On Type-Reduction Versus Direct Defuzzification for Type-2 Fuzzy Logic Systems

Jerry M. Mendel

Abstract This chapter examines type-reduction and direct defuzzification for interval type-2 fuzzy logic systems. It provides critiques of type-reduction as an end to itself as well as of direct defuzzification, and concludes that: (1) a good way to categorize type-reduction/direct defuzzification algorithm papers is as papers that either focus on algorithms that lead to a type-reduced set, or directly to a defuzzified value; (2) research on type-reduction as an end to itself has led to results that are arguably of very little value; and, (3) the practice of base-lining an IT2 FLS that uses direct defuzzification against one that uses type-reduction followed by defuzzification is unnecessary.

1 Introduction

A *type-2 fuzzy set* (Fig. 1) (T2 FS) can be thought of as a fuzzy-fuzzy set. Its membership function (MF) no longer has a single value at each value of the primary variable, but instead is a blurred version of that function, i.e., at each value of the primary variable the membership is itself a function, called a *secondary MF*. When the secondary MF is a constant equal to 1, the T2 FS is called an *interval type-2 fuzzy set* (IT2 FS) or an *interval-valued fuzzy set*; otherwise, it is called a *general type-2 fuzzy set* (GT2 FS).

The MF of a T2 FS is three-dimensional, with *x-axis* called the *primary variable*, *y-axis* called the *secondary variable* and *z-axis* called the *MF value* (or *secondary grade*). A *vertical slice* is a plane that is parallel to the MF-value z-axis. The *footprint of uncertainty* (*FOU*) of a T2 FS lies on the *x-y plane* and includes the closure of all points on that plane for which the MF value is non-zero; it is the 2D-domain on which

J.M. Mendel (✉)
Ming Hsieh Department of Electrical Engineering, Signal & Image Processing Institute, University of Southern California, Los Angeles, CA 90089-2564, USA
e-mail: mendel@sipi.usc.edu

D.E. Tamir et al. (eds.), *Fifty Years of Fuzzy Logic and its Applications*,
Studies in Fuzziness and Soft Computing 326,
DOI 10.1007/978-3-319-19683-1_20

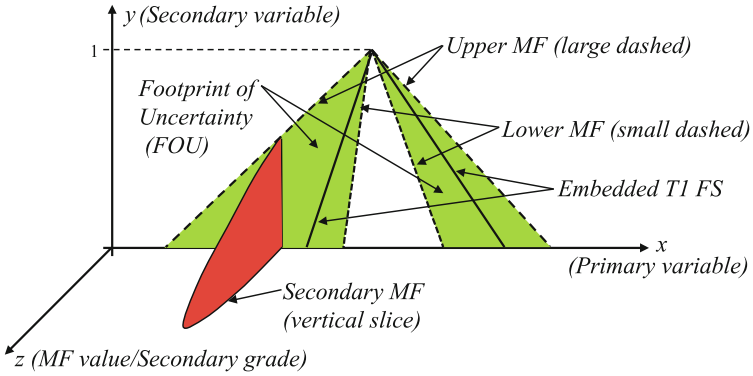

Fig. 1 Components of a General type-2 fuzzy set

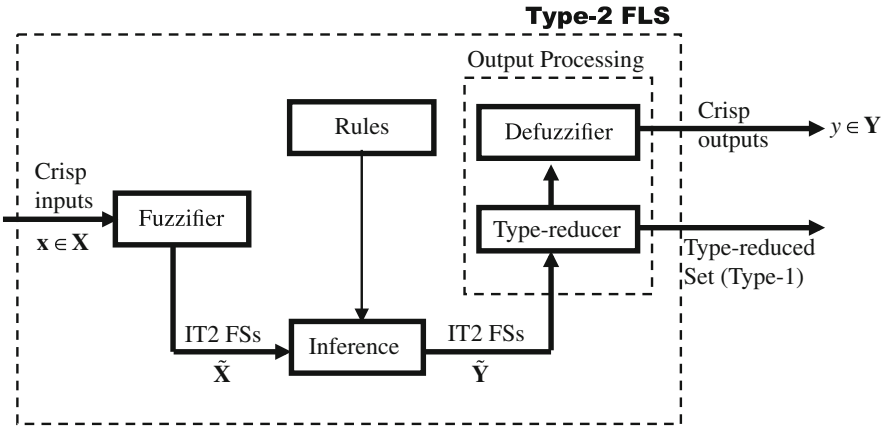

Fig. 2 Type-2 fuzzy logic system [23]. When all T2 FSs are IT2 FSs, the GT2 FLS becomes an IT2 FLS

sit the secondary grades. The FOU is lower and upper bounded by a *lower membership function* (LMF) and an *upper membership function* (UMF), both of which are type-1 fuzzy sets (T1 FSs). The FOU can be completely covered by T1 FSs that are called *embedded T1 FSs*.

T2 FSs are used in type-2 fuzzy logic systems (T2 FLSs) (Fig. 2). A general T2 FLS (GT2 FLS) was originally called a T2 FLS; however, because most of the works about T2 FLSs have focused on IT2 FSs, and only more recently on GT2 FSs, we now view the field of T2 FLSs as the union of the sub-fields of IT2 FLSs and GT2 FLSs.

Practical applications of T2 FLSs (e.g., fuzzy logic control [26]) require a number at the output of the FLS and not a FS. Readers of this book know that for a T1 FLS such a number is obtained by a process called *defuzzification*, which can be

interpreted as the projection of a T1 FS into a type-0 FS. Many kinds of defuzzifiers are possible, including center of gravity (centroid), height and center of sets.

So, how does one go from a T2 FS to a number? Nilesh Karnik and I struggled with how to do this for months and finally we came up with a *two-stage approach* [14, 23]: type-reduction (TR)—a new concept for FSs—followed by defuzzification.

TR projects a T2 FS into a T1 FS, after which that T1 FS is projected into a type-0-FS by defuzzification to obtain a number. Because a type-reduced set is a T1 FS, defuzzification is achieved by computing the centroid of that set. For an IT2 FLS the type-reduced set is an interval set whose center of gravity is the average value of its two end-points; so, defuzzification for an IT2 FLS is trivial.

Just as there can be different kinds of defuzzification methods for a T1 FLS, there can be different kinds of TR methods for a T2 FLS. It is generally agreed that all TR methods must satisfy Karnik and Mendel's [23] *fundamental design requirement* for a T2 FLS, namely: *When all sources of [membership function] uncertainty disappear, a T2 FLS must reduce to a comparable T1 FLS*. This design requirement seems as reasonable today (in 2015) as it did to Karnik and Mendel in 2001, and is analogous to what happens to a probability density function when random uncertainties disappear. In that case, the variance of the pdf goes to zero, and a probability analysis reduces to a deterministic analysis. So, just as the capability for a deterministic analysis is embedded within a probability analysis, the capability for a T1 FLS is embedded within a T2 FLS.

As is stated in [25]: "TR became burned into the architecture of a T2 FLS because Karnik and Mendel first developed all of their T2 concepts and calculations for a general T2 FS and a GT2 FLS. Although TR was originally developed for a GT2 FLS, because it was so simple to perform for an IT2 FLS it was kept in the architecture of an IT2 FLS. There is nothing wrong with doing this; however, in retrospect we may have been blind-sided by the need for TR in a GT2 FLS from asking the question 'Is TR really needed in an IT2 FLS?'

"In fact, there are many ways to go from an IT2 FLS to a number that bypass TR and still satisfy the fundamental design requirement."

"A student in a class that I taught some years ago asked: 'Instead of performing TR, why can't we just use a combination of two T1 FLSs, one that uses only the lower membership functions and the other that uses only the upper membership functions?' My answer at that time was: 'You can't do this because each end-point of the type-reduced set uses a mixture of lower and upper membership function information.' While my answer was technically correct, it was predicated on using type-reduction, rather than on what the student had suggested. My answer today would be: 'You can do what you are suggesting, and this can be done in different ways; however, by bypassing TR you may not be able to provide a measure of the uncertainties that have flowed through all of the IT2 FLS computations (analogous to a standard deviation).' For example, you could begin with the architecture of an IT2 FLS as a linear combination of two T1 FLSs, as in [1], or as the centroid of the average of the lower and upper membership functions of the aggregated rule fired sets, as in [29]. All of these IT2 FLSs go directly to the defuzzified output value and they all satisfy the fundamental design requirement."

Unfortunately, TR cannot be performed using a closed-form mathematical formula. Its calculations led to two simple iterative algorithms that have been and continue to be called *KM-Algorithms* (or Karnik-Mendel Algorithms) [14, 23]. Because the computational complexity of using GT2 FSs was so great in the late 1990s, until around 2008 we as well as all others focused on TR for IT2 FLSs. It is fair to say that all thinking about type-reduction was in the context of IT2 FLSs.

Karnik and Mendel's two-step approach for going from a T2 FS to a number spawned a plethora of research and subsequent publications that fall into two categories: (1) type-reduction as an end to itself (i.e., TR viewed as a mathematical problem but with no application in mind, focusing on finding improved ways to perform TR); and, (2) direct defuzzification (i.e., methods for bypassing TR that project a T2 FS directly into a number). Unfortunately, some authors use the phrase "type-reduction" in the titles of papers that are about direct defuzzification, which is (in the opinion of this author) arguably incorrect.

Two journal articles that cover all aspects of this appeared in 2013, [24, 33]. The article by Wu [33] categorizes type-reduction and direct defuzzification algorithms into the following two categories:

- Enhancements to the KM TR algorithms (W1) (these are about type-reduction) and
- Alternative TR algorithms (W2) (these are almost all about direct defuzzification)

The article by Mendel [24] categorizes type-reduction algorithms into the following four categories:

- Improved KM algorithms (M1) (these are about type-reduction)
- Understanding the KM/EKM algorithms leading to further improved algorithms (M2) (these are about type-reduction)
- Non-KM algorithms that preserve the ability to approximate the centroid or type-reduced set (M3) (these are about type-reduction), and,
- Non-KM algorithms that do not preserve the ability to approximate the centroid or type-reduced set (M4) (these are about direct defuzzification)

While both articles contain a wealth of information about type-reduction and direct defuzzification algorithms, it requires a considerable effort by readers to relate an algorithm to its different categorizations across these two papers. In order to help with this, I will now classify the TR algorithms differently from both of these articles, as:

- Algorithms that lead to a type-reduced set (Table 1)
- Algorithms that lead directly to a defuzzified value (Table 2)

The connections between these two classes of algorithms and the classifications in [24, 33] are given in these tables.

Table 1 Algorithms that lead to a type-reduced set

Authors/references	Year	Wu Class [33]	Mendel Class [24]	Emphasizes		FLS Application?	Baselines against
				Speedup	Accuracy		
Wu and Mendel [37]	2002	W2	M3	No	No	Time-series forecasting	Nothing
Niewiadomski, Ocheleska and Szczepaniak [30]	2006	None[a]	M3	No	No	Linguistic summarization	Nothing
Melgarejo [22]	2007	W1	M3	Yes	No[b]	None	KM
Duran, Bernal and Melgarejo [5]	2008	W1	M3	Yes	No[c]	None	EKM
Li, Yi and Zhao [17]	2008	W2	M3[d]	Yes	Yes	Fuzzy logic control	COS TR
Wu and Mendel [34]	2009	W1	M1	Yes	No	None	KM
Hu, Zhao and Yang [13]	2010	W1	M3	Yes	Yes	None	EKM
Wu and Nie [35]	2011	W1	M1	Yes	No	None	KM and EKM
Liu andMendel [19]	2011	W1[e]	M2	No	Yes	None	Nothing
Liu, Qin and Wu [21]	2012	W1[e]	M2	Yes	No	None	Continuous EKM
Liu, Mendel and Wu [20]	2012	F[f]	M2	Yes	Yes	None	Nothing
Hu, Wang and Cai [12]	2012	W1	M3	Yes	No	None	EKM
Ulu, Guzelkaya and Eksin [32]	2013	W2	M3	Yes	Yes	None	EKM

[a] Wu includes this reference but does not include its algorithm in the main body of his paper
[b] Same accuracy achieved for their algorithms and KM algorithms
[c] Same accuracy achieved for their algorithms and EKM algorithms
[d] Although this paper is not listed in [24], if it had been it would be in M3
[e] Although this paper is not listed in [33], if it had been it would be in W1
[f] Mentioned only in a footnote in [33]

Table 2 Algorithms that lead directly to a defuzzified value

Authors/references	Year	Wu Class [33]	Mendel Class [24]	Emphasizes		FLS Application?	Baselines against
				Speedup	Accuracy		
Dziech and Gorzalczany [6]	1987	W2[a]	M4	No	No	Signal transmission[b]	Nothing
Gorzalczany [7]	1988	W2	M4	No	No	Fuzzy logic control[c]	Nothing
Liang and Mendel [18]	2000	W2	M4[d]	No	No	Equalization	Nothing
Wu and Tan [36]	2005	W2	M4	Yes	No[e]	PI Control	WM uncertainty bounds + defuzzification
Coupland and John [3]	2007	W2	M4[d]	Yes	No	None	Nothing
Nie andTan [29]	2008	W2	M4	Yes	Yes[f]	Fuzzy logic control	COS TR + defuzzification and WM uncertainty bounds + defuzzification
Greenfield et al. [9]	2009	W2	M4	Yes	Yes	None	KM
Greenfield, Chiclana and John [10]	2009	W2	M4	No	Yes	None	Exhaustive defuzzification
Greenfield, Chiclana and John [11]	2009	W2	M4	Yes	Yes	None	Exhaustive defuzzification
Du and Ying [4]	2010	W2	M4[d]	No	No	Fuzzy logic control	COS TR + defuzzification
Biglarbegian, Melek and Mendel [1]	2010	W2	M4	No	No	Fuzzy logic control	Nothing
Biglarbegian, Melek and Mendel [2]	2011	W2	M4	No	No	Function approx., identification	T1 FLS
Greenfield and Chiclana [8][g]	2011	W2	M4	No	Yes	None	Exhaustive defuzzification
Mendel and Liu [27][a, d]	2012	W2	M4	Yes	Yes	None	Nie-Tan
Tau, et al. [31]	2012	W2	M4[d]	Yes	No	Fuzzy logic control	Other controllers
Mendel and Liu [28][a, d]	2013	W2	M4	Yes	Yes	None	Nie-Tan
Khosravi, Nahavandi & Khosravi [15]	2013	W2[a]	M4[d]	Yes	Yes	Time-series prediction, identification	Five other methods
Khosravi & Nahavandi [16]	2014	W2[a]	M4[d]	Yes	Yes	Load forecasting	Five other methods

[a] Although this paper is not listed in [33], if it had been it would be in W2
[b] Not much detail is provided for this application
[c] This application is only suggested, but it is not examined in the paper
[d] Although this paper is not listed in [24], if it had been in would be in M4
[e] Commonly used control metrics, IAE, ISE and ITAE are compared
[f] Uses ITAE to do this
[g] Although "type-reduction" appears in the title of this paper, no type-reduced set is obtained in it

2 Explanations of the Tables

Tables 1 and 2 organize the papers chronologically, so that one can see some sort of continuity in thought as time progresses. Unfortunately, no standards existed when these studies were performed (nor do they exist today, in 2015) for how to compare TR or direct defuzzification algorithms, so it is very difficult to draw statistically meaningful conclusions from the papers that are in these two tables. Most of the time the authors used different IT2 FSs (FOUs) (some of which are sampled versions of continuous FOUs), or randomly chosen samples for the LMFs and UMFs of discrete FOUs, or FOUs for which mathematical formulas are provided for their LMF and UMF.

Fortunately, [33] provides statistically meaningful comparisons for two kinds of FOUs: (1) randomly chosen samples for the LMFs and UMFs of discrete FOUs, and (2) evolutionary fuzzy logic controller design. If one is interested in which of the algorithms are the fastest or most accurate, then [33] is the paper to go to.

3 Critique of Type-Reduction as an End to Itself

Research on *type-reduction as an end to itself*, as summarized in Table 1, uses the (oft unstated) assumption that one begins with an IT2 FS to perform TR on. It then focuses on how to improve (or approximate) the KM Algorithms to perform type-reduction on that given IT2 FS. One or both of two performance measures are used to evaluate improvements: computing time and accuracy. All of this research (except for [30, 37]) is done outside of the original context of a T2 FLS, and is therefore open to some critical examinations.

For starters, the assumption that *one begins with an IT2 FS to perform TR on* needs to be questioned. I have explained in [24] that there are two distinctly different situations that can occur in an IT2 FLS:

1. Fired-rule IT2 FSs are first aggregated by means of the union operation resulting in an aggregated IT2 FS (so that the assumption is valid), after which this aggregated IT2 FS is type-reduced (this is called *centroid type-reduction*); and,
2. Fired-rule IT2 FSs are aggregated as part of type-reduction (this is called *center-of sets type-reduction*), in which case one does not begin with an aggregated IT2 FS, and so the assumption is invalid.

Obviously the people who are researching *type-reduction as an end to itself* are focusing on Situation 1; however, in real-world applications of FLSs it is Situation 2 that is more often used than Situation 1, because (this is also true for T1 FLSs) performing the aggregation of the fired-rule IT2 FSs is too time consuming. Consequently, research on *type-reduction as an end to itself* is about things that arguably will be of very little direct value or use in a T2 FLS.

One is then led to question whether or not the two metrics that are used in this kind of research for Situation 1 are important, namely computing time and accuracy.

Consider first the metric of *computing time*. If the results of this research are not going to be used in a real-time FLS then does it really matter if the algorithms that perform type-reduction take 10^{-3}, 10^{-4}, 10^{-5}, etc. sec? KM or EKM [34] algorithms are already quite fast; they converge quadratically [19] and have been observed to take 10^{-5} to 10^{-3} s to converge (see, e.g., [34]). The EIASC algorithms [35] are even faster and they are very easy to understand. So, focusing on convergence time for Situation 1 is to me arguably a red herring because it makes no perceptual difference to a human when it is already quite small.

Consider next the metric of *accuracy*. To me an important question is: How much accuracy is required by the type-reduced set? Unfortunately, this question is never asked in the papers that use the metric of accuracy. Instead, one is left with an impression that higher accuracy is always better. Unless one knows what the type-reduced set will be used for then, I would like to ask: "What is higher accuracy better for?" To answer this question there are again two different situations/cases that need to be examined:

- The LMF and UMF are given by mathematical formulas and so they can be sampled at any desired rate (C1), and
- The LMF and UMF are given only by a collection of samples (C2).

Consider C1, in which the LMF and UMF are given by mathematical formulas, and so they can be sampled at any desired rate. Then KM algorithms will give very accurate results. To counter this some authors perform studies in which they reduce the sampling rate to see how more accurate their algorithms are than the KM algorithms. The Catch-22 to this work is that they are comparing the results from their algorithms and the KM algorithms both of which use the reduced sampled data with so-called *true results*, where the latter use the highly sampled data. To me this is circular reasoning, because if one has access to highly sampled data and is performing the type-reduction computations off-line then why not use all of the highly sampled data to perform the type-reduction?

Consider next C2 in which the LMF and UMF are given only by a collection of samples, as would occur if perchance fired rule output sets were aggregated by using the union (although, as explained above, this is usually avoided). The reason that the resulting aggregated IT2 FS is given only by a collection of samples is that it has been computed using an algorithm for the union that only uses sampled values. So, what exactly does accuracy mean in this situation? In this case the KM algorithms compute the true values of the type-reduced set and so they are 100 % accurate. To get around this fact, the authors pretend that they have access to the LMF and UMF given by mathematical formulas and have created a set of sampled values for the LMF and the UMF from those formulas. This throws us back to the previous case, but now that case is reached by violating the assumption that the LMF and UMF are given only by a collection of samples. This again is circular reasoning.

Regrettably, the conclusion I am led to about research on *type-reduction as an end to itself* is that it has led to results that are arguably of very little value.

4 Critique of Direct Defuzzification

All research on direct defuzzification should be applauded as long as the direct defuzzification method obeys the already-mentioned *fundamental design requirement* for a T2 FLS, namely: *When all sources of [membership function] uncertainty disappear, a T2 FLS must reduce to a comparable T1 FLS*. It should be applauded because its researchers have had the courage and conviction to challenge the need for type-reduction in an IT2 FLS. To me, this is how real progress occurs. These authors (although they may not have actually said it this way or at all) asked: "Why is type reduction needed in an IT2 FLS? Why can't we go directly to a defuzzified output?"

As is demonstrated by the large number of papers in Table 2, there are many ways to go directly to a defuzzified output; however, many of the authors of direct defuzzification papers compare their numerical results with the ones obtained from using type-reduction followed by defuzzification, as though the latter was a baseline approach. So another natural question to ask is:

- Should an IT2 FLS that uses type-reduction followed by defuzzification be the baseline IT2 FLS against which all others must be compared?

My answer to this question is "No," and my reason is that maybe it should be the other way around, i.e. maybe an IT2 FLS that uses type-reduction followed by defuzzification should be compared against an IT2 FLS that uses direct defuzzification. This may sound like double talk or the chicken before the egg parable, but I am quite serious about my answer. Let me explain.

Real-world FLSs are designed with application-dependent performance specifications in mind. Unfortunately, authors (myself included) do not usually state what those specs are. Instead they assume that the goal is to optimize the performance metrics rather than meet those metrics, something that is not done in the business world.

Many years ago I attended a one-day seminar on entrepreneurship given by world-famous Peter Drucker. It just so happened that we were sitting at the same table for lunch. Everyone around the table introduced themselves. When it came to my turn and I told everyone that I was an engineer, Drucker shook his head. I asked him why he was doing that, to which he replied:

> You engineers are always trying to optimize something, which is why you make such poor businessmen. A business person develops a product with a certain level of performance as quickly as possible, manufactures it, sells it and makes a profit. He or she then improves the product a bit and sells the next version, making even more profit. In the meantime, you engineers have not gotten off of the block; you are still trying to optimize your product, and have nothing to show for it.

So if an application's performance metrics can be met by an IT2 FLS that uses direct defuzzification, that should be the end of the story. If, however, those performance metrics cannot be met by such an IT2 FLS then one could try either a different IT2 FLS that uses direct defuzzification, or, perhaps as a last resort, an IT2

FLS that uses type reduction followed by defuzzification. There is no a priori guarantee, however, that even the latter will be able to meet the performance metrics. If it cannot, then one may need to use a GT2 FLS.

5 Conclusions

Ever since Karnik and Mendel introduced the concept of type-reduction, which leads to solving two optimization problems, one for the left-end and one for the right end of the type-reduced set, a cottage industry has emerged that has focused on better ways to perform TR or to bypass it entirely. This paper has tried to provide a critical examination of the research that has been performed on these two topics.

It has explained that:

1. A good way to categorize type-reduction/direct defuzzification algorithm papers is, as: papers that focus on algorithms that lead to
   a. A type-reduced set, or
   b. Directly to a defuzzified value.
2. Research on type-reduction as an end to itself has led to results that are arguably of very little value.
3. The practice of base-lining an IT2 FLS that uses direct defuzzification against one that uses type-reduction followed by defuzzification is unnecessary.

Note that if Karnik and Mendel had never introduced type-reduction followed by defuzzification, and instead had achieved their design requirement by using direct defuzzification, then we would not be having this conversation, and people would be comparing results obtained by using one kind of direct defuzzification method against those obtained by using at least one other kind of direct defuzzification method.

As a final thought, note that type reduction was invented because Karnik and Mendel thought that the type-reduced set would itself be of value, in that it would provide a valuable measure of the flow of MF uncertainties through the FLS. Although it does do this, regrettably, to the best of knowledge of this author, there is not one application paper that makes use of the type-reduced set in this way. On the other hand, type-reduction did lead to the KM Algorithms (as well as others) which are very useful for solving other non-FLS problems, as is explained in [24].

References

1. Biglarbegian, M., Melek, W.W., Mendel, J.M.: On the stability of interval type-2 TSK fuzzy logic control systems. IEEE Trans. Syst. Man Cybern. Part B: Cybern. **40**, 798–818 (2010)
2. Biglarbegian, M., Melek, W.W., Mendel, J.M.: On the robustness of type-1 and interval type-2 fuzzy logic systems in modeling. Inf. Sci. **181**, 1325–1347 (2011)

3. Coupland, S., John, R.I.: Geometric type-1 and type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. **15**(1), 3–15 (2007)
4. Du, X., Ying, H.: Derivation and analysis of the analytical structures of the interval type-2 fuzzy-PI and PD controllers. IEEE Trans. Fuzzy Syst. **18**(4), 802–814 (2010)
5. Duran, K., Bernal, H., Melgarejo, M.: Improved iterative algorithm for computing the generalized centroid of an interval type-2 fuzzy set. In: Proceedings of NAFIPS Conference, New York, Paper 50056 (2008)
6. Dziech, A., Gorzalczany, M.B.: Decision making in signal transmission problems with interval-valued fuzzy sets. Fuzzy Sets Syst. **23**, 191–203 (1987)
7. Gorzalczany, M.B.: Interval-valued fuzzy controller based on verbal model of object. Fuzzy Sets Syst. **28**, 45–53 (1988)
8. Greenfield, S., Chiclana, F.: Type-reduction of the discretised interval type-2 fuzzy set: what happens as discretisation becomes finer? In: Proceedings of IEEE Symposium on Advances in Type-2 Fuzzy Logic System, Paris, pp. 102–109,(2011)
9. Greenfield, S., Chiclana, F., Coupland, S., John, R.I.: The collapsing method for defuzzification of discretized interval type-2 fuzzy sets. Inf. Sci. **179**, 2055–2069 (2009)
10. Greenfield, S., Chiclana, F., John, R.I.: The collapsing method: does the direction of collapse affect accuracy? In: Proceeding of IFSA-EUSFLAT, Lisbon, pp. 980–985 (2009)
11. Greenfield, S., Chiclana, F., John, R.I.: Type-reduction of the discretised interval type-2 fuzzy set. In: Proceedings of IEEE FUZZ Conference, JeJu Island, pp. 738–743 (2009)
12. Hu, H., Wang, Y., Cai, Y.: Advantages of the enhanced opposite direction searching algorithm for computing the centroid of an interval type-2 fuzzy set. Asian J. Control **14**(6), 1–9 (2012)
13. Hu, H., Zhao, G., Yang, H.N.: Fast algorithm to calculate generalized centroid of interval type-2 fuzzy set. Control Decis. **25**(4), 637–640 (2010)
14. Karnik, N., Mendel, J.M.: Centroid of a type-2 fuzzy set. Inf. Sci. **132**, 195–220 (2001)
15. Khosravi, A., Nahavandi, S., Khosravi, R.: A new neural network-based type reduction algorithm for interval type-2 fuzzy logic systems. In: Proceeding of IEEE FUZZ Conf., Hyderabad, Paper 1116, (2013)
16. Khosravi, A., Nahavandi, S.: Load forecasting using interval type-2 fuzzy logic systems: optimal type-reduction. IEEE Trans.Ind. Inf. **10**(2), 1055–1063 (2014)
17. Li, C., Yi, J., Zhao, D.: A novel type-reduction method for interval type-2 fuzzy logic systems. In: Proceedings 5th International Conference Fuzzy Systems Knowledge Discovery, Jinan, pp. 157–161 (2008)
18. Liang, Q., Mendel, J.M.: Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters. IEEE Trans. Fuzzy Syst. **8**(5), 551–563 (2000)
19. Liu, X., Mendel, J.M.: Connect Karnik-Mendel algorithms to root-finding for computing the centroid of an interval type-2 fuzzy set. IEEE Trans. Fuzzy Syst. **19**(4), 652–665 (2011)
20. Liu, X., Mendel, J.M., Wu, D.: Study on enhanced Karnik-Mendel algorithms: initialization explanations and computation improvements. Inf. Sci. **187**, 75–91 (2012)
21. Liu, X., Qin, Y., Wu, L.: Fast and direct Karnik-Mendel algorithm computation for the centroid of an interval type-2 fuzzy set. In: Proceedings IEEE FUZZ Conference, Brisbane, pp. 1058–1065 (2012)
22. Melgarejo, M.C.A.: A fast recursive method to compute the generalized centroid of an interval type-2 fuzzy set. In: Proceedings of NAFIPS Conference, San Diego, pp. 190–194 (2007)
23. Mendel, J.M.: Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall, Upper Saddle River (2001)
24. Mendel, J.M.: On KM algorithms for solving type-2 fuzzy set problems. IEEE Trans. Fuzzy Syst. **21**(3), 426–446 (2013)
25. Mendel, J.M.: Type-2 fuzzy sets and beyond. In: Seising, R., Trillas, E., Moraga, C., Termini, S. (eds.) On Fuzziness: A Homage to Lotfi A. Zadeh, vol. 2. Ch. 34. Springer (2013)
26. Mendel, J.M., Hagras, H., Tan, W.-W., Melek, W.M., Ying, H.: Introduction to Type-2 Fuzzy Logic Control. IEEE Press and Wiley, Hoboken (2014)

27. Mendel, J.M., Liu, X.: New closed-form solutions for Karnik-Mendel algorithm +defuzzification of an interval type-2 fuzzy set. In: Proceedings IEEE FUZZ Conference, Brisbane, pp. 1610–1617 (2012)
28. Mendel, J.M., Liu, X.: Simplified interval type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. **21**(6), 1056–1069 (2013)
29. Nie, M., Tan, W.W.: Towards an efficient type-reduction method for interval type-2 fuzzy logic systems. In: Proceeding of IEEE FUZZ Conference, Hong Kong, Paper FS0339 (2008)
30. Niewiadomski, A., Ochelska, J., Szczepaniak, P.S.: Interval-valued linguistic summaries of databases. Control Cybern. **35**(2), 415–444 (2006)
31. Tau, C.W., Taur, J.S., Chang, C.-W., Chang, Y.-H.: Simplified type-2 fuzzy sliding controller for wing rock system. Fuzzy Sets Syst. **207**, 111–129 (2012)
32. Ulu, C., Guzelkaya, M., Eskin, I.: A closed form type-reduction method for piecewise linear interval type-2 fuzzy sets. Int. J. Approx. Reason. **54**(9), 1421–1433 (2013)
33. Wu, D.: Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: overview and comparisons. IEEE Trans. Fuzzy Syst. **21**(1), 80–99 (2013)
34. Wu, D., Mendel, J.M.: Enhanced Karnik-Mendel algorithms. IEEE Trans. Fuzzy Syst. **17**(4), 923–934 (2009)
35. Wu, D. and Nie, M.: Comparison and practical implementations of type-reduction algorithms for type-2 fuzzy sets and systems. In: Proceedings of IEEE FUZZ Conference, Taipei, pp. 2131–2138 (2011)
36. Wu, D., Tan, W.W.: Computationally efficient type-reduction strategies for a type-2 fuzzy logic controller. In: Proceedings of IEEE FUZZ Conference, Reno, pp. 353–358, (2005)
37. Wu, H., Mendel, J.M.: Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. **10**(5), 622–639 (2002)

Author Biography

Jerry M. Mendel was born in New York City and received the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, NY. Currently he is Professor of Electrical Engineering at the University of Southern California in Los Angeles, where he has been since 1974.

He has published over 550 technical papers and is author and/or co-author of 11 books, including *Uncertain Rule-based Fuzzy Logic Systems: Introduction and New Directions* (Prentice-Hall, 2001), *Perceptual Computing: Aiding People in Making Subjective Judgments* (Wiley & IEEE Press, 2010), and *Introduction to Type-2 Fuzzy Logic Control: Theory and Application* (Wiley & IEEE Press, 2014). His present 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 Life Fellow of the IEEE, a Distinguished Member of the IEEE Control Systems Society, and a Fellow of the International Fuzzy Systems Association. He was President of the IEEE Control Systems Society in 1986. He was a member of the Administrative Committee of the IEEE Computational Intelligence Society for nine years, and was Chairman of its Fuzzy Systems Technical Committee and 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.