the first test with triangular secondary MF derived a triangular-looking T1 MF, and the second test with trapezoid secondary MF derived a trapezoid-looking T1 MF. It can be easily deduced from the α-plane representation theory; (3) for



(a) (b)

FIGURE 7. (a) The FOU for Case B and (b) its corresponding vertical slice when $x = 2$

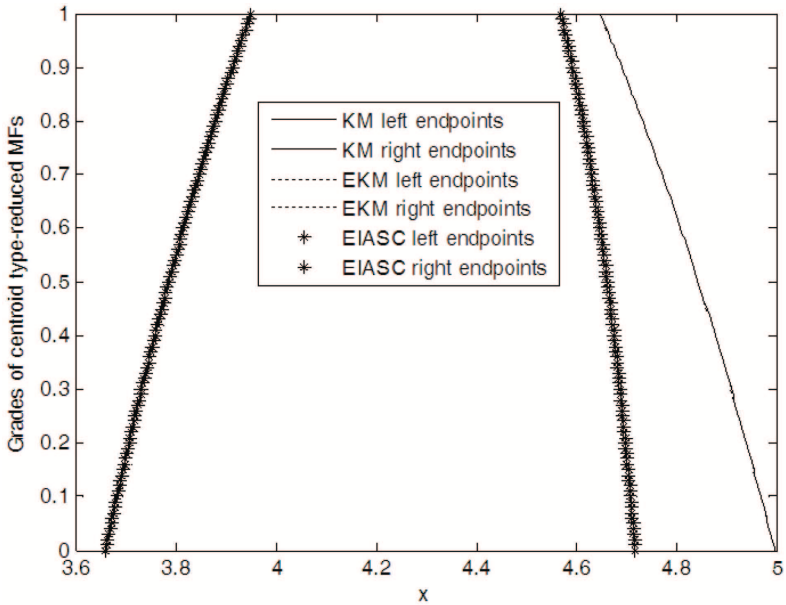


FIGURE 8. Three types of type-reduction algorithms of type-reduced MFs when $\Delta = 100$ for Case B

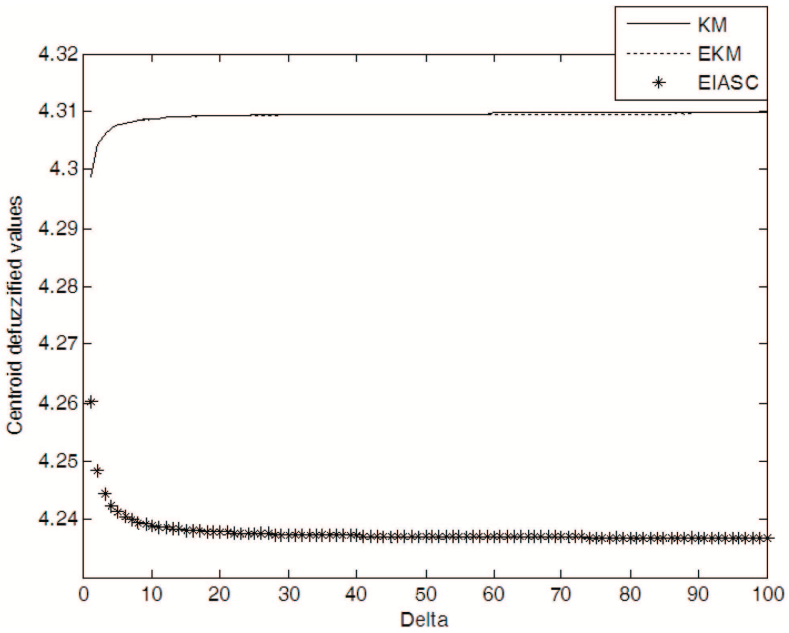


FIGURE 9. Three types of type-reduction algorithms of centroid defuzzified values of Δ ranging from 1 to 100 for Case B

three types of type-reduction algorithms, as Δ increases, the centroid defuzzified values all converge to real values, and the real values can be viewed as the defuzzified values of the continuous T2 FSs; (4) for three types of type-reduction algorithms, from Figures 5, 6, 8, 9, we can find that the grades of centroid type-reduced MFs and the centroid defuzzified values are different in Case A, the grades of left endpoints of centroid intervals are almost the same, and the centroid defuzzified values for KM algorithms and EKM algorithms are almost the same in Case B (this shows that KM and EKM are very similar); (5) for three types of type-reduction algorithms, they demand only one α-planes when the error between a defuzzified value and the constant is smaller than 0.05.

Additionally, compared with the exhaustive computation method α-plane theory based type-reduction algorithms are more efficient for centroid type-reduction. Assume that the primary variable x is sampled for N points, and $u(x)$ is sampled for M points. There are M^N embedded T2 FSs included in the GT2 FS. For each embedded T2 FS, N multiplications, $2(N-1)$ additions and one division are needed for centroid computation, and $N-1$ comparisons are needed for the t-norm operation. Therefore, the computation complexity of exhaustive computation method is about $O(4M^N N)$. For the proposed strategy (α-plane theory), no matter which of the above three types of type-reduction algorithms we are choosing, N samples are needed for x, and k samples are needed for α-plane. From [12,26], for each α-plane, $O(2 \times 4N \times n)$, where n (the number of iterations) is usually smaller than 6. So the proposed strategy requires about $O(8N \times n \times k)$ to confirm the centroid FS, which tremendously decreases the computation complexity from exponential into linear. Because the computation for each α-plane is completely independent, the computations can be performed in parallel. Besides, these two lower and upper bounds of the interval are also independent. Therefore, if $2k$ parallel processors are used to compute in parallel, the complexity of the proposed strategy is only $O(4N \times n)$.

5. **Conclusions and Expectations.** This paper investigates three types of type-reduction algorithms for GT2 FLSs based on α-plane theory. According to two numerical simulation examples, as Δ increases, the centroid defuzzified values for three types of

type-reduction algorithms all converge to real values. Compared with the exhaustive computation method, the proposed method can greatly decrease the computation complexity from exponential into linear.

There are much interesting works that lie ahead, including the study of type-reduction [10] of GT2 FLSs, the parametric optimization of GT2 FLSs [1,27]. Future research will be concentrated on GT2 FLSs design, algorithms that are based on [24,27-31] and this paper.

Acknowledgment. This paper is partially supported by the National Science Foundation ("Uncertain Nonlinear System Adaptive Control with Non-smooth and Nonlinear Inputs", No. 61104017). The authors greatly appreciate Prof. J. M. Mendel, who has given the author some important suggestions.

REFERENCES

[1] 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, 2010.

[2] 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.

[3] 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, 1999.

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

[5] N. N. Karnik and J. M. Mendel, Centroid of a type-2 fuzzy set, *Information Sciences*, vol.132, no.1, pp.195-220, 2001.

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

[7] M. C. A. Melgarejo, A fast recursive method to compute the generalized centroid of an interval type-2 fuzzy set, *Proc. of North Amer. Fuzzy Info. Process. Soc.*, San Diego, pp.190-194, 2007.

[8] K. Duran, H. Bernal and M. Melgarejo, Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set, *Annu. Meeting North Amer. Fuzzy Inf. Process. Soc.*, New York, pp.1-5, 2008.

[9] D. Wu and M. Nie, Comparison and practical implementations of type reduction algorithms for type-2 fuzzy sets and systems, *Proc. of IEEE Int. Conf. Fuzzy Syst.*, Taipei, Taiwan, pp.2131-2138, 2011.

[10] F. Liu, An efficient centroid type reduction strategy for general type-2 fuzzy logic system, *Information Sciences*, vol.178, no.9, pp.2224-2236, 2008.

[11] W. M. Dong and F. S. Wong, Fuzzy weighted averages and implementation of the extension principle, *Fuzzy Sets and Systems*, vo.21, no.2, pp.183-199, 1987.

[12] F. Liu and J. M. Mendel, Aggregation using the fuzzy weighted average as computed by the Karnik-Mendel algorithms, *IEEE Trans. Fuzzy Syst.*, vol.16, no.1, pp.1-12, 2008.

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

[14] 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, 2006.

[15] 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, no.3, pp.137-209, 2000.

[16] J. Aisbett, J. T. Rickard and D. G. Morgenthaler, Type-2 fuzzy sets as functions on spaces, *IEEE Trans. Fuzzy Syst.*, vol.18, no.8, pp.841-844, 2010.

[17] J. M. Mendel, Type-2 fuzzy sets and systems: An overview, *IEEE Computational Intelligent Magazine*, vol.2, no.2, pp.20-29, 2007.

[18] J. M. Mendel and H. Wu, Type-2 fuzzistics for symmetric interval type-2 fuzzy sets: Part 1, forward problems, *IEEE Trans. Fuzzy Syst.*, vol.14, no.6, pp.781-792, 2006.

[19] J. M. Mendel and H. Wu, Type-2 fuzzistics for non-symmetric interval type-2 fuzzy sets: Forward problems, *IEEE Trans. Fuzzy Syst.*, vol.15, no.5, pp.916-930, 2007.

[20] X. Liu and J. M. Mendel, Connect Karnik-Mendel algorithms to root finding for computing the centroid of an interval type-2 fuzzy set, *IEEE Trans. Fuzzy Syst.*, vol.19, no.4, pp.652-665, 2011.
[21] J. M. Mendel, Advances in type-2 fuzzy sets and systems, *Information Sciences*, vol.177, no.1, pp.84-110, 2007.
[22] J. M. Mendel and D. Wu, *Perceptual Computing: Aiding People in Making Subjective Judgments*, Wiley, Hoboken, NJ, 2010.
[23] 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, 2002.
[24] J. M. Mendel, F. Liu and D. Zhai, α-plane representation for type-2 fuzzy sets: Theory and applications, *IEEE Trans. Fuzzy Syst.*, vol.17, no.5, pp.1189-1207, 2009.
[25] J. M. Mendel, On KM algorithms for solving type-2 fuzzy set problems, *IEEE Trans. Fuzzy Syst.*, vol.21, no.3, pp.426-446, 2013.
[26] J. M. Mendel and F. Liu, Super-exponential convergence of the Karnik-Mendel algorithms for computing the centroid of an interval type-2 fuzzy set, *IEEE Trans. Fuzzy Syst.*, vol.15, no.2, pp.309-320, 2007.
[27] J. M. Mendel, General type-2 fuzzy logic systems made simple: A tutorial, *IEEE Trans. Fuzzy Syst.*, vol.22, no.5, pp.1162-1182, 2014.
[28] 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, 2011.
[29] 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, 2012.
[30] T. Wang, Y. Chen and S. Tong, Fuzzy reasoning models and algorithms on type-2 fuzzy sets, *International Journal of Innovative Computing, Information and Control*, vol.4, no.10, pp.2451-2460, 2008.
[31] Y. Chen and T. Wang, Gaussian interval type-2 interpolative fuzzy reasoning, *ICIC Express Letters*, vol.6, no.1, pp.229-234, 2012.
[32] Q. Lu, P. Shi, H. K. Lam and Y. Zhao, Interval type-2 fuzzy model predictive control of nonlinear networked control systems, *IEEE Trans. Fuzzy Syst.*, 2015.
[33] H. Li, C. Wu, P. Shi and Y. Gao, Control of nonlinear networked systems with packet dropouts: Interval type-2 fuzzy model-based approach, *IEEE Trans. Fuzzy Syst.*, 2014.
[34] H. Li, X. Sun, P. Shi and H. K. Lam, Control design of interval type-2 fuzzy systems with actuator fault: Sampled-data control approach, *Information Sciences*, vol.302, pp.1-13, 2015.