6-2010

Towards a More Adequate Defuzzification of Interval-Valued Fuzzy Sets

Vladik Kreinovich
The University of Texas at El Paso, vladik@utep.edu

Van Nam Huynh

Yoshiteru Nakamori

Comments:
Technical Report: UTEP-CS-08-33a
To appear in the *Proceedings of the 7th International Conference on Modeling Decisions for Artificial Intelligence MDAI'2010*, Perpignan, French Catalonia, France, October 27-29, 2010

Towards a More Adequate Defuzzification of Interval-Valued Fuzzy Sets

Vladik Kreinovich[1], Van Nam Huynh[2], and Yohiteru Nakamori[2]

[1] Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
vladik@utep.edu
[2] School of Knowledge Science
Japan Advanced Institute of Science and Technology, JAIST
1-1 Asahidai, Nomi, Ishikawa 923-1292 Japan
{huynh,nakamori}@jaist.ac.jp

Abstract. It is known that interval-valued fuzzy sets $[\underline{m}(x), \overline{m}(x)]$ provide a more adequate description of expert uncertainty than the more traditional "type-1" (number-valued) fuzzy techniques. Specifically, an interval-valued fuzzy set can be viewed as a class of possible fuzzy sets $m(x) \in [\underline{m}(x), \overline{m}(x)]$. In this case, as a result of defuzzification, it is natural to return the range $[\underline{u}, \overline{u}]$ of all possible values $u(m)$ that can be obtained by defuzzifying membership functions $m(x)$ from this class.
In practice, it is reasonable to restrict ourselves only to *fuzzy numbers* $m(x)$, i.e., to "unimodal" fuzzy sets. Under this restriction, in general, we get a narrower range $[\underline{u}_a, \overline{u}_a]$ of possible values of $u(m)$. In this paper, we describe a feasible algorithm for computing the new range $[\underline{u}_a, \overline{u}_a]$.

Keywords: interval-valued fuzzy sets, fuzzy numbers, unimodality, defuzzification, feasible algorithm, fuzzy control

1 Introduction: Need for Defuzzification of Interval-Valued Fuzzy Sets

Need for intelligent control. In many practical control situations, there is a small number of experts skilled in the corresponding control. Since there are only a few such skilled experts, they are unable to personally control all needed situations. It is therefore desirable to design an automated system that would implement their expertise.

Need to use fuzzy sets. Experts are often only able to describe their control by using imprecise (*fuzzy*) words from natural language such as "small" or "close to 0". To translate such knowledge into a numerical strategy, Zadeh invented fuzzy logic. For each natural-language property P like "small" and for every possible value x of the corresponding quantity, an expert is often not 100% certain whether x satisfies the property P. We describe his or her certainty by a

degree $m(x)$ from the interval $[0,1]$. These degrees form a *fuzzy set*. To be a more precise, a fuzzy set is usually defined as a function m which maps all possible values of the corresponding quantity into the interval $[0,1]$; see, e.g., [2, 7]. The function m is also called a *membership function*.

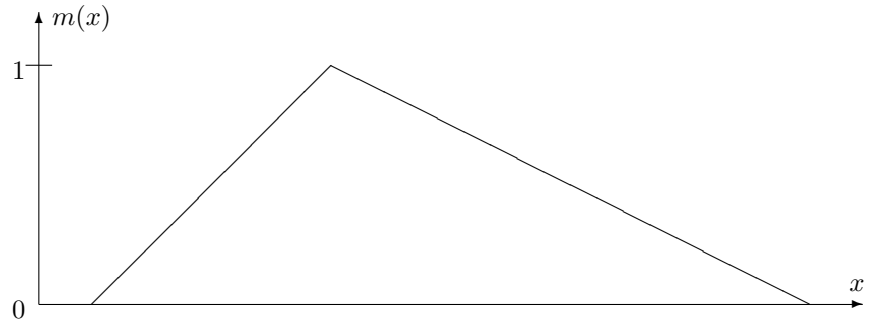
How can we represent a generic membership function in a computer. We are interested in producing an algorithm for control. The input to this algorithm is a membership function. So, to describe the algorithm, we first need to describe how we can represent a generic membership function $m(x)$ in a computer.

Some membership functions are determined by their analytical (or algorithmic) expression. For example, a piece-wise linear membership function pictured below can be represented by explicit formulas for its linear parts. However, for a generic membership function, there are no analytical or algorithmic expressions. Instead, from experts, we get the degrees $m(x_i)$ to which different values x_i are possible. In practice, we can only ask a finite number of questions to an expert, so we have only finitely many values x_i.

It is therefore reasonable to represent the "input" membership functions – describing such terms as "small" – by their values at a finite number of points.

Usually, a membership function $m(x)$ is represented by its values $m_i \stackrel{\text{def}}{=} m(x_i)$ on a uniform grid $x_i = x_0 + i \cdot h$ for some $h > 0$, i.e., as an array $m = (m_1, m_2, \ldots)$.

"Unimodular" fuzzy sets – fuzzy numbers. Usually, fuzzy sets are "unimodular" in the sense that the corresponding membership function $m(x)$ first (non-strictly) increases (usually, from 0 to 1), and then (non-strictly) decreases (usually, from 1 to 0). Such fuzzy sets are also known as *fuzzy numbers*.



Need for a defuzzification. Based on the expert's rules and the formulas of fuzzy logic, we translate the fuzzy sets corresponding to the natural-language terms into a fuzzy set that describes reasonable control values. Since we want a single control value, we must use a special *defuzzification* procedure.

Usually, a *centroid* defuzzification is used, in which we transform a membership function $m(x)$ into the "centroid" value

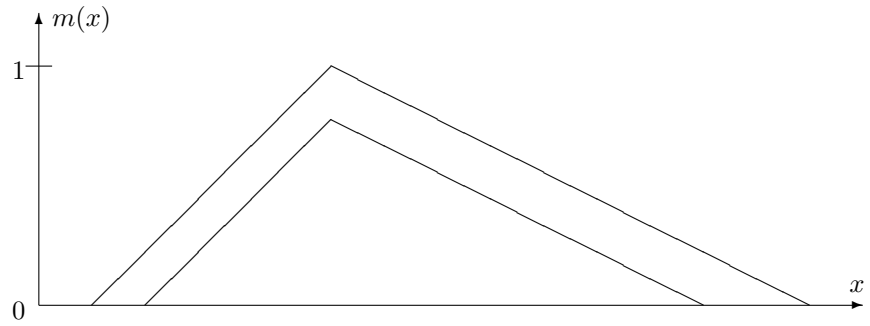
$$u(m) = \frac{\int x \cdot m(x)\, dx}{\int m(x)\, dx}. \tag{1}$$

How to describe centroid defuzzification under the generic computer representation of a membership function. When we know the values $m_i = m(x_i)$ of a function $m(x)$ on a grid, a natural way to approximate an integral $\int m(x)\,dx$ of this function is by using the corresponding integral sum: $\int m(x)\,dx \approx \sum\limits_{i=1}^{n} m(x_i) \cdot \Delta x_i$, where $\Delta x_i = x_{i+1} - x_i = h$. In other words, the resulting integral sum is simply proportional to the sum of the corresponding values: $\int m(x)\,dx \approx h \cdot \sum\limits_{i=1}^{n} m_i$.

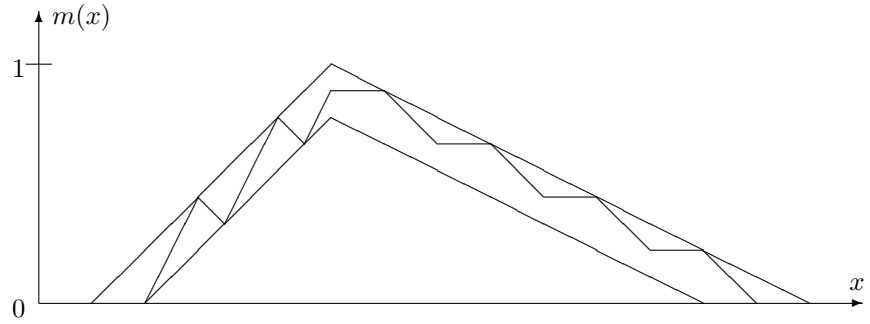
Similarly, the integral in the numerator of the centroid formula can be approximated as $\int x \cdot m(x)\,dx \approx h \cdot \sum\limits_{i=1}^{n} x_i \cdot m_i$. When we divide this integral sum by the previous one, the factors h in the numerator and in the denominator cancel each other, so we end up with the following formula for the result $u(m)$ of centroid defuzzification:

$$u(m) = \frac{\sum\limits_{i=1}^{n} x_i \cdot m_i}{\sum\limits_{i=1}^{n} m_i}. \tag{2}$$

Need for interval-valued fuzzy sets. In practice, just like an expert cannot be 100% sure whether a given value x is small, this same expert cannot describe her degree of certainty by an exact number. At best, she can produce an interval $[\underline{m}(x), \overline{m}(x)]$ of possible values. As a result, we get *interval-valued* fuzzy sets; see, e.g., [5, 6].



The resulting interval-valued fuzzy set can be viewed as a class of all fuzzy sets m for which, for every x, the value $m(x)$ is within this interval:



A computer representation of interval-valued fuzzy sets. In the interval-valued case, for every i, instead of the exact value of m_i, we only know the interval $[\underline{m}_i, \overline{m}_i]$ of possible values of m_i, where $\underline{m}_i \stackrel{\text{def}}{=} \underline{m}(x_i)$ and $\overline{m}_i \stackrel{\text{def}}{=} \overline{m}(x_i)$.

Defuzzification of interval-valued fuzzy sets. For different values $m_i \in [\underline{m}_i, \overline{m}_i]$, we get, in general, different values of $u(m)$. Our objective is to find the range of possible value of $u(m)$ when $m_i \in [\underline{m}_i, \overline{m}_i]$.

The function (2) is continuous; thus, its range on a connected closed bounded box $[\underline{m}_1, \overline{m}_1] \times \ldots \times [\underline{m}_n, \overline{m}_n]$ is an interval. We will denote the endpoints of this interval by $\underline{u}$ and $\overline{u}$. Thus, to find the range, it is sufficient to find the smallest possible and the largest possible values of the expression (2) under the condition $m_i \in [\underline{m}_i, \overline{m}_i]$.

How to defuzzify an interval-valued fuzzy set. As a result of defuzzifying an interval-valued fuzzy set $[\underline{m}(x), \overline{m}(x)]$, it is thus reasonable to take the interval $[\underline{u}, \overline{u}]$ formed by the results of defuzzifying all fuzzy sets $m(x) \in [\underline{m}(x), \overline{m}(x)]$.

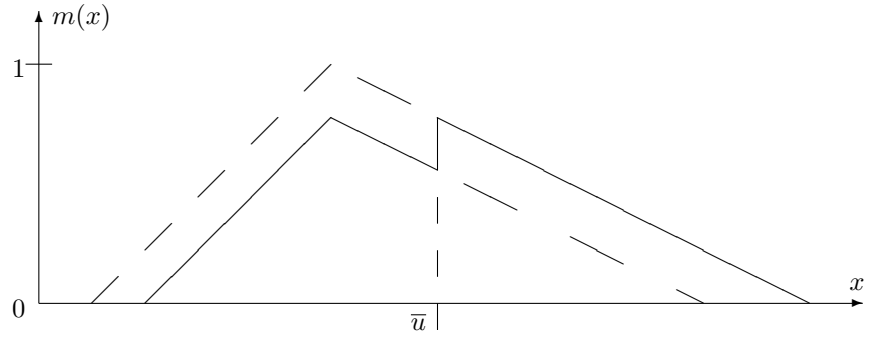
Analytical expressions and efficient algorithms have been designed for computing this interval; see, e.g., [3–6, 8].

2 Formulation of the Problem: Need for a More Adequate Defuzzification of Interval-Valued Fuzzy Sets

The existing defuzzification of interval-valued fuzzy sets: reminder. For an interval-valued fuzzy set $[\underline{m}(x), \overline{m}(x)]$, we need to find the interval $[\underline{u}, \overline{u}]$ formed by the results of defuzzifying all fuzzy sets $m(x) \in [\underline{m}(x), \overline{m}(x)]$.

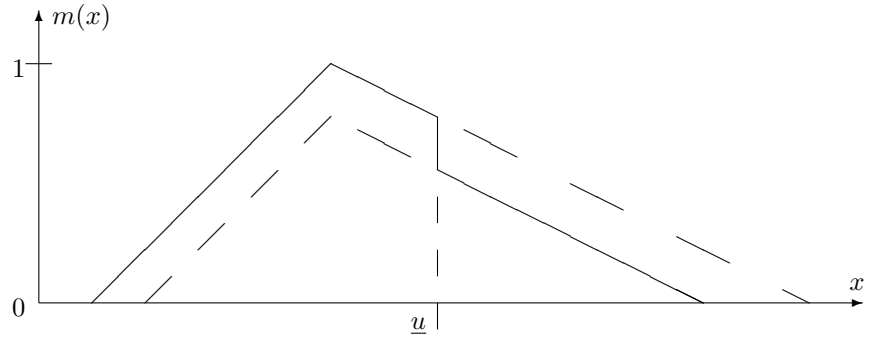
It is known [3–6, 8] that the maximum $\overline{u}$ is attained when we choose $m(x) = \underline{m}(x)$ for all $x < \overline{u}$ and $m(x) = \overline{m}(x)$ for all $x \geq \overline{u}$:

$$\overline{u} = \frac{\int\limits_{-\infty}^{\overline{u}} x \cdot \underline{m}(x)\, dx + \int\limits_{\overline{u}}^{\infty} x \cdot \overline{m}(x)\, dx}{\int\limits_{-\infty}^{\overline{u}} \underline{m}(x)\, dx + \int\limits_{\overline{u}}^{\infty} \overline{m}(x)\, dx}. \tag{3}$$



Similarly, the minimum $\underline{u}$ is attained when we choose $m(x) = \overline{m}(x)$ for all $x < \underline{u}$ and $m(x) = \underline{m}(x)$ for all $x \geq \underline{u}$:

$$\underline{u} = \frac{\int\limits_{-\infty}^{\underline{u}} x \cdot \overline{m}(x)\,dx + \int\limits_{\underline{u}}^{\infty} x \cdot \underline{m}(x)\,dx}{\int\limits_{-\infty}^{\underline{u}} \overline{m}(x)\,dx + \int\limits_{\underline{u}}^{\infty} \underline{m}(x)\,dx}. \tag{4}$$



Problem. As we have seen from the above pictures, sometimes, the endpoints $\underline{u}$ and $\overline{u}$ of the desired defuzzification interval $[\underline{u}, \overline{u}]$ are only attained for un-natural fuzzy sets – namely, for fuzzy sets which are not unimodal.

Natural idea. Instead of considering the defuzzification results $u(m)$ for *all possible* membership functions $m(x) \in [\underline{m}(x), \overline{m}(x)]$, let us consider the range $[\underline{u}_a, \overline{u}_a]$ of possible values $u(m)$ for all *unimodal* fuzzy sets $m(x) \in [\underline{m}(x), \overline{m}(x)]$.

Since we exclude some membership functions, we should, in general, get a narrower interval of values $u(m)$: $[\underline{u}_a, \overline{u}_a] \subseteq [\underline{u}, \overline{u}]$.

Comment. Here, *a* stands for "more *a*dequate".

What is needed. We need to compute the interval $[\underline{u}_a, \overline{u}_a]$ of all the values that can be obtained by defuzzifying all unimodal membership functions $m(x) \in [\underline{m}(x), \overline{m}(x)]$.

What we do in this paper. In this paper, we design a feasible algorithm for computing the desired (narrower) interval $[\underline{u}_a, \overline{u}_a]$.

3 Towards Precise Mathematical Formulation of the Problem

The problem. The problem is that,

- as we have mentioned earlier, it is often reasonable to restrict ourselves to fuzzy numbers (unimodal fuzzy sets),
- while, as we have seen, the maximum and/or minimum of the value $u(m)$ is sometimes attained at a membership function $m(x)$ which is not unimodal.

It is therefore desirable to find the maximum and the minimum of $u(m)$ only among *unimodal* values m_i, i.e., values which m_i first (non-strictly) increases and then (non-strictly) decreases.

The problem reformulated in precise mathematical terms. In precise terms, we are only interested in finding the maximum and the minimum of the expression (2) among all the values $m_1, \ldots, m_n$ for which, for some "mode location" $\ell = 1, 2 \ldots, n$, we have

$$m_1 \le m_2 \ldots \le m_{\ell-1} < m_\ell \ge m_{\ell+1} \ge \ldots \ge m_{n-1} \ge m_n.$$

Let us denote the corresponding minimum and maximum by $\underline{u}_a$ and $\overline{u}_a$.

4 Our Main Result: The New Defuzzication Algorithm for Interval-Valued Fuzzy Sets

First stage: auxiliary computations. First, for all i and $j \ge i$, we compute $m^-_{i,j} \stackrel{\text{def}}{=} \max(\underline{m}_i, \ldots, \underline{m}_j)$ and $m^+_{i,j} \stackrel{\text{def}}{=} \min(\overline{m}_i, \ldots, \overline{m}_j)$. Specifically, for each i,

- we first take $m^-_{i,i} = \underline{m}_i$ and $m^+_{i,i} = \overline{m}_i$, and then
- for each $j = i+1, i+2, \ldots$, we compute

$$m^-_{i,j} = \max(m^-_{i,j-1}, \underline{m}_j)$$

and

$$m^+_{i,j} = \min(m^+_{i,j-1}, \overline{m}_j).$$

Second stage: computing $\overline{u}_a$.

- For each of n^3 possible combinations of three integers $\ell \le s < e$, we take $m_i = m^-_{1,i}$ for $i < \ell$, $m_i = m^-_{i,s}$ for $\ell \le i \le s$, $m_i = m^+_{i,n}$ for $i \ge e$, and $m_i = \text{const} \in [m_e, m_s]$ for $i \in (e, s)$.
- We check whether all these values satisfy the conditions $m_i \in [\underline{m}_i, \overline{m}_i]$, and if yes, we compute the ratio $u(m)$.
- The largest of these values is returned as the desired upper bound $\overline{u}_a$.

Third stage: computing $\underline{u}_a$.

- For each of n^3 possible combinations of three integers $s \le e \le \ell$, we take $m_i = m^-_{i,n}$ for $i > \ell$, $m_i = m^-_{s,i}$ for $s \le i \le \ell$, $m_i = m^+_{1,i}$ for $i \le s$, and $m_i \in [m_e, m_s]$ for $i \in (e, s)$.
- We check whether all these values satisfy the conditions $m_i \in [\underline{m}_i, \overline{m}_i]$, and if yes, we compute the ratio $u(m)$.
- The smallest of these values is returned as the desired lower bound $\underline{u}_a$.

Computational complexity. For each i, computing the next value $m^{\pm}_{i,j}$ from the previous one requires requires one step, so all these values can be computed in time $O(n^2)$.

For each of $O(n^3)$ combinations of values, we need linear time to compute the ratio $u(m)$. Thus, totally, we need $O(n^3) \cdot O(n) = O(n^4)$ steps. This is polynomial time, i.e., this algorithm is feasible; see, e.g., [1].

5 Justification of the New Algorithm

Plan. The justification of our algorithm is based on the ideas used in the justification of the existing algorithms for defuzzifying interval-values fuzzy sets. So, to explain our justification, we will first recall these ideas, and then explain how they can be modified to compute $\underline{u}_a$ and $\overline{u}_a$ instead of $\underline{u}$ and $\overline{u}$.

Derivation of the original formula for $\overline{u}$: reminder. Let us start with the maximum. Let $\widetilde{m}_1, \ldots, \widetilde{m}_n$ be the values at which the maximum is attained. It is well known, from calculus, that when maximum is attained inside the interval $\widetilde{m}_i \in (\underline{m}_i, \overline{m}_i)$, then the corresponding partial derivative $\dfrac{\partial u}{\partial m_i}$ is equal to 0.

When the maximum is attained at $\widetilde{m}_i = \underline{m}_i$, then we cannot have $\dfrac{\partial u}{\partial m_i} > 0$, since then, for some small $\varepsilon > 0$, the value at $m_i = \widetilde{m}_i + \varepsilon$ will be even larger. Thus, we must have $\dfrac{\partial u}{\partial m_i} \le 0$.

Similarly, when the maximum is attained at $\widetilde{m}_i = \overline{m}_i$, then we cannot have $\dfrac{\partial u}{\partial m_i} < 0$, since then, for some small $\varepsilon > 0$, the value at $m_i = \widetilde{m}_i - \varepsilon$ will be even larger. Thus, we must have $\dfrac{\partial u}{\partial m_i} \ge 0$.

The partial derivative of the expression (2) is straightforward to compute: it is equal to

$$\frac{\partial}{\partial m_i}\left(\frac{\sum\limits_{j=1}^{n} x_j \cdot m_j}{\sum\limits_{j=1}^{n} m_j}\right) = \frac{x_i \cdot \left(\sum\limits_{j=1}^{n} m_j\right) - \sum\limits_{j=1}^{n} x_j \cdot m_j}{\left(\sum\limits_{j=1}^{n} m_j\right)^2} = \frac{x_i - u}{\sum\limits_{j=1}^{n} m_j}.$$

Since all the values of m_j of the membership function are non-negative, the sign of the partial derivative coincides with the sign of the difference $x_i - \overline{u}$.

Thus, we arrive at the following conclusions:

- if $\underline{m}_i < \widetilde{m}_i < \overline{m}_i$, then $x_i = \overline{u}$;
- if $\widetilde{m}_i = \underline{m}_i$, then $x_i \le \overline{u}$;
- if $\widetilde{m}_i = \overline{m}_i$, then $x_i \ge \overline{u}$.

So, if $x_i < \overline{u}$, we cannot have $\underline{m}_i < \widetilde{m}_i < \overline{m}_i$ and we cannot have $\widetilde{m}_i = \overline{m}_i$, so the only remaining possibility is $\widetilde{m}_i = \underline{m}_i$.

Similarly, if $x_i > \overline{u}$, we cannot have $\underline{m}_i < \widetilde{m}_i < \overline{m}_i$ and we cannot have $\widetilde{m}_i = \underline{m}_i$, so the only remaining possibility is $\widetilde{m}_i = \overline{m}_i$.

It should be mentioned that when $x_i = \overline{u}$, then replacing $\widetilde{m}_i$ with any other value $m_i \in [\underline{m}_i, \overline{m}_i]$ does not change the expression (2) and thus, for this particular i, we can pick any value $m_i \in [\underline{m}_i, \overline{m}_i]$.

Thus, we arrive at the following formula.

Resulting formula for $\overline{u}$. In the discrete case, the maximum $\overline{u}$ is attained when we choose $m_i = \underline{m}_i$ for all i for which $x_i < \overline{u}$ and $m_i = \overline{m}_i$ for all i for which $x_i \ge \overline{u}$:

$$\overline{u} = \frac{\sum\limits_{i:x_i<\overline{u}} x_i \cdot \underline{m}_i + \sum\limits_{j:x_j\ge\overline{u}} x_j \cdot \overline{m}_j}{\sum\limits_{i:x_i<\overline{u}} \underline{m}_i + \sum\limits_{j:x_j\ge\overline{u}} \overline{m}_j}. \tag{5}$$

Comment. Similarly, in the continuous case, we arrive at the formula (3).

A similar formula for $\underline{u}$. Similarly, we can conclude that in the discrete case, the minimum $\underline{u}$ is attained when we choose $m_i = \overline{m}_i$ for all i for which $x_i < \underline{u}$ and $m_i = \underline{m}_i$ for all i for which $x_i \ge \underline{u}$:

$$\underline{u} = \frac{\sum\limits_{i:x_i<\underline{u}} x_i \cdot \overline{m}_i + \sum\limits_{j:x_j\ge\underline{u}} x_j \cdot \underline{m}_j}{\sum\limits_{i:x_i<\underline{u}} \overline{m}_i + \sum\limits_{j:x_j\ge\underline{u}} \underline{m}_j}. \tag{6}$$

How can we actually compute $\overline{u}$ and $\underline{u}$: towards an algorithm for the general case. How can we perform these computations in the general case? The above formulas (5) and (6) require that we know $\overline{u}$ and $\underline{u}$ in order to find the appropriate values $m_i \in [\underline{m}_i, \overline{m}_i]$. Thus, the above formulas do not directly lead to an efficient algorithm for computing $\overline{u}$ and $\underline{u}$.

The possibility to efficiently compute $\overline{u}$ and $\underline{u}$ comes from the fact that, e.g., in the formula (5), all we need to know is where exactly $\overline{u}$ is in comparison with the values $x_1 < x_2 < \ldots < x_n$. For simplicity, let us supplement these values with $x_0 = -\infty$ and $x_{n+1} = +\infty$. Then, the real line is divided into $n+1$ (finite or infinite) intervals $(x_k, x_{k+1}]$, $k = 0, 1, \ldots, n$. So, to find $\overline{u}$, it is sufficient to try all these $n+1$ intervals.

We will describe the arguments in details for the case of the maximum. For the minimum, the arguments are similar.

If $x_k < \overline{u} \le x_{k+1}$, then the formula (5) can be rewritten as $\overline{u} = \overline{u}_k \stackrel{\text{def}}{=} \dfrac{\overline{N}_k}{\overline{D}_k}$, where

$$\overline{N}_k \stackrel{\text{def}}{=} \sum_{i=1}^{k} x_i \cdot \underline{m}_i + \sum_{j=k+1}^{n} x_j \cdot \overline{m}_j,$$

and

$$\overline{D}_k \stackrel{\text{def}}{=} \sum_{i=1}^{k} \underline{m}_i + \sum_{j=k+1}^{n} \overline{m}_i.$$

We only need to consider values k for which $x_k < \overline{u}_k \le x_{k+1}$.

So, we compute the ratios $\overline{u}_k$ for all k, keep only those ratios for which the inequality $x_k < \overline{u}_k \le x_{k+1}$ is satisfied, and then return the largest of the kept ratios $\overline{u}_k$ as the desired value of $\overline{u}$.

Additional ideas for speeding up the computation of $\underline{u}$ and $\overline{u}$: reminder. For a standard defuzzification (2), we need to perform a liner number of steps $O(n)$: n multiplications and $n-1$ additions to compute the numerator, $n-1$ additions to compute the denominator, and 1 division to compute the ratio $u(m)$. Let us show that we can compute $\overline{u}$ in linear time as well.

For $k = 0$, we can compute $\overline{N}_0$ and $\overline{D}_0$ in linear time. Then, when we move from $\overline{N}_k$ to $\overline{N}_{k+1}$ (or from $\overline{D}_k$ to $\overline{D}_{k+1}$), we only to change one term, so we only need a finite number of steps. Thus, to find all n ratios, we only need a linear number of steps.

Let us summarize the resulting algorithm.

Algorithm for computing $\overline{u}$: reminder.

- First, we compute $\overline{N}_0 = \sum\limits_{j=1}^{n} x_j \cdot \overline{m}_j$ and $\overline{D}_0 = \sum\limits_{j=1}^{n} \overline{m}_j$.
- Then, for $k = 1, 2, \ldots, n$, we compute $\overline{N}_{k+1} = \overline{N}_k - x_k \cdot (\overline{m}_k - \underline{m}_k)$ and $\overline{D}_{k+1} = \overline{D}_k - (\overline{m}_k - \underline{m}_k)$.
- For each k, we compute the ratio $\overline{u}_k = \dfrac{\overline{N}_k}{\overline{D}_k}$, and check whether

$$x_k < \overline{u}_k \le u_{k+1};$$

  if this inequality is satisfied, we keep $\overline{u}_k$ as a possible value.
- The largest of these possible values is then returned as $\overline{u}$.

Comment. A similar efficient (linear time) algorithm can be used to compute $\underline{u}$.

Algorithm for computing $\underline{u}$: reminder.

- First, we compute $\underline{N}_0 = \sum\limits_{j=1}^{n} x_j \cdot \underline{m}_j$ and $\underline{D}_0 = \sum\limits_{j=1}^{n} \underline{m}_j$.

– Then, for $k = 1, 2, \ldots, n$, we compute $\underline{N}_{k+1} = \underline{N}_k + x_k \cdot (\overline{m}_k - \underline{m}_k)$ and $\underline{D}_{k+1} = \underline{D}_k + (\overline{m}_k - \underline{m}_k)$.
– For each k, we compute the ratio $\underline{u}_k = \dfrac{\underline{N}_k}{\underline{D}_k}$, and check whether

$$x_k < \underline{u}_k \le u_{k+1};$$

if this inequality is satisfied, we keep $\underline{u}_k$ as a possible value.
– The smallest of these possible values is then returned as $\underline{u}$.

Comment: How to compute $\overline{u}$ and $\underline{u}$ in the analytical case. For the case when $\underline{m}(x)$ and $\overline{m}(x)$ are given by analytical formulas, we can explicitly integrate both numerator and denominator and get algebraic equations for the unknown values $\overline{u}$ or $\underline{u}$.

Towards a solution for the new problem. We want to find a sequence m_i that attains the largest possible value $\overline{u}_a$ among all unimodal sequences. Let ℓ be the mode location for this sequence.

Let us fix the values $\overline{u}_a$ and ℓ and see how we can use the inequalities similar to the ones used in the justification of the known algorithm.

When $x_i < \overline{u}_a$, then, as we mentioned earlier, the derivative $\dfrac{\partial u}{\partial m_i}$ is negative and thus, we cannot decrease $\widetilde{m}_i$. In the past, we only had one restriction: that $m_i \ge \underline{m}_i$. Now, we have additional restrictions: e.g., for $i \le \ell$, that $m_i \ge m_j$ for all $j < i$. Thus, the fact that we cannot decrease m_i means that either $\widetilde{m}_i = \underline{m}_i$ or that $\widetilde{m}_i = \widetilde{m}_j$ for some $j < i$. In the second case, for $\widetilde{m}_j$, we can repeat the same argument, and eventually, we will find that $\widetilde{m}_i = \widetilde{m}_j$ for some value j which cannot be decreased because it is equal to $\widetilde{m}_j = \underline{m}_j$. Thus, we have $\widetilde{m}_i = \underline{m}_j$.

In general, since $i \le \ell$, we have $\widetilde{m}_i \ge \widetilde{m}_j \ge \underline{m}_j$. Thus, we have $\widetilde{m}_i \ge \max(\underline{m}_1, \ldots, \underline{m}_i)$. Since we concluded that $\widetilde{m}_i$ is equal to one of these lower endpoints, it cannot be larger than the largest of them, so we have $\widetilde{m}_i = \max(\underline{m}_1, \ldots, \underline{m}_i)$.

For $i > \ell$, we may also have $\widetilde{m}_i = \overline{m}_j$ for some j for which $x_j > \overline{v}_a$. In this case, the values m_k between i and j are constant.

Thus, the "past-mode" part ($i > \ell$) of the optimal solution can be divided into three zones:

– first, there is a zone $[\ell, s]$ (s for *start*) before $\overline{v}_a$ where we have

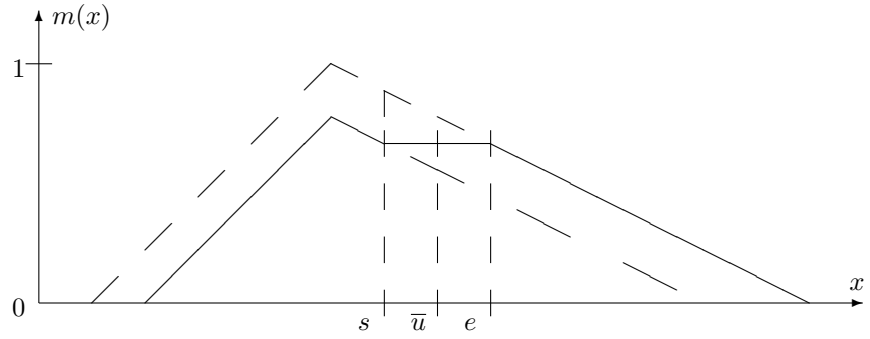
$$\widetilde{m}_i = \max(\underline{m}_i, \ldots, \underline{m}_s);$$

– then, there is a zone $[e, n]$ (e for *end*) past $\overline{v}_a$ where we have

$$\widetilde{m}_i = \min(\overline{m}_i, \ldots, \overline{m}_n);$$

– finally, in the zone between s and e, the values are constant.

So, to describe all such solutions, it is sufficient to try all possible values of three indices: ℓ, s and e.



Thus, we arrive at the above algorithm for computing $\overline{u}_a$. An algorithm for $\underline{u}_a$ can be similarly justified.

6 Remaining Open Problems

How to speed up computations? The existing algorithms for defuzzifying interval-values fuzzy sets requires linear time $O(n)$ to compute both endpoints $\underline{u}$ and $\overline{u}$ of the defuzzification interval $[\underline{u}, \overline{u}]$. In contrast, the new algorithm for computing the "more adequate" defuzzification interval $[\underline{u}_a, \overline{u}_a]$ requires a much longer time $O(n^4)$. It is therefore desirable to come up with a faster way of computing the new interval $[\underline{u}_a, \overline{u}_a]$.

How to take normality into consideration. In addition to unimodularity, another reasonable restriction on fuzzy sets is *normality* – that there exists a value m_i which is equal to 1. It is desirable to extend our algorithm to the case when instead of limiting ourselves simply to unimodal membership functions $m(x) \in [\underline{m}(x), \overline{m}(x)]$, we limit ourselves to unimodal *and* normal membership functions. Since we introduce an additional limitation on $m(x)$, we should get an even narrower interval $[\underline{u}_b, \overline{u}_b] \subseteq [\underline{u}_a, \overline{u}_a]$. How to compute this interval is an interesting open problem.

Acknowledgments

This work was supported in part by NSF grant HRD-0734825, by Grant 1 T36 GM078000-01 from the National Institutes of Health, and by the Japan Advanced Institute of Science and Technology (JAIST) International Joint Research Grant 2006-08. The authors are thankful to the anonymous referees for valuable suggestions.

References

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C.: *Introduction to Algorithms*, MIT Press, Cambridge, Massachusetts, 2001.
2. Klir, G., Yuan, B.: *Fuzzy Sets and Fuzzy Logic*, Prentice Hall, New Jersey, 1995.
3. Kreinovich, V., Lakeyev, A., Rohn, J., Kahl, P.: *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Kluwer Academic Press, 1998.
4. Lea, R., Kreinovich, V., Trejo, R.: "Optimal interval enclosures for fractionally-linear functions, and their application to intelligent control", *Reliable Computing* 2(3):265–286, 1996.
5. Mendel, J. M.: *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice-Hall, 2001.
6. Mendel, J. M.: "Type-2 fuzzy sets and systems: an overview", *IEEE Computational Intelligence Magazine* 2:20–29, 2007.
7. Nguyen, H. T., Walker, E. A.: *First Course in Fuzzy Logic*, CRC Press, Boca Raton, Florida, 2006.
8. Wu, D. R., Mendel, J. M. "Enhanced Karnik-Mendel Algorithms", *IEEE Transactions on Fuzzy Systems* 17:923–934, 2009.