

Error-free Simplification of Transparent Mamdani Systems

Andri Riid

Laboratory of Proactive Technologies
Tallinn University of Technology
Ehitajate tee 5, 19086, Tallinn, Estonia
e-mail: andri@dcc.ttu.ee

Kalle Saastamoinen

Laboratory of Applied Mathematics
Lappeenranta University of Technology
P.O. Box 20, 53851, Lappeenranta, Finland
e-mail: kalle.saastamoinen@lut.fi

Ennu Rüstern

Department of Computer Control
Tallinn University of Technology
Ehitajate tee 5, 19086, Tallinn, Estonia
e-mail: ennu.rustern@dcc.ttu.ee

Abstract— This paper shows that combinatorial complexity of fuzzy systems is at least in part caused by redundancy in these systems and presents the algorithm and its implementation for detection and removal of such redundancy for a special class of Mamdani systems. Performance of the simplification algorithm is demonstrated with uniformly impressive results on acknowledged benchmarks coming from different areas of engineering - truck backer-upper control, Mackey-Glass time series prediction and Iris data classification.

I. MOTIVATION

One evident problem that is marring the large scale applications of fuzzy logic is the combinatorial explosion of rules (curse of dimensionality). As the number of MFs and/or input variables increases, the upper bound of fuzzy rules grows exponentially:

$$R_{max} = \prod_{i=1}^N S_i, \quad (1)$$

where S_i is the number of MFs per i -th input variable ($i = 1, \dots, N$).

In the ideal fuzzy system the number of fuzzy rules $R = R_{max}$, meaning that the rule base of the system is fully defined and contains all possible antecedent combinations. Situation $R > R_{max}$ indicates a failure in fuzzy system design - either redundant or contradictory rules are present, both of which distort the original goal of the designer. In real life applications, however, the number of rules usually remains well below R_{max} for several reasons.

First of all, commonly there is not enough material (data) or immaterial (knowledge) evidence to cover the input space universally, not only because it would be too time consuming to collect exhaustive evidence in large scale applications but also because of potential inconsistency that certain antecedent combinations may present (an antecedent "IF sun is bright AND rain is heavy" could be one such example).

Moreover, it is common practice that for the sake of compactness, the rules with little relevance are excluded from the model (for all we know they may be based on few noisy samples). The exclusion decision of a given rule may be based on its contribution to approximation properties (using singular value decomposition, orthogonal transforms, etc. [1]) or on

how often or to what degree a given rule is contributing to the output (this can be easily evaluated by computing cumulative rule activation degrees on available data).

On the whole, rule base simplification can be fitted under two categories - error-free simplification or degrading simplification. Error-free simplification searches for existing redundancies in the model. In other words, if error-free simplification is effective, it is the sign of a shortcoming of the initial designer.

With degrading simplification, the model is made less complex by removing non-redundant system parameters. Incidentally, this is achieved at the expense of system flexibility, accuracy etc.

Typically, simplification is carried out on initial complex model. However, with certain design methodologies unnecessary complexity is avoided by the model design procedure. A typical example is the application of tree partitioning of the input space (instead of more common grid partitioning) but the most common constructive compactness-friendly approach these days (related primarily to 1st order Takagi-Sugeno systems [2]) is fuzzy clustering. With clustering, the rules are created in product space only in regions where data concentration is high. Interestingly enough, the side effect of that is the redundancy of cluster projections that are used as the prototypes for MFs of the model. The projections that become fuzzy sets may be highly similar to each other, similar to the universal set or reduced to singleton sets, which calls for adequate methods to sort out the mess [3]. Another feature of product space clustering is that R is always much less than R_{max} (in fact $R = S_i$ before simplification). For this reason and also from interpolational aspect, product space clustering is not very well suited for Mamdani modeling.

For Mamdani modeling, (Mamdani systems and their properties are observed in Sect. II), where output MFs are typically shared among rules we therefore propose an error-free simplification algorithm in this paper. The algorithm is based on three simple enough principles covered in Sect. III. Sect. IV considers the typical implementation issues when designing a computer program for the simplification of Mamdani systems. In the remainder of the paper, the algorithm is applied successfully to different typical engineering problems - control,

identification and classification.

II. MAMDANI SYSTEMS

Generally, fuzzy rules in Mamdani-type fuzzy systems are based on the disjunctive rule format

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1r} \text{ AND } x_2 \text{ is } A_{2r} \text{ AND } \dots \\ &\dots \text{ AND } x_N \text{ is } A_{Nr} \text{ THEN } y \text{ is } B_r \\ &\text{OR } \dots \end{aligned} \quad (2)$$

where A_{ir} denote the linguistic labels of the i -th input variable associated with the r -th rule ($i = 1, \dots, N$), and B_r is the linguistic label of the output variable, associated with the r -th rule.

Each A_{ir} has its representation in the numerical domain - the membership function μ_{ir} (the same applies to B_r that is represented by γ_r) and in general case the inference function that computes the fuzzy output $F(y)$ of the system (2) has the following form

$$F(y) = \bigcup_{r=1}^R \left(\left(\bigcap_{i=1}^N \mu_{ir}(x_i) \right) \cap \gamma_r \right), \quad (3)$$

where \bigcup_r^R denotes the aggregation operator (corresponds to OR in (2)), \cap is the implication operator (THEN) and \bigcap_i^N is the conjunction operator (AND). In order to obtain crisp output, (3) is generally defuzzified with center-of-gravity method

$$y = Y_{cog}(F(y)) = \frac{\int_Y y F(y) dy}{\int_Y F(y) dy}. \quad (4)$$

The results obtained in current paper are valid for a subclass of Mamdani systems that match the following requirements:

- The inference operators used here are product and sum. They are distributive, associative, commutative and they have neutral element. Distributivity means that in a given universe X binary operation \cap is distributive over \cup if $a \cap (b \cup c) = (a \cap b) \cup (a \cap c)$ and vice versa for operation \cup . Associativity means that $a \cap (b \cap c) = (a \cap b) \cap c$. Commutativity means that $a \cap b = b \cap a$ and vice versa for operation \cup , neutral element means that $a \cap 1 = a$ and $a \cup 0 = a$ for all elements $a, b, c \in X$.

With product-sum inference (4) reduces to

$$y = \frac{\sum_{r=1}^R \tau_r c_r s_r}{\sum_{r=1}^R \tau_r s_r}, \quad (5)$$

where τ_r is the activation degree of r -th rule (computed with the conjunction operator (product)) and c_r and s_r are the center-of-gravity and area of γ_r , respectively (see [4]).

- The input MFs ($s = 1, \dots, S_i$) are given by the following definition:

$$\mu_i^s(x_i) = \begin{cases} \frac{x_i - a_i^{s-1}}{a_i^s - a_i^{s-1}}, & a_i^{s-1} < x_i < a_i^s \\ \frac{a_i^{s+1} - x_i}{a_i^{s+1} - a_i^s}, & a_i^s < x_i < a_i^{s+1} \\ 0, & a_i^{s+1} \leq x_i \leq a_i^{s+1} \end{cases}, \quad (6)$$

Such definition of input MFs satisfies input transparency condition assumed for correct interpretation of Mamdani

rules (see [5] for further details), however, in current paper we are more interested in its other property, namely

$$\sum_{s=1}^{S_i} \mu_i^s = 1. \quad (7)$$

- The number of output MFs is relatively small and they are shared among rules (this is, in fact, typical for Mamdani systems).

III. ERROR-FREE RULE BASE SIMPLIFICATION TECHNIQUES

In this section the rule base simplification principles will be reviewed.

Lemma 1: If a subset of fuzzy rules consisting of S_i rules (8) share the same output MF B_*

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1*} \text{ AND } \dots \text{ AND } x_i \text{ is } A_i^s \dots \\ &\dots \text{ AND } x_N \text{ is } A_{N*} \text{ THEN } y \text{ is } B_* \\ &s = 1, \dots, S_i \end{aligned} \quad (8)$$

then this group of rules can be replaced by a following single rule.

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1*} \text{ AND } \dots \text{ AND } x_{i-1} \text{ is } A_{i-1,*} \text{ AND} \\ &x_{i+1} \text{ is } A_{i+1,*} \dots \text{ AND } x_N \text{ is } A_{N*} \text{ THEN } y \text{ is } B_* \end{aligned} \quad (9)$$

Proof: To prove that (this derives from (5)) we need to show that

$$\sum_{s=1}^{S_i} \mu_i^s \prod_{j=1, j \neq i}^N \mu_{j*} = \prod_{j=1, j \neq i}^N \mu_{j*}, \quad (10)$$

which is obvious when

$$\sum_{s=1}^{S_i} \mu_i^s = 1, \quad (11)$$

which is ensured by (6) that concludes the proof. \blacksquare

Example 1. Consider three rules of a two-input Mamdani system:

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is } A_{21} \text{ THEN } y \text{ is } B_1 \\ &\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is } A_{22} \text{ THEN } y \text{ is } B_1 \\ &\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is } A_{23} \text{ THEN } y \text{ is } B_1 \end{aligned} \quad (12)$$

If there are no more linguistic labels of x_2 as Fig. 1 clearly

	A_{21}	A_{22}	A_{23}
A_{11}	B_1	B_1	B_1
A_{12}			
A_{13}			

Fig. 1. Redundancy of rules that makes rule compression possible.

implies, it is quite obvious that if x_1 is A_{11} , output is in

fact independent from the value of x_2 that could as well be expressed by the following single rule

$$\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is } \textit{whatever} \text{ THEN } y \text{ is } B_1, \quad (13)$$

where "whatever" (or "don't care") describes the situation that x_2 may have any value in its domain without a slightest effect to the output and can thus be removed from the rule, resulting in a nice compressed formulation

$$\text{IF } x_1 \text{ is } A_{11} \text{ THEN } y \text{ is } B_1 \quad (14)$$

Lemma 2: If a subset of fuzzy rules consisting of $S_i - 1$ rules share the same output MF

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1*} \text{ AND } \dots \text{ AND } x_i \text{ is } A_i^s \dots \\ &\dots \text{ AND } x_N \text{ is } A_{N*} \text{ THEN } y \text{ is } B_* \\ &s = 1, \dots, S_i, s \neq t \end{aligned} \quad (15)$$

then this group of rules can be replaced by a following single rule.

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{1*} \text{ AND } \dots \text{ AND } x_i \text{ is NOT } A_i^t \dots \\ &\dots \text{ AND } x_N \text{ is } A_{N*} \text{ THEN } y \text{ is } B_* \end{aligned} \quad (16)$$

Proof: To prove that we need to show that

$$\sum_{s=1, s \neq t}^{S_i} \mu_i^s \prod_{j=1, j \neq i}^N \mu_{j*} = (1 - \mu_i^t) \prod_{j=1, j \neq i}^N \mu_{j*}, \quad (17)$$

where $1 - \mu_i^t$ represents the negation of A_i^t .

It is easy to see that $\sum_{s=1, s \neq t}^{S_i} \mu_i^s = 1 - \mu_i^t$ if MFs of the i -th input variable add up to one (7), which completes the proof. ■

Example 2: Consider two rules of a hypothetical fuzzy system

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is } A_{21} \text{ THEN } y \text{ is } B_1 \\ &\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is } A_{23} \text{ THEN } y \text{ is } B_1 \end{aligned} \quad (18)$$

	A_{21}	A_{22}	A_{23}
A_{11}	B_1		B_1
A_{12}			
A_{13}			

Fig. 2. Rule base configuration allowing NOT construction.

(18) implies that y is independent from x_2 if x_1 is A_{11} and x_2 is "anything else than A_{22} " or simply "NOT A_{22} ":

$$\text{IF } x_1 \text{ is } A_{11} \text{ AND } x_2 \text{ is NOT } A_{22} \text{ THEN } y \text{ is } B_1 \quad (19)$$

Note that it would be also possible to write (19) as

$$\text{IF } x_1 \text{ is } A_{11} \text{ THEN } y \text{ is } B_1 \text{ UNLESS } x_2 \text{ is } A_{22} \quad (20)$$

Lemma 3. Let us consider a subset of rules containing all occurrences of consecutive MFs (μ_i^s and μ_i^{s+1}) of a given i -th variable

If the rule base is complete, it is possible to find a pair of rules from the subset represented by $c^* \cdot s^* \cdot \mu_{1*} \cdot \dots \cdot \mu_i^s \cdot \dots \cdot \mu_{N*}$ and $c^{**} \cdot s^{**} \cdot \mu_{1*} \cdot \dots \cdot \mu_i^{s+1} \cdot \dots \cdot \mu_{N*}$ (as each rule is represented by a $c_r s_r \prod_{i=1}^N \mu_{ir}$ according to (5)).

If $\gamma^* \equiv \gamma^{**}$, the above rules can be written as

$$(\mu_i^s + \mu_i^{s+1}) \cdot c^* \cdot s^* \cdot \mu_{1*} \cdot \dots \cdot \mu_{i-1,*} \cdot \mu_{i+1,*} \cdot \dots \cdot \mu_{N*} \quad (21)$$

If similar thing can be validated for for all remaining pairs of the subset it means in fact that we can replace the original MFs with a new one $\mu_i' = \mu_i^s + \mu_i^{s+1}$.

Proof: Obvious. ■

Example 3. Consider the situation depicted in Fig. 3: The

	A_{21}	A_{22}	A_{23}
A_{11}	B_1	B_1	
A_{12}	B_3	B_3	
A_{13}	B_2	B_2	

Fig. 3. Redundancy of MFs revealed by rule base analysis.

segment of (7) corresponding to this situation:

$$\begin{aligned} &c_1 s_1 \mu_{11} \mu_{21} + c_3 s_3 \mu_{12} \mu_{21} + c_2 s_2 \mu_{13} \mu_{21} + \\ &+ c_1 s_1 \mu_{11} \mu_{22} + c_3 s_3 \mu_{12} \mu_{22} + c_2 s_2 \mu_{13} \mu_{22} = \\ &= (\mu_{21} + \mu_{22})(c_1 s_1 \mu_{11} + c_2 s_2 \mu_{13} + c_3 s_3 \mu_{12}) \end{aligned} \quad (22)$$

Because these six rules cover all occurrences of μ_{21} and μ_{22} , we can define a new MF replacing μ_{21} and μ_{22} and thus can get rid of three redundant rules in original rule base. Note that summing up two triangles of (6) would result in a trapezoid MF and the updated partition would still satisfy (7).

IV. IMPLEMENTATION.

For fuzzy logic software tools the rule base information is generally stored in a separate $R \times (N + 1)$ dimensional matrix (Matlab's Fuzzy Logic Toolbox uses a variant of this) that accommodates the identifiers of MFs associated with fuzzy rules. Each row in the matrix represents an individual rule and column specifies the input variable (output variable in the last column, which is written in bold in the examples below) to which the identifier in current column is assigned to. Note that NOT-operator is represented by a minus sign and 0 represents a removed antecedent variable, e.g. a line -1 2 0 4 5 would be equivalent to a rule

$$\begin{aligned} &\text{IF } x_1 \text{ is NOT } A_{11} \text{ AND } x_2 \text{ is } A_{22} \text{ AND} \\ &x_4 \text{ is } A_{44} \text{ THEN } y \text{ is } B_5 \end{aligned} \quad (23)$$

Implementation of rule base reduction schemes described in Sect. III is based on the analysis of the rule matrix and

subsequent manipulation of it, which is described with the following algorithm (except the detection of redundant MFs that follows the logic under Lemma 3).

- 1) Fix an input variable (e.g. input No. 3)
- 2) Delete (temporarily) the indices corresponding to this variable from the rule matrix.
- 3) Split the rule matrix into submatrices containing groups of rules where other input variables have fixed MFs throughout the group
- 4) If the output MF associated with the rules is the same throughout the submatrix combine this group into a new rule and restore deleted indices.
- 5) Combine submatrices into one rule matrix.

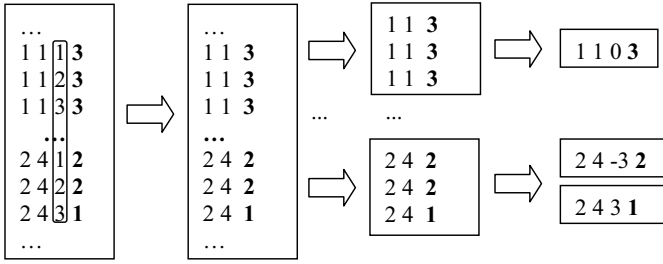


Fig. 4. Four steps of the simplification algorithm.

Simplification mechanism employing NOT-constructions is equivalent up to the step 4, where rules in the group (on the assumption that there are no more than two unique output MFs associated with the group and one of them is used only once) that are associated with the more "popular" output MF can be combined into one.

One of the important aspects of the approach is that the rule that is once used in the simplification as the raw material for the improved rule, cannot be "recycled". This is illustrated with the following simple example. From the initial rule set in

	A_{21}	A_{22}	A_{23}
A_{11}	B_1	B_1	B_1
A_{12}	B_1	B_2	B_1
A_{13}	B_1	B_2	B_2

Fig. 5. Conflicting simplification scenarios.

Fig. 5 we can extract a compressed rule "IF x_1 is A_{11} THEN y is B_1 " or "IF x_2 is A_{21} THEN y is B_1 " but not both because that means that the rule "IF x_1 is A_{11} AND x_2 is A_{21} THEN y is B_1 " would appear twice in the rule base. Similarly, if we decide in favor of the first compressed rule then next we choose "IF x_1 is NOT A_{11} AND x_2 is A_{21} THEN y is B_1 " and from the remainder we can extract "IF x_1 is NOT A_{11} and

x_2 is A_{22} THEN y is B_2 " but not simultaneously with "IF x_1 is A_{13} AND x_2 is NOT A_{21} THEN y is B_2 ". Incidentally, the simplification tool must not only be considerate to flag all used rules with "don't touch" signs for itself but also intelligent enough to be able to choose between different simplification scenarios to maximize the rule reduction rate. This presents quite a challenge to the developer of the software.

Everything of the above is based on the assumption that the rule base is complete. Incompleteness of the rule base, however, arises the question how to treat the blank spots in the rule base. Could we use them to our advantage so as to minimize the number of rules? Or should we maintain status quo and integrity of the rule base? To explain this dilemma in finer detail let us look at the first column in Fig. 6. In first (optimistic) approach we may combine these rules into IF x_2 is A_{21} THEN y is B_2 , ignoring the fact that it would actually mean writing B_2 into the blank spot and thus changing the original rule base. In conservative approach, we combine existing rules into IF x_1 is NOT A_{13} AND x_2 is A_{21} THEN y is B_2 , leaving the blank spot untouched. We encounter further such dilemmas when analyzing the rest of the rule base. Conservative approach indeed seems closer to the spirit of error-free simplification, however, when we look at the problem at numerical level, the pros and cons are not so obvious.

Each undefined rule means that there is some area in the input space for which we cannot compute matching output values. In practice some pre-specified value (usually average of the domain of the output variable) is used in this case to maintain continuity. If we use the blank spot to our advantage we typically use a neighboring rule as the prototype. Both may and may not be adequate guesses for the missing rule, neither is clearly better. Therefore, the simplification tool has a built-in option to determine if we take optimistic or conservative approach when treating incomplete rule bases. In the following, we use the notation $I_c = 1$ for the conservative and $I_c = 0$ for the optimistic approach.

	A_{21}	A_{22}	A_{23}	A_{24}
A_{11}	B_3	B_1	B_1	B_3
A_{12}	B_3	B_2		B_3
A_{13}			B_2	
A_{14}	B_3	B_3	B_3	B_2

Fig. 6. An incomplete rule base: another challenge.

V. APPLICATIONS

The proposed approach is validated on three applications coming from different areas of engineering - truck backer-

upper control, Mackey-Glass time series identification and Iris data classification that are presented below.

A. Truck backer-upper controller simplification

In this application the simplification algorithm is applied to the the fuzzy trajectory management unit (TMU) of truck backer-upper control system from [6] that originally uses 28 rules that specify the optimal truck angle Φ_r in respect to its coordinates x and y . Application of the algorithm reveals that the original controller is heavily redundant as the number of its rules can be reduced to 11 without any loss in control quality that means almost 60% reduction in size (see Fig. 7). Incidentally, the biggest contribution to size reduction comes from detection and merging redundant MFs (13 rules), rule compression removes 2 and NOT-construction further 2 rules. As the original rule base is complete, applying the simplification tool with either (optimistic or conservative) option produces the same exact result

B. Mackey-Glass time series prediction

The next example includes prediction of a time series that is generated by the Mackey-Glass [7] time-delay differential equation

$$\dot{x} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t), \quad (24)$$

and subsequent simplification of the prediction model.

To obtain the time series value at integer points, the numerical solution to the above MG equation is found using the fourth-order Runge-Kutta method. We assume $x(0) = 1.2$, $\tau = 17$, and $x(t) = 0$ for $t < 0$.

We use up 4 to known values of the time series in time, to predict the value in the future. For each t , the input training data is a four-dimensional vector of the following form.

$$x(t+6) = f(x(t-18), x(t-12), x(t-6), x(t)) \quad (25)$$

There are 1000 input/output data values. We use the first 500 data values for training, while the others are used as checking data for validating the identified fuzzy model.

To obtain the prediction model we apply a simplistic modeling algorithm [8] that provides a crude predictor of the phenomenon (we are more interested in the performance of our simplification algorithm than in modeling accuracy). This method assumes predefined input-output partition - we are using an uniform one for the sake of simplicity - and finds out the best matching set of fuzzy rules for this partition on the basis of training data samples. For each potential rule we identify the sample $[x_1(k)x_2(k)x_3(k)x_4(k)y(k)]$ that yields maximum rule activation degree $\tau_r(k)$ and use $y(k)$ to determine matching output MF $\gamma_j, j = 1, \dots, T$ (the one that produces $\max(\gamma_j(y(k)))$).

To observe the effects of simplification we employ models of different sizes by varying the number of MFs (and even input variables - both 3- and 4-input models are being used). The results are given in Table I, where first column specifies the S_i s of each input variable, the second the number of output MFs (T). Further columns contain modeling errors on training

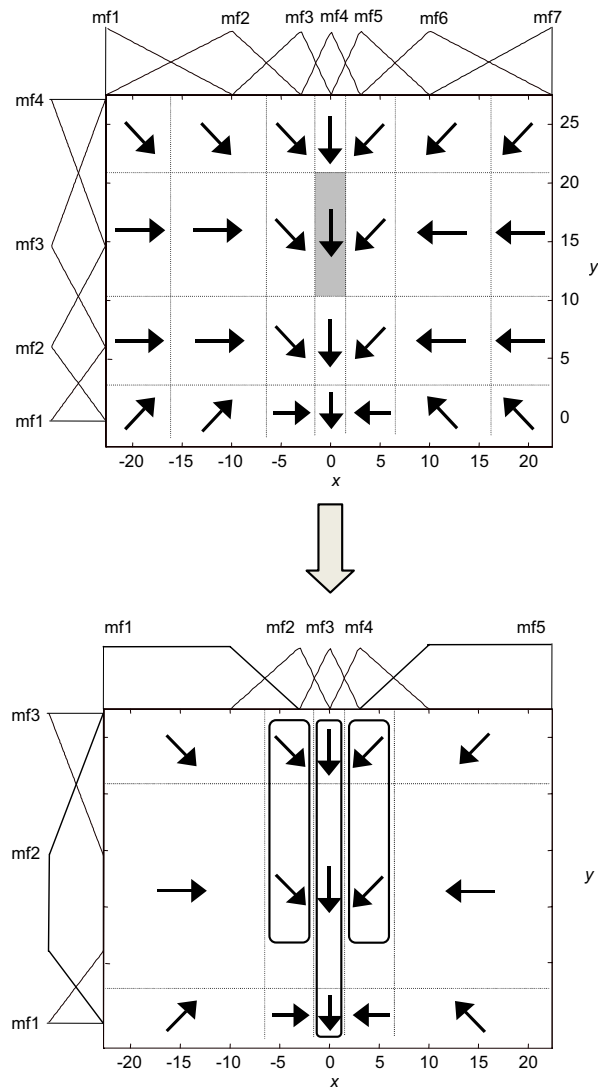


Fig. 7. TMU of the truck backer-upper before (above) and after (below) the simplification.

and checking data (ϵ_{tr} and ϵ_{ch} , respectively), the number of rules before (R_0) and after simplification (R_f) and rule reduction rates (η).

The results reveal some general characteristics of the simplification algorithm. It can be seen that rule reduction rate is higher if the number of output MFs is small and number of input MFs is high, which is rather logical, because with a small number of output MFs these are shared more extensively and thus redundancy is more likely to exist. Large number of input MFs on the other hand means that the number of rules that can be replaced by a single rule is generally larger. However, with incomplete rule bases, the large number of input MFs increases the number of undefined rules thus limiting algorithm's capability when we take conservative approach regarding completeness of the rule base. It is, however, clearly evident that this option does not have any effect to the modeling error neither with training nor checking data.

TABLE I

RESULTS OF MG TIME SERIES PREDICTION AND MODEL SIMPLIFICATION

S_i	T	ϵ_{tr}	ϵ_{ch}	R_0	R_f	I_c	η
4 × 4 × 4	5	0.0691	0.0687	57	29	1	49.1%
4 × 4 × 4	5	0.0691	0.0687	57	26	0	54.4%
4 × 4 × 4	9	0.0623	0.0620	57	44	1	22.8%
4 × 4 × 4	9	0.0623	0.0620	57	40	0	30.0%
5 × 5 × 5	5	0.0599	0.0593	88	56	1	36.4%
5 × 5 × 5	5	0.0599	0.0593	88	37	0	58.0%
5 × 5 × 5	9	0.0426	0.0420	88	79	1	10.2%
5 × 5 × 5	9	0.0426	0.0420	88	60	0	31.8%
6 × 6 × 6	5	0.0483	0.0479	128	112	1	12.5%
6 × 6 × 6	5	0.0483	0.0479	128	60	0	53.1%
6 × 6 × 6	9	0.0347	0.0346	128	128	1	0%
6 × 6 × 6	9	0.0347	0.0346	128	90	0	29.7%
3 × 3 × 3 × 3	5	0.0639	0.0627	68	36	1	47.1%
3 × 3 × 3 × 3	5	0.0639	0.0627	68	41	0	39.7%
3 × 3 × 3 × 3	9	0.0619	0.0607	68	43	1	36.8%
3 × 3 × 3 × 3	9	0.0619	0.0607	68	44	0	35.3%
4 × 4 × 4 × 4	5	0.0384	0.0376	181	110	1	39.2%
4 × 4 × 4 × 4	5	0.0384	0.0376	181	96	0	47.0%
4 × 4 × 4 × 4	9	0.0404	0.0397	181	131	1	27.6%
4 × 4 × 4 × 4	9	0.0404	0.0397	181	115	0	36.5%
5 × 5 × 5 × 5	5	0.0451	0.0442	275	227	1	17.5%
5 × 5 × 5 × 5	5	0.0451	0.0442	275	128	0	53.5%
5 × 5 × 5 × 5	9	0.0354	0.0345	275	254	1	7.6%
5 × 5 × 5 × 5	9	0.0354	0.0345	275	162	0	41.1%

It also turns out that redundancy of input MFs is a rather rare phenomenon as it was detected in none of the above models. Rule compression contributes more to η if $I_c = 1$ and NOT operator is mostly responsible for redundancy removal if $I_c = 0$.

As for comparison of the modeling accuracy - ANFIS [9] with two generalized bell membership functions on each of the four inputs and containing 16 Takagi-Sugeno rules shows $\epsilon_{tr} = \epsilon_{ch} = 0.0025$ after 10 training epochs.

C. Iris data classification

The Iris flower data set is a multivariate data set introduced by Sir Ronald Aylmer Fisher [10]. The dataset consists of 50 samples from each of three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor). Four features are available for each sample, they are the length and the width of sepal and petal. The task is to develop a classifier that determines the species based on the combination of the four features.

We use the same algorithm as for time-series prediction, however, because there are 3 species there will be always 3 output MFs each representing the different species. After rule base extraction the center-of-gravity defuzzification is replaced by mean-of-maxima method that produces neatly discretized output values $\{1,2,3\}$ corresponding to three classes.

The results are given in Table II where ϵ_{tr} represents the number of misclassified samples from the training data set and ϵ_{ch} represents the number of misclassified samples from the training data set (both contain 75 samples in total).

It turns out that if the number of input variables is smaller than 4 MF redundancy is detected. In other ways the results resemble those we obtained with time series prediction. As for comparison of classification accuracy, 96.67% is reported in [11] using the classifiers with 4 inputs (and 7 rules) and

TABLE II

RESULTS OF IRIS CLASSIFICATION AND MODEL SIMPLIFICATION

S_i	ϵ_{tr}	ϵ_{ch}	R_0	R_f	I_c	η
3 × 3	3	3	8	5	1	37.5%
3 × 3	3	3	8	5	0	37.5%
4 × 4	8	13	12	6	1	50.0%
4 × 4	8	13	12	6	0	50.0%
3 × 3 × 3	2	4	23	9	1	60.9%
3 × 3 × 3	2	4	23	9	0	60.9%
4 × 4 × 4	3	3	35	17	1	57.1%
4 × 4 × 4	3	3	35	12	0	65.7%
3 × 3 × 3 × 3	2	4	58	19	1	60.2%
3 × 3 × 3 × 3	2	4	58	22	0	62.1%
4 × 4 × 4 × 4	6	8	107	42	1	60.7%
4 × 4 × 4 × 4	6	8	107	23	0	78.5%

2 inputs (and 5 rules). This is very close to our result (96%) with comparable configurations.

VI. CONCLUSIONS

In this paper we presented some ideas how to get rid of redundancy in a special class of Mamdani systems and also demonstrated that the implementation of these ideas indeed reduces complexity of fuzzy systems from different areas of engineering by 30-60%.

REFERENCES

- [1] J. Yen and L. Wang, "Simplifying Fuzzy Rule-Based Models Using Orthogonal Transformation Methods," *IEEE Trans. Systems, Man, Cybern. Part B*, vol. 29, no. 1, pp. 13-24, 1999.
- [2] T. Takagi, M. Sugeno and "Fuzzy identification of systems and its applications to modeling and control", *IEEE Trans. Systems Man and Cybern.*, vol. 15, pp. 116-132, 1985.
- [3] H. Roubos and M. Setnes, "Compact and Transparent Fuzzy Models and Classifiers Through Iterative Complexity Reduction," *IEEE Trans. Fuzzy Systems*, vol. 9, no. 4, pp. 516-524, 2001.
- [4] A. Riid and E. Rüstern, "On the Interpretability and Representation of Linguistic Fuzzy Systems," *Proc. IASTED International Conference on Artificial Intelligence and Applications*, Benalmadena, Spain, pp. 88-93, 2003.
- [5] A. Riid E. Rüstern, "Transparent Fuzzy Systems in Modeling and Control" in *Interpretability Issues in Fuzzy Modeling*, J. Casillas, O. Cordon, F. Herrera and L. Magdalena (editors), New York: Springer, pp. 452-476, 2003.
- [6] A. Riid and E. Rüstern, "Fuzzy logic in control: truck backer-upper problem revisited," *Proc. IEEE Int. Conf. Fuzzy Systems*, vol. 1, pp. 513-516, 2001.
- [7] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science*, vol. 197, pp. 287-289, 1977.
- [8] L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. on Systems, Man and Cybern.*, vol. 22, no. 6, pp. 1414-1427, 1992.
- [9] J.-S. R. Jang, "ANFIS: Adaptive-Network-Based Fuzzy Inference System," *IEEE Trans. on Systems, Man and Cybern.*, vol. 23 no. 3, pp. 665-685, 1993.
- [10] R.A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems," *Annals of Eugenics*, vol. 7, pp. 179-188, 1936.
- [11] D. Nauck, U. Nauck and R. Kruse, "Generating Classification Rules with the Neuro-Fuzzy System NEFCLASS," *Proc. Biennial Conf. North Am. Fuzzy Inf. Process. Soc. (NAFIPS)*, Berkeley, CA, 1996.