AIMS Mathematics, 5(6): 7719–7745.
DOI: 10.3934/math.2020494
Received: 03 August 2020
Accepted: 07 October 2020
Published: 14 October 2020

http://www.aimspress.com/journal/Math

Research article

Study on weighted-based noniterative algorithms for centroid type-reduction of interval type-2 fuzzy logic systems

Yang Chen*, Jinxia Wu and Jie Lan

College of Science, Liaoning University of Technology, Jinzhou, Liaoning, 121001, P. R. China

*** Correspondence:** Email: lxychenyang@lnut.edu.cn; Tel: +8613897856294; Fax: +864164199415.

Abstract: Interval type-2 fuzzy logic systems (IT2 FLSs) have been widely used in many areas. Among which, type-reduction (TR) is an important block for theoretical study. Noniterative algorithms do not involve the complicated iteration process and obtain the system output directly. By discovering the inner relations between discrete and continuous noniterative algorithms, this paper proposes three types of weighted-based noniterative according to the Newton-Cotes quadrature formulas in numerical integration techniques. Moreover, the continuous noniterative algorithms are considered as the benchmarks for computing. Four simulation experiments are provided to illustrate the performances of weighted-based noniterative algorithms for computing the defuzzified values of IT2 FLSs. Compared with the original noniterative algorithms, the proposed weighted-based algorithms can obtain smaller absolute errors and faster convergence speeds under the same sampling rate, which afford the potential values for designing T2 FLSs.

Keywords: type-reduction; weighted Nagar-Bardini algorithms; weighted Nie-Tan algorithms; weighted Begian-Melek-Mendel algorithms; absolute errors
Mathematics Subject Classification: 68XX, 68Uxx

1. Introduction

The computationally relative simple interval type-2 fuzzy sets (IT2 FSs) are currently the most commonly used T2 FSs. Therefore, interval type-2 fuzzy logic systems (IT2 FLSs [1,2,57]) based on IT2 FSs have superior ability to cope with uncertainties environments like power systems [3], permanent magnetic drive [4–6], intelligent controllers [7], medical systems [8], pattern recognition systems [9], database and information systems [10] and so on. The footprint of uncertainty (FOU) of

an IT2 FS [11] make it own more design degrees of freedom compared with a T1 FS. Generally speaking, a T2 FLS (see Figure 1) is composed of fuzzifier, rules, inference, type-reducer and defuzzifier. Among which, the block of type-reduction plays the central role of transforming the T2 to the T1 FS. Finally the defuzzification changes the T1 FS to the crisp output.

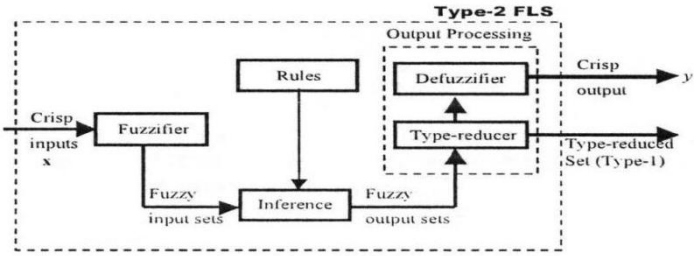

Figure 1. A T2 FLS [12].

In the past decades, many types of type-reduction (TR) were proposed gradually. Among which, the most famous one is the Karnik-Mendel (KM) algorithms [13]. This type of algorithms has the advantages of preserving the uncertainties flow between the upper and lower membership functions (MFs) of T2 FLSs. Then the continuous KM (CKM) algorithms [14] were put forward, in addition, the monotoncity and super convergence property of them were proved. For the sake of improving the calculation efficiency, Wu and Mendel proposed the enhanced KM (EKM) algorithms [15]. Extensive simulation experiments show that the EKM algorithms can save two iterations on average compared with the KM algorithms. Then Liu et al. gave the theoretical explanations for the initialization of EKM algorithms and extended the EKM algorithms to three different forms of weighted-based EKM (WEKM) algorithms [16,17] to calculate the more accurate centroids of type-2 fuzzy sets.

However, the iterative natures of KM types of algorithms make the applications of corresponding IT2 FLSs more challenge. Therefore, other types of noniterative algorithms [18] are proposed gradually, and they are Nagar-Bardini (NB) algorithms [19], Nie-Tan (NT) algorithms [20], Begian-Melek-Mendel (BMM) algorithms [21,22] and so on. Among which, IT2 FLSs based on the NB algorithms are proved to have superior performances to respond to the affect of uncertainties in systems' parameters to the IT2 FLSs according to other algorithms like EKM, BMM, Greenfield-Chiclana Collapsing Defuzzifier (GCCD [24]) and Wu-Mendel Uncertainty Bound (UB [24]). Moreover, Nie-Tan (NT) algorithms have the simple closed form. Recent studies prove that the continuous NT (CNT) algorithms [20] to be accurate algorithms to calculate the centroid of IT2 FSs. In addition, BMM algorithms [18] are proved to be more generalized forms of NB and NT algorithms. All these works have laid theoretical foundations for studying the TR of T2 FLSs.

In order to obtain more accurate centroid TR of IT2 FLSs, this paper extends the NB, NT, and BMM algorithms to the corresponding WNB, WNT, and WBMM algorithms according to the Newton-Cotes quadrature formulas. The rest of this paper is organized as follows. Section 2 introduces the background of IT2 FLSs. Section 3 provides the Newton-Cotes formulas, the weighted-based noniterative algorithms, and how to adopt them to perform the centroid TR of IT2 FLSs. Section 4 gives four computer simulation examples to illustrate the performances of weighted-based noniterative algorithms. Finally Section 5 is the conclusions and expectations.

2. IT2 FLSs

Definition 1. A T2 FS $\tilde{A}$ can be characterized by its T2 MF $\mu_{\tilde{A}}(x,u)$, i.e.,

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

where the primary variable $x\in X$, the secondary variable $u\in[0,1]$, here Eq (1) is called as the point-value expression, whose compact form is as:

$$\tilde{A}=\int_{x\in X}\int_{u\in[0,1]}\mu_{\tilde{A}}(x,u)/(x,u) \tag{2}$$

Definition 2. The secondary MF of $\tilde{A}$ is a vertical slice of $\mu_{\tilde{A}}(x,u)$, i.e.,

$$\mu_{\tilde{A}}(x=x',u)\equiv\mu_{\tilde{A}}(x')=\int_{\forall u\in[0,1]}f_{x'}(u)/u\,. \tag{3}$$

Definition 3. The two dimensional support of $\mu_{\tilde{A}}(x,u)$ is referred to as the footprint of uncertainty (FOU) of $\tilde{A}$, i.e.,

$$\mathrm{FOU}(\tilde{A})=\bigcup\nolimits_{x\in X}J_x=\{(x,u)\in X\times[0,1]\,|\,\mu_{\tilde{A}}(x,u)>0\} \tag{4}$$

The upper and lower bounds of $\mathrm{FOU}(\tilde{A})$ are called as the upper MF (UMF) and lower MF (LMF), respectively, i.e.,

$$\mathrm{UMF}(\tilde{A})=\overline{\mu}_{\tilde{A}}(x)=\overline{\mathrm{FOU}(\tilde{A})},\quad \mathrm{LMF}(\tilde{A})=\underline{\mu}_{\tilde{A}}(x)=\underline{\mathrm{FOU}(\tilde{A})}\,. \tag{5}$$

Because the secondary membership grades of IT2 FSs are all uniformly equal to 1, i.e., $f_x(u)\equiv1$, every IT2 FS can be completely described by its UMF and LMF. Then the vertical slices representation of $\tilde{A}$ can be as:

$$\tilde{A}=\int_{\forall x\in X}\mu_{\tilde{A}}(x)/x=\int_{\forall x\in X}\tilde{A}(x)/x \tag{6}$$

Here $\mu_{\tilde{A}}(x)$ can be denoted as $\tilde{A}(x)$ for simplicity.

Definition 4. An embedded T1 FS A_e depends on $\mu_{\tilde{A}}(x,u)$, i.e.,

$$A_e=\{(x,u(x)\,|\,\forall x\in X,u\in J_x\}\,. \tag{7}$$

Definition 5. An IT2 FS can be considered as the union of all its embedded T2 FSs $\tilde{A}_e^j$, i.e.,

$$\tilde{A}=\sum_{j=1}^{m}\tilde{A}_e^j \tag{8}$$

where m denotes the number of embedded T2 FSs, $\tilde{A}_e^j=1/A_e^j$, and Eq (8) is referred to as the wavy slices representation [28] of $\tilde{A}$.

From the aspect of inference structure, IT2 FLSs can usually be divided into two categories: Mamdani type [3,6,12,18,25] and Takagi-Sugeno-Kang type [4,25–27]. Without loss of generality,

here we consider a Mamdani IT2 FLS with n inputs $x_1 \in X_1, \cdots x_n \in X_n$ and one output $y \in Y$. The IT2 FLS can be characterized by M fuzzy rules, where the sth fuzzy rule is of the form:

$$\widetilde{R}^s: \text{If } x_1 \text{ is } \widetilde{F}_1^s \text{ and } \cdots \text{ and } x_n \text{ is } \widetilde{F}_n^s, \text{ then } y \text{ is } \widetilde{G}^s (s = 1,2,\cdots,M) \tag{9}$$

in which $\widetilde{F}_i^s (i = 1,\cdots,n; s = 1,\cdots,M)$ is the antecedent IT2 FS, and $\widetilde{G}^s (s = 1,\cdots,M)$ is the consequent IT2 FS.

For simplicity, here we use the singleton fuzzifier, i.e., the input measurements are modeled as crisp sets (type-0 FSs). As $x = x'$, the firing interval of each fuzzy rule is computed as:

$$F^s: \begin{cases} F^s(x') \equiv [\underline{f}^s(x'), \overline{f}^s(x')], \\ \underline{f}^s(x') \equiv T_{i=1}^n \underline{\mu}_{\widetilde{F}_i^s}(x_i'), \\ \overline{f}^s(x') \equiv T_{i=1}^n \overline{\mu}_{\widetilde{F}_i^s}(x_i') \end{cases} \tag{10}$$

where T denotes the minimum or product t-norm, and $\underline{f}^s(x')$ and $\bar{f}^s(x')$ are the left and right end points of the firing interval, respectively.

As for the centroid TR, we combine the firing interval of each rule with its consequent IT2 FS to obtain the fired-rule output FS $\widetilde{B}^l$ (which can be described by its FOU):

$$\widetilde{B}^s: \begin{cases} \text{FOU}(\widetilde{B}^s) = [\underline{\mu}_{\widetilde{B}^s}(y \mid x'), \overline{\mu}_{\widetilde{B}^s}(y \mid x')], \\ \underline{\mu}_{\widetilde{B}^s}(y \mid x') = \underline{f}^s(x') * \underline{\mu}_{\widetilde{G}^s}(y), \\ \overline{\mu}_{\widetilde{B}^s}(y \mid x') = \overline{f}^s(x') * \overline{\mu}_{\widetilde{G}^s}(y) \end{cases} \tag{11}$$

where $*$ represents the minimum or product t-norm.

Then the output IT2 FS $\widetilde{B}$ can be obtained by aggregating all the fired-rule output FSs as :

$$\widetilde{B}: \begin{cases} \text{FOU}(\widetilde{B}) = [\underline{\mu}_{\widetilde{B}}(y \mid x'), \overline{\mu}_{\widetilde{B}}(y \mid x')], \\ \underline{\mu}_{\widetilde{B}}(y \mid x') = \underline{\mu}_{\widetilde{B}^1}(y \mid x') \vee \underline{\mu}_{\widetilde{B}^2}(y \mid x') \vee \cdots \vee \underline{\mu}_{\widetilde{B}^M}(y \mid x'), \\ \overline{\mu}_{\widetilde{B}}(y \mid x') = \overline{\mu}_{\widetilde{B}^1}(y \mid x') \vee \overline{\mu}_{\widetilde{B}^2}(y \mid x') \vee \cdots \vee \overline{\mu}_{\widetilde{B}^M}(y \mid x') \end{cases} \tag{12}$$

where $\vee$ denotes the maximum operation.

Finally the type-reduced set $Y_C(x')$ can be obtained by computing the centroid of $\widetilde{B}$, i.e.,

$$Y_C(x') = 1/[l_{\widetilde{B}}(x'), r_{\widetilde{B}}(x')] \tag{13}$$

where the two end points $l_{\widetilde{B}}(x')$ and $r_{\widetilde{B}}(x')$ can be calculated by different types of TR algorithms [13–24,29].

3. Weighted-based noniterative algorithms

Before introducing the weighted-based noniterative algorithms, we first give the preliminary knowledge: Newton-Cotes quadrature formulas [16,30].

3.1. Newton-Cotes quadrature formulas

Generally speaking, the numerical integration is an approach that approximates the definite integral $\int_a^b f(x)dx$ according to the functional value $f(x_i)$ on some certain discrete points. Therefore, the calculation of definite integral can be ascribed to computing functional values.

Definition 6. (Quadrature formula [16,17,46,47]) Let the discrete points satisfy that $a = x_0 < x_1 < x_2 < \cdots < x_N = b$, and the definite integral $\int_a^b f(x)dx = Q(f) + E(f)$ with the property:

$$Q(f) = \sum_{i=0}^{N} w_i f(x_i) = w_0 f(x_0) + \cdots + w_N f(x_N) \tag{14}$$

then Eq (14) is referred to as the numerical integration or quadrature formula, where $E(f)$ is called as the remainder or truncation error of integration, $\{w_i\}_{i=0}^N$ is called as the weight coefficient, and $\{x_i\}_{i=0}^N$ is the integration node.

Next, the composite trapezoidal rule, composite Simpson rule, and composite Simpson 3/8 rule to adopted to approximate $f(x)$ as the straight line, quadratic polynomial function, and cubic polynomial function, respectively.

Theorem 1. (Composite trapezoidal rule [16,17,46,47]) Let $y = f(x)$ be a function defined on $[a,b]$. Divide the interval $[a,b]$ into N subintervals $\{x_{i-1}, x_i\}_{i=1}^N$ with the equidistance $h = \frac{b-a}{N}$, where the equidistance node is as $x_i = x_0 + ih (i = 0,1,\cdots,N)$, then the numerical approximation of definite integral with the composite trapezoidal rule is as

$$\int_a^b f(x)dx = \frac{h}{2}[f(a) + f(b) + 2\sum_{i=1}^{N-1} f(x_i)] + E_T(f,h) \tag{15}$$

Suppose that f be second order continuous differentiable on $[a,b]$, then the remainder $E_T(f,h) = -\frac{(b-a)f''(\zeta)}{12}h^2$, where $\zeta \in (a,b)$.

Theorem 2. (Composite Simpson rule [16,17,46,47]) Let $y = f(x)$ be a function defined on $[a,b]$. Divide the interval $[a,b]$ into $2N$ subintervals $\{x_{i-1}, x_i\}_{i=1}^{2N}$ with the equidistance $h = \frac{b-a}{2N}$, where the equidistance node is as $x_i = x_0 + ih (i = 0,1,\cdots,2N)$, then the numerical approximation of definite integral with the composite Simpson rule is as

$$\int_a^b f(x)dx = \frac{h}{3}[f(a) + f(b) + 2\sum_{i=1}^{N-1} f(x_{2i}) + 4\sum_{i=0}^{N-1} f(x_{2i+1})] + E_S(f,h) \tag{16}$$

Suppose that f be fourth order continuous differentiable on $[a,b]$, then the remainder $E_S(f,h) = -\frac{(b-a)f^{(4)}(\zeta)}{180}h^4$, where $\zeta \in (a,b)$.

Theorem 3. (Composite Simpson 3/8 rule [16,17,46,47]) Let $y = f(x)$ be a function defined on

$[a,b]$. Divide the interval $[a,b]$ into $3N$ subintervals $\{x_{i-1},x_i\}_{i=1}^{3N}$ with the equidistance $h=\frac{b-a}{3N}$, where the equidistance node is as $x_i=x_0+ih(i=0,1,\cdots,3N)$, then the numerical approximation of definite integral with the composite Simpson 3/8 rule is as

$$\int_a^b f(x)dx=\frac{3h}{8}[f(a)+f(b)+\sum_{i=1}^{N}2f(x_{3i})+\sum_{i=1}^{N}3f(x_{3i-2})+\sum_{i=1}^{N}3f(x_{3i-1})]+E_{SC}(f,h) \tag{17}$$

Suppose that f be fourth order continuous differentiable on $[a,b]$, then the remainder $E_{SC}(f,h)=-\frac{(b-a)f^{(4)}(\zeta)}{80}h^4$, where $\zeta\in(a,b)$.

Here all the integrals are measured in the Lebesgue sense.

3.2. Weighted-based noniterative algorithms

3.2.1. Weighted Nagar-Bardini algorithms

IT2 FLSs based on the closed form of Nagar-Bardini (NB) algorithms [19,46,47] can make distinctly improvement on coping with uncertainties. For the centroid output IT2 FS $\widetilde{B}$, suppose that the primary variable y be equally discretized into N points, i.e., $y_1<y_2<\cdots<y_N$, then the left and right centroid interval can be computed as:

$$l_{\widetilde{B}}=\frac{\sum_{i=1}^{N}y_i\underline{\mu}_{\widetilde{B}}(y_i)}{\sum_{i=1}^{N}\underline{\mu}_{\widetilde{B}}(y_i)},\ \text{and}\ \ r_{\widetilde{B}}=\frac{\sum_{i=1}^{N}y_i\overline{\mu}_{\widetilde{B}}(y_i)}{\sum_{i=1}^{N}\overline{\mu}_{\widetilde{B}}(y_i)}. \tag{18}$$

Then the defuzzified output can be obtained as:

$$y_{NB}=\frac{l_{\widetilde{B}}+r_{\widetilde{B}}}{2}. \tag{19}$$

Similar to the continuous KM types of algorithms [13–17,31], the continuous NB (CNB) algorithms can be adopted for studying the theoretical property of centroid TR and defuzzification of IT2 FLSs.

Let $a=y_1<y_2<\cdots<y_N=b$, where a and b are the left and right end point of y, respectively, then the CNB algorithms calculate the centroids as:

$$l_{\widetilde{B}}=\frac{\int_a^b y\underline{\mu}_{\widetilde{B}}(y)dy}{\int_a^b\underline{\mu}_{\widetilde{B}}(y)dy},\ \text{and}\ \ r_{\widetilde{B}}=\frac{\int_a^b y\overline{\mu}_{\widetilde{B}}(y)dy}{\int_a^b\overline{\mu}_{\widetilde{B}}(y)dy}. \tag{20}$$

Here the output centroid two end points can be computed without iterations. Furthermore, the output is a linear combination of the output of two T1 FLSs: one constructed from the LMFs, and the other constructed from the UMFs.

In this section, we propose a type of weighted NB (WNB) algorithms, i.e.,

$$l_{\tilde{B}} = \frac{\sum_{i=1}^{N} w_i y_i \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} w_i \underline{\mu}_{\tilde{B}}(y_i)}, \text{ and } r_{\tilde{B}} = \frac{\sum_{i=1}^{N} w_i y_i \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} w_i \overline{\mu}_{\tilde{B}}(y_i)}. \tag{21}$$

WNB algorithms can be considered as the numerical implementation of CNB algorithms. Comparing Eqs (18) and (20), it is found that the CNB and NB algorithms are very similar, i.e., the sum operations in the discrete version are transformed into the definite integral operations in the continuous version, i.e., the sum operations for the sampling points y_i play the role of integrating for the integrand.

Table 1. Weights assignment for the weighted-based noniterative algorithms.

Algorithms	Integration rule	Weights
NB,NT,BMM	———————	$w_i = 1 (i = 1, \cdots, N)$
TWNB,TWNT,TWBMM	Composite Trapezoidal rule	$w_i = \begin{cases} 1/2, i = 1, N, \\ 1, i \neq 1, N. \end{cases}$
SWNB,SWNT,SWBMM	Composite Simpson rule	$w_i = \begin{cases} 1/2, i = 1, N, \\ 1, i = 1 \bmod(2), i \neq 1, N, \\ 2, i = 0 \bmod(2), i \neq N. \end{cases}$
S3/8WNB,S3/8WNT,S3/8WBMM	Composite Simpson 3/8 rule	$w_i = \begin{cases} 1/3, i = 1, N, \\ 2/3, i = 1 \bmod(3), i \neq 1, N, \\ 1, i = 2 \bmod(3), i \neq N, \\ 1, i = 0 \bmod(2), i \neq N. \end{cases}$

According to the quadrature formula (see Eq (14)), the corresponding weights w_i for every MF of sampling points y_i can be assigned appropriately, then the more accurate computational results may be obtained. Actually, many types of weights assignment method can be used. However, this paper only considers the numerical integration methods based on the Composite trapezoidal rule, Composite Simpson rule, and Composite Simpson 3/8 rule in Newton-Cotes quadrature formulas. And the corresponding WNB algorithms are referred to as the TWNB, SWNB, and S3/8WNB algorithms, respectively. Table 1 provides the weights assignment approach for the three types of weighted-based noniterative algorithms.

3.2.2. Weighted Nie-Tan algorithms

The closed form of discrete Nie-Tan (NT) algorithms can compute the centroid output of $\tilde{B}$ straightly as:

$$y_{NT} = \frac{\sum_{i=1}^{N} y_i [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]}{\sum_{i=1}^{N} [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]}. \tag{22}$$

Most recent studies prove the continuous NT (CNT) algorithms [20] to be an accurate method for computing the centroids of IT2 FSs, i.e.,

$$y_{CNT} = \frac{\int_a^b y [\underline{\mu}_{\tilde{B}}(y) + \overline{\mu}_{\tilde{B}}(y)] dy}{\int_a^b [\underline{\mu}_{\tilde{B}}(y) + \overline{\mu}_{\tilde{B}}(y)] dy}. \tag{23}$$

This section proposes a type of weighted NT (WNT) algorithms, i.e.,

$$y_{WNT} = \frac{\sum_{i=1}^{N} w_i y_i [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]}{\sum_{i=1}^{N} w_i [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]}. \tag{24}$$

WNT algorithms can be viewed as the numerical implementation of CNT algorithms. The sum operations in the discrete NT algorithms are transformed into the definite integral operations in the continuous NT (CNT) algorithms, i.e., the sum operations for the sampling points y_i act as integrating for the integrand. Therefore, we can assign weights for every MF of sampling points y_i to try to obtain more accurate computational results. Here the corresponding WNT algorithms are referred to as the TWNT, SWNT, and S3/8WNT algorithms, respectively.

3.2.3. Weighted Begian-Melek-Mendel algorithms

Begian-Melek-Mendel (BMM) algorithms can also obtain the output of IT2 FLSs directly, i.e.,

$$y_{BMM} = \alpha \frac{\sum_{i=1}^{N} y_i \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} \underline{\mu}_{\tilde{B}}(y_i)} + \beta \frac{\sum_{i=1}^{N} y_i \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} \overline{\mu}_{\tilde{B}}(y_i)} \tag{25}$$

where α and β are two adjustable coefficients.

IT2 FLSs based on BMM algorithms [13,21,22] are superior to T1 FLSs counterparts on both robustness and stability. In addition, the BMM algorithms are more generalized form of NB and NT algorithms. Observing the Eqs (19) and (25), it can be found that BMM and NB algorithms are exactly the same while $\alpha = \alpha_{NB} = \frac{1}{2}$, and $\beta = \beta_{NB} = \frac{1}{2}$. For the NT algorithms, the Eq (22) can be simply transformed to:

$$
\begin{aligned}
y_{NT} &= \frac{\sum_{i=1}^{N} \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]} \times \frac{\sum_{i=1}^{N} y_i \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} \underline{\mu}_{\tilde{B}}(y_i)} + \frac{\sum_{i=1}^{N} \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]} \times \frac{\sum_{i=1}^{N} y_i \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} \overline{\mu}_{\tilde{B}}(y_i)} \\
&= \alpha_{NT} \frac{\sum_{i=1}^{N} y_i \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} \underline{\mu}_{\tilde{B}}(y_i)} + \beta_{NT} \frac{\sum_{i=1}^{N} y_i \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} \overline{\mu}_{\tilde{B}}(y_i)}
\end{aligned} \tag{26}
$$

in which

$$
\alpha_{NT} = \frac{\sum_{i=1}^{N} \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]}, \text{ and } \beta_{NT} = \frac{\sum_{i=1}^{N} \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} [\underline{\mu}_{\tilde{B}}(y_i) + \overline{\mu}_{\tilde{B}}(y_i)]}. \tag{27}
$$

Comparing the Eqs (25) and (26), it can be found that, as $\alpha = \alpha_{NT}$ and $\beta = \beta_{NT}$, BMM and NT algorithms become the same.

Continuous BMM (CBMM) algorithms can also be used for studying the theoretical properties of TR of IT2 FLSs, i.e.,

$$
y_{CBMM} = \alpha \frac{\int_a^b y \underline{\mu}_{\tilde{B}}(y) dy}{\int_a^b \underline{\mu}_{\tilde{B}}(y) dy} + \beta \frac{\int_a^b y \overline{\mu}_{\tilde{B}}(y) dy}{\int_a^b \overline{\mu}_{\tilde{B}}(y) dy}. \tag{28}
$$

Based on the quadrature formula, this section gives a type of weighted BMM (WBMM) algorithms, i.e.,

$$
y_{WBMM} = \alpha \frac{\sum_{i=1}^{N} w_i y_i \underline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} w_i \underline{\mu}_{\tilde{B}}(y_i)} + \beta \frac{\sum_{i=1}^{N} w_i y_i \overline{\mu}_{\tilde{B}}(y_i)}{\sum_{i=1}^{N} w_i \overline{\mu}_{\tilde{B}}(y_i)}. \tag{29}
$$

Here WBMM algorithms can be considered as the numerical implementation of CBMM algorithms. Comparing Eqs (25) and (28), it is found that the CBMM and BMM algorithms are very similar, i.e., the sum operations in the discrete version are transformed into the definite integral operations in the continuous version, i.e., the sum operations for the sampling points y_i play the role of integrating for the integrand. This paper considers the numerical integration methods based on the Composite trapezoidal rule, Composite Simpson rule, and Composite Simpson 3/8 rule in Newton-Cotes quadrature formulas. And the corresponding WBMM algorithms are referred to as the TWBMM, SWBMM, and S3/8WBMM algorithms, respectively.

For the above three types of weighed-based noniterative algorithms (see Table 1), suppose that the primary variable be the letter x, and x is uniformly sampled on $[a,b]$, i.e., $x_i = a + \frac{i-1}{N-1}(b-a)$, $i = 1, \cdots, N$, and mod represents the modular arithmetic.

Next we can make conclusions about assigning weights for three types of weighted-based noniterative algorithms as:

1) Substitute $x_i(i=0,1,\cdots,N)$, $x_0=a$, $x_N=b$ in Eq (15), $x_i(i=0,1,\cdots,2N)$, $x_0=a$, $x_{2N}=b$ in Eq (16), and $x_i(i=0,1,\cdots,3N)$, $x_0=a$, $x_{3N}=b$ in Eq (17) all by $x_i(i=1,\cdots,N)$, $x_1=a$, $x_N=b$.
2) The coefficients $h/2$, $h/3$, $3h/8$ in Eqs (15)-(17) can be cancelled straightly by the quotient of two definite integrals.
3) In Tables 1–3, for the TWNB, TWNT, TWBMM, and SWNB, SWNT, SWBMM algorithms, the weights are assigned as $1/2$ of the right of Eqs (15) and (16); while for the S3/8WNB, S3/8WNT, and S3/8WBMM algorithms, the weighted are assigned as $1/3$ of the right of Eq (17).
4) In Tables 1–3, for the SWNB, SWNT, and SWBMM, and S3/8WNB, S3/8WNT, and S3/8WBMM algorithms, the number of sampling N is not only restricted to $N=2n+1$ and $N=3n+1$ (as the requirements of $N=1\bmod(2)$ and $N=1\bmod(3)$ in Eqs (16) and (17), where n is an integer).

Finally the inner relations between weighted-based noniterative algorithms and continuous noniterative algorithms for performing the centroid TR of IT2 FLSs can be made as:

1) Weighted-based noniterative algorithms calculate the type-reduced set $Y_{WNontierative}$ based on the linear combinations of functional values on sampling points. While the benchmark continuous noniterative algorithms compute according to the integral operations. In theory, the solutions of weighted-based noniterative algorithms will approach to the continuous noniterative algorithms as the number of sampling $N\to\infty$.
2) As the number of sampling increases, weighted-based noniterative algorithms may obtain more accurate computational results.
3) Weighted-based noniterative algorithms perform the numerical calculations according to the sum operation, whereas the continuous noniterative algorithms perform the calculations symbolically by means of the integral operations. On the whole, weighted-based noniterative algorithms can be viewed as the numerical implementation of continuous noniterative algorithms according to the numerical integration approaches.

4. Simulations

Four computer simulation examples are provided in this section. Here we suppose that the centroid output IT2 FS [16,18,29,32] has been obtained by merging or weighting fuzzy rules under the guidance the inference before the TR and defuzzification. For the first example, the FOU is bounded by the piece-wise linear functions. For the second example, the FOU is bounded by both the piece-wise linear functions and Gaussian functions. For the third example, the FOU is bounded by the Gaussian functions. For the last example, the FOU is defined as the symmetric Gaussian primary MF with uncertainty derivations. Then Figure 2 and Table 2 show the defined FOUs for four examples. In examples 1, 3 and 4, here we choose the primary variable $x\in[0,10]$ for tests. In example 2, the primary variable is selected as $x\in[-5,15]$ for test.

Firstly, the CNB, CNT, and CBMM algorithms are considered the benchmarks to compute the centroid defuzzified values for four examples. And they are provided in Table 3. Here the adjustable

coefficient α for CBMM algorithms is chosen as the average of 1000 random numbers distributed on [0, 1], and the other adjustable coefficient is defined as $\beta = 1-\alpha$.

Table 2. MF expressions for FOUs.

Num	Expression
1	$\underline{\mu}_{\tilde{A}_1}(x) = \max\{\begin{bmatrix} \frac{x-1}{6}, & 1 \le x \le 4 \\ \frac{7-x}{6}, & 4 < x \le 7 \\ 0, & \text{otherwise} \end{bmatrix}, \begin{bmatrix} \frac{x-3}{6}, & 3 \le x \le 5 \\ \frac{8-x}{9}, & 5 < x \le 8 \\ 0, & \text{otherwise} \end{bmatrix}\},$ $\overline{\mu}_{\tilde{A}_1}(x) = \max\{\begin{bmatrix} \frac{x-1}{2}, & 1 \le x \le 3 \\ \frac{7-x}{4}, & 3 < x \le 7 \\ 0, & \text{otherwise} \end{bmatrix}, \begin{bmatrix} \frac{x-2}{5}, & 2 \le x \le 6 \\ \frac{16-2x}{5}, & 6 < x \le 8 \\ 0, & \text{otherwise} \end{bmatrix}\}$
2	$\underline{\mu}_{\tilde{A}_2}(x) = \begin{cases} \frac{0.6(x+5)}{19}, -5 \le x \le 2.6 \\ \frac{0.4(14-x)}{19}, 2.6 < x \le 14 \end{cases}, \overline{\mu}_{\tilde{A}_2}(x) = \begin{cases} \exp[-\frac{1}{2}(\frac{x-2}{5})^2], -5 \le x \le 7.185 \\ \exp[-\frac{1}{2}(\frac{x-9}{1.75})^2], 7.185 < x \le 14 \end{cases}$
3	$\underline{\mu}_{\tilde{A}_3}(x) = \max\{0.5\exp[-\frac{(x-3)^2}{2}], 0.4\exp[-\frac{(x-6)^2}{2}]\},$ $\overline{\mu}_{\tilde{A}_3}(x) = \max\{\exp[-0.5\frac{(x-3)^2}{4}], 0.8\exp[-0.5\frac{(x-6)^2}{4}]\},$
4	$\underline{\mu}_{\tilde{A}_4}(x) = \exp[-\frac{1}{2}(\frac{x-3}{0.25})^2], \ \overline{\mu}_{\tilde{A}_4}(x) = \exp[-\frac{1}{2}(\frac{x-3}{1.75})^2],$

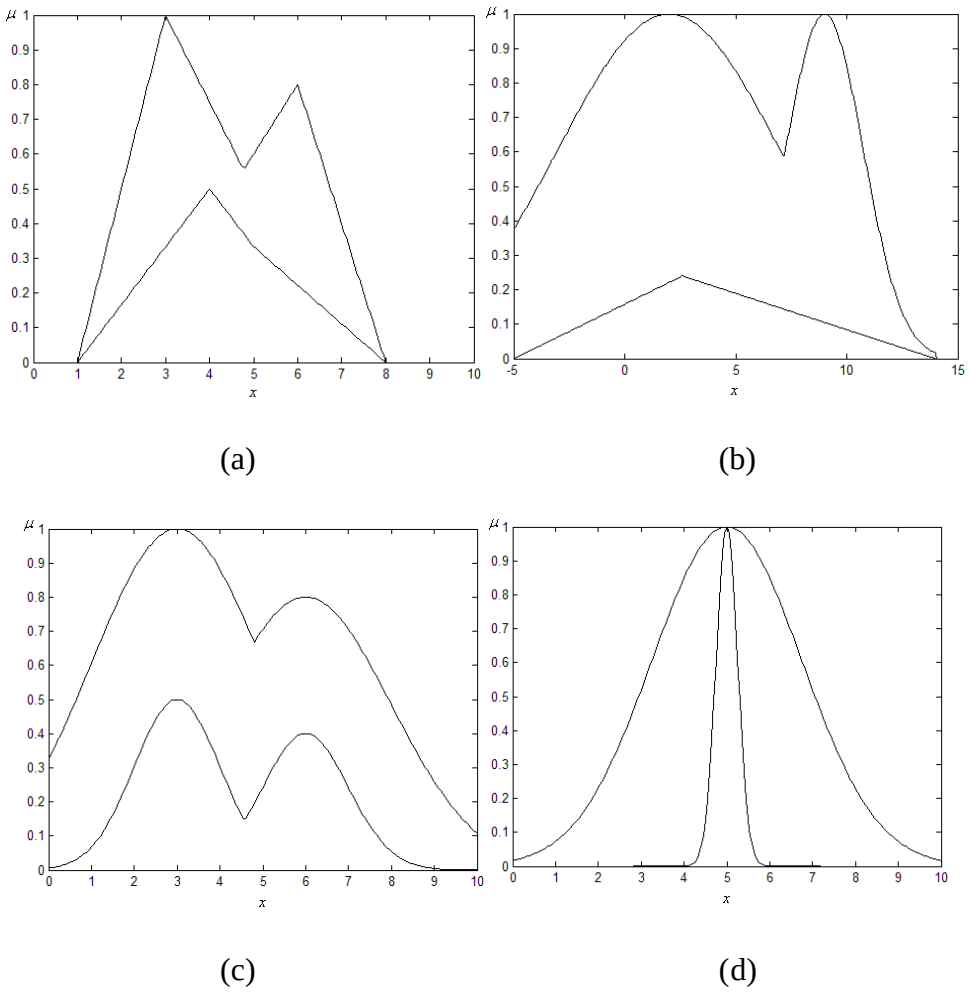

Figure 2. Graphs of FOUs, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

Table 3. Defuzzified values obtained by three types of continuous noniterative algorithms.

Num	Approach		
	CNB	CNT	CBMM
1	4.3050	4.3208	4.3041
2	3.7774	3.7141	3.7747
3	4.3702	4.3953	4.3690
4	5.0000	5.0000	5.0000

Next, we study the performances of proposed three types of weighted-based noniteraitve algorithms, respectively. Here the number of sampling is chosen as $N = 50:50:4000$ for tests, then the graphs of centroid defuzzified values computed by weighted-based noniterative algorithms are shown in Figures 3–5, respectively. Furthermore, the functional graphs of absolute errors between the continuous noniteraive algorithms and weighted-based noniteraive algorithms are shown in Figures 6–8.

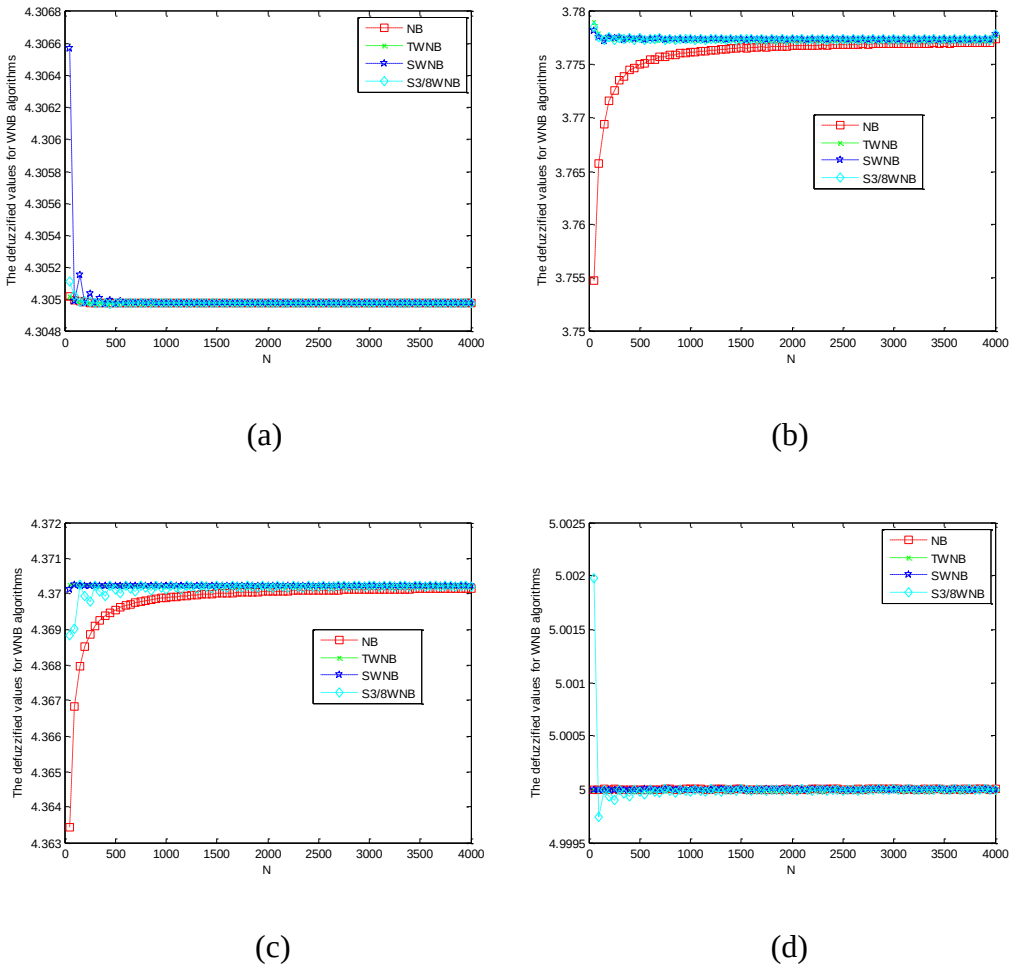

Figure 3. Defuzzified values computed by the WNB algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

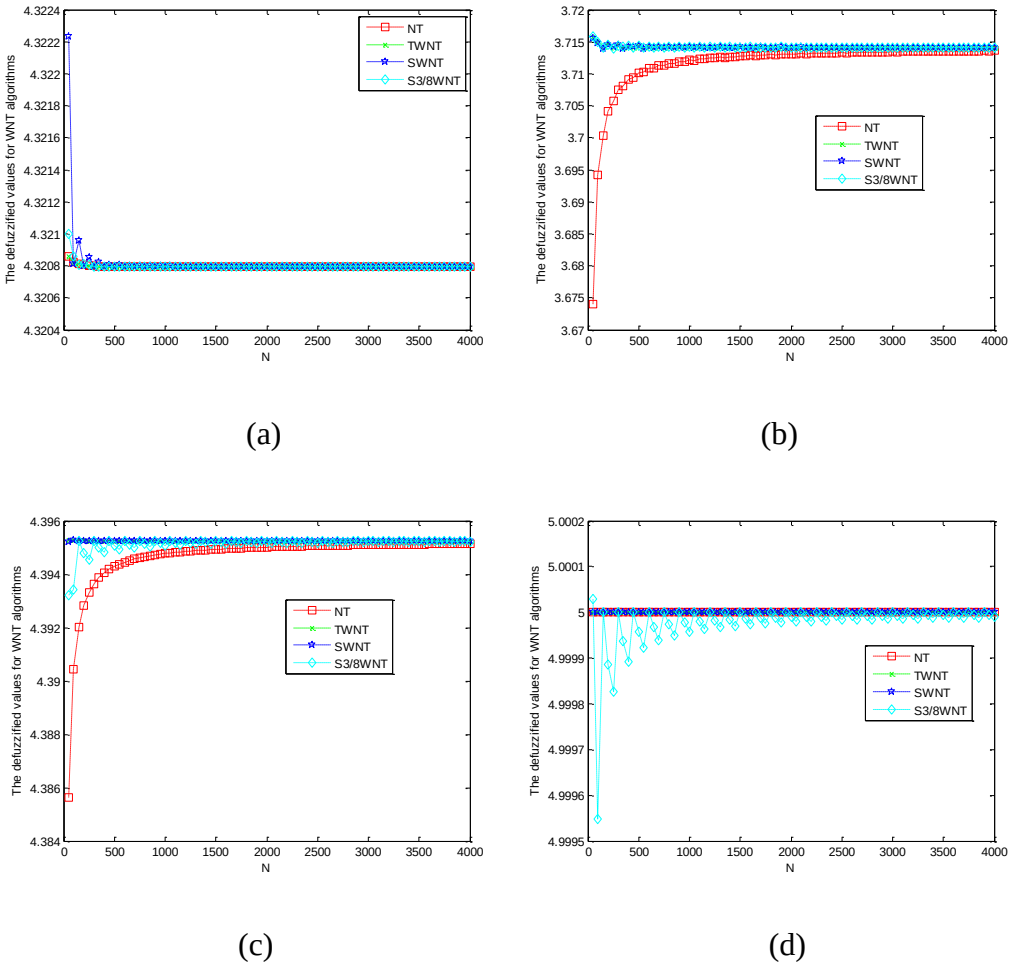

Figure 4. Defuzzified values computed by the WNT algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

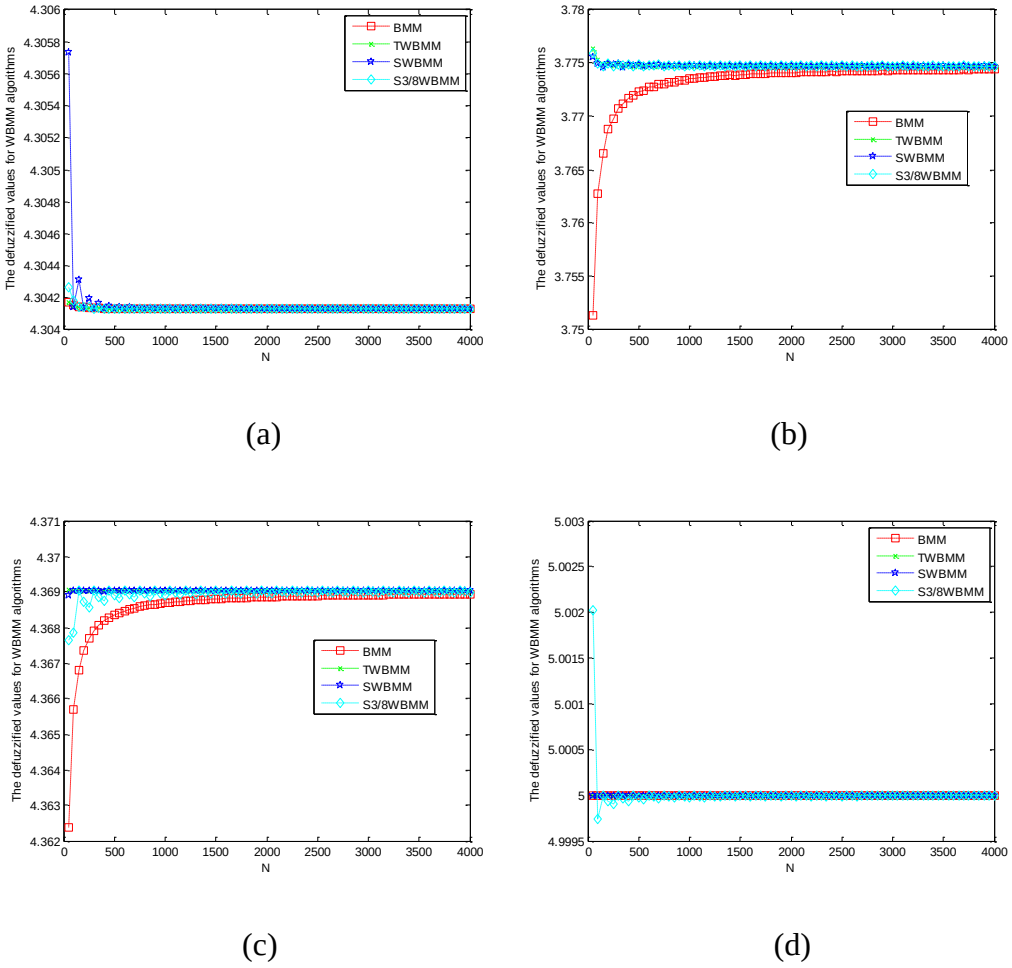

Figure 5. Defuzzified values computed by the WBMM algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

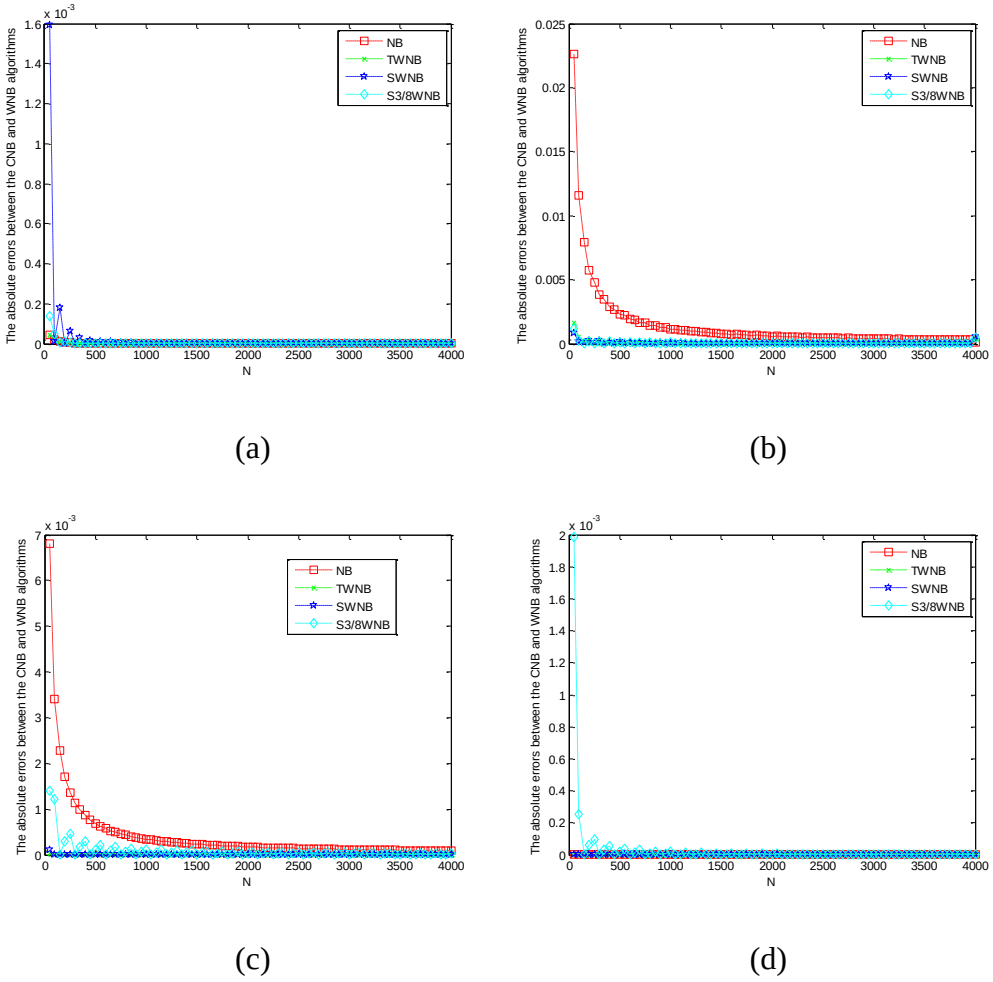

Figure 6. Absolute errors of defuzzified values between the CNB and WNB algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

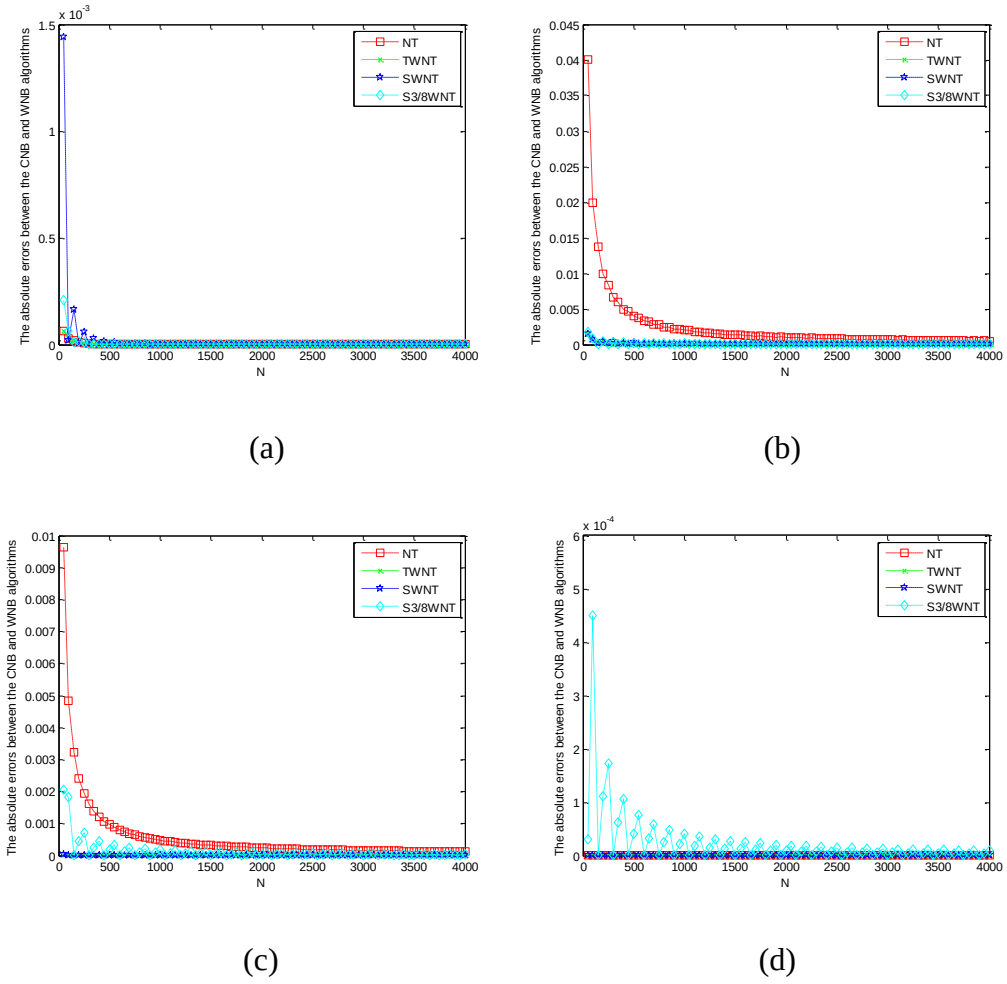

Figure 7. Absolute errors of defuzzified values between the CNT and WNT algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

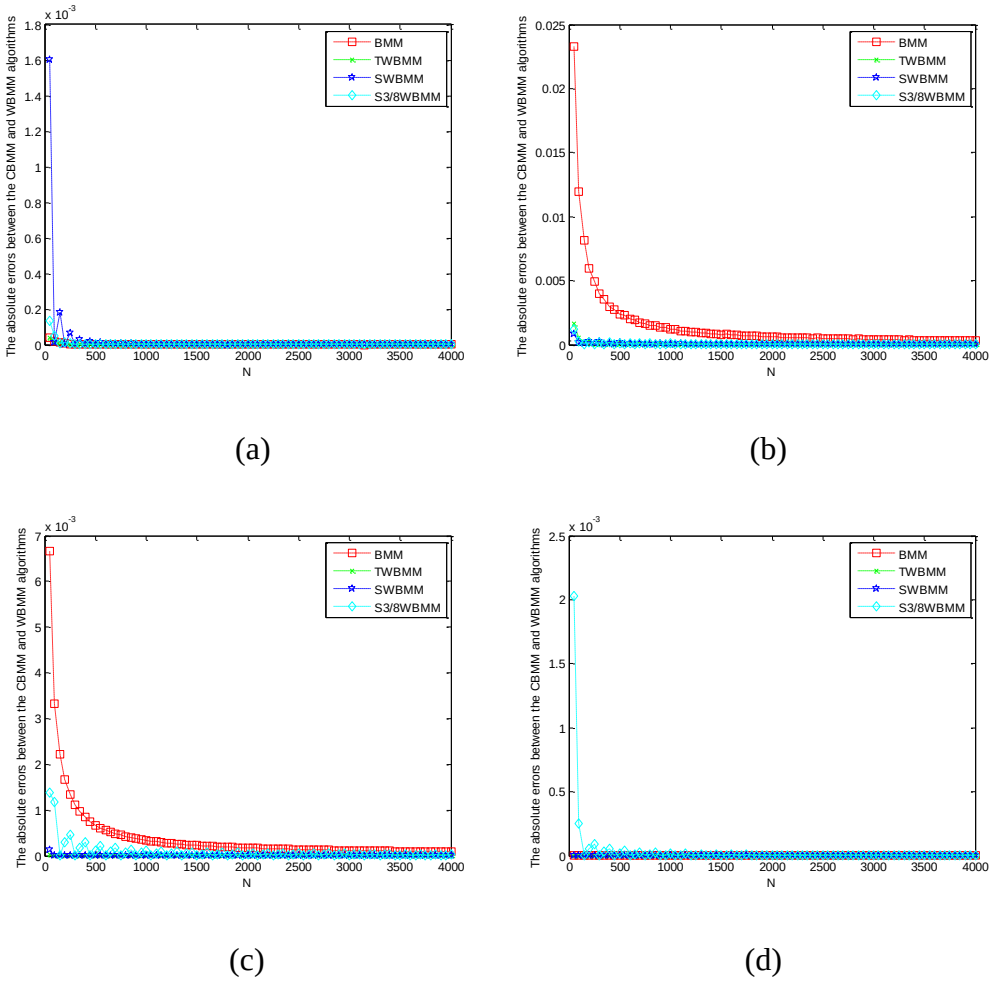

Figure 8. Absolute errors of defuzzified values between the CBMM and WBMM algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

Next, let's quantitatively measure the calculation accuracies of proposed weighted-based noniterative algorithms. For the number of sampling $N = 50:50:4000$, we compute the defined average relative errors $|y_{WNoniterivei} - y^*_{CNoniterativei}| / |y^*_{CNoniterativei}| (i = 1,\cdots,4)$. Then Tables 4–6 provide the values of average of relative errors with respect to N, in which the last line represents the total averages for four examples.

Table 4. Averages of relative errors for $(|y_{WNB_i} - y_{CNB_i}| / |y_{CNB_i}| (i = 1,\cdots,4)$.

Algorithm	NB	TWNB	SWNB	S3/8WNB
Example 1	0.000029	0.000029	0.000572	0.000067
Example 2	0.038400	0.001690	0.001450	0.001390
Example 3	0.009800	0.000094	0.000122	0.001908
Example 4	0.000110	0.000110	0.000110	0.000731
Total average	0.016110	0.000640	0.000750	0.001370

Table 5. Averages of relative errors for $|y_{WNT_i} - y_{CNT_i}|/|y_{CNT_i}|(i=1,\cdots,4)$.

Algorithm	NT	TWNT	SWNT	S3/8WNT
Example 1	0.000043	0.000043	0.000519	0.00096
Example 2	0.067610	0.002170	0.002430	0.002280
Example 3	0.013730	0.000050	0.000050	0.002770
Example 4	0.000028	0.000028	0.000028	0.000458
Total average	0.027140	0.000760	0.001010	0.001870

Table 6. Averages of relative errors for $|y_{WBMM_i} - y_{CBMM_i}|/|y_{CBMM_i}|(i=1,\cdots,4)$.

Algorithm	BMM	TWBMM	SWBMM	S3/8WBMM
Example 1	0.000028	0.000028	0.000576	0.000056
Example 2	0.039650	0.001590	0.001320	0.001290
Example 3	0.009598	0.000096	0.000125	0.001866
Example 4	0.000112	0.000112	0.000112	0.000738
Total average	0.016460	0.000610	0.000710	0.001320

Then we study the specific computation times of weighted-based noniterative algorithms for better applications. The number of sampling is still chosen as $N = 50:50:4000$. Here the unrepeatable computation times depend on the hardware and software environments. The simulations are performed by a dual-core CPU dell desktop with E5300@2.60GHz and 2.00 GB memory. The programs are operated by Matlab 2013a on Windows XP. Figures 9–11 provide the comparisons of computation times of weighted-based noniteraitve algorithms for these four examples.

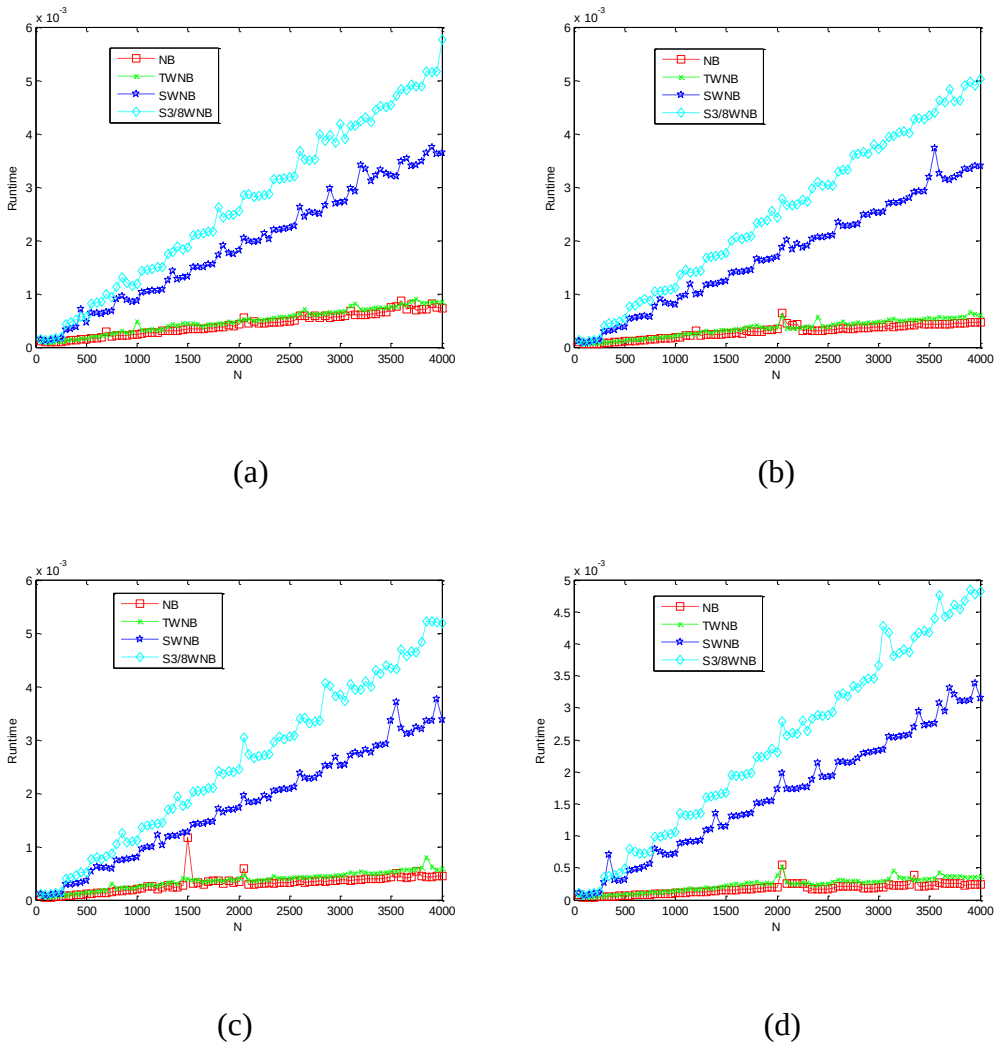

Figure 9. Comparsions of computation times for WNB algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

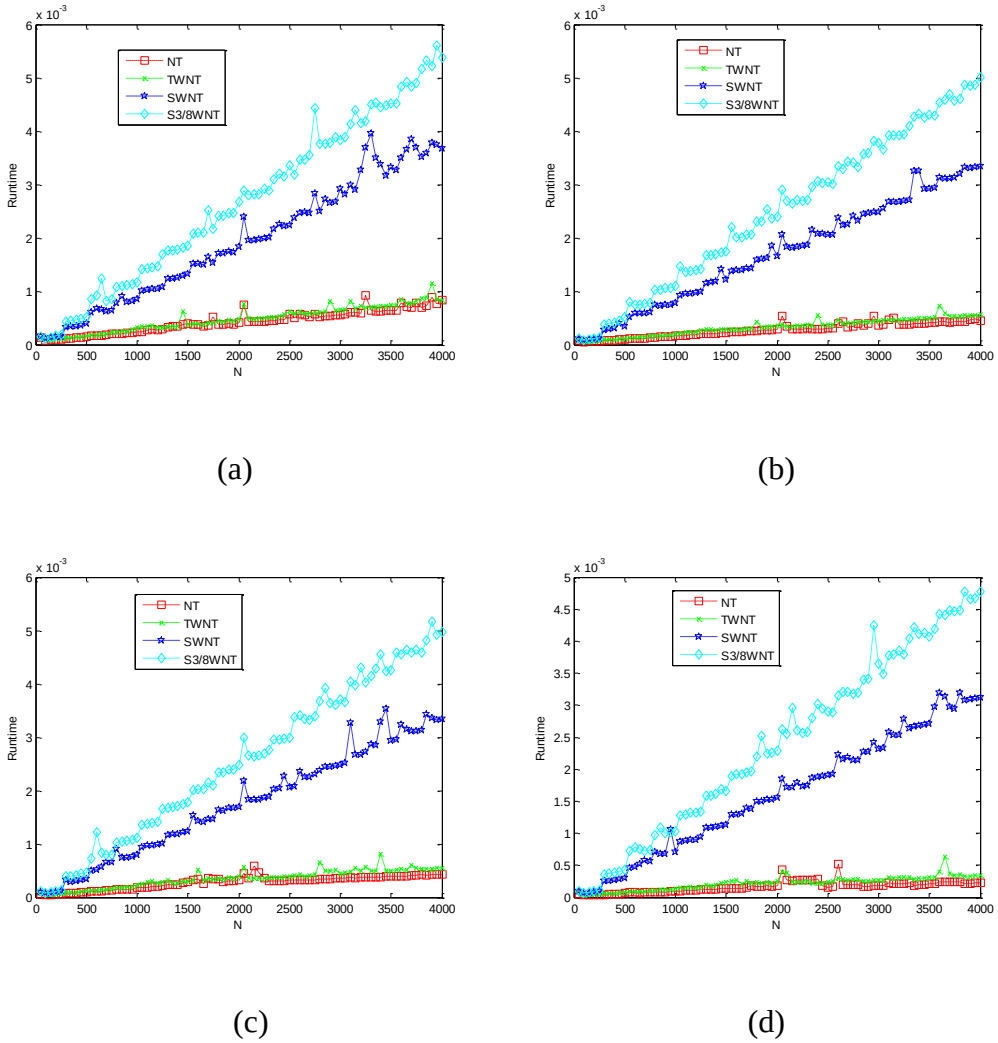

Figure 10. Comparsions of computation times for WNT algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

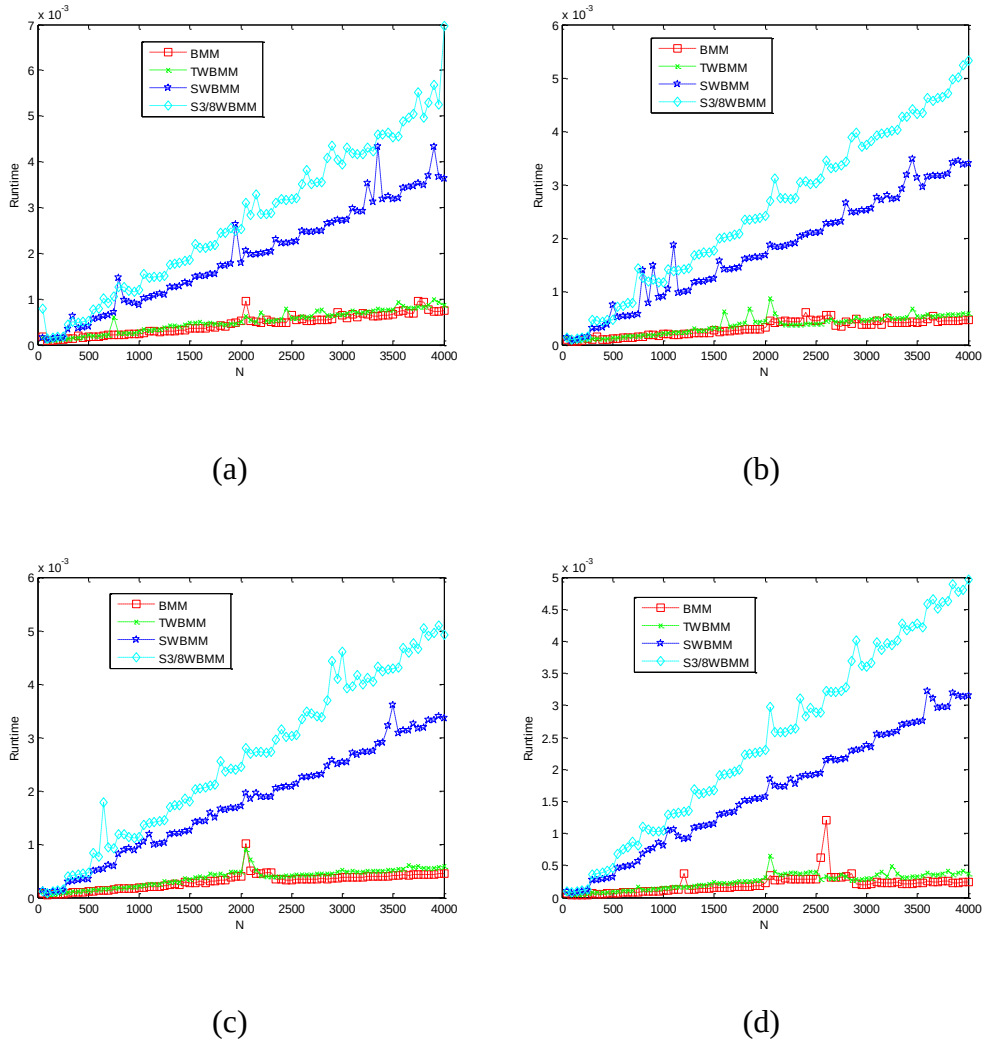

Figure 11. Comparsions of computation times for WBMM algorithms, (a) example 1; (b) example 2; (c) example 3; and (d) example 4.

Considering the Figures 3–8, Tables 4–6, and Figures 9–11 comprehensively, the following conclusions can be obtained:

1) In these four examples, the absolute errors of three types of weighted-based noniterative algorithms all converge as the number of sampling increases. In example 1, NB, TWNB, NT, TWNT, and BMM, TWBMM algorithms can obtain the smallest absolute errors and errors amplitudes of variation, while SNB, SNT, and SBMM algorithms get the largest absolute errors and errors amplitude of variation. In both examples 2 and 3, the proposed weighted-based noniterative can obtain the values of absolute errors and errors amplitudes of variation that are obviously less than their corresponding original noniterative algorithms. In the last example, the first three types of weighted-based noniterative algorithms can get almost the same absolute errors and errors amplitudes of variation, while the last type of weighted-based noniterative algorithms obtain the larger ones.
2) For the NB algorithms, the largest average of relative error is 3.84%. While the largest average

of relative error of proposed WNB algorithms is only 0.169%. For the NT algorithms, the largest average of relative error is 6.761%. The largest average of relative error of proposed WNT algorithms is only 0.243%. For the BMM algorithms, the largest average of relative error is 3.965%. The largest average of relative error of proposed WBMM algorithms is only 0.159%.

3) The total mean of average of relative error of NB algorithms is 1.611%. While the largest total mean of average of relative error of proposed WNB algorithms is only 0.137%. The total mean of average of relative error of NT algorithms is 2.714%. While the largest total mean of average of relative error of proposed WNT algorithms is only 0.187%. The total mean of average of relative error of BMM algorithms is 1.646%. While the largest total mean of average of relative error of proposed WBMM algorithms is only 0.132%.
4) In general, see Figures 9–11, the computational speeds of original noniteraive algorithms are faster than their corresponding weighted-based noniterative algorithms. However, the computational speeds first two types on weighted-based noniterative algorithms are almost completely the same. As the number of sampling is fixed, the size relation of computation times is as: S3/8WNoiteraive > SWNoiteraive > TWNoniterative > Noniterative. It may just because the weights of proposed weighted-based noniterative algorithms are more complex than the noniterative algorithms. In other words, the convergence speeds of proposed weighted-based noniteraive algorithms are faster than the original noniterative algorithms.
5) From the above analysis, it can be found that, by choosing the proposed weighted-based noniteraive algorithms appropriately, which can improve both the calculation accuracies and convergence speeds.

The proposed weighted-based noniteraive algorithms can be used to investigate the TR and defuzzification of IT2 FLSs. If only the computational accuracy were considered, the proposed three types of weighted-based noniteraive algorithms outperformed the original noniteraive algorithms, in which the second type of weighed-based noniteraive algorithms were the best. Moreover, the computation time of proposed weighted-based noniteraive algorithms were not much different from the original noniteraive algorithms. Considering the above analysis comprehensively, we advise to use the second or third types of weighted-based noniterative algorithms for the TR and defuzzification of IT2 FLSs with the combination of linear functions and nonlinear functions as in examples 1 and 4, and adopt the first or second types of weighted-based noniteraive algorithms for the TR and defuzzification of IT2 FLSs with the combination of linear and nonlinear functions and nonlinear functions as in examples 2 and 3.

Finally, it is important to point out that, we only focus on the experimental performances of weighted-based noniteraive algorithms. It can be obtained from the simulation examples that, compare with the noniteraive algorithms, the proposed weighted-based noniteraive algorithms can improve the computational accuracies. However, if the requirements of computational accuracy were not high, the weighted-based noniteraive algorithms can not show their advantages, as the simplest noniteraive algorithms could attain well results.

5. Conclusions and expectations

This paper compares the operations between three types of discrete noniterative algorithms with their corresponding continuous versions. According to the Newton-Cotes quadrature formulas in the

numerical integration technique, three types of noniterative algorithms are extended to the weighted-based noniterative algorithms. The continuous noniterative algorithms are considered as the benchmarks for performing the centroid TR and defuzzification of IT2 FLSs. Four simulation examples illustrate and analyze the computational accuracies and computation times of the proposed algorithms. Compared with the original noniterative algorithms, the proposed weighted-based noniteraive algorithms can obtain both higher calculation accuracies and faster convergence speeds.

In the future work, we will concentrate on designing the centroid TR of T2 FLSs [13–24,29,31,33,56] with weighted-based reasonable initialization enhanced Karnik-Mendel algorithms, the center-of-sets TR [12,13,34,45–49] of T2 FLSs, and seeking for global optimization algorithms [3–6,25–27,35–44,52,53] for designing and applying IT2 or GT2 FLSs in real world problems like forecasting, control [50,51,54,55] and so on.

Acknowledgments

The paper is supported by the National Natural Science Foundation of China (No. 61973146, No. 61773188, No. 61903167, No. 61803189), the Liaoning Province Natural Science Foundation Guidance Project (No. 20180550056), and Talent Fund Project of Liaoning University of Technology (No. xr2020002). The author is very thankful to Professor Jerry Mendel, who has given the author some important advices.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. O. Castillo, P. Melin, *Type-2 fuzzy logic theory and applications*, Berlin, Germany: Springer-Verlag, 2008.
2. H. Hagras, C. Wagner, Towards the wide spread use of type-2 fuzzy logic systems in real world applications, *IEEE Comput. Intell. M.,* **7** (2012), 14–24.
3. Y. Chen, D. Z. Wang, S. C. Tong, Forecasting studies by designing Mamdani interval type-2 fuzzy logic systems: with combination of BP algorithms and KM algorithms, *Neurocomputing*, **174** (2016), 1133–1146.
4. D. Z. Wang, Y. Chen, Study on permanent magnetic drive forecasting by designing Takagi Sugeno Kang type interval type-2 fuzzy logic systems, *T. I. Meas. Control*, **40** (2018), 2011–2023.
5. S. Barkat, A. Tlemcani, H. Nouri, Noninteracting adaptive control of PMSM using interval type-2 fuzzy logic systems, *IEEE T. Fuzzy Syst.,* **19** (2011), 925–936.
6. Y. Chen, D. Z. Wang, Forecasting by designing Mamdani general type-2 fuzzy logic systems optimized with quantum particle swarm optimization algorithms, *T. I. Meas. Control*, **41** (2019), 2886–2896.
7. B. Safarinejadian, P. Ghane, H. Monirvaghefi, Fault detection in non-linear systems based on type-2 fuzzy logic, *International Journal of Systems Sciences*, **46** (2015), 394–404.

8. C. S. Lee, M. H. Wang, H. Hagras, Type-2 fuzzy ontology and its application to personal diabetic-diet recommendation, *IEEE T. Fuzzy Syst.*, **18** (2010), 316–328.
9. O. Mendoza, P. Melin, O. Castillo, Interval type-2 fuzzy logic and modular networks for face recognition applications, *Appl. Soft Comput*., **9** (2009), 1377–1387.
10. A. Niewiadomski, On finity, countability, cardinalities, and cylindric extensions of type-2 fuzzy sets in linguistic summarization of databases, *IEEE T. Fuzzy Syst.*, **18** (2010), 532–545.
11. J. M. Mendel, R. I. John, F. L. Liu, Interval type-2 fuzzy logic systems made simple, *IEEE T. Fuzzy Syst.*, **14** (2006), 808–821.
12. J. M. Mendel, General type-2 fuzzy logic systems made simple: a tutorial, *IEEE T. Fuzzy Syst.*, **22** (2014), 1162–1182.
13. J. M. Mendel, On KM algorithms for solving type-2 fuzzy set problems, *IEEE T. Fuzzy Syst.*, **21** (2013), 426–446.
14. J. M. Mendel, F. L. Liu, Super-exponential convergence of the Karnik-Mendel algorithms for computing the centroid of an interval type-2 fuzzy set, *IEEE T. Fuzzy Syst.*, **15** (2007), 309–320.
15. D. R. Wu, J. M. Mendel, Enhanced Karnik-Mendel algorithms, *IEEE T. Fuzzy Syst.*, **17** (2009), 923–934.
16. X. W. Liu, J. M. Mendel, D. R. Wu, Study on enhanced Karnik-Mendel algorithms: initialization explanations and computation improvements, *Inform. Sciences*, **184** (2012), 75–91.
17. Y. Chen, D. Z. Wang, Study on centroid type-reduction of general type-2 fuzzy logic systems with weighted enhanced Karnik-Mendel algorithms, *Soft Comput.*, **22** (2018), 1361–1380.
18. Y. Chen, Study on centroid type-reduction of interval type-2 fuzzy logic systems based on noniterative algorithms, *Complexity*, **2019** (2019), 1–12.
19. A. M. EI-Nagar, M. EI-Bardini, Simplified interval type-2 fuzzy logic system based on new type-reduction, *J. Intell. Fuzzy Syst.*, **27** (2014), 1999–2010.
20. J. W. Li, R. John, S. Coupland, G. Kendall, On Nie-Tan operator and type-reduction of interval type-2 fuzzy sets, *IEEE T. Fuzzy Syst.*, **26** (2018), 1036–1039.
21. M. Biglarbegian, W. W. Melek, J. M. Mendel, On the robustness of type-1 and interval type-2 fuzzy logic systems in modeling, *Inform. Sciences*, **181** (2011), 1325–1347.
22. M. Biglarbegian, W. W. Melek, J. M. Mendel, On the stability of interval type-2 TSK fuzzy logic systems, *IEEE T. Cybernetics*, **40** (2010), 798–818.
23. S. Greenfield, F. Chiclana, S. Coupland, R. John, The collapsing method of defuzzification for discretised interval type-2 fuzzy sets, *Inform. Sciences*, **179** (2009), 2055–2069.
24. H. W. Wu, J. M. Mendel, Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems, *IEEE T. Fuzzy Syst.*, **10** (2002), 622–639.
25. M. d. l. A. Hernandez, P. Melin, G. M. Méndez, O. Castillo, I. López-Juarez, A hybrid learning method composed by the orthogonal least-squares and the back-propagation learning algorithms for interval A2-C1 type-1 non-singleton type-2 TSK fuzzy logic systems, *Soft Comput*., **19** (2015), 661–678.
26. Y. Chen, D. Z. Wang, W. Ning, Forecasting by TSK general type-2 fuzzy logic systems optimized with genetic algorithms, *Optim. Contr. Appl. Met.*, **39** (2018), 393–409.
27. A. Khosravi, S. Nahavandi, Load forecasting using interval type-2 fuzzy logic systems: optimal type reduction, *IEEE T. Ind. Inform.*, **10** (2014), 1055–1063.
28. C. Wagner, H. Hagras, Towards general type-2 fuzzy logic systems based on zSlices, *IEEE T. Fuzzy Syst.*, **18** (2010), 637–660.

29. S. Greenfield, F. Chiclana, Accuracy and complexity evaluation of defuzzification strategies for the discretised interval type-2 fuzzy set, *Int. J. Approx. Reason.,* **54** (2013), 1013–1033.
30. J. H. Mathews, K. K. Fink, *Numerical Methods Using Matlab*, Prentice-Hall Inc, Upper Saddle River, NJ, 2004.
31. X. W. Liu, J. M. Mendel, Connect Karnik-Mendel algorithms to root-finding for computing the centroid of an interval type-2 fuzzy set, *IEEE T. Fuzzy Syst.,* **19** (2011), 652–665.
32. Y. Chen, Study on sampling based discrete Nie-Tan algorithms for computing the centroids of general type-2 fuzzy sets, *IEEE Access*, **7** (2019), 156984–156992.
33. D. R. Wu, Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: overview and comparisons, *IEEE T. Fuzzy Syst.*, **21** (2013), 80–99.
34. M. A. Khanesar, A. Jalalian, O. Kaynak, Improving the speed of center of set type-reduction in interval type-2 fuzzy systems by eliminating the need for sorting, *IEEE T. Fuzzy Syst.,* **25** (2017), 1193–1206.
35. D. R. Wu, J. M. Mendel, Recommendations on designing practical interval type-2 fuzzy systems, *Eng. Appl. Artif. Intel.,* **85** (2019), 182–193.
36. F. Gaxiola, P. Melin, F. Valdez, J. R. Castro, O. Castillo, Optimization of type-2 fuzzy weights in backpropagation learning for neural networks using GAs and PSO, *Appl. Soft Comput.,* **38** (2016), 860–871.
37. C. H. Hsu, C. F. Juang, Evolutionary robot wall-following control using type- 2 fuzzy controller with species-de-activated continuous ACO, *IEEE T. Fuzzy Syst.,* **21** (2013), 100–112.
38. A. Khosravi, S. Nahavandi, D. Creighton, D. Srinivasan, Interval type-2 fuzzy logic systems for load forecasting: a comparative study, *IEEE T. Power Syst.,* **27** (2012), 1274–1282.
39. C. W. Tao, J. S. Taur, C. W. Chang, Y. H. Chang, Simplified type-2 fuzzy sliding controller for wing rocket system, *Fuzzy Sets Syst.,* **207** (2012), 111–129.
40. M. A. Sanchez, O. Castillo, J. R. Castro, Generalized type-2 fuzzy systems for controlling a mobile robot and a performance comparison with interval type-2 and type-1 fuzzy systems, *Expert Syst. Appl.,* **42** (2015), 5904–5914.
41. L. Cervantes, O. Castillo, Type-2 fuzzy logic aggregation of multiple fuzzy controllers for airplane flight control, *Inform. Sciences*, **324** (2015), 247–256.
42. O. Castillo, L. Amador-Angulo, J. R. Castro, M. Garcia-Valdez, A comparative study of type-1 fuzzy logic systems, interval type-2 fuzzy logic systems and generalized type-2 fuzzy logic systems in control problems, *Inform. Sciences*, **354** (2016), 257–274.
43. O. Castillo, P. Melin, E. Ontiveros, C. Peraza, P. Ochoa, F. Valdez, J. Soria, A high-speed interval type 2 fuzzy system approach for dynamic parameter adaptation in metaheuristics, *Eng. Appl. Artif. Intel.,* **85** (2019), 666–680.
44. E. Ontiveros-Robles, P. Melin, O. Castillo, Comparative analysis of noise robustness of type 2 fuzzy logic controllers, *Kybernetika*, **54** (2018), 175–201.
45. E. Ontiveros-Robles, P. Melin, O. Castillo, New methodology to approximate type-reduction based on a continuous root-finding karnik mendel algorithm, *Algorithms*, **10** (2017), 77–96.
46. Y. Chen, Study on weighted Nagar-Bardini algorithms for centroid type-reduction of interval type-2 fuzzy logic systems, *J. Intell. Fuzzy Syst.,* **34** (2018), 2417–2428.
47. Y. Chen, Study on weighted Nagar-Bardini algorithms for centroid type-reduction of general type-2 fuzzy logic systems, *J. Intell. Fuzzy Syst.,* **37** (2019), 6527–6544.

48. T. Kumbasar, Revisiting Karnik-Mendel algorithms in the framework of linear fractional programming, *Int. J. Approx. Reason.*, **82** (2017), 1–21.
49. Y. Chen, J. X. Wu, J. Lan, Study on reasonable initialization enhanced Karnik-Mendel algorithms for centroid type-reduction of interval type-2 fuzzy logic systems, *AIMS Math.*, **5** (2020), 6149–6168.
50. S. C. Tong, Y. M. Li, Robust adaptive fuzzy backstepping output feedback tracking control for nonlinear system with dynamic uncertainties, *Science China Information Sciences*, **53** (2010), 307–324.
51. S. C. Tong, Y. M. Li, Observer-based adaptive fuzzy backstepping control of uncertain pure-feedback systems, *Science China Information Sciences*, **57** (2014), 1–14.
52. Q. F. Fan, T. Wang, Y. Chen, et al, Design and application of interval type-2 fuzzy logic system based on QPSO algorithm, *Int. J. Fuzzy Syst.*, **20** (2018), 835–846.
53. M. Deveci, I. Z. Akyurt, S. Yavuz, GIS-based interval type-2 fuzzy set for public bread factory site selection, *Journal of Enterprise Information Management*, **31** (2018), 820–847.
54. L. Liu, Y. J. Liu, S. C. Tong, C. L. P. Chen, Integral barrier Lyapunov function based adaptive control for switched nonlinear systems, *Science China Information Sciences*, **63** (2020), 1–14.
55. L. Liu, Y. J. Liu, D. P. Li, S. C. Tong, Z. S. Wang, Barrier Lyapunov function based adaptive fuzzy FTC for switched systems and its applications to resistance inductance capacitance circuit system, *IEEE T. Cybernetics*, **50** (2020), 3491–3502.
56. F. Chiclana, S. M. Zhou, Type-reduction of general type-2 fuzzy sets: The type-1 OWA approach, *Int. J. Intell. Syst.*, **28** (2013), 505–522.
57. J. M. Mendel, H. Hagars, W. W. Tan, W. W. Melek, H. Ying, *Introduction to type-2 fuzzy logic control: theory and applications*,Wiley-IEEE Press, 2014.

AIMS Press

© 2020 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0)