On Fusion of Soft and Hard Computing: Traditional ("Hard Computing") Optimal Rescaling Techniques Simplify Fuzzy Control

Hugh F. VanLandingham, Vladik Kreinovich

Abstract— **One of the main objectives of fuzzy control is to translate expert rules – formulated in imprecise ("fuzzy") words from natural language – into a precise control strategy. This translation is usually done is two steps. First, we apply a fuzzy control methodology to get a rough *approximation* to the expert's control strategy, and then we *tune* the resulting fuzzy control system. The first step (getting a rough approximation) is well-analyzed, and the fact that we have expert's intuitive understanding enables us to use soft computing techniques to perform this step. The second (tuning) step is much more difficult: we no longer have any expert understanding of which tuning is better, and therefore, soft computing techniques are not that helpful. In this paper, we show that we can formulate an important particular case of the tuning problem as a traditional optimization problem and solve it by using traditional ("hard computing") techniques. We show, on a practical industrial control example, that the resulting fusion of soft computing (for a rough approximation) and a hard computing (for tuning) leads to a high quality control.**

Keywords— **Fuzzy control, tuning, rescaling, optimal rescaling.**

I. Introduction

A. Fuzzy Control: One of the Most Successful Soft Computing Techniques

In most industrial applications, we want to control the corresponding industrial processes in such a way as to maximize the output within certain (physical and economical) restrictions. When the corresponding mathematical description is linear, we can use well-known optimal control techniques to find the optimal control strategy. In reality, however, most industrial processes are non-linear. For non-linear control problems, the situation is much more complicated: there are good recipes which often work but, alas, there is still no general methods of generating an optimal (or even a reasonably good) control; see, e.g., [7]. (For a formal proof that the corresponding optimization problems are computationally difficult (NP-hard), see, e.g., [4] and references therein.)

If for a certain industrial process, no known technique leads to a good quality control, what can we do? Usually, the very fact that this process is actually used in industry means that this process is reasonably well controlled by human controllers. Therefore, if we want to automate this control, we must somehow transform the knowledge of these expert controllers (operators) into an automatic control strategy.

The necessity for such a transformation was one of the main motivations behind one of the most successful soft computing techniques – fuzzy control. Specifically, our goal is to describe a function which takes the sensor inputs $x_1, \ldots, x_n$ (numbers) and generates the (numerical) value of the control effort u. Unfortunately, expert operators cannot formulate their expertise in these terms. Instead, they describe their control strategy by using uncertain ("fuzzy") statements of the type "if the obstacle is straight ahead, the distance to it is small, and the velocity of the car is medium, press the brakes hard". Fuzzy control is a methodology which translates such statements into precise formulas for control. Fuzzy control was started by L. Zadeh and E. H. Mamdani [2], [6], [17], [18] in the framework of Zadeh's *fuzzy set theory* [16]. For the current state of fuzzy control the reader is referred, e.g., to the volume [11] (or to [12]).

B. Tuning Is Necessary

Fuzzy control methodology usually consists of two steps (see, e.g., [10]:

- first, we apply a routine fuzzy control methodology to get a rough *approximation* to the expert's control strategy;
- the resulting control is usually not that good, so we have to *tune* the resulting fuzzy control system.

The first step usually starts with assigning membership functions to all the terms that the expert uses in his rules (in our sample phrase these words are "small", "medium", and "hard"). Most software packages for fuzzy control are based on (usually triangular) membership functions whose domains have equally spaced endpoints. For example, we can fix a neutral value N (usually, $N = 0$), and a number Δ, and take:

- "negligible" with the domain $[N - \Delta, N + \Delta]$;
- "small positive" with the domain $[N, N + 2\Delta]$;
- "medium positive" with the domain $[N + \Delta, N + 3\Delta]$, etc.

Correspondingly:

- "small negative" has the domain $[N - 2\Delta, N]$;
- "medium negative" has the domain $[N - 3\Delta, N - \Delta]$, etc.

Once an interval $[a - \Delta, a + \Delta]$ is given, then we can take a triangular membership function $\mu(x)$ which:

- is equal to 0 outside this interval;
- is equal to 1 for $x = a$, and
- is linear on each of the intervals $[a - \Delta]$ and $[a, a + \Delta]$.

H. F. VanLandingham is with The Bradley Department of Electrical Engineering, Virginia Polytechnic Institute and State University, Blacksburg VA 24061, USA, email hughv@vt.edu.

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

The resulting control is often not perfect, so a further tuning is necessary.

C. Usually, Soft Computing Techniques Are Used For Tuning, But This May Not Be The Best Idea

Usually, soft computing techniques such as neural networks or genetic algorithms are used for tuning fuzzy control. This is done mainly by tuning the corresponding membership functions. The results are usually reasonable, but this tuning often takes lots of time; for example, several thousands iterations are typical for neural networks.

How come soft computing techniques are so good for getting a rough approximation, but these same techniques are not so good for improving (tuning) this approximation? The explanation is very simple:

On the first step (getting a rough approximation), the fact that we have an expert's intuitive understanding enables us to use soft computing techniques to perform this step.

On the second (tuning) step, we no longer have any expert understanding of which tuning is better; as a result, soft computing techniques are not that helpful.

D. Natural Idea: Let's Use Hard Computing for Tuning

Since soft computing techniques do not work that well for tuning, we propose to supplement them with more traditional ("hard computing") optimization techniques. In this paper, we show that we can formulate an important particular case of the tuning problem as a traditional optimization problem and solve it by using traditional ("hard computing") techniques. We also show, on a practical industrial control example, that the resulting fusion of

- soft computing (for a rough approximation) and
- hard computing (for tuning)

does lead to a high quality control.

Comment. Preliminary results of our research first appeared in [15]

II. Rescaling: An Important Particular Case of Tuning

A. Rescaling: Physical Motivations

In some cases, there are physical reasons why the use of membership functions with equally spaced domains does not work well. For example, if the control variable u is always positive (e.g., if we control the flow of some substance into a reactor), then negative values (that will be eventually generated by an equal spacing method) simply make no sense.

A natural idea is to choose another scale $\widetilde{u} = f(u)$ to represent the control variable u, so that equal spacing will work fine for $\widetilde{u}$. This idea is in good accordance with our common-sense description of physical processes; let us give a few examples.

From the physical viewpoint it is quite possible to describe the strength of an *earthquake* by its energy, but, when we talk about its consequences, it is much more convenient to use a logarithmic scale (called *Richter scale*).

Non-linear scales are used to describe amplifiers and noise (decibels, in electrical engineering).

A non-linear scale is used to describe hardness of minerals in geosciences, etc.

For a general survey of different scales and rescalings, see [13].

In our case, we want to design such a scale that for $f(u)$ the equally spaced endpoints $N - k \cdot \Delta$ and $N + k \cdot \Delta$ would make sense for all integers k. Therefore, we are looking for a function $f(u)$, whose domain is the set of all positive values, and whose range is the set of all possible real numbers. In mathematical notations, f must map $(0, \infty)$ onto $(-\infty, \infty)$. There are lots of such functions, and evidently not all of them will improve the control. So we arrive at the following problem:

B. The Main Problem: Informally

Which rescaling $f : (0, \infty) \to (-\infty, \infty)$ should we choose?

C. What We Are Planning To Do

In this paper, we do the following:

first, we formulate the problem of choosing the best rescaling function $f(u)$ as a mathematical optimization problem;

then, we solve this optimization problem under some reasonable optimality criteria; as a result, we get an optimal function $f(u)$;

finally, we show that the use of this optimal re-scaling function really improves fuzzy control.

III. Towards the Use of Hard Computing: Motivations of the Proposed Formal Description of the Problem

A. Why Is This Problem Difficult?

We want to find a scaling function $f(u)$ that is the best in some reasonable sense. In other words, we want to find a scaling function for which some characteristic I attains the value that corresponds to the best performance of the resulting fuzzy control.

As examples of such characteristics, we can take:

- an average running time of the algorithm,
- smoothness of the resulitng control;
- stability of the resulting control, etc.

A seemingly natural approach is to describe this characteristic in precise terms and solve the corresponding optimization problem. Alas, life is not so simple. The problem is that even for the simplest linear plants (controlled systems), we do not know how to compute any of these possible characteristics for a give rescaling $f(u)$. How can we find $f(u)$ for which $I(f(u))$ is optimal if we cannot compute $I(f(u))$ even for a single function $f(u)$? There does not seem to be a likely answer.

However, we will show that this problem is solvable (and give the solution).

Comment. To solve this problem, we use a general idea described in the book [9]; this book also contains applica-

tions of similar optimization methods to other soft computing techniques such as fuzzy logic, neural networks, genetic algorithms, etc.

B. Some Rescalings Preserve Equal Spacing

Let us first show that not all physically meaningful rescalings help.

Indeed, in order to get numerical values of the variable u (e.g., of the spatial coordinate x), we must fix a starting point (origin) and a measuring unit (e.g., meter). In principle, we could as well choose feet to describe length.

If we change the unit, then some things change, e.g., the *numerical values* of all the coordinates change: x meters are equal to $\lambda \cdot x$ feet, where λ is the number of feet in 1 meter.

On the other hand, some things do not change: e.g., when we change the measuring unit, equally spaced intervals remain equally spaced.

Similarly, we could choose a different initial point for measuring the x coordinate. If we take, as a new initial point, a point which previously had a coordinate x_0 (so that now its coordinate is 0), then, similarly, on one hand, the numerical values of the points' coordinates change from x to $x - x_0$; on the other hand, intervals that had equal length in the old scale (x) will still have equal length if we measure then in the new scale ($x - x_0$).

We can also change *both* the measuring unit and the starting point. This way we arrive at a transformation $x \to \lambda \cdot x + x_0$.

Summarizing: if x is a reasonable scale – in the sense that equally spaced membership functions lead to a reasonably good control – then the same is true for an arbitrary scale of the type $\lambda \cdot x + x_0$, where $\lambda > 0$, and x_0 is a real number. The reason is that if we have a sequence of equally spaced intervals $[N + k \cdot \Delta, N + (k+1) \cdot \Delta]$, then these intervals will remain equally spaces after these linear rescalings $x \to \lambda \cdot x + x_0$: namely, these intervals will turn into intervals $[\widetilde{N} + k \cdot \widetilde{\Delta}, \widetilde{N} + (k+1) \cdot \widetilde{\Delta}]$, where $\widetilde{N} = \lambda \cdot N + x_0$ and $\widetilde{\Delta} = \lambda \Delta$.

C. We Must Choose a Family of Scaling Functions, Not a Single Function

Let us now consider a scale u, for which equal spacing does not work. Assume that $u \to f(u)$ is a transformation after which equal spacing becomes applicable. This means that if we use $f(u)$ as a new scale, then equal spacings work fine. But as we have just shown, for any $\lambda > 0$ and x_0 equal spacing will also work fine for the scale $\lambda \cdot f(u) + x_0$.

Therefore, if $f(u)$ is a function that transforms the initial scale into a scale, for which equal spacing works fine, then for every $\lambda > 0$ and x_0 the function $\widetilde{f}(u) = \lambda \cdot f(u) + x_0$ has the same desired property.

This means that there is no way to pick one function $f(u)$, because with any function $f(u)$, the whole family of functions $\lambda \cdot f(u) + x_0$ has the same property. Therefore, desired functions form a *family* $\{\lambda \cdot f(u) + x_0\}_{\lambda > 0, x_0}$. Hence, instead of choosing a single function, we must formulate a problem of choosing a family.

D. Which Family Is the Best?

Among all such families, we want to choose the best one. In formalizing what "the best" means, we follow the general idea described in [9]. The criteria to choose may be computational simplicity, stability or smoothness of the resulting control, etc.

In mathematical optimization problems, numerical criteria are most frequently used, where to every family we assign some value expressing its performance, and choose a family for which this value is maximal.

However, it is not necessary to restrict ourselves to such numeric criteria only. For example, if we have several different families that lead to the same average stability characteristics T, we can choose between them the one that leads to the maximal smoothness characteristics P. In this case, the actual criterion that we use to compare two families is not numerical, but more complicated. For example, we may say that a family Φ_1 is better than the family Φ_2 if and only if either $T(\Phi_1) < T(\Phi_2)$, or $T(\Phi_1) = T(\Phi_2)$ and $P(\Phi_1) < P(\Phi_2)$.

A criterion can be even more complicated. What a criterion *must* do is to allow us for every pair of families to tell whether the first family is better with respect to this criterion (we'll denote it by $\Phi_2 < \Phi_1$), or the second is better ($\Phi_1 < \Phi_2$) or these families have the same quality in the sense of this criterion (we'll denote it by $\Phi_1 \sim \Phi_2$).

E. The Criterion for Choosing the Best Family Must Be Consistent

Of course, it is necessary to demand that these choices be consistent: e.g., if $\Phi_1 < \Phi_2$ and $\Phi_2 < \Phi_3$ then $\Phi_1 < \Phi_3$.

F. The Criterion Must Be Final

Another natural demand is that this criterion must be *final* in the sense that it 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 a problem choosing the absolute "best" family. Therefore, we need some additional criterion for that choice.

For example, if several families turn out to have the same stability characteristics, we can choose among them a family with maximal smoothness. 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: Φ_1 is better than Φ_2 according to this new criterion if either it was better according to the old criterion, or according to the old criterion they had the same quality, and Φ_1 is better than Φ_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

ultimate; we have to modify it until we arrive at a final criterion that will have that property.

G. The Criterion Must Be Reasonably Invariant

We have already discussed the effect of changing units in a new scale $f(u)$. But it is also possible to change units in the original scale, in which the control u is described. If we use a unit that is c times smaller, then a control whose numeric value in the original scale was u, will now have the numeric value cu. For example, if we initially measured the flux of a substance (e.g., rocket fuel) into the reactor by kg/sec, we can now switch to lb/sec.

Comment. There is no physical sense in changing the starting point for u, because we consider the control variable that takes only positive values, and so 0 is a fixed value, corresponding to the minimal possible control.

We are looking for the universal rescaling method, that will be applicable to any reasonable situation (we do not want it to be adjustable to the situation, because the whole purpose of this rescaling is to avoid time-consuming adjustments). Suppose now that we first used kg/sec, compared two different scaling functions $f(u)$ and $\widetilde{f}(u)$, and it turned out that $f(u)$ is better (or, to be more precise, that the family $\Phi = \{\lambda \cdot f(u) + x_0\}$ is better than the family $\widetilde{\Phi} = \{\lambda \cdot \widetilde{f}(u) + x_0\}$). It sounds reasonable to expect that the relative quality of the two scaling functions should not depend on what units we used for u. So we expect that when we apply the same methods, but with the values of control expressed in lb/sec, then the results of applying $f(u)$ will still be better than the results of applying $\widetilde{f}(u)$.

The result of applying the function $f(u)$ to the control in lb/sec can be expressed in old units (kg/sec) as $f(c \cdot u)$, where c is a ratio of these two units. So the result of applying the rescaling function $f(u)$ to the data in new units (lb/sec) coincides with the result of applying a new scaling function $f_c(u) = f(c \cdot u)$ to the control in old units (kg/sec). So, we conclude that if $f(u)$ is better than $\widetilde{f}(u)$, then $f_c(u)$ must be better than $\widetilde{f}_c(u)$, where $f_c(u) = f(c \cdot u)$ and $\widetilde{f}_c(u) = \widetilde{f}(c \cdot u)$. This must be true for every c because we could use not only kg/sec or lb/sec, but arbitrary units as well.

Now we are ready for the formal definitions.

IV. Definitions and the Main Result

Definition 1: By a *rescaling function* (or a *rescaling*, for short), we mean a strictly monotonic function that maps the set of all positive real numbers $(0, \infty)$ onto the set of all real numbers $(-\infty, +\infty)$.

Definition 2: We say that two rescalings $f(u)$ and $\widetilde{f}(u)$ are *equivalent* if $\widetilde{f}(u) = \lambda \cdot f(u) + x_0$ for some positive constant λ and for some real number x_0.

Comment. As we have already mentioned, if we apply two equivalent rescalings, we will get two scales that are either both leading to a good control, or are both inadequate.

Definition 3: By a *family* we mean the set of functions $\{\lambda \cdot f(u) + x_0\}$, where $f(u)$ is a fixed rescaling, λ runs over all positive real numbers, and x_0 runs over all real numbers. The set of all families will be denoted by S.

Definition 4: A pair of relations $(<, \sim)$ is called *consistent* if it satisfies the following conditions:
- if $F < G$ and $G < H$ then $F < H$;
- $F \sim F$;
- if $F \sim G$ then $G \sim F$;
- if $F \sim G$ and $G \sim H$ then $F \sim H$;
- if $F < G$ and $G \sim H$ then $F < H$;
- if $F \sim G$ and $G < H$ then $F < H$;
- if $F < G$ then it is not true that $G < F$ or $F \sim G$.

Definition 5: Assume a set A is given. Its elements will be called *alternatives*. By an *optimality criterion* we mean a consistent pair $(<, \sim)$ of relations on the set A of all alternatives. If $G < F$, 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.

Definition 6: We say that an alternative F is *optimal* (or the *best*) with respect to a criterion $(<, \sim)$ if for every other alternative G either $G < F$ or $F \sim G$.

Definition 7: We say that a criterion is *final* if there exists an optimal alternative, and this optimal alternative is unique.

Comment. In the present paper, we consider optimality criteria on the set S of all families.

Definition 8: By a *result of a unit change* in a function $f(u)$ to a unit that is $c > 0$ times smaller we mean a function $f_c(u) = f(c \cdot u)$.

Definition 9: By the *result of a unit change* in a family Φ by $c > 0$ we mean the set of all the functions that are obtained by this unit change from $f \in \Phi$. This result will be denoted by $c \cdot \Phi$.

Definition 10: We say that an optimality criterion on F is *unit-invariant* if for every two families Φ and $\widetilde{\Phi}$ and for every number $c > 0$ the following two conditions are true:
- if $\Phi < \widetilde{\Phi}$, then $c \cdot \Phi < c \cdot \widetilde{\Phi}$;
- if $\Phi \sim \widetilde{\Phi}$, then $c \cdot \Phi \sim c \cdot \widetilde{\Phi}$.

Theorem 1: If a family Φ is optimal in the sense of some optimality criterion that is final and unit-invariant, then every rescaling $f(u)$ from Φ is equivalent to $f(u) = \log(u)$.

Comment. This result means that the optimal rescalings are of the type $\gamma \cdot \log(u) + \alpha$ for some real numbers $\gamma > 0$ and α.

Comment. For reader's convenience, the proof is given in the last section.

V. Case Study: Application of the Resulting Optimal Rescaling to Fuzzy Control (Brief Description)

A. Description of a Plant

We design a control for chemical reaction within a constant volume, non-adiabatic, continuously stirred tank reactor (CSTR). The model that describes the CSTR is described by the following system of differential equations

(see, e.g., [8]):

$$\dot{x}_1 = -x_1 + D \cdot a \cdot (1 - x_1) \cdot \exp\left(\frac{x_2}{1 + \frac{x_2}{\gamma}}\right);$$

$$\dot{x}_2 = -x_2 + B \cdot D \cdot a \cdot (1 - x_1) \cdot \exp\left(\frac{x_2}{1 + \frac{x_2}{\gamma}}\right) - u \cdot (x_2 - x_c),$$

where:

- x_1 is the conversion rate;
- x_2 is the (dimensionless) temperature; and
- u is the (dimensionless) heat transfer coefficient.

The objective of the control is to stabilize the system (i.e., bring it closer to the equilibrium point).

B. What We Did

We applied a logarithmic rescaling $x_2 \to X = \log(x_2)$, and used membership functions with equal spacing for X. No further adjustment of membership functions was made.

C. Results

Even without any further adjustment the results of this control were comparable to the results of applying the intelligent "gain scheduled" (non-linear) PID controller [3], [8]. In other words, we got the control that was as good as the one generated by the state-of-art traditional control theory with respect to stability and controllability of the plant.

With respect to the computational complexity our fuzzy controller is much simpler.

D. Rescaling Is Necessary

Without the rescaling, we got a fuzzy control whose quality was much worse than that of a PID controller.

Comment. The details of this case study were published in [14].

VI. Proof of the Main Result

The idea of this proof is as follows: first we prove that the optimal family is unit-invariant (in Part 1), and from that, in Part 2, we conclude that an arbitrary function f from Φ satisfies a certain functional equation; the solutions to this equations are known, and this completes the proof.

1. Let us first prove that the optimal family Φ_{opt} exists and is *unit-invariant* in the sense that $\Phi_{opt} = c \cdot \Phi_{opt}$ for all $c > 0$.

Indeed, we assumed that the optimality criterion is final, therefore there exists a unique optimal family Φ_{opt}. Let's now prove that this optimal family is unit-invariant (this proof is practically the same as in [9]). The fact that Φ_{opt} is optimal means that for every other Φ, either $\Phi < \Phi_{opt}$ or $\Phi_{opt} \sim \Phi$. If $\Phi_{opt} \sim \Phi$ for some $\Phi \neq \Phi_{opt}$, then from the definition of the optimality criterion we can easily deduce that Φ is also optimal, which contradicts the fact that there is only one optimal family. So for every Φ either $\Phi < \Phi_{opt}$ or $\Phi_{opt} = \Phi$.

Take an arbitrary c and apply this conclusion to $\Phi = c \cdot \Phi_{opt}$. If $c \cdot \Phi_{opt} = \Phi < \Phi_{opt}$, then from the invariance of the optimality criterion (condition ii)) we conclude that $\Phi_{opt} < c^{-1} \cdot \Phi_{opt}$, and that conclusion contradicts the choice of Φ_{opt} as the optimal family. So $\Phi = c \cdot \Phi_{opt} < \Phi_{opt}$ is impossible, and therefore $\Phi_{opt} = \Phi$, i.e., $\Phi_{opt} = c \cdot \Phi_{opt}$, and the optimal family is really unit-invariant.

2. Let us now deduce the actual form of the functions $f(u)$ from the optimal family Φ_{opt}.

If $f(u)$ is such a function, then the result $f(c \cdot u)$ of changing the unit of u to a c times smaller unit belongs to $c \cdot \Phi_{opt}$; so, due to Part 1 of this proof, the function $f(c \cdot u)$ also belongs to the family Φ_{opt}.

By the definition of a family, all its functions can be obtained from each other by a linear transformation $\lambda \cdot f(u) + x_0$; therefore, $f(c \cdot u) = \lambda \cdot f(u) + x_0$ for some λ and x_0. The corresponding values λ and x_0 depend on c; so, we arrive at the following functional equation for $f(u)$:

$$f(c \cdot u) = \lambda(c) \cdot f(u) + x_0(c).$$

In the survey on functional equations [1], the solutions of this equation are not explicitly given, but a for a similar functional equation

$$f(x + y) = f(x) \cdot h(y) + k(y),$$

all solutions are enumerated in Corollary 1 to Theorem 1 from Section 3.1.2 of [1]: they are $f(x) = \gamma \cdot x + \alpha$ and $f(x) = \gamma \cdot \exp(c \cdot x) + \alpha$, where $\gamma \neq 0$, $c \neq 0$ and α are arbitrary constants. To use this result, let us reduce our equation to the one with known solutions.

The only difference between these two equations is that we have a product, and we need a sum. There is a well known way to reduce product to a sum: turn to logarithms, because $\log(ab) = \log(a) + \log(b)$. For simplicity, let us use natural logarithms $\ln(x)$. Let us introduce new variables $X = \ln(u)$ and $Y = \ln(c)$. In terms of these new variables, $x = \exp(X)$ and $c = \exp(Y)$. Substituting these values into our functional equation, and taking into consideration that

$$\exp(X) \cdot \exp(Y) = \exp(X + Y),$$

we conclude that

$$F(X + Y) = H(Y) \cdot F(X) + K(Y),$$

where we denoted

$$F(X) \stackrel{\text{def}}{=} f(\exp(X)), \quad H(Y) \stackrel{\text{def}}{=} \lambda(\exp(Y)),$$

$$K(Y) \stackrel{\text{def}}{=} x_0(\exp(Y)).$$

So, according to the above-cited result, either

$$F(X) = \gamma \cdot X + \alpha,$$

or $F(X) = \gamma \cdot \exp(c \cdot X) + \alpha$.

From $F(X) = f(\exp(X))$, we conclude that $f(u) = F(\ln(u))$, therefore either $f(u) = \gamma \cdot \ln(u) + \alpha$, or $f(u) = \gamma \cdot \exp(c \cdot \ln(u)) + \alpha = \gamma \cdot u^c + \alpha$. In the second case the

function $f(u)$ maps $(0,\infty)$ onto the interval (α,∞), and we defined a rescaling as a function whose values run over all possible real numbers. So the second case is impossible, and $f(x) = \gamma \cdot \ln(u) + \alpha$, which means that $f(u)$ is equivalent to a logarithm. Q.E.D.

VII. Conclusions

One of the most successful examples of soft computing is fuzzy control. One of the important steps in designing a fuzzy control is the choice of the membership functions for all the terms that the experts use. This choice strongly influences the quality of the resulting control.

For simple controlled systems, it is sufficient to have equally spaced membership functions, i.e., functions that have similar shape (usually triangular or trapezoid), and are located in intervals of equal length

$$\ldots, [N-\Delta, N+\Delta], [N, N+2\Delta], [N+\Delta, N+3\Delta], \ldots$$

For complicated systems this choice does not lead to a good fuzzy control, so it is necessary to tune the membership functions. This tuning is usually done by using soft computing techniques such as neural networks or genetic algorithms. Such tuning is, however, a very time-consuming procedure. We show that traditional ("hard computing") optimization techniques lead to a faster tuning.

Specifically, we consider the case when equally spaced membership functions are inadequate because the control variable u can take only positive values. Such situations occur, for example, when we control the flux of the substances into a chemical reactor (e.g., the flux of fuel into an engine). Our idea is to "rescale" this variable, i.e., to use a new variable $\widetilde{u} = f(u)$, and to choose a function $f(u)$ in such a way that we can apply membership functions, that are equally spaced in $\widetilde{u}$.

We give a mathematical proof that the optimal rescaling is logarithmic ($f(u) = a \cdot \log(u) + b$). We also show on a real-life example of a non-linear chemical reactor that the resulting fuzzy control, without any further tuning of membership functions, can be comparable in quality with the best state-of-art non-linear controls of traditional control theory.

Acknowledgments

This work was supported in part by NASA under cooperative agreement NCC5-209, by the 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 F49620-00-1-0365, and by Grant No. W-00016 from the U.S.-Czech Science and Technology Joint Fund.

References

[1] J. Aczel, *Lectures on functional equations and their applications*, Academic Press, NY–London, 1966.

[2] S.S.L. Chang and L.A. Zadeh, "On fuzzy mapping and control", *IEEE Transactions on Systems, Man and Cybernetics*, 1972, Vol. SMC-2, pp. 30–34.

[3] K.A. Hoo and J.C. Kantor, *Chemical Engineering Communications*, 1985, Vol. 37, No. 1.

[4] V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational complexity and feasibility of data processing and interval computations*, Kluwer, Dordrecht, 1998.

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

[6] E.H. Mamdani, "Application of fuzzy algorithms for control of simple dynamic plant", *Proceedings of the IEE*, 1974, Vol. 121, No. 12, pp. 1585–1588.

[7] R.R. Mohler, *Nonlinear systems. Vol. I. Dynamics and control. Vol. II. Applications to bilinear control,* Prentice Hall, Englewood Cliffs, NJ, 1991.

[8] N.D. Murray. *Nonlinear PID controllers*, Unpublished Master's Thesis, Virginia Polytechnic Institute and State University, September 1990.

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

[10] H.T. Nguyen and V. Kreinovich, "Methodology of fuzzy control: an introduction", In: H.T. Nguyen and M. Sugeno (eds.), *Fuzzy Systems: Modeling and Control*, Kluwer, Boston, MA, 1998, pp. 19–62.

[11] H.T. Nguyen and M. Sugeno (eds.), *Fuzzy Systems: Modeling and Control*, Kluwer, Boston, MA, 1998.

[12] M. Sugeno (editor), *Industrial Applications of Fuzzy Control*, North Holland, Amsterdam, 1985.

[13] P. Suppes, D.H. Krant, R.D. Luce, and A. Tversky, *Foundations of measurement.* Academic Press, San Diego, CA. Vol. I, 1971; Vol. II, 1989, Vol. III, 1989.

[14] H.F. VanLangingham and A. Tsoukkas, "Application of fuzzy logic control to nonlinear process control", *Proceedings of the 1992 International Fuzzy Systems and Intelligent Control Conference*, Louisville, KY, 1992, pp. 8–17.

[15] H.F. VanLandingham, A. Tsoukkas, V. Kreinovich, and C. Quintana, "Nonlinear rescaling of control values simplifies fuzzy control", *Proceedings of the Third International Workshop on Neural Networks and Fuzzy Logic, Houston, TX, June 1–3, 1992*, NASA, January 1993, Vol. I (NASA Conference Publication No. 10111), pp. 174–182.

[16] L.A. Zadeh, "Fuzzy sets", *Information and control*, 1965, Vol. 8, pp. 338–353.

[17] L.A. Zadeh. "Towards a theory of fuzzy systems", In: R.E. Kalman, N. DecLaris (eds.), *Aspects of Network and Systems Theory*, Holt, Rinehart, Winston, 1971.

[18] L.A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes", *IEEE Transactions on Systems, Man and Cybernetics*, 1973, Vol. 3, pp. 28–44.