

Learning Maximal Structure Fuzzy Rules with Exceptions

P. Carmona

Dpto. Informática.
Universidad de Extremadura
pablo@unex.es

J.L. Castro

Dpto. C. Computación e I.A.
Universidad de Granada
castro@decsai.ugr.es

J.M. Zurita

Dpto. C. Computación e I.A.
Universidad de Granada
zurita@decsai.ugr.es

Abstract

This paper proposes a method to solve the conflicts that arise in the framework of fuzzy model identification with maximal rules [3]. This resolution is expressed including exceptions in the rules, that way achieving a higher model interpretability with respect to other techniques. Besides, several methods are presented to improve the interpretability, based on reducing the number of rules or exceptions of the model. Finally, the method is applied to an example.

Keywords: Fuzzy identification, interpretability, maximal rules.

1 Introduction

Fuzzy model identification [1, 5, 7] is noted for representing the model of a system from a set of examples by means of fuzzy rules. This model, which is a universal approximator [6, 2], enables to describe linguistically the relation between the input and the output of the system, thus taking care of the interpretability of the result.

In order to achieve a high interpretability, we must try to identify rules as general as possible, so that each rule covers the highest number of examples and, this way, the size of the rule base diminishes. Nevertheless, obtaining those general rules can provoke the appearance of conflicting zones where rules with a different consequent coexist, something that negatively affects the aforementioned interpretability.

In this paper, a strategy is proposed in order to solve these conflicts, making use of the information available from the examples in the conflicting zone. This solution takes the form of exceptions in the rules, diminishing the number of rules in the model and increasing its interpretability.

2 Learning Maximal Structure Fuzzy Rules

In [3], Castro et al. present a strategy to learn MISO systems ($\Omega : X^n \rightarrow Y$) from a set of examples $\theta = \{e_1, \dots, e_m\}$. Each example has the form $e_i = ((x_1^i, \dots, x_n^i), y^i)$, where x_j^i is the value of the j -th input variable and y^i is the output value of the system. The identified model will be represented by means of maximal rules of the form:

$$R^i: \text{if } X_1 \text{ is } A_1^i \text{ and } \dots \text{ and } X_n \text{ is } A_n^i \text{ then } Y \text{ is } LY^i \quad (1)$$

where each A_j^i is a set of labels associated disjunctively with the j -th input variable and taken from their respective fuzzy domain $DX_j = \{LX_{j,1}, \dots, LX_{j,t_j}\}$, and LY^i is the label associated with the output variable and taken from its fuzzy domain $DY = \{LY_1, \dots, LY_{t_{n+1}}\}$.

The learning algorithm proposed in [3] is:

1. To transform the examples in initial rules
2. For each initial rule:
 - 2.1. If the rule does not conflict with any definitive rule:
 - 2.1.1. For each label in each input variable:
 - 2.1.1.1. If the amplification of the rule is possible, amplify it.
 - 2.1.1.2. If the amplified rule does not subsume in any definitive rule, store it in the set of definitive rules.

In the step 2.1, we introduce a small modification trying to amplify initial rules that subsume in definitive rules (which was not done in the original algorithm), since the amplification could reach regions of the input space not covered with the existing definitive rules.

The translation from examples to initial rules consists in associating each value x_j^i and y^i with the label that present the highest membership degree out of all contained in its respective fuzzy domain. Amplifying a rule consists in adding a label to one of its input variables. An amplification from R^i to $R^{i'}$ is possible if $R^{i'}$ does not conflicts with any initial rule, i.e., if no initial rule R^j exists so that $A_k^j \in A_k^{i'}$, for all k , and $LY^j \neq LY^{i'}$.

3 Adding Exceptions to Fuzzy Rules

In the above algorithm, the search of maximal rules provokes that different consequents can co-exist in some input fuzzy regions. Next, a strategy is proposed to solve these conflicts.

During the learning process, the information contained in the examples is used only for the extraction of the initial rules. From that moment, the process of amplification of a rule to a certain input fuzzy region only verifies whether this region is or not occupied with an initial rule. Therefore, this process ignores the information that could be contained in the training set about such region. The basis that will support the approach proposed here to solve the conflicts consists in taking advantage of this information.

A compound rule of the form presented in (1) is equivalent to a conjunction of simple rules with one label associated to each input variable. Therefore, the set of simple rules involved in a conflict can be isolated in order to select one of them based on certain criterion. With that goal, a certainty degree for each simple rule involved in the conflict will be calculated from the number of positive and negative examples that each rule presents in the training set. An example of this type of measure is proposed in [4], where the concepts *positive example* and *negative example* are defined by means of fuzzy sets and where the certainty degree ranges from 0 to 1.

However, it must be noted that the main goal of

Table 1: A two-inputs/one-output fuzzy model

		X_1		
		N	Z	P
X_2	N	Z	Z	Z
	Z	Z	Z	P
	P	N	P	P

the amplification is for the amplified rules to be as general as possible. This way, the finally obtained consequents in an input subspace does not assure these consequents are the best, since their values are determined by initial rules that can be far away from the subspace under consideration.

Therefore, when solving a conflict, although it must be tried to restrict the selection of the best consequent to those involved in the conflicting rules in order to obtain maximal rules, it seems desirable to extend the space of selection if none of those rules has a sufficient degree of certainty. For that reason, a threshold μ will be established on the certainty degree in order to decide when the search of the best rule must be extended to all the possible rules for the conflicting region.

Once the best rule is selected, it is necessary to modify the rest of compound rules involved in the conflict. In this respect, when several rules with a different consequent have the highest certainty degree, the consequent with the highest number of occurrences will be selected, since it can exist more than one rule with the same consequent between the rules in conflict. This strategy tries to reduce the number of compound rules to be modified as much as possible.

The procedure to modify compound rules consists in the addition of exceptions. An exception is an n -tuple of labels $(LX_{1,i_1}, \dots, LX_{n,i_n})$ that defines the fuzzy region of the input subspace where the compound rule is not applied.

The use of exceptions entails an improvement in the model expressiveness with respect to the traditional description methods. This fact can be observed in the example in table 1, where a fuzzy model is shown and where the fuzzy domain of every variable is $\{N, Z, P\}$. The number of simple rules describing the model is $3 \times 3 = 9$ rules. A description using the usual technique that as-

sociates an input subspace with the same output (consequent) to the antecedent of each rule gives the following 5 fuzzy rules:

- R^1 : if X_2 is $\{N\}$ then Y is Z
- R^2 : if X_1 is $\{N, Z\}$ and X_2 is $\{Z\}$ then Y is Z
- R^3 : if X_1 is $\{P\}$ and X_2 is $\{Z\}$ then Y is P
- R^4 : if X_1 is $\{Z, P\}$ and X_2 is $\{P\}$ then Y is P
- R^5 : if X_1 is $\{N\}$ and X_2 is $\{P\}$ then Y is N

However, the same model can be described with only 3 rules using exceptions:

- R^1 : if X_2 is $\{N, Z\}$ then Y is Z
except if X_1 is P and X_2 is Z
- R^2 : if X_1 is $\{Z, P\}$ and X_2 is $\{Z, P\}$ then Y is P
except if X_1 is Z and X_2 is Z
- R^3 : if X_1 is $\{N\}$ and X_2 is $\{P\}$ then Y is N

It must be noticed that the addition of an exception to a compound rule will cancel it if this rule equals a simple rule.

Therefore, the proposed method to solve conflicts is finally described with the following algorithm:

1. For each fuzzy region of the input space where two or more consequents coexist:
 - 1.1. Work out the certainty degrees of the simple rules involved and select the highest (w_1).
 - 1.2. If w_1 reaches a threshold μ , go to step 1.5.
 - 1.3. Search on the rest of possible rules for one with a certainty degree higher than w_1 .
 - 1.4. If that rule exists, select it as the best rule (adding a new compound rule) and go to step 1.6.
 - 1.5. If there are more than one different rule with the highest certainty degree (w_1) between the conflicting rules, select the one appearing more times in the conflicting region. If all appear the same times, select one of them randomly.
 - 1.6. For each deleted simple rule, form the appropriate exception.
 - 1.6.1. If the exception cancels the associated compound rule, delete the rule.
 - 1.6.2. Otherwise, add the exception to the set of exceptions of the rule.

4 Improving the Interpretability

The model generated with the algorithm described in the previous section can still improve its interpretability in different ways. Next, several strategies for that goal are described.

4.1 Reducing Fuzzy Rules

The interpretability of the rules can increase if the exceptions of a rule are reduced by deleting labels from the antecedent. For example, the rule

- if X_1 is $\{N, P\}$ and X_2 is $\{N, Z, P\}$ then Y is Z
except if X_1 is $\{N\}$ and X_2 is $\{Z\}$
or X_1 is $\{P\}$ and X_2 is $\{Z\}$

could be reduced to the rule

- if X_1 is $\{N, P\}$ and X_2 is $\{N, P\}$ then Y is Z

The following algorithm decides if a reduction can be carried out after the addition of a new exception and, if that is the case, accomplishes it.

1. Given the rule $R^i : A_1^i, \dots, A_n^i \rightarrow LY^i$ with exceptions $E^i = \{E_1^i, \dots, E_p^i\}$, where the new exception added to that rule is $E_p^i = (LX_{1,p_1}, \dots, LX_{n,p_n})$.
2. For each d from 1 to n :
 - 2.1. Set up a set of exceptions E^* taking LX_{d,p_d} in the d -th element of every exception and taking the different combinations of the labels from $A_1^i, \dots, A_{d-1}^i, A_{d+1}^i, \dots, A_n^i$ in the rest of elements. That is, $E^* = A_1^i \times \dots \times A_{d-1}^i \times LX_{d,p_d} \times A_{d+1}^i \times \dots \times A_n^i$.
 - 2.2. If $E^* \subseteq E^i$ then set $E^i = E^i - E^*$ and $A_d^i = A_d^i - \{LX_{d,p_d}\}$.
 3. If the reduced rule subsumes in some other compound rule, delete it.

4.2 Merging Fuzzy Rules

In the algorithm presented in section 3, a rule is added to the set of definitive rules when the selected rule is not one of the conflicting rules (step 1.4). This can lead to a considerable increase in the number of rules with respect to the one obtained by the identification algorithm. In order to minimize this increase, after the addition of a new rule it must be tried to merge that rule with anyone of the existing compound rules.

Proposition 1 A rule $R^i : A_1^i, \dots, A_n^i \rightarrow LY^i$ with exceptions $E^i = \{E_1^i, \dots, E_{p_i}^i\}$ could be merged with another rule $R^j : A_1^j, \dots, A_n^j \rightarrow LY^j$ with exceptions $E^j = \{E_1^j, \dots, E_{p_j}^j\}$ if the following is fulfilled:

1. $LY^i = LY^j$.
2. There exists an r so that $A_r^i \neq A_r^j$.
3. $A_s^i = A_s^j$, for all $s \neq r$.

resulting a rule $R^*: A_1^i, \dots, A_r^i \cup A_r^j, \dots, A_n^i \rightarrow LY^i$ with exceptions $E^* = E^i \cup E^j$.

The merging of rules could be presented after the reduction of rules, since the merging condition could be satisfied if the rule loses a label in the antecedent along the reduction process. Because of that, it could be useful to try this merging in the step 3 of the algorithm presented in the previous subsection, once it has been verified that the rule does not subsume in other rules.

The following algorithm describes the method for merging rules:

1. Given the compound rule trying to be merged $R^i: A_1^i, \dots, A_n^i \rightarrow LY^i$ with exceptions $E^i = \{E_1^i, \dots, E_{p_i}^i\}$.
2. If there is another rule $R^j: A_1^j, \dots, A_n^j \rightarrow LY^j$ with exceptions $E^j = \{E_1^j, \dots, E_{p_j}^j\}$ in the set of definitive rules that can be merged with R^i :
 - 2.1. Replace the rules R^i and R^j by the rule $R^*: A_1^i, \dots, A_r^i \cup A_r^j, \dots, A_n^i \rightarrow LY^i$ with exceptions $E^* = E^i \cup E^j$.
 - 2.2. Try to merge R^* .

This is a recursive method, since the merged rule could satisfy the merging condition with respect to some other existing in the rule base.

4.3 Merging Exceptions

Until now, exceptions has been described as n -tuples of labels that define fuzzy regions in the input space similar to the ones defined by the antecedents of simple rules. Therefore, the exceptions expressed in that way can be considered *simple exceptions*.

Trying to increase the model interpretability, the concept of compound rule can be translated to the representation of exceptions, giving rise to *compound exceptions*. Thus, a compound exception can be defined as an n -tuple $E_i = (E_{i,1}, \dots, E_{i,n})$, where $E_{i,k} \subseteq DX_k$.

In order to obtain a description as compact as possible by means of exceptions, it is necessary to state a mechanism for merging exceptions in a similar way to that for merging rules explained in the previous subsection. Nevertheless, whereas the merger of rules runs on line (i.e., it is done during the process to solve conflicts), the merg-

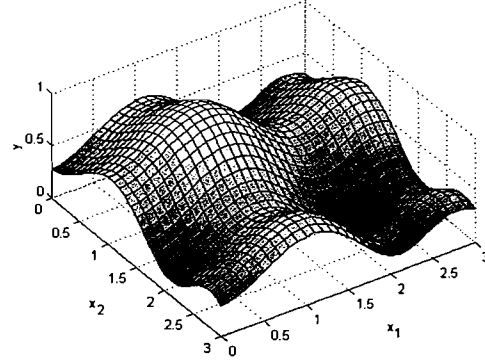


Figure 1: $y = (\sin(x_1^2) \cdot e^{-x_1} + \sin(x_2^2) \cdot e^{-x_2} + c_1) / c_2$

ing of exceptions will run off line (i.e., once the final exceptions of every rule have been obtained). This is due to the use of simple exceptions in the reduction of rules.

Proposition 2 An exception $E_i = (E_{i,1}, \dots, E_{i,n})$ could merge with another one $E_j = (E_{j,1}, \dots, E_{j,n})$ if the following is fulfilled:

1. There exists an r so that $E_{i,r} \neq E_{j,r}$.
2. $E_{i,s} = E_{j,s}$, for all $s \neq r$.

The result of the merging will be a new exception $E_* = (E_{i,1}, \dots, E_{i,r} \cup E_{j,r}, \dots, E_{i,n})$.

The following algorithm describes the method for merging exceptions:

1. Given the set of exceptions $E = \{E_1, \dots, E_p\}$ and the exception trying to be merged $E_i = (E_{i,1}, \dots, E_{i,n})$.
2. If there exists a $j \neq i$, so that it is possible to merge E_i and E_j :
 - 2.1. Replace the exceptions E_i and E_j by the exception $E_* = (E_{i,1}, \dots, E_{i,r} \cup E_{j,r}, \dots, E_{i,n})$.
 - 2.2. Try to merge E_* .

This recursive algorithm will be called iteratively for every rule while exception merging is possible.

5 Experimental Results

The proposed method was applied to the identification of the following system (figure 1):

$$f: [0, 3] \times [0, 3] \rightarrow [0, 1]$$

$$y = (\sin(x_1^2) \cdot e^{-x_1} + \sin(x_2^2) \cdot e^{-x_2} + c_1) / c_2$$

where $c_1 = 0.2338$ and $c_2 = 0.8567$ restricts the

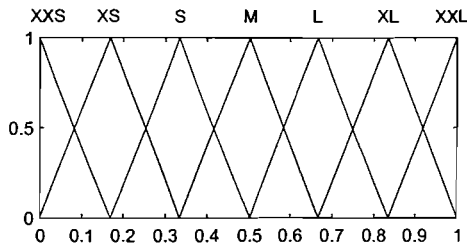


Figure 2: Fuzzy domains of x_1 , x_2 , and y .

Table 2: Comparative results

	Rules	MSE
Optimal model	49	0.0022
With conflicts	16	0.0132
Without conflicts	17	0.0093

function values to the interval $[0, 1]$.

The fuzzy domains of x_1 , x_2 and y were defined as shown in Figure 2, and scale factors $sf_{x_1} = 3$, $sf_{x_2} = 3$, and $sf_y = 1$ were used.

A set with 20 randomly generated examples was used in the identification process in order to analyse the generalization capacity of the method.

Table 2 shows the number of rules and the mean square error (MSE) for the optimal model, the model obtained with the identification process without resolution of conflicts ([3]), and the one obtained once the conflicts are solved with the method proposed here ($\mu = 0.5$). A significant improvement in the accuracy of the model can be observed. Figure 3 shows the rule base finally obtained together with the model surface it represents.

References

- [1] R. Babuska. *Fuzzy Modeling for Control*. Kluwer Academic Publishers, Boston, 1998.
- [2] J. Castro. Fuzzy logic controllers are universal approximators. *IEEE Transactions on System, Man and Cybernetics*, 25(4):629–635, 1995.
- [3] J. Castro, J. Castro-Schez, and J. Zurita. Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert sys-

```

R01: {XL}, {XL}->XXS
R02: {L}, {L}->XXS
R03: {M, XL}, {L}->XS
R04: {XL}, {XS}->XS
R05: {L}, {XXS, M}->XS
R06: {XL, XXL}, {XXS}->S
R07: {M, L, XXL}, {XS, XL}->S
     Exc: {M}, {XS}
R08: {XXS, XS, M, L, XXL}, {XL, XXL}->S
     Exc: {M, L, XXL}, {XXL}
R09: {XXS, XL, XXL}, {XS, S}->M
     Exc: {XL, XXL}, {XS}
R10: {XXS, XS, S, M, XXL}, {XXS, L}->M
     Exc: {S, XXL}, {XXS}
     Exc: {M}, {L}
R11: {XXS, L, XL, XXL}, {S}->M
R12: {XXS}, {M}->M
R13: {S, M, L, XL, XXL}, {XXL}->L
R14: {S}, {XXS, XL}->L
R15: {M, L, XL, XXL}, {M, XXL}->L
     Exc: {M, L}, {M}
R16: {XS, S, M}, {XS, S}->XL
R17: {XS, S, M}, {M}->XXL

```

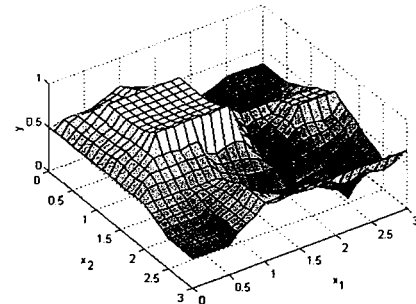


Figure 3: Final model without conflicts

tems. *Fuzzy Sets and Systems*, 101:331–342, 1999.

- [4] A. González and R. Pérez. SLAVE: A genetic learning system based on the iterative approach. *IEEE Transactions on Fuzzy Systems*, 7(2):176–191, 1999.
- [5] H. Hellendorn and D. Driankov, editors. *Fuzzy Model Identification*. Springer, Berlin, 1997.
- [6] B. Kosko. Fuzzy systems as universal approximators. *Proceedings of IEEE International Conference on Fuzzy Systems*, pages 1153–1162, 1992.
- [7] H. T. Nguyen and M. Sugeno. *Fuzzy Systems Modeling and Control*. Kluwer Academic Publishers, Boston, 1998.