From Semi-Heuristic Fuzzy Techniques To Optimal Fuzzy Methods: Mathematical Foundations and Applications

Vladik Kreinovich

Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
email vladik@cs.utep.edu

Abstract

Fuzzy techniques have been successfully used in various application areas ranging from control to image processing to decision making. In all these applications, there is usually:

- a general idea, and then
- there are several possible implementations of this idea; e.g., we can use:
  - different membership functions,
  - different "and" and "or" operations,
  - different defuzzifications, etc.

In the first approximation, the results are usually reasonably robust and independent on this choice, so any heuristic or semi-heuristic choice works OK. However:

- if we want to further improve the semi-heuristic "good enough" control or image processing techniques,
- we must actually make the selection that would lead to an optimal fuzzy method.

Even if we know the exact optimality criterion, the resulting optimization problem is so complicatedly non-linear that, often, no known optimization techniques can be used. In many real-life situations, we do not even know the exact criterion: the relative quality of different controls or image processing techniques may be not by exact numerical criteria, but rather by expert opinions which are, themselves, fuzzy.

In this paper, we describe a general mathematical technique for solving such optimization problems. This technique is based on the group-theoretic (symmetry) methodology, a methodology which is one of the main successful tools of modern physics. This methodology has lead us to the justification of many heuristic formulas in fuzzy logic, fuzzy control, neural networks, etc. (many examples are presented in our 1997 Kluwer book *Applications of continuous mathematics to computer science*).

We also present new applications of this techniques, including applications:

- to optimal fuzzy control, and
- to optimal fuzzy image processing.

1 Introduction

Main objectives of science and engineering, and computational problems generates by these objectives. What are the main objectives of science and engineering in general?

- first, we want to *find out* what is happening in the world;
- second, based on the information on the current state of the world, we want to be able to *predict* what will be happening in the future; and
- finally, if the predicted situation is not perfect for us, we would like to be able to *control* the situation and thus, make it more preferable for us.

These three tasks are the tasks of different difficulty:

- To find out what is actually going on, in principle, all we need is *measure* (or otherwise *estimate*) the corresponding quantities.
  - If the direct measurement or estimation is possible, then the only problem that we may face is that we may have *several* different results of measuring (or estimating) of the same quantity. So, we face a problem of *combining* different measurement and/or estimation results, i.e., the problem of *data fusion*.
  - In some cases, direct measurement or direct estimation of the desired quantity is not possible, so we must first measure (or estimate) some related quantities, and then apply some *data processing* to estimate the value of the desired quantity.
- If we also want to predict and to control, it is not enough to be able to measure and to estimate. To be able to predict and to control, we need to develop *knowledge* which would allow us to *predict* the future situations, both:

– *without* our interference (for the purposes of pure prediction), and

– *with* a certain control strategy in place (for control problems).

One possible way to acquire this *dynamical* knowledge is to record and analyze how different real-life situations actually changed, and then try to find some general laws which describe these changes.

Unfortunately, different situations are often so drastically different, that their dynamic behaviors have seemingly nothing in common; therefore, to be able to extract dynamic laws from the observed dynamics, we must first *divide* these situations into clusters with similar behavior; thus, in order to be able to predict and control, we must first be able to *cluster*. Therefore, to serve the main objectives of science and engineering, we must be able to perform the following computation-related tasks:

- *data fusion*;
- *data processing*;
- *clustering*;
- *prediction*; and
- *control*.

The need for fuzzy methods. Traditionally, all above computational tasks were based on *measurement* results. In many situations, the traditional engineering measurement-based methods lead to very accurate predictions and to a very efficient control. In some other situations, however, the quality of the measurement-based control and prediction is not that satisfactory, while human experts and controllers can achieve much better results. In these situations, it is desirable to add *human expertise*, i.e., expert estimates and expert rules, to the measurement results. For example, a human drives a car much better than an automatic controller (of course, it is worth mentioning that the performance of a good driver is greatly enhanced by the processors in the car which choose the best gear and/or the best fuel injection strategy, etc.).

The better the expert, the better the results. It is therefore desirable to have a top expert in all the situations. However, in every field, be it medicine, geology, or driving, there are few top experts, and it is impossible to use them in all the cases. Hence, it is desirable to design *automated* systems which would incorporate the knowledge of these experts and thus achieve better prediction, better control, etc.

To *incorporate* human knowledge into computer-based systems, we must first *describe* this knowledge in terms which are understandable to a computer, and then teach the computer how to *process* thus represented expert knowledge. One problem with describing the human knowledge to a computer is that:

- computers were originally desired to handle *numbers* (i.e., numerical, measurement-related data), while
- expert knowledge often comes in terms of *words* from natural language; for example:
  - when an expert estimates a *size* of the object, he does not usually say that the size is exactly between 1 and 3 cm; he may say that the size is "around 2 cm", or just that "the object is small";
  - similarly, when an expert controller formulates a control rule for a car, he does not formulate it in exact numerical terms; these rules sound more like "if you going fast, and the road goes bad so that the ride gets bumpy, then slow down".

In 1965, L. Zadeh has proposed a methodology, which he called *fuzzy* methodology, for translating this expert knowledge into the terms which are understandable to a computer. The main idea of this methodology is as follows:

- When we used precise properties like "ℓ is between 1 cm and 3 cm", then for each object (i.e., for each value of ℓ''), this statement is either true or false.

  In the computer:

  * "true" is normally represented as 1, and
  * "false" normally represented as 0.

- When we use imprecise words instead of precisely defined properties, e.g., when we say that "ℓ is small", then we get more possibilities than before:
  - if an object is small enough, then we are 100% sure that the corresponding value of ℓ is indeed small;
  - if an object is large enough, then we are 100% sure that the corresponding value of ℓ is *not* small;
  - however, there are many values of ℓ for which we are not 100% sure that it is small, nor we are 100% sure that the object is not small; we have, instead, some *degree of belief* (or *degree of certainty*) with which we believe that this object is small.

  These *degrees of belief* describe the situations intermediate between "absolutely true" (described by 1) and "absolutely false" (described by 0). It is therefore reasonable to describe such intermediate degrees of belief by numbers between 0 and 1.

Combining these numbers with the numbers 0 and 1 (which correspond to "absolutely false" and "absolutely true"), we conclude that:

- instead of the statements with truth values from the 2-element set $\{0,1\}$,
- we now have statements with "truth values" from the entire interval $[0,1]$.

Similarly:

- in contrast to "crisp" *properties* $P(x)$ which can be described as functions assigning to every object x from the Universe of discourse U a value "true" or "false" (0 or 1),
- words from natural language (like "small") correspond to *fuzzy properties*, i.e., functions which assign to every element $x \in U$ a number from the interval $[0,1]$, i.e., the "degree of truth" that this object x satisfies the property P; such a function is usually called a *membership function* and denoted by $\mu_P(x)$.

In our processing of expert estimates, we may need to combine several pieces of evidence. Thus, we need to be able, given the truth values of two statements A and B, to estimate the degrees of belief in the *logical combinations* of these statements, such as $A\&B$ or $A \vee B$. If we do not know the interrelation between A and B, only our degrees of belief $d(A)$ and $d(B)$ in A and B, then this is the only information that we can use to compute the estimated degrees of belief $d(A\&B)$ and $d(A \vee B)$. Thus, we must describe *functions* $f_\&$ and $f_\vee$ which transform $d(A)$ and $d(B)$ into, correspondingly, an estimate $f_\&(d(A), d(B))$ for $d(A\&B)$ or an estimate $f_\vee(d(A), d(B))$ for $d(A \vee B)$. These functions $f_\& : [0,1] \times [0,1] \to [0,1]$ and $f_\vee : [0,1] \times [0,1] \to [0,1]$ are called "and" and "or" operations, or, alternatively, *t-norms* and *t-conorms*.

As a result of processing fuzzy data, we do not immediately get a *crisp* decision, we may get a *fuzzy* decision of the type "swerve a little bit to the left" or "slow down a little bit". These decisions are easily understood by *human* operators, but, since our goal is an *automated* control, we must somehow transform these *fuzzy* instructions into *crisp* ones, i.e., we must somehow *defuzzify* the fuzzy instructions.

Thus, to apply fuzzy methodology to above problems, we must make the following choices:

- first, we must choose membership functions,
- second, we must choose "and" and "or" operations,
- third, we must choose a defuzzifications.

Towards optimal fuzzy methods. Fuzzy techniques have been successfully used in various application areas ranging from control to image processing to decision making (see, e.g., [24, 61]). In all these applications, the corresponding implementation of fuzzy methodology (i.e., the choice of membership functions, "and" and "or" operations, and of a defuzzification) is usually selected on a heuristic (or semi-heuristic) basis.

It is known that in principle, different implementations lead to different results, but in the *first approximation*, the results are usually reasonably robust and independent on this choice, so any heuristic or semi-heuristic choice works OK. However:

- if we want to further improve the semi-heuristic "good enough" control or image processing techniques,
- we must actually make the selection that would lead to an *optimal* fuzzy method, i.e., a method which best suits the particular problem.

Thus, there is a need to go from fuzzy methodology to *optimal* fuzzy methodology.

Finding the optimal fuzzy methodology is a very difficult problem.

- Even *if we know* the exact optimality criterion, the resulting optimization problem is so complicatedly non-linear that, often, no known optimization techniques can be used.
- In many real-life situations, we *do not* even *know* the exact criterion: the relative quality of different controls or image processing techniques may be determined not by exact numerical criteria, but rather by expert opinions which are, themselves, fuzzy. In such situations, the problem of choosing the best methodology is even more difficult to solve.

What we are planning to do. In this talk, we describe a general mathematical technique for solving the optimization problems related to choosing the best fuzzy methodology.

This technique is based on the group-theoretic (symmetry) methodology, a methodology which is one of the main successful tools of modern physics. This methodology has lead us to the justification of many heuristic formulas in fuzzy logic, fuzzy control, neural networks, etc.; many examples are presented in our 1997 Kluwer book [55].

The methods described in [55], however, mainly concentrate on the *continuous* and *smooth* (*differentiable*) situations.

- This is quite acceptable for *neural networks*, because neural networks are based on the elementary neurons, in which, typically, the input-output relation is *smooth*: e.g., in the most widely used neuron, the output y is related to the inputs $x_1, \ldots, x_n$ by a formula

  $$y = s_0(w_1 \cdot x_1 + \ldots + w_n \cdot x_n - w_0),$$

  where w_i are real numbers, and

  $$s_0(z) = \frac{1}{1 + e^{-z}}.$$

- In *fuzzy logic*, however, non-smooth and even discontinuous functions are very common; let us just give two examples:
  - The cases when all the data comes from expert estimates is extremely rare. Typically, some data comes from measurements, and some data – from expert estimates. So, if we use fuzzy methodology for data processing, we must consider not only *fuzzy* properties (corresponding to expert estimates), but also *crisp* properties (corresponding to measurements).
    * A membership function which describes a *fuzzy* property is usually *continuous* (and often smooth).
    * On the other hand, a *crisp* property, by definition, is described by a membership function which can only take values 0 and 1 ("true" and "false") but which cannot take any intermediate values. Thus, such a function cannot continually change its value from 0 to 1, and it has to be *discontinuous*.
  - Even when all the properties are fuzzy, and represented by *smooth* membership functions, non-smoothness often stem from the "and" and "or" operations. Indeed, the two most widely used pairs of "and" and "or" operations (originally proposed by Zadeh himself) are:
    * $f_\&(a,b)=\max(a,b)$ and $f_\vee(a,b)=\max(a,b)$;
    * $f_\&(a,b)=a\cdot b$ and $f_\vee(a,b)=a+b-a\cdot b$.

    Operations from the second pair are *differentiable*, but the operations from the first pair are *not* differentiable for $a=b$.

There are other examples of fuzzy-related discontinuous and non-differentiable functions.

In this paper, we will show, on several important examples, how a technique from [55] can be extended to such fuzzy-related non-smooth and non-differentiable functions. This exposition will be illustrated by the examples from all five applications areas of fuzzy methodology: data fusion, data processing, clustering, prediction, and control.

2 Optimal choice of fuzzy methodology in data fusion and data processing

Let us first briefly describe what is already known about the best choice of fuzzy methodology for data fusion and data processing:

Choice of membership functions: general results.

- The authors of [48, 49] compared the quality of the approximation achieved by using different shapes of membership functions. Their numerical experiments have shown that in almost all test situations, the *best approximation* if we use the "sinc" membership function $\sin(x)/x$.

  The paper [25] contains a partial explanation of this result: namely, it is proven that in linear approximation, the function $\sin(x)/x$ is indeed the best (in some reasonable sense). It is desirable to extend this explanation to the general (non-linear) case.

- The most *robust* membership functions (i.e., the least sensitive to the inaccuracy of the input data) are piecewise-linear ones [53, 57].

  This result explains why the piecewise-linear membership functions are, at present, most frequently used.

Choice of "and" and "or" operations: general results. (These results are (mainly) summarized in [40, 41, 53, 57, 67, 10].)

- If we are looking for the operations which are *most robust* (i.e., least sensitive to the inaccuracy with which we measure the membership functions), then, depending on what exactly we are looking for, we can get two different results:
  - if we are looking for the operations which are the most robust *in the the worst case*, then the best choice is to use $f_\&(a,b)=\min(a,b)$ and $f_\vee(a,b)=\max(a,b)$ [56, 59, 53, 57, 61];
  - if we are looking for the operations which are the most robust *in the average*, then the best choice is to use $f_\&(a,b)=a\cdot b$ and $f_\vee(a,b)=a+b-a\cdot b$ [60, 53, 57, 61];
  - instead of minimizing the *average* error, we can try to minimize the corresponding *entropy* [63, 64, 65, 38, 32, 33]:
    * if we use the *average* entropy (in some reasonable sense), we get the same pair of optimal functions as for average error;
    * for an appropriately defined *worst-case* entropy the optimal operations are $f_\&(a,b)=\min(a,b)$ and $f_\vee(a,b)=a+b-a\cdot b$.
- If we are looking for the operations which are the *fastest to compute*, then the best choice is to use $f_\&(a,b)=\min(a,b)$ and $f_\vee(a,b)=\max(a,b)$ [45].

In all these cases, the optimal "and" and "or" operations are the ones which are listed above as the most frequently used:

- for "and", we get $f_\&(a,b) = \min(a,b)$ or $f_\&(a,b) = a \cdot b$;
- for "or", we get $f_\vee(a,b) = \max(a,b)$ or $f_\&(a,b) = a + b - a \cdot b$.

Comment. These optimization results are in good accordance with the general group-theoretic approach that enables us to classify techniques that are optimal relative to arbitrary reasonable criteria [10, 40, 41, 55, 67].

Optimal choice of membership functions for data fusion: main idea. The above results gave a very general description of which membership functions are the best, a description which does not depend on whether we consider data fusion, data processing, or control. Let us now be more specific and consider specific requirements of *data fusion*.

Optimal choice of membership functions for data fusion: we need a family of membership functions. We are interested in having a family F of membership functions that would enable us to describe any possible knowledge. These functions must be representable in a computer; in any computer, we can only represent finitely many parameters. Therefore, we are looking for a *finite-parametric* (*finite-dimensional*) family of membership functions (finite-parametric in the sense that fixing the values of finitely many (m) parameters would be sufficient to pick any function from this family).

Let us describe which requirements are natural to impose on this family F if we want it to be adequate for fuzzy data fusion.

Optimal choice of membership functions for data fusion: the family must be unit-invariant. In data fusion, we have to combine knowledge coming from different experts, and this knowledge may be represented in *different measuring units*.

> For example, one expert may estimate the length of an object in feet, while another expert may have his estimate in meters.

To combine this knowledge, we have to *transform* these expert estimates into a single measuring unit. In mathematical terms, changing to a new unit which is λ times smaller means that every numerical value x of the estimated quantity changes from x to $\widetilde{x} = \lambda \cdot x$. Thus, the expert estimate which was expressed by a membership function $\mu(x)$ from a function F is now transformed into a new membership function $\widetilde{\mu}(x) = \mu(\lambda \cdot x)$. It is reasonable to require that this new function also belong to the function F.

> Otherwise, if this function does not belong to the family F of computer-representable membership functions:

– we will *not* be able to directly *represent* the new membership function $\widetilde{\mu}(x)$ in a computer;
– instead, we will need to *approximate* this function by a computer-representable membership function which will be, in general, *different* from $\widetilde{\mu}(x)$, and will, thus, be not 100% adequate in representing the fused knowledge.

To avoid this inadequacy, we will, therefore, need the above requirement.

Optimal choice of membership functions for data fusion: the family must be shift-invariant. Similarly, different experts may use different *starting points* for the estimated quantity x.

> For example, when estimating temperature, one expert may use Celsius, while another expert may use Fahrenheit. It is known that the difference in these two scales is not only that they have different units – their starting points are also different.

In mathematical terms, changing a starting point means that every numerical value x of the estimated quantity changes from x to $\widetilde{x} = x + s$. Thus, the expert estimate which was expressed by a membership function $\mu(x)$ (from a family F) is now transformed into a new membership function $\widetilde{\mu}(x) = \mu(x + s)$. Similarly to the above argument about changing units, it is reasonable to require that this new function also belong to the family F.

Optimal choice of membership functions for data fusion: the family must support combination of evidence (data fusion). Finally, after we have transformed all expert estimates to the same scale, we must *combine* them. The simplest possible case is when all expert estimates are consistent, so all we have to do is to say that the first expert's statement is true, *and* the second expert's statement is true, etc. The statement of i-th expert is expressed by a membership function $\mu_i(x)$ from the family F; therefore, the resulting combined knowledge can be expressed by applying an "and" operation to these membership function, i.e., by a function $\mu(x) = f_\&(\mu_1(x), \mu_2(x), \ldots)$. Similarly to the first two requirements, it is reasonable to require that this function belong to the same family F (because otherwise, if this function does not belong to F and we need to approximate this function to represent it in a computer, we will be adding an extra inadequacy to our computer representation).

Thus, we arrive at the following definitions:

Definition 2.1. *Let m be a positive integer. By an m-dimensional family F of membership functions, we mean a continuous mapping $F : U \to M$, where U is an open region in an m-dimensional space R^m, and M is a set of all membership function (with topology corresponding to Hausdorff metric). We say that a function $\mu(x)$ belongs to the family F (and denote it by $\mu(x) \in F$) if $\mu(x) = F(\vec{c})$ for some $\vec{c} \in U$.*

Definition 2.2. *Let* $f_\&: [0,1] \times [0,1] \to [0,1]$. *We say that a family is appropriate for data fusion if it satisfies the following three properties:*

- *if* $\mu(x) \in F$ *and* $\lambda > 0$, *then* $\mu(\lambda \cdot x) \in F$;
- *if* $\mu(x) \in F$ *and* $s \in R$, *then* $\mu(x+s) \in F$;
- *if* $\mu_1(x) \in F$, $\mu_2(x) \in F$, $\ldots$, $\mu_n(x) \in F$, *then* $f_\&(\mu_1(x), \ldots, \mu_n(x)) \in F$.

Smooth case: main result. We have already mentioned that according to most optimality criteria, two "and"-operations are optimal:

$$f_\&(a,b) = a \cdot b \text{ and } f_\&(a,b) = \min(a,b).$$

The first of these two operations is *smooth* and it is, therefore, reasonable to require that membership functions are *smooth* (differentiable) as well. For technical reasons, it is also necessary to require that these membership functions are everywhere positive (from the computer viewpoint, it does not matter much, because 0 can be always approximated, within an arbitrary accuracy, by a small positive real number). For such a smooth case, we have the following result:

Theorem 2.1. [34, 41, 42, 43] *Let* $f_\&(a,b) = a \cdot b$, *and let* F *be a finite-dimensional family of smooth everywhere positive membership functions which is appropriate for data fusion. Then, every membership function* $\mu(x)$ *from the family* F *has the form* $\mu(x) = \exp(-P(x))$ *for some polynomial* $P(x)$.

Smooth case: simplest membership functions appropriate for data fusion are Gaussian. The smaller the degree of the polynomial $P(x)$, the fewer parameters we need to represent the corresponding membership functions.

- We cannot only have polynomials of 0-th degree (constants), because such a polynomial corresponds to a constant membership function, i.e., to the case when all possible values x have the same degree of possibilities, and thus, the experts know nothing at all.
- We cannot have polynomials of 1-st degree (linear functions), because every non-degenerate linear function $P(x)$ takes negative values for some x, and thus, the corresponding value $\mu(x) = \exp(-P(x))$ will be > 1, which is impossible for a membership function.

Thus, the simplest possible case is when $P(x)$ is a second-order polynomial, i.e., when the membership functions $\mu(x)$ are *Gaussian* membership functions.

Gaussian membership functions are indeed actively used in fuzzy methodology. Their additional advantage is that the corresponding data fusion is *easy to compute*: namely, if we use $f_\&(a,b) = a \cdot b$ to combine Gaussian membership functions $\mu_i(x) = \exp((x-a_i)^2/(\sigma_i)^2)$, $1 \le i \le n$, then we the combination $\mu(x) = \mu_1(x) \cdot \ldots \cdot \mu_n(x)$ is also a Gaussian membership function

$\mu(x) = \exp((x-a)^2/\sigma^2$ with

$$a = \frac{a_1 \cdot (\sigma_1)^{-2} + \ldots + a_n \cdot (\sigma_n)^{-2}}{(\sigma_1)^{-2} + \ldots + (\sigma_n)^{-2}}$$

and $\sigma^{-2} = (\sigma_1)^{-2} + \ldots + (\sigma_n)^{-2}$ [24, 34]. These are computationally very simple formulas to implement.

In addition to adequacy for *data fusion*, Gaussian membership functions have advantages for other parts of fuzzy methodology as well:

Gaussian membership functions are good for fuzzy data processing. [26, 27, 52] The main objective of fuzzy data processing is as follows:

- we know the expert estimates for the quantities x_i (expressed by membership functions $\mu_i(x_i)$);
- we know that the desired quantity y is related to x_i by a formula $y = f(x_1, \ldots, x_n)$ with a known algorithm f; and
- we want to compute the resulting estimate for y.

The formula for the desired membership function is expressed by the *extension principle* (see, e.g., [24]):

$$\mu(y) = \sup_{x_1,\ldots,x_n : y = f(x_1,\ldots,x_n)} f_\&(\mu_1(x_1), \ldots, \mu_n(x_n)). \tag{2.1}$$

When the estimates are accurate enough, i.e., when $x_i \approx a_i$ for some known a_i, we can neglect the quadratic and higher order terms in the Taylor expansion of the function $f(x_1, \ldots, x_n)$, and use a simplified linear formula

$$f(x_1, \ldots, x_n) \approx c_0 + c_1 \cdot (x_1 - a_1) + \ldots + c_n \cdot (x_n - a_n),$$

where

$$c_0 = f(a_1, \ldots, a_n) \text{ and } c_i = \frac{\partial f(x_1, \ldots, x_n)}{\partial x_i}_{|x_1 = a_1, \ldots, x_n = a_n}.$$

For such linear functions, if we use $f_\&(a,b) = a \cdot b$ and Gaussian membership functions $\mu_i(x) = \exp(-(x-a_i)^2/(\sigma_i)^2)$, then $\mu(y)$ is also a Gaussian membership function $\mu(y) = \exp(-(x-a)^2/\sigma^2$, with $a = c_0$ and $\sigma = \sqrt{c_1^2 \cdot \sigma_1^2 + \ldots + c_n^2 \cdot \sigma_n^2}$. These formulas are easy to compute.

Gaussian membership functions are good for clustering. (see next section).

Gaussian membership functions are good for fuzzy control. Indeed, as shown in [70, 71, 71, 73, 74], they have a *universal approximation* property in the sense that every control strategy can be approximated by a fuzzy control strategy based on Gaussian membership functions and $f_\&(a,b) = a \cdot b$.

In traditional fuzzy rule based modeling, the condition of each rule is a *conjunction* of several conditions related to different inputs, i.e., each condition is of the type $A_1(x_1)\& \ldots \& A_n(x_n)$. Crudely speaking, we can say that in these conditions, different inputs are *independent* in the sense that the degree with which each input x_i satisfies the condition does not depend on the values of the other inputs. In particular, when each condition $A_i(x_i)$ is described by a Gaussian membership function, and we use product as "and", the resulting membership function for the condition $\mu(x) = \mu_1(x_1) \cdot \ldots \cdot \mu_n(x_n)$ is also Gaussian, and its α-cut (i.e., the set of all values $x = (x_1, \ldots, x_n)$ for which $\mu(x) \geq \alpha$) is an *ellipsoid* with axes coinciding with the coordinate axes.

In some real-life cases, however, there *is* a dependency. As a result, some conditions A_i^j in some expert rules may describe not a single input, but a *combination* of such inputs. The simplest case is when we have a *linear combination*. In this case, each expert rule has the form

$$\text{If } x_1^{(j)} \text{ is } A_1^j \text{ and } x_2^{(j)} \text{ is } A_2^j \text{ and } \ldots \text{ and } x_n^{(j)} \text{ is } A_n^j, \text{ then } u \text{ is } B^j,$$

where $x_i^{(j)} = w_{i1}^{(j)} \cdot x_1 + \ldots + w_{in}^{(j)} \cdot x_n$ is a linear combination of the (measured) inputs x_i. If we use Gaussian membership functions and $a \cdot b$ as "and", then the resulting membership function is still Gaussian, and its α-cut is an *arbitrarily* oriented ellipsoid. For rules with such ellipsoid-based conditions, a universal approximation property is proven in [13, 14, 15, 23, 28, 29, 30, 31]

(For a general overview of universal approximation results for fuzzy control, see [35].)

Smooth case: non-Gaussian membership functions. A Gaussian membership function $\mu(x) = \exp(-(x-a)^2/\sigma^2)$ describes a simple piece of knowledge about the quantity x, the knowledge of the type "x is approximately equal to a, with an error of size σ". To describe more complicated knowledge, we need functions $\mu(x) = \exp(-P(x))$ with polynomials $P(x)$ of higher degree. For example, the knowledge of the type "x is either approximately equal to a_1 or to a_2" can be represented by a membership function $\exp(-P(x))$ with a 4-th order polynomial $P(x) = (x - a_1)^2 \cdot (x - a^2)^2/\sigma^2$.

Non-smooth case. According to most optimality criteria, two "and"-operations are optimal:

- a *smooth* operation $f_\&(a, b) = a \cdot b$ and
- a *non-smooth* operation $f_\&(a, b) = \min(a, b)$.

We have described appropriate families for the smooth "and"-operation. Let us now describe appropriate families for the non-smooth "and"-operation.

Before we formulate our result, we need a technical comment. When we considered smooth operations, it was natural to require that the membership functions are also smooth; now, we no longer require smoothness or even continuity of membership functions (moreover, as we will see, we have to abandon continuity). It is known, from mathematics, that there are many weird discontinuous functions, e.g., a function $f(x)$ which is equal to 1 if x is rational number and 0 otherwise. These weird function clearly do not represent any reasonable expert estimates. Therefore, to avoid such weird functions, we will require some *regularity*: namely, we will require that a membership function be *piece-wise monotonic*.

Definition 2.3. *We say that a function $f(x) : R \to [0,1]$ is piece-wise monotonic if the real line can be divided into finitely many intervals (finite or infinite) on each of which this function is either non-decreasing, or non-increasing.*

We will prove that for $f_{\&} = \min$, the appropriate membership functions are *piecewise-constant* in the following precise sense:

Definition 2.4. *We say that a function $f(x) : R \to [0,1]$ is piece-wise constant if the real line can be divided into finitely many intervals (finite or infinite) on each of which this function is constant.*

Theorem 2.2. *Let $f_{\&}(a,b) = \min(a,b)$, and let F be a finite-dimensional family of piece-wise monotonic membership functions which is appropriate for data fusion. Then, every membership function $\mu(x)$ from the family F is piece-wise constant.*

Comments.

- Piece-wise constant fuzzy sets are known and actively used under the name of *nested intervals* and *nested sets*; see, e.g., the survey [54] and references therein.
- This theorem was proved in collaboration with Hung T. Nguyen and Leonid K. Reznik.
- For reader's convenience, all the proofs are placed in the special (last) proofs section.

3 Optimal choice of fuzzy methodology in clustering

The main results of this section are described in detail in [36, 37, 77].

Clustering is important. The idea of clustering is very natural in science: The analysis of every new phenomenon starts with *classification*, when instead of *numerous* different *examples*, we have a *few classes*. Classification helped to analyze chemical elements, elementary particles, living organisms, astronomical objects, etc.

In some situations, where assumptions about structure of data can be formulated in statistical terms, statistical techniques (see, e.g., [19]) are appropriate if we have sufficiently many data. In other situations, we must use heuristic classification methods, in particular, methods that use fuzzy logic. The main idea of fuzzy clustering is described in [4, 5, 6, 7, 8, 9, 11, 12, 16, 22, 75, 76].

The goal of fuzzy clustering: "typical" representatives and how to use them. We start with objects which we want to classify (i.e., to cluster). To classify, we use several (numerical) characteristics of these object. Let us denote the total number of these characteristics by s. The s real numbers that characterize each object can be naturally viewed as a point in s-dimensional space R^s. Thus, having n objects means that we have n points $\vec{x}_1, \ldots, \vec{x}_n$ in this space. These n points are the input for clustering.

As a result of clustering, we want to describe several clusters. Each cluster can be characterized by its "typical" element $\vec{t}_j \in R^s$. After these typical elements $\vec{t}_1, \ldots, \vec{t}_q$ a re found, we can then classify each object $\vec{x} \in R^s$ according to which typical element it is closest to. This "classification" is a fuzzy notion:

- if an element $\vec{x}$ is very close to, say, $\vec{t}_1$, and not close to any other typical representative, then it is reasonable to conclude that $\vec{x}$ belongs to class 1;
- however, if an object $\vec{x} \in R^s$ is almost equally close to two different representatives $\vec{t}_1$ and $\vec{t}_2$, then it is reasonable to conclude that this object belongs, to some extent, to *both* clusters 1 and 2.

To express this idea in precise terms, we select a function $f(\vec{x})$ (called *potential function*) such that for every two point $\vec{x}$ and $\vec{y}$ from R^s, the value $f(\vec{x} - \vec{y})$ describes to what extent $\vec{x}$ and $\vec{y}$ are close. This function is usually non-negative, and the closer $\vec{x}$ and $\vec{y}$, the larger the value of the potential function. Potentially, as a potential function, we can use a *membership function* which describes the relation "$\vec{x}$ and $\vec{y}$ are close"; however, from the mathematical viewpoint, the choice of membership function would mean that we only allow $f(\vec{x})$ to take values from the interval $[0, 1]$, and sometimes, more general values are needed (in our main text, we will explain why we need such values).

When the potential function is selected, then we can say that an object $\vec{x}$ belongs to 1-st cluster with a degree $f(\vec{x} - \vec{t}_1)$, to the 2-nd cluster with the degree $f(\vec{x} - \vec{t}_2)$, ..., and to q-th cluster with the degree $f(\vec{x} - \vec{t}_q)$. Since we do not require any normalization of the function $f(\vec{x})$, it is convenient to *normalize* these values so that they will add up to 1, in other words, to describe the degree to which x belongs to j-th cluster as

$$d_j(\vec{x}) = \frac{f(\vec{x} - \vec{t}_j)}{f(\vec{x} - \vec{t}_1) + \ldots + f(\vec{x} - \vec{t}_q)}. \tag{3.1}$$

How to find "typical" representatives? The most widely used approach. We have described how to classify an object when the clusters (or, to be more precise, their typical representatives) have already been found. How can we find these representatives?

The most widely used fuzzy clustering method is the method of Fuzzy C-Means (Fuzzy ISODATA) [4, 5, 6, 7, 8, 9, 16, 22]. This method is based on the natural idea that each characteristic of a typical representative should be equal to an average over all elements of the corresponding cluster. If we have *crisp* clustering, then we would simply take the arithmetic average. However, since we have *fuzzy* clustering, it is natural to count, in this average, each element $\vec{x}_i$ with the weight $d_j(\vec{x}_i)$ that is proportional to this element's degree of belonging to the cluster. In other words, it is natural to require that for each j,

$$\vec{t}_j = \frac{d_j(\vec{x}_1) \cdot \vec{x}_1 + \ldots + d_j(\vec{x}_n) \cdot \vec{x}_n}{d_j(\vec{x}_1) + \ldots + d_j(\vec{x}_n)}. \tag{3.2}$$

This method leads to *good quality* clustering. Its main *disadvantage* is that since the values $d_j(\vec{x}_i)$, in their turn, depend on $\vec{t}_j$, the equation (3.2) is, actually, a non-linear system of equations for determining the cluster "centers" $\vec{t}_1, \ldots, \vec{t}_q$, and solving this system of equations often requires lots of *computation time.*

How to find "typical" representatives? Recent approaches. To simplify computations, a new method has been recently proposed [75, 76] (see also [11, 12]). This method is based on the following idea: when we say that an element $\vec{t}_j$ is a typical representative of the cluster that consists of elements $\vec{x}_{i_1}, \ldots, \vec{x}_{i_k}$, we mean that for each element $\vec{x} \in R^s$, the degree $f(\vec{x} - \vec{t}_j)$ with which $\vec{x}$ is close to $\vec{t}_j$ is equal to the average of the degrees $f(\vec{x} - \vec{x}_{i_1}), \ldots, f(\vec{x} - \vec{x}_{i_k})$ with which $\vec{x}$ is close to all elements of this cluster:

$$f(\vec{x} - \vec{x}_{i_1}) + \ldots + f(\vec{x} - \vec{x}_{i_k}) = k \cdot f(\vec{x} - \vec{t}_j). \tag{3.3}$$

If we have a *crisp* classification, then each of the original data points $\vec{x}_1, \ldots, \vec{x}_n$ belongs to one and only one cluster and therefore, by adding equalities (3.3) for all q clusters, we would conclude that

$$\sum_{i=1}^{n} f(\vec{x} - \vec{x}_i) = \sum_{j=1}^{q} k_j \cdot f(\vec{x} - \vec{t}_j), \tag{3.4}$$

where k_j is the total number of elements in j-th cluster (i.e., the *cardinality* of j-th cluster).

For a *fuzzy* clustering, it is reasonable to expect a similar formula, with k_j being the *fuzzy cardinality* of j-th cluster (see, e.g., [24]). So, to find $\vec{t}_j$, we can do the following:

- compute, for all $\vec{x}$ (from a grid), the function

$$M(\vec{x}) = \sum_{i=1}^{n} f(\vec{x} - \vec{x}_i).$$

- represent this function $M(\vec{x})$ as a sum

$$M(\vec{x}) = \sum_{j=1}^{q} k_j \cdot f(\vec{x} - \vec{t}_j)$$

for the smallest possible number of clusters.

Theoretically, the smallest possible number of clusters is 1, in which case $M(\vec{x}) = k_1 \cdot f(\vec{x} - \vec{t}_1)$. If one cluster is indeed sufficient, then, due to the properties of the "closeness" function $f(\vec{x})$, we can find $\vec{t}_1$ easily: it is the value for which $M(\vec{x})$ is the largest possible. In this case, if $f(\vec{x})$ is normalized in such a way that $f(\vec{0}) = 1$ (i.e., if $f(\vec{x})$ is a membership function, and $\vec{x}$ is close to $\vec{x}$ with degree of truth 1), we can take $k_1 = M(\vec{t}_1)$.

In view of this observation, it is reasonable to select, as $\vec{t}_1$, the value for which $M(\vec{x})$ is the largest possible. In this case, we cannot take $k_1 = M(\vec{t}_1)$, because other clusters are also contributing to this value $M(\vec{t}_1)$. Instead, we can take $k_1 = q \cdot M(\vec{t}_1)$ for some number $q \in (0, 1)$. After that, we can subtract $k_1 \cdot f(\vec{x} - \vec{t}_1)$ from the original function $M(\vec{x})$, and use a similar method to represent the new function $M_1(\vec{x}) = M(\vec{x}) - k_1 \cdot f(\vec{x} - \vec{t}_1)$ as a sum

$$M_1(\vec{x}) = \sum_{j=2}^{q} k_j \cdot f(\vec{x} - \vec{t}_j);$$

etc. We stop when the remainder becomes small enough.

This method is very similar to a very successful method of image reconstruction used in radio astronomy under the name of *CLEAN* (see, e.g., [20]). Due to the success of the CLEAN method, it is not surprising that this clustering method also turned out to be reasonably successful.

Main problem: how to choose a potential function? We have mentioned that the above fuzzy clustering methods turned out to be very successful, but we must clarify this statement: these methods are very successful provided we appropriately choose the potential function $f(\vec{x})$. For a different choice of $f(\vec{x})$, the resulting clustering may not be that good.

To the best of our knowledge, so far, the choice of the potential function was mainly done either empirically or heuristically. The following three families of potential functions are most widely used:

- in the original Fuzzy C-Means method, the function $f(\vec{x}) = |\vec{x}|^{-m}$ is used, where $|\vec{x}|$ is the norm of a vector $\vec{x}$, and $m > 0$ is a positive real number;
- in [75, 76], the potential function $f(\vec{x}) = \exp(-\alpha \cdot |\vec{x}|)$ is used; and
- in [11, 12], the Gaussian potential function $f(\vec{x}) = \exp(-\alpha \cdot |\vec{x}|^2)$ is used.

The first choice is used when we have no information about the typical cluster radius; the second and third choices presuppose that an approximate cluster radius is already known.

In this section, we show that these three choices are indeed optimal in some reasonable sense. Thus, we provide a theoretical justification of these empirical and heuristic choices.

Optimal in what sense? The main idea. We are looking for the *best* (*optimal*) choice of a potential function.

Normally, the word "best" is understood in the sense of some *numerical* optimality criterion. However, in our case of *fuzzy* choice, it is often difficult to formulate the exact *numerical* criterion. Instead, we assume that there is an *ordinal* criterion, i.e., that we can compare arbitrary two choices, but that we cannot assign numerical values to these choices.

It turns out that in many cases, there are reasonable *symmetries*, and it is natural to assume that the (ordinal) optimality criterion is invariant with respect to these symmetries. Then, we are able to describe all choices that are optimal with respect to some invariant ordinal optimality criteria.

This general approach was described and used in [10, 40, 41, 55, 67], in particular, for fuzzy control. In this section, we will show that this approach is applicable to fuzzy clustering as well.

Let us borrow from the experience of modern physics and use symmetries. In modern physics, symmetry groups are a tool that enables to compress complicated differential equations into compact form (see, e.g., [68]). Moreover, the very differential equations themselves can be uniquely deduced from the corresponding symmetry requirements (see, e.g., [18, 17]).

It is possible to use symmetry. As we have mentioned, in our previous papers, we have shown that the symmetry group approach can be used to find optimal membership functions, optimal t-norms and t-conorms, and optimal defuzzification procedures.

It is therefore reasonable to expect that the same approach can also be used to choose the best potential function for fuzzy clustering.

We must choose a family of functions. We must select a potential function $f(\vec{x})$. The only way the potential function $f(\vec{x})$ is used in clustering is through the normalized formula (3.1). Because of the normalization, if we re-scale the values of the potential function, i.e., if we choose a constant $C > 0$ and consider a new potential function $\widetilde{f}(\vec{x}) = C \cdot f(\vec{x})$, this new potential function will lead to exactly the same values $d_j(\vec{x})$ as the old one. Therefore, from the viewpoint of fuzzy clustering, there is no way to distinguish between the functions $f(\vec{x})$ and $\widetilde{f}(\vec{x}) = C \cdot f(\vec{x})$. So, based on clustering behavior, we cannot choose a *single* function $f(\vec{x})$; we can only choose a 1-parametric *family* of functions $\{C \cdot f(\vec{x})\}$ that is characterized by a parameter C.

Comment about notations. In the following text, we will denote families of functions by capital letters, such as F, F', G, etc.

We must choose the best family of functions. We want to select the *best* family of functions.

What is a criterion for choosing a family of functions? What does it mean to choose a *best* family of functions? It means that we have some *criterion* that enables us to choose between the two families.

Traditionally, optimality criteria are *numerical*, i.e., to every family F, we assign some value $J(F)$ expressing its quality, and choose a family for which this value is maximal (i.e., when $J(F) \geq J(G)$ for every other alternative G). However, it is not necessary to restrict ourselves to such numeric criteria only.

For example, if we have several different families F that have the same classification ability $P(F)$, we can choose between them the one that has the minimal computational complexity $C(F)$. In this case, the actual criterion that we use to compare two families is not numeric, but more complicated:

> *A family F_1 is better than the family F_2 if and only if*
>
> – *either $P(F_1) > P(F_2)$,*
>
> – *or $P(F_1) = P(F_2)$ and $C(F_1) < C(F_2)$.*

A criterion can be even more complicated.

The only thing that a criterion *must* do is to allow us, for every pair of families (F_1, F_2), to make one of the following conclusions:

- the first family is better with respect to this criterion (we'll denote it by $F_1 \succ F_2$, or $F_2 \prec F_1$);
- with respect to the given criterion, the second family is better ($F_2 \succ F_1$);
- with respect to this criterion, the two families have the same quality (we'll denote it by $F_1 \sim F_2$);
- this criterion does not allow us to compare the two families.

Of course, it is necessary to demand that these choices be consistent.

> For example, if $F_1 \succ F_2$ and $F_2 \succ F_3$ then $F_1 \succ F_3$.

The criterion must be final, i.e., it must pick the unique family as the best one. A natural demand is that this criterion must choose a *unique* optimal family (i.e., a family that is better with respect to this criterion than any other family).

The reason for this demand is very simple: If a criterion *does not choose* any family at all, then it is of no use. If *several* different families are the best according to this criterion, then we still have the problem of choosing the best among them. Therefore we need some additional criterion for that choice, like in the above example:

If several families $F_1, F_2, \ldots$ turn out to have the same classification ability ($P(F_1) = P(F_2) = \ldots$), we can choose among them a family with minimal computational complexity ($C(F_i) \to \min$).

So what we actually do in this case is abandon that criterion for which there were several "best" families, and consider a new "composite" criterion instead: F_1 is better than F_2 according to this new criterion if either it was better according to the old criterion, or they had the same quality according to the old criterion and F_1 is better than F_2 according to the additional criterion.

In other words, if a criterion does not allow us to choose a unique best family, it means that this criterion is not final, we'll have to modify it until we come to a final criterion that will have that property.

The criterion must not change if we change the measuring unit for $\vec{x}$. The exact mathematical form of a function $f(\vec{x})$ depends on the exact choice of units for measuring the s coordinates $x_1, \ldots, x_s$ of $\vec{x} \in R^s$. If we replace each of these units by a new unit that is λ times larger, then the same physical value that was previously described by a numerical value x_k will now be described, in the new units, by a new numerical value $\widetilde{x}_k = x_k/\lambda_j$. For example, if we replace centimeters by inches, with $\lambda = 2.54$, then $x_k = 5.08$ cm becomes $\widetilde{x}_k = x_k/\lambda = 2$ in. After this transformation, $\vec{x}$ changes to $\widetilde{\vec{x}} = \vec{x}/\lambda$.

How will the expression for closeness $f(\vec{x})$ change if we use the new units? In terms of $\widetilde{\vec{x}}$, we have $\vec{x} = \lambda \cdot \widetilde{\vec{x}}$. Thus, if we change the measuring unit for $\vec{x}$, the same dynamics that was originally represented by a function $f(\vec{x})$, will be described, in the new units, by a function $\widetilde{f}(\vec{x}) = f(\lambda \cdot \vec{x})$.

Since we assumed that we have no information about the cluster radii, there is no reason why one choice of unit should be preferable to the other. Therefore, it is reasonable to assume that the relative quality of different families should not change if we simply change the units, i.e., if the family F is better than a family G, then the transformed family $\widetilde{F}$ should also be better than the family $\widetilde{G}$.

The criterion must not change if we apply a rotation. Similarly, it is reasonable to require that the relative quality of two different families of functions do not change if we apply an arbitrary *rotation* around 0 in s-dimensional space R^s.

We are now ready for the formal definitions.

Definition 3.1.

- *By a family F, we mean a differentiable function $f(\vec{x})$ from R^s to R.*
- *We say that a function $e(\vec{x})$ belongs to the family $f(\vec{x})$ (or that $f(\vec{x})$ contains the function $e(\vec{x})$) if $e(\vec{x}) = C \cdot f(\vec{x})$ for some $C > 0$.*
- *Two families F and G are considered equal if they contain the same functions.*

Let's denote the set of all possible families by Φ.

- The set of all pairs (F_1, F_2) of elements $F_1 \in \Phi$, $F_2 \in \Phi$, is usually denoted by $\Phi \times \Phi$.
- An arbitrary subset R of a set of pairs $\Phi \times \Phi$ is called a *relation* on the set Φ. If $(F_1, F_2) \in R$, it is said that F_1 and F_2 are in relation R; this fact is denoted by $F_1 R F_2$.

Definition 3.2. *A pair of relations $(\prec, \sim)$ on a set Φ is called consistent if it satisfies the following conditions, for every $F, G, H \in \Phi$:*

(1) *if $F \prec G$ and $G \prec H$ then $F \prec H$;*

(2) *$F \sim F$;*

(3) *if $F \sim G$ then $G \sim F$;*

(4) *if $F \sim G$ and $G \sim H$ then $F \sim H$;*

(5) *if $F \prec G$ and $G \sim H$ then $F \prec H$;*

(6) *if $F \sim G$ and $G \prec H$ then $F \prec H$;*

(7) *if $F \prec G$ then it is not true that $G \prec F$, and it is not true that $F \sim G$.*

Definition 3.3. *Assume a set Φ is given. Its elements will be called alternatives.*

- *By an optimality criterion, we mean a consistent pair $(\prec, \sim)$ of relations on the set Φ of all alternatives.*
  - *If $F \succ G$ we say that F is better than G;*
  - *if $F \sim G$ we say that the alternatives F and G are equivalent with respect to this criterion.*
- *We say that an alternative F is optimal (or best) with respect to a criterion $(\prec, \sim)$ if for every other alternative G either $F \succ G$ or $F \sim G$.*
- *We say that a criterion is final if there exists an optimal alternative, and this optimal alternative is unique.*

Comment. In this section, we will consider optimality criteria on the set Φ of all families.

Definition 3.4. *Let $\lambda > 0$ be a positive real number.*

- *By a λ-rescaling of a function $f(\vec{x})$ we mean a function $\widetilde{f}(\vec{x}) = f(\lambda \cdot \vec{x})$.*
- *By a λ-rescaling of a family of functions F we mean the family consisting of λ-rescalings of all functions from F.*

Denotation. *λ-rescaling of a family F will be denoted by $R_\lambda(F)$.*

Definition 3.5. *We say that an optimality criterion on Φ is* unit-invariant *if for every two families F and G and for every number $\lambda > 0$, the following two conditions are true:*

i) if F is better than G in the sense of this criterion (i.e., $F \succ G$), then $R_\lambda(F) \succ R_\lambda(G)$;

ii) if F is equivalent to G in the sense of this criterion (i.e., $F \sim G$), then $R_\lambda(F) \sim R_\lambda(G)$.

Definition 3.6. *Let $T : R^s \to R^s$ be a rotation around 0 in s-dimensional space.*

- *By a* T-rotation *of a function $f(\vec{x})$ we mean a function $\widetilde{f}(\vec{x}) = f(T\vec{x})$.*
- *By a* T-rotation *of a family of functions F we mean the family consisting of T-rotations of all functions from F.*

Denotation. *T-rotation of a family F around 0 will be denoted by $T(F)$.*

Definition 3.7. *We say that an optimality criterion on Φ is* rotation-invariant *if for every two families F and G and for every rotation T, the following two conditions are true:*

i) if F is better than G in the sense of this criterion (i.e., $F \succ G$), then $T(F) \succ T(G)$;

ii) if F is equivalent to G in the sense of this criterion (i.e., $F \sim G$), then $T(F) \sim T(G)$.

Comment. As we have already remarked, the demands that the optimality criterion is final, unit-invariant, and rotation invariant are quite reasonable. At first glance they may seem rather trivial and therefore weak, because these demands do not specify the exact optimality criterion. However, these demands are strong enough, as the following theorem shows:

Theorem 3.1. *If a family F is optimal in the sense of some optimality criterion that is final, unit-invariant, and rotation-invariant, then every function $f(\vec{x})$ from this family F has the form $C \cdot |\vec{x}|^\alpha$ for some real numbers C and α.*

Comments.

- Thus, our general approach provides a precise mathematical justification for the (highly successful) potential functions used in Fuzzy C-Means approach.
- Since none of the optimal functions are from the interval $[0,1]$, our result explains why we cannot restrict ourselves to membership functions $f(\vec{x})$, and why we need to consider the potential functions which can attain values outside the interval $[0,1]$.
- For the case when we *have* the prior knowledge of the cluster radius, a similar approach explains the potential functions $f(\vec{x}) = \exp(-\alpha \cdot |\vec{x}|)$ and $f(\vec{x}) = \exp(-\alpha \cdot |\vec{x}|^2)$. The proof of this result is presented in detail in our Technical Report [37].
- The proof of Theorem 3.1 actively uses the differentiability and the related differential equations. This relation with differentiability is not surprising if we recall that the symmetry method comes from physics where differential equations are the main way of describing dynamics.

4 Optimal choice of fuzzy methodology in prediction

We must choose a family of functions. At any given moment of time, the current state of a system can be describe by the values of finitely many parameters $x_1, \ldots, x_n$. To be able to predict how the system changes, we must be able to describe how each parameter changes with time, i.e., we must be able to describe n functions $f_i(x_1, \ldots, x_n)$ $(1 \le i \le n)$ which describes the dependence of the rate change

$$\dot{x}_i = \frac{dx_i}{dt}$$

on the current values $x_1, \ldots, x_n$:

$$\frac{dx_i}{dt} = f_i(x_1, \ldots, x_n).$$

We are talking about the situations in which we do not have the exact knowledge of these functions, we only have *expert estimates* for them. So, we must describe a family of functions which would correspond to different types of expert knowledge.

Comment about notations. In the following text, we will denote families of functions by capital letters, such as F, F', G, etc.

Reasonable conditions on the desired family of functions, and what these conditions lead to. For a complex system, we usually have many independent processes that lead to the change in x_i. These processes can be present separately or at the same time.

> For example, the increase in ozone pollution can be caused by industrial pollution, or by frequent thunderstorms.

If the first factor leads to the rate $f(x_1, \ldots, x_n)$, and the second factor leads to the rate $f'(x_1, \ldots, x_n)$, then both factors together lead to the rate $f(x_1, \ldots, x_n) + f'(x_1, \ldots, x_n)$.

Thus, if two functions are reasonable (i.e., belong to the desired family F), their sum should also be reasonable (i.e., should also belong to the same family F). In mathematical terms, the family F should be *closed under addition.*

The second condition on the desired family F follows from the fact that the intensity of a process can change. Thus, if $f(x_1, \ldots, x_n)$ is a reasonable rate of change, then for every real number λ, the product $\lambda \cdot f(x_1, \ldots, x_n)$ is also a reasonable rate of change:

- the values $\lambda \in (0, 1)$ describe the *decreased* intensity;
- the values $\lambda > 1$ describe the *increased* intensity; and
- the values $\lambda < 0$ describe the *reversed* process.

Thus, if $f \in F$, then $\lambda \cdot f \in F$.

Together with the first condition, we can conclude that if the functions $f_1, \ldots, f_m$ belong to F and $c_1, \ldots, c_m$ are real numbers, then the linear combination $f = c_1 \cdot f_1 + \ldots + c_m \cdot f_m$ must also belong to the family F. In mathematical terms, the family $\mathcal{F}$ must be a *linear space.*

It is known, from linear algebra, that linear spaces can be described as follows: every linear space has a subset $\{e_1, e_2, \ldots\}$ called a *basis*, such that every element e from the linear space can be represented as a linear combination of elements from this basis: $e = c_1 \cdot e_1 + c_2 \cdot e_2 + \ldots$ The smallest possible number of elements in this basis is called a *dimension* of the linear space.

In principle, some spaces are infinite-dimensional, but with an infinite basis, we can represent an arbitrary function of n variables; so, if we want our family to be meaningful, we must restrict ourselves only to *finite-dimensional* linear spaces, i.e., to linear spaces F formed by functions of the type $f(x_1, \ldots, x_n) = c_1 \cdot e_1(x_, \ldots, x_n) + \ldots + e_m \cdot e_m(x_1, \ldots, x_m)$, where $e_j(x_1, \ldots, x_n)$ are fixed functions, and c_j are arbitrary real numbers.

For such families, choosing the family means choosing the corresponding m functions $e_1(x_1, \ldots, x_n), \ldots, e_m(x_1, \ldots, x_n)$.

We must choose the best family of functions. We want to select the *best* transformation from expert words to functions. This means, in particular, that we are interested in choosing the *best* family of functions.

The criterion must not change if we change the measuring units for one of the variables x_i. The exact mathematical form of a function $f_i(x_1, \ldots, x_n)$ depends on the exact choice of units for measuring $x_1, \ldots, x_n$. If, for some j, we replace a unit for measuring x_j by a new unit that is λ_j times larger, then the same physical value that was previously described by a numerical value x_j will now be described, in the new units, by a new numerical value $\widetilde{x}_j = x_j/\lambda_j$. For example, if we replace centimeters by inches, with $\lambda_j = 2.54$, then $x_j = 5.08$ cm becomes $\widetilde{x}_j = x_j/\lambda_j = 2$ in.

How will the dynamical equations $\dot{x}_i = f_i(x_1, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_n)$ change if we use the new unit? In terms of $\widetilde{x}_j$, we have $x_j = \lambda_j \cdot \widetilde{x}_j$, and thus, we have $\dot{x}_i = f_i(x_1, \ldots, x_{j-1}, \lambda_j \cdot x_j, x_{j+1}, \ldots, x_n)$. In other words, if we change the measuring unit for x_j, the same dynamics that was originally represented by a function $f_i(x_1, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_n)$, will be described, in the new units, by a function $\widetilde{f}(x_1, \ldots, x_n) = f_i(x_1, \ldots, x_{j-1}, \lambda_j \cdot x_j, x_{j+1}, \ldots, x_n)$.

If we make a similar replacement of the measuring units for several quantities x_j, so that x_1 is replaced by a unit that is λ_1 times larger, x_2 by a unit that is λ_2 times larger, etc., then each function $f_i(x_1, \ldots, x_n)$ will be replaced by a new function $\widetilde{f}_i(x_1, \ldots, x_n) = \lambda_i^{-1} \cdot f_i(\lambda_1 \cdot x_1, \ldots, \lambda_n \cdot x_n)$.

It is reasonable to assume that the relative quality of different families should not change if we simply change the units, i.e., if the family F is better than a family G, then the transformed family $\widetilde{F}$ should also be better than the family $\widetilde{G}$.

We are now ready for the formal definitions.

Definition 4.1. *Let two positive integers $n, m \geq 1$ be fixed, and let $i \leq n$.*

- *By a family F, we mean a collection of m differentiable function $e_1(x_1, \ldots, x_n), \ldots, e_m(x_1, \ldots, x_n)$.*
- *We say that a function $e(x_1, \ldots, x_n)$ belongs to the family F (and that F contains the function $e(x_1, \ldots, x_n)$) if this function can be represented as a linear combination of the functions e_j, i.e., if there exist m real numbers $c_1, \ldots, c_m$ for which, for all x_k, $e(x_1, \ldots, x_n) = c_1 \cdot e_1(x_1, \ldots, x_n) + \ldots + c_m \cdot e_m(x_1, \ldots, x_m)$.*
- *Two families F and G are considered equal if they contain the same functions.*

Denotation. *Let's denote the set of all possible families by Φ.*

Definition 4.2. *Let $\vec{\lambda} = (\lambda_1, \ldots, \lambda_n)$ be a tuple of positive real numbers.*

- *By a $\vec{\lambda}$-rescaling of a function $f(x_1, \ldots, x_n)$ we mean a function $\widetilde{f}(x_1, \ldots, x_n) = \lambda_i^{-1} \cdot f_i(\lambda_1 \cdot x_1, \ldots, \lambda_n \cdot x_n)$.*
- *By a $\vec{\lambda}$-rescaling of a family of functions F we mean the family consisting of $\vec{\lambda}$-rescalings of all functions from F.*

Denotation. *$\vec{\lambda}$-rescaling of a family F will be denoted by $R_{\vec{\lambda}}(F)$.*

Definition 4.3. *We say that an optimality criterion on Φ is unit-invariant if for every two families F and G and for every vector $\vec{\lambda}$, the following two conditions are true:*

i) if F is better than G in the sense of this criterion (i.e., $F \succ G$), then $R_{\vec{\lambda}}(F) \succ R_{\vec{\lambda}}(G)$;

ii) if F is equivalent to G in the sense of this criterion (i.e., $F \sim G$), then $R_{\vec{\lambda}}(F) \sim R_{\vec{\lambda}}(G)$.

Theorem 4.1. *If a family F is optimal in the sense of some optimality criterion that is final and unit-invariant, then every function $f_i(x_1, \ldots, x_n)$ from this family F is a linear combination of the functions of the type*

$$x_1^{\alpha_1} \cdot \ldots \cdot x_n^{\alpha_n} \cdot \ln^{k_1}(x_1) \cdot \ldots \cdot \ln^{k_n}(x_n),$$

where α_j are complex numbers, and k_j are non-negative integers.

Comments.

- This result was proven in collaboration with Brian Penn and Scott A. Starks.
- The above expression can be re-formulated without complex numbers; in this case, the basic functions are of the type

  $$x_1^{\alpha_1} \cdot \sin(\beta_1 \cdot \ln(x_1) + \varphi_1) \cdot \ldots \cdot x_n^{\alpha_n} \cdot \sin(\beta_n \cdot \ln(x_n) + \varphi_n) \cdot \ln^{k_1}(x_1) \cdot \ldots \cdot \ln^{k_n}(x_n),$$

  where α_j, β_j, and φ_j are real numbers, and k_j are non-negative integers.

In particular, for $n = 1$, we get the following result:

Corollary 4.1. *For $n = 1$, if an m-dimensional family F is optimal in the sense of some optimality criterion that is final and unit-invariant, then every function $f(x)$ from the family F is equal to a linear combination of the functions of the type $\ln^p(x) \cdot x^\alpha \cdot \sin(\beta \cdot \ln(x) + \varphi)$, where p is a non-negative integer, α, β and φ are real numbers.*

Corollary 4.2. *Let a 2-dimensional family F be optimal in the sense of some optimality criterion that is final and unit-invariant. Then, every function $f(x)$ from the family F has one of the following forms:*

1. $f(x) = C_1 \cdot x^{\alpha_1} + C_2 \cdot x^{\alpha_2}$;
2. $f(x) = C_1 \cdot x^\alpha + C_2 \cdot x^\alpha \cdot \ln(x)$;
3. $f(x) = C \cdot x^\alpha \cdot \sin(\beta \cdot \ln(x) + \varphi)$.

Comment. The optimal families that we have just described are exactly the ones that were described, on a semi-heuristic basis, by Ludwig von Bertalanffy in his General System Theory (see, e.g., his books [2, 3]).

Bertalanffy mainly considered equations of the *first* type. These so-called *Bertalanffy equations* turned out to be very adequate for describing growth in biology (namely, the growth of individual organisms and of their organs), so adequate that they are routinely used by fisheries in England and Japan and by by the Food and Agriculture Organization (FAO) of the United Nations. The following particular cases of the Bertalanffy equation describe the simplest growth processes:

- For $\alpha_1 = 1$, $C_1 > 0$, and $C_1 = 0$, we get the equation $\dot{x} = C_1 \cdot x$ that describe an *exponential* growth $x(t) = C \cdot \exp(C_1 \cdot t)$.
- For $\alpha_1 = 1$, $\alpha_2 = 2$, $C_1 > 0$, and $C_2 < 0$, we get the equation $\dot{x} = C_1 \cdot x - |C_2| \cdot x^2$ that describes a so-called *logistic curve* that starts with an exponential growth but then flatters out. For this particular growth function, the growth equation also admits an explicit solution

$$L(t) = \frac{1}{K + A \cdot b^t}.$$

Equations of the *second* type were originally proposed by Gompertz (for $\alpha = 1$). These equations describe, e.g., such growth processes as *population dynamics* (see, e.g., [62]),

Thus, our general approach provides a precise mathematical justification for the (highly successful) semi-heuristic formulas of von Bertalanffy's general system theory.

How to use this result: examples.

- If an increase in x_1 leads to a slower increase rate of x_2, this means that we have a term in $\dot{x}_2 = f_2(x_1, \ldots, x_n)$ that is decreasing with x_1. Since this terms should be monotonic, it should not contain sines, and therefore, it should be of the form $C \cdot x_2^\alpha \cdot \ln^k(x_2)$. The exact values of the coefficients must be determined in one of the following two ways:
  - either by showing the expert the results of different values and asking this expert to choose the most appropriate value;
  - or by tuning the resulting simulation to the actual recorded behavior of the system that we are simulating.
- Similarly, if we know that, e.g., x_3 starts decreasing if both x_1 and x_2 are present, then we should add, to f_3, terms of the type $-x_1^{\alpha_1} \cdot x_2^{\alpha_2}$, maybe with logarithmic terms as well.

What did we gain? At first glance, there still seems to be a lot of freedom of choice, and this is inevitable, because we are developing a general formalism that should cover many different systems. However, we did gain a lot:

- initially, we had the choice of choosing several arbitrary *functions*;
- now, we only need to choose a few *parameters*.

In some real-life situations, dynamics is not smooth. The above theorem describes the situations when the dynamics is *smooth*, i.e., when:

- the dependence of x_i on time is differentiable, and
- the dependence of the rate change $\dot{x}_i$ on the variables x_j is also differentiable.

In some real-life situations, however, the dynamics is *not* smooth (see, e.g., [46, 47, 51, 66]):

- Experiments with human learning shows that the learned performance x_i does not increase smoothly with time, but rather increases by sudden "jumps" ("leaps", "learning insights"), followed by periods of relatively constant performance.
- Similarly, human growth (and similar growth of other living beings) is not continuous, but comes in almost discontinuous *bursts* (*spurts*), between which the growth is almost 0.

How can we describe such dynamics?

Main idea: let us use generalized function (Schwartz distributions). From the practical viewpoint, a discontinuous dynamics ("jump"), a change which occurs momentarily, is indistinguishable from a continuous but very fast dynamical change (which occurs during a very small period of time T). Of course, if we fix T, and increase the accuracy of measuring time, we will be able to distinguish between the instantaneous jump and the change which takes time T. Therefore, we can consider the instantaneous change as a *limit* of continuous changes which occur during the time period T as $T \to 0$. When $T \to 0$, the rate change, which is equal to const$/T$, tends to ∞. So, this rate change ("derivative") is no longer a normal function, with finite values for all moments of time t, it is a "generalized" function which may take *infinite* values.

The need for such "generalized" functions has been felt by physicists for quite some time, and in the 1940s-1950s, a mathematically consistent theory of generalized functions (also called *Schwartz distributions*) was developed (see. e.g., [21, 69]).

It is worth mentioning that this theory was used not only in physics, but in fuzzy systems as well:

- In [60], this theory is used to describe average sensitivity of "and" and "or" operations, and to find the operations with the smallest possible values of average sensitivity.
- In [41], this theory is used to describe smoothness of membership functions and controls, and to find the "and" and "or" operations and membership functions which guarantee the largest possible smoothness.

Basic examples of generalized functions include:

- a *step* function $\theta(x)$ which is equal to 0 when $x < 0$ and equal to 1 when $x > 0$;
- a *delta* function $\delta(x)$ which can be defined as the derivative of the step function; this function is equal to 0 when $x \neq 0$ and equal to ∞ when $x = 0$;
- we can differentiate the delta function and get its first, second, etc., derivatives $\delta'(x)$, $\delta''(x)$, etc.

In the theory of generalized functions, every continuous function $f(x)$ has a derivative, which may be a generalized function (e.g., the derivative of a step function is a delta function which is not a regular function). Thus, to describe discontinuous and non-smooth dynamics, we must look for equations of the type $\dot{x}_i = f_i(x_1, \ldots, x_n)$ in which f_i can be *generalized* functions. To cover such dynamics, we must therefore generalize Theorem 4.1 to families which consist of possibly generalized functions:

Definition 4.4. *By a basic generalized function, we mean one of the following functions: 1, step function $\theta(x)$, delta function $\delta(x)$, and the first, second, etc., derivatives of the delta function.*

Theorem 4.2. *If a family F of generalized functions is optimal in the sense of some optimality criterion that is final and unit-invariant, then every function $f_i(x_1, \ldots, x_n)$ from this family F is a linear combination of the functions of the type*

$$b_1(x_1) \cdot \ldots \cdot b_n(x_n) \cdot x_1^{\alpha_1} \cdot \ldots \cdot x_n^{\alpha_n} \cdot \ln^{k_1}(x_1) \cdot \ldots \cdot \ln^{k_n}(x_n),$$

where b_j are basic generalized functions, α_j are complex numbers, k_j are non-negative integers.

Comments.

- This result was also obtained in collaboration with Brian Penn and Scott A. Starks.
- Regular functions correspond to the case when $b_j(x) = 1$ for all j. In this case, Theorem 4.2 reduces to Theorem 4.1.

- With non-regular generalized functions, we can represent discontinuous and non-smooth dynamics: e.g., a "jump" in learning ability x_1 at a moment t_0, accompanied by the steady increase of the amount of learned material x_2, can be described by the following system of differential equations: $\dot{x}_1 = \Delta \cdot \delta(x_2)$, $\dot{x}_2 = \text{const}$. When the value x_2 reaches the level 0, the value of x_1 immediately increases by Δ.

5 Optimal choice of fuzzy methodology in fuzzy control

What properties do we require from control? In order to find out what implementation of fuzzy methodology is the best, let us briefly enumerate the basic properties that we can require of control.

First property: stability (control must control). The main objective of control is that it should control. For example, if we control a car on the road, then, for the largest part of the trip, one of the main objectives of this control is to make sure that it stays in its lane with the desired speed, i.e., that whenever it will accidentally deviate from the straight course, the steering control will return it back on course, and when the speed would deviate, the acceleration or deceleration would bring it back to the optimal cruise speed.

In the general case, we want a control that, after an initial deviation, will bring the controlled system back "on track". This property is called *stability* of the control (and of the controlled system).

Stability is a matter of closed loop analysis: one and the same control strategy can be stable for one controlled system (described by a certain set of differential equations), but become unstable for a slightly different system.

Even for a fixed system, stability is a generic term. There are many different particular notions of stability, depending on how big initial deviations we allow (usually, only small ones), whether we want the system to be stabilized for a potentially infinite amount of time or only for a given finite interval, etc.

Second property: smoothness. Stability is not all we expect from a control.

For example, when driving a car, stability means, in particular, that once the car swerved, it should return to the original trajectory. The faster it returns, the more stable is the system. Therefore, from the viewpoint of stability only, the ideal (optimal) control would be the one that brings the car back on track in the shortest possible time (i.e., with the largest possible acceleration). The resulting driving with sudden accelerations may be good on a racetrack or for a car chase, but it is very uncomfortable for passengers. From the passenger viewpoint, we prefer the resulting trajectory to be *smooth*.

The non-smoothness of the optimal control is not a peculiar feature of the car example: in control theory, there are general theorems that show that under certain (reasonably general) conditions, the optimal control is indeed of the above-described “bang-bang” type (see, e.g., [50]; not incidentally, the word “bang-bang” is an “official”, well-defined and widely used term in control theory).

Just like stability, smoothness is a generic notion; there are several different understandings (and formalizations) of what “smooth” means. In mathematical terms, *smoothness* is typically formalized as the existence of first, second, or higher order derivatives.

Third property: computational simplicity. Stability and smoothness are typical examples of the *idealized* goals. When, in mathematical control theory, we look for the optimal control strategy, we look for the optimal mathematical function, without taking into consideration how exactly we are going to implement this function.

In *real life*, however, the computational ability of the processor that actually computes the desired control is limited, so some very good control strategies may be too complicated for it. Moreover, in many control situations, we need the control *fast* (e.g., for a car control, if we spend too much time on the computation of the optimal control, the car may, by then, have already wrecked).

In principle, we can always replace the existing processor with a faster one, add extra memory, add an additional processor, etc., but this addition is *not* always *physically possible*:

- For example, in controlling a *space mission*, we are usually very much limited both in terms of the weight and the space required for a processor, and especially, in terms of the power feed; every increase in the computational ability of the controlling processor can only come at the (undesirable) expense of the decrease in the useful payload.

- In commercial applications, e.g., in controlling an *appliance* (which one of the main areas of application for fuzzy control), one of the major considerations is *cost*. Every increase in the computational ability of the processor increases the cost of the simple appliance, and this cost increase is only reasonable if it leads to even better savings in performance[1].

In view of all this, we would like our control to be *computationally simple*. (Computational simplicity is what sometimes makes engineers use fuzzy control even in the situations when the system is well defined, and the optimal control is known.)

These properties are not exactly consistent: a trade-off is needed. At first glance, it may seem that we should require *all three* properties of our control. However, as we have mentioned, these properties are not exactly consistent:

- the most stable control is often not smooth at all;

[1] We are greatly thankful to Piero Bonissone who attracted out attention to this example.

- the most stable and the smoothest controls can be computationally complicated.

Therefore, in real life, we must either choose one of these properties that we desire the most, or, even better, look for some trade-off between these three properties.

With this description in mind, let us describe which implementations of fuzzy methodology guarantee the smoothest, stablest, etc., control.

Choice of "and" and "or" operations. (These results are (mainly) summarized in [10, 40, 41, 53, 57, 67].)

- If we are looking for the *smoothest* control, then the best choice is to use $f_\&(a,b) = a \cdot b$ and $f_\vee(a,b) = \min(a,b)$ [40, 41, 67].

- If we are looking for the control that is *most robust* (i.e., least sensitive to the inaccuracy with which we measure the membership functions), then, depending on what exactly we are looking for, we can get two different results:

  - if we are looking for the control that is the most robust *in the the worst case*, then the best choice is to use $f_\&(a,b) = \min(a,b)$ and $f_\vee(a,b) = \max(a,b)$ [56, 59, 53, 57, 61];
  - if we are looking for the control that is the most robust *in the average*, then the best choice is to use $f_\&(a,b) = a \cdot b$ and $f_\vee(a,b) = a + b - a \cdot b$ [60, 53, 57, 61];
  - instead of minimizing the *average* error, we can try to minimize the corresponding *entropy* [63, 64, 65, 38, 32, 33]:

    * if we use the *average* entropy (in some reasonable sense), we get the same pair of optimal functions as for average error;
    * for an appropriately defined *worst-case* entropy the optimal operations are $f_\&(a,b) = \min(a,b)$ and $f_\vee(a,b) = a + b - a \cdot b$.

- If we are looking for the operations which are the *fastest to compute*, then the best choice is to use $f_\&(a,b) = \min(a,b)$ and $f_\vee(a,b) = \max(a,b)$ [45].

- Finally, if, in control applications, we are looking for the *most stable* control for a given system, then the best choice is to use $f_\&(a,b) = \min(a,b)$ and $f_\vee(a,b) = a + b - a \cdot b$ [40, 41, 67, 38].

Choice of defuzzification. In [40, 41, 38, 32], we show that the optimal defuzzification is given by the centroid formula.

Comment. These optimization results are in good accordance with the general group-theoretic approach that enables us to classify techniques that are optimal relative to arbitrary reasonable criteria [10, 40, 41, 55, 67].

6 Proofs

6.1 Proof of Theorem 2.2

Let us show that each membership function $\mu(x)$ from the desired appropriate m-dimensional family F can have, on each of its intervals of monotonicity I, at most $2m+6$ different values.

> Since the function $\mu(x)$ is monotonic on the interval I, it will then follow that it is piece-wise constant on this interval I. Then, since each function from F can have only finitely many intervals of monotonicity, it will follow that this function $\mu(x)$ is piece-wise constant on the entire real line, i.e., exactly what we want to prove.

We will prove this result by reduction to a contradiction. Indeed, let us assume that a function $\mu(x)$ from the appropriate family F attains $N > 2m+6$ different values $v_1 < \ldots < v_N$ on the interval I.

By the choice of I, the function $\mu(x)$ is either non-decreasing, or non-increasing on this interval. Without losing generality, we can assume that $\mu(x)$ is non-decreasing. This means, in particular, that if $\mu(x) = \mu(x')$ for some $x < x'$ ($x, x' \in I$), then $\mu(x'') = \mu(x)$ for all intermediate points $x'' \in (x, x')$ as well. Thus, for each value v, the set of all x for which $\mu(x) = v$ is *convex*, and is, therefore, a (finite or infinite) *interval* (open, closed, or semi-closed). In particular, we have such intervals I_j for all values v_j, $1 \le j \le N$. Let us denote, by u_j, the upper end of the interval I_j. We will call u_j the *right critical value corresponding to* v_j. All these critical values (except maybe the last one) are finite. Then:

- either $\mu(u_j) = v_j$ (if the interval I_j contains its upper endpoint),
- or $\mu(u_j - \varepsilon) = v_j$ for all sufficiently small $\varepsilon > 0$; we will denote this case as $\mu(u_j-) = v_j$.

For a non-decreasing function $\mu(x)$, it is possible that $u_j = u_{j+1}$ for some j: namely, it is possible when $\mu(u_j-) = v_j$ and $\mu(u_j) = v_{j+1}$. One can easily see that only two immediate neighbors u_j, u_{j+1} in the sequence $u_1, \ldots, u_N$ can coincide. If there are such coinciding pairs, we discard one element of each pair. We started with $N > 2m+6$ values; even in the worst case, when all elements belong to coinciding pairs, after we discard one element from each pair, we get at least $N/2 > (2m+6)/2 = m+3$ values. If the last value u_N is infinite, we discard it too, and get $> m+2$ *different* finite values u_j. Let us denote the number of these new values by $\widetilde{N}$, the values themselves by $\widetilde{u}_1, \ldots, \widetilde{u}_{\widetilde{N}}$, and the corresponding values v_j by $\widetilde{v}_j$. From non-decreasing character of μ, our choice of $\widetilde{u}_j$, and from the fact that $v_1 < v_2 < \ldots < v_N$, we conclude that $\widetilde{u}_1 < \widetilde{u}_2 < \ldots < \widetilde{u}_{\widetilde{N}}$ and $\widetilde{v}_1 < \widetilde{v}_2 < \ldots < \widetilde{v}_{\widetilde{N}}$.

By definition of appropriateness, since $\mu(x) \in F$, for every $\lambda > 0$ and s, the function $\mu(\lambda \cdot x + s)$ also belongs to the family F. Similarly, if we have several values $\lambda_1, \ldots, \lambda_{\widetilde{N}} > 0$ and $s_1, \ldots, s_{\widetilde{N}}$, then the function

$$\mu'(x) = \min(\mu(\lambda_1 \cdot x + s_1), \ldots, \mu(\lambda_{\widetilde{N}} \cdot x + s_{\widetilde{N}}))$$

also belongs to the family F. In particular, if all λ_j and s_j are chosen in such a way that $\lambda_j \cdot x + s_j \in I$ for all $x \in I$, then, since $\mu(x)$ is non-decreasing on I, we get a new membership function

$$\widetilde{\mu}(x) = \mu(t(x)),$$

where

$$t(x) = \min(\lambda_1 \cdot x + s_1, \ldots, \lambda_{\widetilde{N}} \cdot x + s_{\widetilde{N}}).$$

We will show that by choosing appropriate values of λ_j and s_j, we will get a $(\widetilde{N} - 1)$-parameteric family of functions within F. Since $\widetilde{N} > m + 2$, we get $\widetilde{N} - 1 > m$, and the existence of a $(\widetilde{N} - 1)$-dimensional family within the family F contradicts to our definition of m as the number of parameters in the family F.

Indeed, let us take $\lambda_1 = 1$ and $s_1 = 0$; then, $\lambda_1 \cdot x + s_1 = x$. Then, for every $j > 1$, we:

- select a small positive number δ_j;
- take $\lambda_j = \lambda_{j-1} - \delta_j$, and
- take such s_j that the values of the $(j-1)$-st and the j-th linear functions coincide at $\widetilde{u}_j$, i.e., for which the following equation holds: $\lambda_{j-1} \cdot \widetilde{u}_j + s_{j-1} = \lambda_j \cdot \widetilde{u}_j + s_j$; in other words, we take $s_j = \lambda_{j-1} \cdot \widetilde{u}_j + s_{j-1} - \lambda_j \cdot \widetilde{u}_j = s_{j-1} - \delta_j \cdot \widetilde{u}_j$.

If we choose small enough values of δ_j, then after all N iterations, we still have $\lambda_N > 0$. We have chosen the values in such a way that the j-th linear function is the smallest on the interval from $\widetilde{u}_{j-1}$ to $\widetilde{u}_j$. Thus, for the new function $\mu'(x)$, the critical values corresponding to $\widetilde{v}_1 < \ldots < \widetilde{v}_{\widetilde{N}}$ are the values $u'_1 < \ldots < u'_{\widetilde{N}}$, where $u'_j = \lambda_j \cdot \widetilde{u}_j + s_j$.

If we know these critical values u'_j, then we can reconstruct the values λ_j and s_j; indeed:

- for $j = 1$, we know that $\lambda_1 = 1$ and $s_1 = 0$; and
- for $j > 1$, we can reconstruct λ_j and s_j from the equations

  $$\lambda_j \cdot \widetilde{u}_{j-1} + s_j = u'_{j-1};$$

  $$\lambda_j \cdot \widetilde{u}_j + s_j = u'_j$$

  (with known u'_{j-1} and u'_j).

Therefore, from the critical values u'_j, we can uniquely reconstruct the values of the parameters $\delta_2, \ldots, \delta_{\widetilde{N}}$. Thus, different vectors $\vec{\delta} = (\delta_2, \ldots, \delta_{\widetilde{N}})$ correspond to different membership functions from the family F. Thus, the m-dimensional family F has a sub-family of dimension $\widetilde{N} - 1 > m$. The contradiction shows that our initial assumption (that a function $\mu(x)$ from the appropriate family F can attain $N > 2m + 6$ different values $v_1 < \ldots < v_N$ on the interval I) is false. Thus, any function from F can attain no more that $2m+6$ different values. The statement is proven, and so is the theorem.

6.2 Proof of Theorem 3.1

This proof of Theorem 3.1 is based on the following auxiliary result of independent interest:

Proposition 3.1. *If an optimality criterion is final and unit-invariant, then the optimal family F_{opt} is also unit-invariant, i.e., $R_\lambda(F_{opt}) = F_{opt}$ for every number λ.*

Proof of Proposition 3.1. Since the optimality criterion is final, there exists a unique family F_{opt} that is optimal with respect to this criterion, i.e., for every other F:

- either $F_{opt} \succ F$
- or $F_{opt} \sim F$.

To prove that $F_{opt} = R_\lambda(F_{opt})$, we will first show that the re-scaled family $R_\lambda(F_{opt})$ is also optimal, i.e., that for every family F:

- either $R_\lambda(F_{opt}) \succ F$
- or $R_\lambda(F_{opt}) \sim F$.

If we prove this optimality, then the desired equality will follow from the fact that our optimality criterion is final and therefore, there is only one optimal family (so, since the families F_{opt} and $R_\lambda(F_{opt})$ are both optimal, they must be the same family).

Let us show that $R_\lambda(F_{opt})$ is indeed optimal. How can we, e.g., prove that $R_\lambda(F_{opt}) \succ F$? Since the optimality criterion is unit-invariant, the desired relation is equivalent to $F_{opt} \succ R_{\lambda^{-1}}(F)$. Similarly, the relation $R_\lambda(F_{opt}) \sim F$ is equivalent to $F_{opt} \sim R_{\lambda^{-1}}(F)$.

These two equivalences allow us to complete the proof of the proposition. Indeed, since F_{opt} is optimal, we have one of the two possibilities:

- either $F_{opt} \succ R_{\lambda^{-1}}(F)$,
- or $F_{opt} \sim R_{\lambda^{-1}}(F)$.

In the first case, we have $R_\lambda(F_{opt}) \succ F$; in the second case, we have $R_\lambda(F_{opt}) \sim F$.

Thus, whatever family F we take, we always have either $R_\lambda(F_{opt}) \succ F$, or $R_\lambda(F_{opt}) \sim F$. Hence, $R_\lambda(F_{opt})$ is indeed optimal and thence, $R_\lambda(F_{opt}) = F_{opt}$. The proposition is proven.

A similar statement is true for rotation invariance. Similarly, we can prove that if an optimality criterion is final and rotation-invariant, then the optimal family F_{opt} is also rotation-invariant, i.e., $T(F_{opt}) = F_{opt}$ for every rotation T.

Conclusions: the optimal family is unit-invariant and rotation invariant. Let us now prove Theorem 3.1. Since the criterion is final, there exists an optimal family $F_{opt} = \{C \cdot f(\vec{x})\}$. Due to the Proposition, the optimal family is unit-invariant and rotation-invariant.

Using rotation invariance. In particular, since $f(\vec{x}) \in F_{\rm opt}$, rotation invariance means that for every rotation T, the function $f(T\vec{x})$ also belongs to the optimal family, i.e., that for every T, there exists a real number $C(T)$ such that

$$f(T\vec{x}) = C(T) \cdot f(\vec{x})$$

for all $\vec{x} \in R^s$.

Since the function $f(\vec{x})$ is assumed to be differentiable (hence, continuous), we can conclude that the ratio $C(T) = f(T\vec{x})/f(\vec{x})$ is a continuous function of T.

Let us consider a rotation T that is a composition of two other rotations $T = T_1 \circ T_2$. Then, the above equality takes the form

$$f(T_1 \circ T_2(x)) = C(T_1 \circ T_2) \cdot f(x).$$

On the other hand, we can apply a similar equality first for T_2, and then for T_1, thus getting

$$f(T_2\vec{x}) = C(T_2) \cdot f(\vec{x})$$

and

$$f(T_1 \circ T_2(\vec{x})) = f(T_1(T_2\vec{x})) = C(T_1) \cdot f(T_2\vec{x}) = C(T_1) \cdot C(T_2) \cdot f(\vec{x}).$$

Comparing the two formulas for $f(T_1 \circ T_2(\vec{x}))$, we conclude that $C(T_1 \circ T_2) = C(T_1) \cdot C(T_2)$.

Similarly, we can conclude that for every sequence of n rotations, we have:

$$C(T_1 \circ \ldots \circ T_n) = C(T_1) \cdot \ldots \cdot C(T_n).$$

In particular, if we take $T_1 = \ldots = T_{2n+1} =$ rotation by an angle $2\pi/(2n+1)$ around the same axis, we have an identity transformation id as $T_1 \circ \ldots \circ T_{2n+1}$ (for which $C(id) = 1$), and therefore, $C^{2n+1}(T_i) = 1$. Hence, $C(T_i) = 1$.

An arbitrary rotation by an angle $2\pi \cdot p/(2n+1)$, with integer p and n, can be represented as a composition of p rotations by an angle $2\pi/(2n+1)$. For each of these angles, as we have already shown, $C = 1$. Therefore, for their composition, we also have $C(T) = 1$.

Let us now show that $C(T) = 1$ for an arbitrary rotation T. Indeed, let α be this rotation's angle. The real number $\alpha/(2\pi)$ can be represented as a limit of rational numbers $p/(2n+1)$; therefore, the angle α is equal to the limit of angles $2\pi \cdot p/(2n+1)$, and hence, the rotation T can be represented as a limit of rotations T_k by angles $2\pi \cdot p/(2n+1)$. We already know that for all these rotations, $C(T_k) = 1$, and since the function $C(T)$ is continuous, we conclude that $C(T) = 1$ for an arbitrary rotation T, i.e., $f(T\vec{x}) = f(\vec{x})$.

Every two points $\vec{x}, \vec{x}'$ from R^s for which $|\vec{x}| = |\vec{x}'|$ can be transformed to each other by an appropriate rotation T around 0, i.e., $T\vec{x} = \vec{x}'$. Hence, $f(\vec{x}') = f(T\vec{x}) = f(\vec{x})$. Thus, the value $f(\vec{x})$ can depend only on the length $|\vec{x}|$ of the vector $\vec{x}$, i.e., $f(\vec{x}) = f_0(|\vec{x}|)$ for some function $f_0(r)$ of one real variable.

Using unit invariance. To determine the exact type of this dependence, let us use the *unit-invariance*. From unit-invariance, it follows that for every λ, there exists a real number $A(\lambda)$ for which $f(\lambda \cdot \vec{x}) = A(\lambda) \cdot f(\vec{x})$. Substituting the expression of $f(\vec{x})$ in terms of $f_0(r)$, we conclude that

$$f_0(\lambda \cdot r) = A(\lambda) \cdot f_0(r)$$

for every two positive real numbers $r, \lambda > 0$.

Since the function $f(\vec{x})$ is differentiable, we can conclude that the ratio $A(\lambda) = f(\lambda \cdot \vec{x})/f(\vec{x})$ is differentiable as well, and that the function $f_0(r)$ is also differentiable for $r > 0$. Thus, we can differentiate both sides of the above equation with respect to λ, and substitute $\lambda = 1$. As a result, we get the following differential equation for the unknown function $f_0(r)$:

$$r \cdot \frac{df_0}{dr} = \alpha \cdot f_0,$$

where by α, we denoted the value of the derivative $dA/d\lambda$ taken at $\lambda = 1$. Moving terms dr and r to the right-hand side and all the term containing f_0 to the left-hand side, we conclude that

$$\frac{df_0}{f_0} = \alpha \cdot \frac{dr}{r}.$$

Integrating both sides of this equation, we conclude that $\ln(f_0) = \alpha \cdot \ln(r) + C$ for some constant C, and therefore, that $f_0(r) = \text{const} \cdot r^\alpha$. Thus, $f(\vec{x}) = f_0(|\vec{x}|) = \text{const} \cdot |\vec{x}|^\alpha$. Theorem 3.1 is proven.

6.3 Proof of Theorem 4.1

According to Proposition 3.1, since an optimality criterion is final and unit-invariant, then the optimal family F_{opt} is also unit-invariant, i.e., $R_{\vec{\lambda}}(F_{opt}) =$

F_{opt} for every vector $\vec{\lambda}$.

Since the criterion is final, there exists an optimal family

$$F_{opt} = \{c_1 \cdot e_1(x_1, \ldots, x_n) + \ldots + c_m \cdot e_m(x_1, \ldots, x_n)\}.$$

Each of the corresponding functions $e_j(x_1, \ldots, x_n)$ belongs to the family F_{opt} (for $c_j = 1$ and $c_k = 0$ for $k \neq j$).

Due to Proposition 3.1, the optimal family is unit-invariant, i.e., $F_{opt} = R_{\vec{\lambda}}(F_{opt})$. In particular, this means that for very j, and for every $\vec{\lambda}$, we have $R_{\vec{\lambda}}(e_j) \in F_{opt}$, i.e.,

$$e_j(\lambda_1 \cdot x_1, \ldots, \lambda_n \cdot x_n) =$$
$$c_{j1}(\vec{\lambda}) \cdot e_1(x_1, \ldots, x_n) + \ldots + c_{jm}(\vec{\lambda}) \cdot e_m(x_1, \ldots, x_n). \tag{4.1}$$

for some values c_{jk}. If we take m different values of $(x_1, \ldots, x_n)$, then the corresponding equations (1) form a system of m linear equations to determine m coefficients $c_{j1}(\vec{\lambda}), \ldots, c_{jm}(\vec{\lambda})$. The well-known Cramer's rule describes the solution of a system of linear equation as a ratio of two determinants and thus, as a differentiable function of the coefficients and right-hand sides of these equations. Since $e_j(x_1, \ldots, x_n)$ are differentiable functions, we can thus conclude that the functions $c_j(\vec{\lambda})$ are differentiable too.

Since both sides of the equation (4.1) is differentiable, let us pick an arbitrary $l = 1, \ldots, n$, differentiate both sides with respect to λ_l, and then substitute $\lambda_l = \ldots = \lambda_n = 1$. As a result, we get the following system of differential equations:

$$x_l \cdot \frac{\partial e_j}{\partial x_l}(x_1, \ldots, x_n) = \sum_{k=1}^{m} c_{jkl} \cdot e_k(x_1, \ldots, x_n), \tag{4.2}$$

where we denoted

$$c_{jkl} = \frac{\partial c_{jk}(\lambda_1, \ldots, \lambda_n)}{\partial \lambda_l}_{|\lambda_1 = \ldots = \lambda_n = 1}.$$

This equation can be further simplified if we use new variables $X_j = \ln(x_j)$, for which $dx_l/x_l = dX_l$. In terms of these new variables, $x_j = \exp(X_j)$, and the values $e_j(x_1, \ldots, x_n)$ take the form $e_j(x_1, \ldots, x_n) = E_j(X_1, \ldots, X_n)$, where we denoted $E_j(X_1, \ldots, X_n) = e_j(\exp(X_1), \ldots, \exp(X_n))$. In terms of the new function $E_j(X_1, \ldots, X_n)$, the equation (4.2) takes the following form:

$$\frac{\partial E_j}{\partial X_l}(X_1, \ldots, X_n) = \sum_{k=1}^{m} c_{jkl} \cdot E_k(X_1, \ldots, X_n). \tag{4.3}$$

If we fix all the variables but one (e.g., except for X_1), we conclude that the functions $E_1(X_1), \ldots, E_m(X_1)$ satisfy a system of linear differential equations with constant coefficients. A general solution of such a system is well known (see, e.g., [1]): it has a form

$$E_j(X_1) = \sum C_{jp} \cdot \exp(\alpha_p \cdot X_1) \cdot X_1^{k_p}, \tag{4.4}$$

where α_p are complex numbers (eigenvalues of the coefficient matrix), C_p are complex numbers, and k_p are non-negative integers.

If we take into consideration the dependence on X_2, then all the coefficients of the formula (4.4) should depend on X_2, i.e.,

$$E_j(X_1, X_2) = \sum C_{jp}(X_2) \cdot \exp(\alpha_p(X_2) \cdot X_1) \cdot X_1^{k_p(X_2)}. \tag{4.5}$$

Since the dependence on X_2 is smooth (hence, continuous), and k_p is an integer, we conclude that k_p is a constant: $k_p(X_2) = k_p$. The dependence on all other coefficients on X_2 can be determined from the fact that, similarly to (4.4), for a fixed X_1, we must have a similar expression in terms of X_2:

$$E_j(X_2) = \sum C'_{jp} \cdot \exp(\alpha'_p \cdot X_2) \cdot X_2^{k'_p}. \tag{4.6}$$

Thus, the only possible dependence of C_{jp} on X_2 is a dependence of the type $\exp(\alpha'_p \cdot X_2) \cdot X_2^{k'_p}$, and the only possible dependence of α_p on X_2 is linear, i.e., we get

$$E_j(X_1, X_2) = \sum C_{jp} \cdot \exp(\alpha_{p1} \cdot X_1 + \alpha_{p2} \cdot X_2 + \alpha'_p \cdot X_1 \cdot X_2) \cdot X_1^{k_{p1}} \cdot X_2^{k_{p2}}. \tag{4.7}$$

We started with the system (4.3). This system remains similar if we make a linear change of variables, e.g., if we replace X_1 and X_2 by $X'_1 = X_1 + X_2$ and $X'_2 = X_1 - X_2$. Therefore, we would like to get a similar formula (4.7) in the new variables. If $\alpha'_p \neq 0$, we get the undesired quadratic term in the exponential expression. Thus, $\alpha'_p = 0$, and (4.7) take the form

$$E_j(X_1, X_2) = \sum C_{jp} \cdot \exp(\alpha_{p1} \cdot X_1 + \alpha_{p2} \cdot X_2) \cdot X_1^{k_{p1}} \cdot X_2^{k_{p2}}. \tag{4.7}$$

Similarly, if we take into consideration the dependence on all n variables X_k, we conclude that

$$E_j(X_1, \ldots, X_n) = \sum C_{jp} \cdot \exp(\alpha_{p1} \cdot X_1 + \ldots + \alpha_{pn} \cdot X_n) \cdot X_1^{k_{p1}} \cdot \ldots \cdot X_n^{k_{pn}}. \tag{4.8}$$

Substituting $X_k = \ln(x_k)$ into this formula (4.8), we get the desired expression for $e_j(x_1, \ldots, x_n)$:

$$e_j(x_1, \ldots, x_n) = \sum C_{jp} \cdot x_1^{\alpha_{p1}} \cdot \ldots \cdot x_n^{\alpha_{pn}} \cdot \ln^{k_{p1}}(x_1) \cdot \ldots \cdot \ln^{k_{pn}}(x_n).$$

The theorem is proven.

6.4 Proof of Theorem 4.2

This proof is very similar to the proof of Theorem 4.1, except that in addition to *regular* solutions of the corresponding differential equations, we have to also consider *generalized* solutions. Methods of describing such solutions (including an explicit description of all unit-invariant generalized functions) are given in [21, 69]; by using these methods, we get the desired result. Basically, on each orthant, we get a formula which is described by a separate expression from Theorem 4.1, and we also get separate expressions for $x_i = 0$.

Acknowledgments. This work was supported in part by NASA under cooperative agreement NCC5-209, by NSF grant No. DUE-9750858, by United Space Alliance under grant No. NAS 9-20000 (PWO C0C67713A6), by Future Aerospace Science and Technology Program (FAST) Center for Structural Integrity of Aerospace Systems, effort sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-95-1-0518, by the National Security Agency under Grant No. MDA904-98-1-0564, and by a donation of Shell Oil Co.

The author is greatly thankful:

- to the conference organizers, especially to Basil Papadopoulos and Apostolos Syropoulos, for their great efforts;
- to Hung T. Nguyen, Brian Penn, Leonid K. Reznik, and Scott A. Starks, for helpful collaboration, and
- to Donald Michie for valuable discussions and learning insights.

References

[1] R. E. Bellman. *Introduction to Matrix Analysis*, McGraw-Hill, New York, 1970.

[2] L. v. Bertalanffy, *Perspectives on general system theory*, G. Braziller, N.Y., 1975.

[3] L. v. Bertalanffy, *General system theory*, G. Braziller, N.Y., 1984.

[4] J. C. Bezdek, "Numerical taxonomy with fuzzy sets", *Journal of Mathematical Biology*, 1974, Vol. 1, pp. 57–71.

[5] J. C. Bezdek, "Cluster validity with fuzzy sets", *Journal of Cybernetics*, 1973, Vol. 3, No. 3, pp. 58–71.

[6] J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*, Plenum, NY, 1981.

[7] J. C. Bezdek, R. Hathaway, M. Sabin, and W. Tucker, "Convergence theory for fuzzy C-Means: counterexample and repairs", *IEEE Trans. Systems, Man, and Cybernetics*, 1987, Vol. SMC-17, pp. 873–877.

[8] J. C. Bezdek, R. Hathaway, M. Sabin, and W. Tucker, "Convergence theory for fuzzy C-Means: counterexample and repairs", In: J. Bezdek (ed.), *The Analysis of Fuzzy Information*, CRC Press, 1987, Vol. 3, Chapter 8.

[9] J. C. Bezdek and S. K. Pal (eds.) *Fuzzy models for pattern recognition*, IEEE Press, N.Y., 1992.

[10] B. Bouchon-Meunier, V. Kreinovich, A. Lokshin, and H. T. Nguyen, "On the formulation of optimization under elastic constraints (with control in mind)", *Fuzzy Sets and Systems,* 1996, Vol. 81, No. 1, pp. 5–29.

[11] S. Chiu, "Fuzzy model identification based on cluster estimation", *J. of Intelligent and Fuzzy Systems*, 1994, Vol. 2, No. 3, pp. 267–278.

[12] S. Chiu, "Selecting input variables for fuzzy models", *J. of Intelligent and Fuzzy Systems*, 1996, Vol. 4, pp. 243–256.

[13] J. A. Dickerson and B. Kosko, "Fuzzy function learning with covariant ellipsoids", *Proceedings of the IEEE International Conference on Neural Networks*, San Francisco, March 28–April 1, 1993, Vol. 3, pp. 1162–1167.

[14] J. A. Dickerson and B. Kosko, "Fuzzy function approximation with supervised ellipsoidal learning", *Proceedings of the World Congress on Neural Networks*, Portland, Oregon, July 11–15, 1993, Vol. 2, pp. 9–17.

[15] J. A. Dickerson and B. Kosko, "Fuzzy function approximation with ellipsoidal rules", *IEEE Trans. Syst., Man, Cybern.*, 1996, Vol. 26, No. 4, pp. 542–560.

[16] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters", *Journal of Cybernetics*, 1973, Vol. 3, pp. 32–57.

[17] A. Finkelstein, O. Kosheleva, and V. Kreinovich, "Astrogeometry: towards mathematical foundations", *International Journal of Theoretical Physics*, 1997, Vol. 36, No. 4, pp. 1009–1020.

[18] A. M. Finkelstein, V. Kreinovich, and R. R. Zapatrin, "Fundamental physical equations uniquely determined by their symmetry groups," *Lecture Notes in Mathematics*, Springer-Verlag, Berlin-Heidelberg-N.Y., Vol. 1214, 1986, pp. 159–170.

[19] K. Fukunaga, *Introduction to statistical pattern recognition*, Academic Press, San Diego, CA, 1990.

[20] *Galactic and extra-galactic radio astronomy*, Springer-Verlag, NY, 1974.

[21] I. M. Gelfand and G. E. Shilov, *Generalized functions*, Vol. 1, Academic Press, N. Y. and London, 1964.

[22] A. Kandel, *Fuzzy techniques in pattern recognition*, Wiley-Interscience, NY, 1982.

[23] H. M. Kim and B. Kosko, "Fuzzy prediction and filtering in impulsive noise", *Fuzzy Sets and Systems*, 1996, Vol. 77, pp. 15–33.

[24] G. Klir and B. Yuan, *Fuzzy sets and fuzzy logic: theory and applications,* Prentice Hall, Upper Saddle River, NJ, 1995.

[25] O. Kosheleva, "Why Sinc? Signal Processing Helps Fuzzy Control", *SC-COSMIC, South Central Computational Sciences in Minority Institutions Consortium, Second Student Conference in Computational Sciences*, October 25–27, 1996, El Paso, TX, Abstracts, pp. 19–21.

[26] O. Kosheleva, S. D. Cabrera, G. A. Gibson, and M. Koshelev, "Fast Implementations of Fuzzy Arithmetic Operations Using Fast Fourier Transform (FFT)", *Proceedings of the 1996 IEEE International Conference on Fuzzy Systems*, New Orleans, September 8–11, 1996, Vol. 3, pp. 1958–1964.

[27] O. Kosheleva, S. D. Cabrera, G. A. Gibson, and M. Koshelev, "Fast Implementations of Fuzzy Arithmetic Operations Using Fast Fourier Transform (FFT)", *Fuzzy Sets and Systems*, 1997, Vol. 91, No. 2, pp. 269–277.

[28] B. Kosko, "Fuzzy Systems as Universal Approximators", *IEEE Trans. on Computers,* 1994, Vol. 43, No. 11, pp. 1329–1333.

[29] B. Kosko, "Optimal fuzzy rules cover extrema", *Proceedings of the World Congress on Neural Networks WCNN'94*, 1994.

[30] B. Kosko, "Optimal fuzzy rules cover extrema", *International Journal of Intelligent Systems*, 1995, Vol. 10, No. 2, pp. 249–255.

[31] B. Kosko, "Additive fuzzy systems: from function approximation to learning", In: C. H. Chen (ed.), *Fuzzy Logic and Neural Network Handbook*, McGraw-Hill, N.Y., 1996, pp. 9-1–9-22.

[32] V. Kreinovich, "Maximum entropy and interval computations", *Reliable Computing*, 1996, Vol. 2, No. 1, pp. 63–79.

[33] V. Kreinovich, "Random sets unify, explain, and aid known uncertainty methods in expert systems", in J. Goutsias, R. P. S. Mahler, and H. T. Nguyen (eds.), *Random Sets: Theory and Applications*, Springer-Verlag, N.Y., 1997, pp. 321–345.

[34] V. Kreinovich, Ching-Chuang Chang, L. Reznik, and G. N. Solopchenko. "Inverse problems: fuzzy representation of uncertainty generates a regularization", *Proceedings of NAFIPS'92: North American Fuzzy Information Processing Society Conference, Puerto Vallarta, Mexico, December 15–17, 1992*, NASA Johnson Space Center, Houston, TX, 1992, pp. 418–426.

[35] V. Kreinovich, G. C. Mouzouris, and H. T. Nguyen, "Fuzzy rule based modeling as a universal control tool", In: H. T. Nguyen and M. Sugeno (eds.), *Fuzzy Systems: Modeling and Control*, Kluwer, Boston, MA, 1998, pp. 135–195.

[36] V. Kreinovich, H. T. Nguyen, S. A. Starks, and Y. Yam, "Decision making based on satellite images: optimal fuzzy clustering approach", *Proceedings of the 37th IEEE Conference on Decision and Control CDC'98*, Tampa, Florida, December 16–18, 1998 (to appear).

[37] V. Kreinovich, H. T. Nguyen, and Y. Yam, "Optimal Choices of Potential Functions in Fuzzy Clustering", *The Chinese University of Hong Kong, Department of Mechanical & Automation Engineering*, Technical Report CUHK-MAE-98-001, January 1998.

[38] V. Kreinovich, H. T. Nguyen, and E. A. Walker, Maximum entropy (MaxEnt) method in expert systems and intelligent control: new possibilities and limitations, In: Hanson K. M. and Silver, R. N. (eds.), *Maximum Entropy and Bayesian Methods*, Kluwer Academic Publishers, Dordrecht, 1996, pp. 93–100.

[39] V. Kreinovich and C. Quintana, "Neural networks: what non-linearity to choose?," *Proceedings of the 4th University of New Brunswick Artificial Intelligence Workshop*, Fredericton, N.B., Canada, 1991, pp. 627–637.

[40] V. Kreinovich, C. Quintana, and R. Lea, "What procedure to choose while designing a fuzzy control? Towards mathematical foundations of fuzzy control", *Working Notes of the 1st International Workshop on Industrial Applications of Fuzzy Control and Intelligent Systems*, College Station, TX, 1991, pp. 123–130.

[41] V. Kreinovich, C. Quintana, R. Lea, O. Fuentes, A. Lokshin, S. Kumar, I. Boricheva, and L. Reznik. "What non-linearity to choose? Mathematical foundations of fuzzy control", *Proceedings of the 1992 International Conference on Fuzzy Systems and Intelligent Control*, Louisville, KY, 1992, pp. 349–412.

[42] V. Kreinovich, C. Quintana, and L. Reznik, "Gaussian membership functions are most adequate in representing uncertainty in measurements".

Proceedings of NAFIPS'92: North American Fuzzy Information Processing Society Conference, Puerto Vallarta, Mexico, December 15–17, 1992, NASA Johnson Space Center, Houston, TX, 1992, pp. 618–625.

[43] V. Kreinovich and L. K. Reznik. "Methods and models of formalizing a priori information (on the example of processing measurements results)," *Analysis and Formalization of Computer Experiments, Proceedings of the Mendeleev Metrology Institute*, 1986, pp. 37–41 (in Russian).

[44] V. Kreinovich, O. Sirisaengtaksin, and H. T. Nguyen, "Sigmoid neurons are the safest against additive errors", *Proceedings of the First International Conference on Neural, Parallel, and Scientific Computations*, Atlanta, GA, May 28–31, 1995, Vol. 1, pp. 419–423.

[45] V. Kreinovich and D. Tolbert, "Minimizing computational complexity as a criterion for choosing fuzzy rules and neural activation functions in intelligent control", In: M. Jamshidi, C. Nguyen, R. Lumia, and J. Yuh (eds.), *Intelligent Automation and Soft Computing. Trends in Research, Development, and Applications. Proceedings of the First World Automation Congress (WAC'94). August 14–17, 1994, Maui, Hawaii*, TSI Press, Albuquerque, NM, Vol. 1, pp. 545–550.

[46] D. Michie, "Learning concepts from data", *Expert Systems with Applications*, 1998 (to appear).

[47] D. Michie, M. Bain, and J. Hayes-Michie, "Cognitive models from subcognitive skills", In: J. McGee, M. J. Grimble, and O. Mowforth (eds.), *Knowledge systems for industrial control*, Peter Peregrinus, London, 1990, pp. 71–99.

[48] S. Mitaim and B. Kosko, "What is the best shape for a fuzzy set in function approximation?", *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems FUZZ-IEEE'96*, New Orleans, September 8–11, 1996, Vol. 2, pp. 1237–1243.

[49] S. Mitaim and B. Kosko, "Fuzzy function approximation and the shape of fuzzy sets", *Proceedings of the World Congress on Neural Networks WCNN'96*, September 1996.

[50] R. R. Mohler, *Nonlinear systems. Vol. 1. Dynamics and control*, Prentice Hall, Englewood Cliff, NJ, 1991.

[51] S. Muggleton and D. Michie, "Machine intelligibility and the duality principle", *BT Technol. J.*, 1996, Vol. 14, No. 4, pp.

[52] H. T. Nguyen, M. Koshelev, O. Kosheleva, V. Kreinovich, and R. Mesiar, "Computational Complexity and Feasibility of Fuzzy Data Processing:

Why Fuzzy Numbers, Which Fuzzy Numbers, Which Operations with Fuzzy Numbers", *Proceedings of the International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'98)*, Paris, France, July 6–10, 1998.

[53] H. T. Nguyen and V. Kreinovich, "Towards theoretical foundations of soft computing applications", *International Journal on Uncertainty, Fuzziness, and Knowledge-Based Systems*, 1995, Vol. 3, No. 3, pp. 341–373.

[54] H. T. Nguyen and V. Kreinovich, "Nested Intervals and Sets: Concepts, Relations to Fuzzy Sets, and Applications", In: R. B. Kearfott et al (eds.), *Applications of Interval Computations*, Kluwer, Dordrecht, 1996, pp. 245–290.

[55] H. T. Nguyen and V. Kreinovich, *Applications of continuous mathematics to computer science*, Kluwer, Dordrecht, 1997.

[56] H. T. Nguyen, V. Kreinovich, B. Lea, and D. Tolbert, "How to control if even experts are not sure: robust fuzzy control", *Proceedings of the Second International Workshop on Industrial Applications of Fuzzy Control and Intelligent Systems*, College Station, December 2–4, 1992, pp. 153–162.

[57] H. T. Nguyen, V. Kreinovich, B. Lea, and D. Tolbert, "Interpolation that leads to the narrowest intervals, and its application to expert systems and intelligent control", *Reliable Computing*, 1995, Vol. 3, No. 1, pp. 299–316.

[58] H. T. Nguyen, V. Kreinovich, and O. Sirisaengtaksin, "Fuzzy control as a universal control tool", *Fuzzy Sets and Systems*, 1996, Vol. 80, No. 1, pp. 71–86.

[59] H. T. Nguyen, V. Kreinovich, and D. Tolbert, "On robustness of fuzzy logics", *Proceedings of IEEE-FUZZ International Conference*, San Francisco, CA, March 1993, Vol. 1, pp. 543–547.

[60] H. T. Nguyen, V. Kreinovich, and D. Tolbert, "A measure of average sensitivity for fuzzy logics", *International Journal on Uncertainty, Fuzziness, and Knowledge-Based Systems*, 1994, Vol. 2, No. 4, pp. 361–375.

[61] H. T. Nguyen and E. A. Walker, *A first course in fuzzy logic*, CRC Press, Boca Raton, FL, 1997.

[62] A. G. Nobile, L. M. Riccardi, L. Sacerdote, "On a class of difference equations modeling growth processes", In: L. M. Riccardi and A. C. Scott, *Biomathematics in 1980*, North-Holland, Amsterdam, 1982, pp. 217–244.

[63] A. Ramer and V. Kreinovich, "Maximum entropy approach to fuzzy control", *Proceedings of the Second International Workshop on Industrial Applications of Fuzzy Control and Intelligent Systems*, College Station, December 2–4, 1992, pp. 113–117.

[64] A. Ramer and V. Kreinovich, "Information complexity and fuzzy control", Chapter 4 in: A. Kandel and G. Langholtz (eds.), *Fuzzy Control Systems*, CRC Press, Boca Raton, FL, 1994, pp. 75–97.

[65] A. Ramer and V. Kreinovich, "Maximum entropy approach to fuzzy control", *Information Sciences*, 1994, Vol. 81, No. 3–4, pp. 235–260.

[66] C. Sammut, S. Hurst, D. Kedzier, and D. Michie, "Learning to fly", In: D. H. Sleeman (ed.), *Proc. of the 9th International Machine Learning Conference ML'92*, Morgan Kaufmann, San Mateo, CA, 1992.

[67] M. H. Smith and V. Kreinovich, "Optimal strategy of switching reasoning methods in fuzzy control", Chapter 6 in H. T. Nguyen, M. Sugeno, R. Tong, and R. Yager (eds.), *Theoretical aspects of fuzzy control*, J. Wiley, N.Y., 1995, pp. 117–146.

[68] *Symmetries in physics*, Springer-Verlag, Berlin, N.Y., 1992.

[69] V. S. Vladimirov, *Equations of Mathematical Physics,* Marcel Dekker, N.Y., 1971.

[70] L.-X. Wang, "Fuzzy Systems Are Universal Approximators", *Proceedings of Second IEEE International Conference on Fuzzy Systems,* San Diego, CA, March 1992, pp. 1163–1170.

[71] L.-X. Wang, *Adaptive Fuzzy Systems and Control,* Prentice-Hall, Englewood-Cliffs, NJ, 1994.

[72] L.-X. Wang and J. M. Mendel, *Generating fuzzy rules from numerical data with applications*, University of Southern California, Signal and Image Processing Institute, Technical Report USC-SIPI # 169, 1991.

[73] L.-X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning", *IEEE Transactions on Neural Networks*, 1992, Vol. 3, pp. 807–814.

[74] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples", *IEEE5 Transactions on Systems, Man, and Cybernetics*, 1992, Vol. 22, pp. 1414–1417.

[75] R. R. Yager and D. P. Filev, "Approximate clustering via the mountain method", *IEEE Trans. Systems, Man and Cybernetics*, 1994, Vol. 24, pp. 1279–1284.

[76] R. R. Yager and D. P. Filev, "Generation of fuzzy rules by mountain clustering", *Journal of Intelligent and Fuzzy Systems*, 1994, Vol. 2, No. 3, pp. 209–219.

[77] Y. Yam, "Extraction of sparse rule base by Cartesian representation and clustering", *These Proceedings.*

[78] X.-J. Zeng and M. G. Singh, "Approximation accuracy analysis of fuzzy systems as function approximators," *IEEE Trans. on Fuzzy Systems,* 1996, Vol. 4, pp. 44–63.