Fuzzy Models and Interpolation

László T. Kóczy, János Botzheim and Tamás D. Gedeon

Abstract This paper focuses on two essential topics of the fuzzy area. The first is the reduction of fuzzy rule bases. The classical inference methods of fuzzy systems deal with dense rule bases where the universe of discourse is fully covered. By applying sparse or hierarchical rule bases the computational complexity can be decreased. The second subject of the paper is the introduction of some fuzzy rule base identification techniques. In some cases the fuzzy rule base might be given by a human expert, but usually there are only numerical patterns available and an automatic method has to be applied to determine the fuzzy rules.

1 Introduction

An overview of fuzzy model identification techniques and fuzzy rule interpolation is presented in this paper. In Sect. 2 the historical background of the fuzzy systems is described. Section 3 introduces the fuzzy rule base reduction methods, including fuzzy rule interpolation, hierarchical fuzzy rule bases, and the combination of these two methods. The next part of the paper deals with fuzzy model identification. In Sect. 4 clustering based fuzzy rule extraction is discussed, and in Sect. 5 our most recently proposed technique, the bacterial memetic algorithm is introduced.

László T. Kóczy
Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Hungary
Institute of Information Technology and Electrical Engineering, Széchenyi István University, Győr, Hungary

János Botzheim
Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Hungary
Department of Computer Science, The Australian National University

Tamás D. Gedeon
Department of Computer Science, The Australian National University

M. Nikravesh et al., (eds.), *Forging New Frontiers: Fuzzy Pioneers I.*

2 Background

Fuzzy rule based models were first proposed by Zadeh (1973) and later practically implemented with some technical innovations by Mamdani and Assilian (1975). If . . . then . . . rules contain an arbitrary number of variables x_i with antecedent descriptors, formulated either by linguistic terms or directly by convex and normal fuzzy sets (fuzzy numbers) in the If . . . part, and one or several output variables y_i and their corresponding consequent terms/membership functions in the then . . . part. An alternative model was proposed by Takagi and Sugeno (1985) where the consequent part was represented by $y = f(x)$ type functions. Later Sugeno and his team proposed the basic idea of hierarchical fuzzy models where the meta-level contained rules with symbolic outputs referring to further sub-rule bases on the lower levels. All these types of fuzzy rule based models might be interpreted as sophisticated and human friendly ways of defining (approximate) mappings from the input space to the output space, the rule bases being technically fuzzy relations of $X \times Y$ but representing virtual "fuzzy graphs" of extended (vague) functions in the state space.

A great advantage of all these models compared to more traditional symbolic rule models is the inherent interpolation property of the fuzzy sets providing a cover of the input space, with the kernel values of the antecedents corresponding to the areas (or points) where supposedly exact information is available – typical points of the graph, and vaguer "grey" areas bridging the gaps between these characteristic values by the partially overlapping membership functions or linguistic terms. It was however necessary in all initial types of fuzzy models that the antecedents formed a mathematically or semantically full cover, without any gaps between the neighboring terms. The "yellow tomato problem" proposed by Kóczy and Hirota pointed out that often it was not really necessary to have a fuzzy cover, since "yellow tomatoes" could be interpreted as something between "red tomatoes" and "green tomatoes", with properties lying also in between, the former category describing "ripe tomatoes", the latter "unripe tomatoes", yellow tomatoes being thus "half ripe". Linear fuzzy rule interpolation proposed in 1990 provided a new algorithm extending the methods of reasoning and control to sparse fuzzy covers where gaps between adjoining terms could be bridged with the help of this new approach, delivering the correct answer both in the case of linguistic rules like the "Case of the yellow tomato" and in more formal situations where membership functions were defined only by mathematical functions. This idea was later extended to many non-linear interpolation methods, some of them providing very practical means to deal with real life systems.

In 1993 and in a more advanced and practically applicable way in 1997 the authors proposed the extension of fuzzy interpolation to a philosophically completely new area, the sub-models of the hierarchical rule base itself. Interpolation of sub-models in different sub-spaces raised new mathematical problems. The solution was given by a projection based approach with local calculations and subsequent substitution into the meta-level rule base, replacing sub-model symbols by actual fuzzy sub-conclusions.

After an overview of these types of fuzzy models we will deal with the problem of model identification. Often human experts provide the approximate rules – like they did in the hierarchical fuzzy helicopter control application by Sugeno referred to above, but more often, there are only input-output data observed on a black box system as the starting point to construct the fuzzy rule structure and the rules themselves.

Sugeno and Yasukawa (1993) proposed the use of fuzzy c-means (FCM) clustering originally introduced by Bezdek (1981) for identifying output clusters in the data base, from where rules could be constructed by best fitting trapezoidal membership functions applied on the input projections of the output clusters. This method has limited applicability, partly because of the conditions of FCM clustering, and partly because of the need to have well structured behavior from the black box under observation.

In the last few years our group has introduced a variety of identification methods for various types of fuzzy models. The FCM based approach could be essentially extended to hierarchical models as well, and was applied to a real life problem (petroleum reservoir characterization (Gedeon et al., 1997; Gedeon et al., 2003)) successfully. Another was the bacterial evolutionary algorithm. The use of Levenberg-Marquardt (LM) optimization for finding the rule parameters was borrowed from the field of neural networks. These latter approaches all had their advantages and disadvantages, partly concerning convergence speed and accuracy, partly locality and globality of the optimum. The most recent results (Botzheim et al., 2005) showed that the combination of bacterial evolutionary algorithm and LM, called Bacterial Memetic Algorithm delivered very good results compared to the previous ones, with surprisingly good approximations even when using very simple fuzzy models. Research on extending this new method to more complicated fuzzy models is going on currently.

3 Fuzzy Rule Base Reduction

The classical approaches of fuzzy control deal with dense rule bases where the universe of discourse is fully covered by the antecedent fuzzy sets of the rule base in each dimension, thus for every input there is at least one activated rule. The main problem is the high computational complexity of these traditional approaches.

If a fuzzy model contains k variables and maximum T linguistic terms in each dimension, the order of the number of necessary rules is $O(T^k)$. This expression can be decreased either by decreasing T, or k, or both. The first method leads to sparse rule bases and rule interpolation, and was first introduced by Kóczy and Hirota (see e.g. Kóczy and Hirota, 1993a; Kóczy and Hirota, 1993b). The second, more effectively, aims to reduce the dimension of the sub-rule bases by using meta-levels or hierarchical fuzzy rule bases. The combination of these two methods leads to the decreasing of both T and k, and was introduced in (Kóczy and Hirota, 1993c).

3.1 Fuzzy Rule Interpolation

The rule bases containing gaps require completely different techniques of reasoning from the traditional ones. The idea of the first interpolation technique, the KH method, was proposed in (Kóczy and Hirota, 1993a; Kóczy and Hirota, 1993b), and considers the representation of a fuzzy set as the union of its α-cuts:

$$A = \bigcup_{\alpha \in [0,1]} \alpha A_\alpha \tag{1}$$

This method calculates the conclusion by its α-cuts. Theoretically all α-cuts should be considered, but for practical reasons only a finite set is taken into consideration during the computation. The KH rule interpolation algorithm requires the following conditions to be fulfilled: the fuzzy sets in both premises and consequences have to be convex and normal (CNF) sets, with bounded support, having continuous membership functions. Further, there should exist a partial ordering among the CNF sets of each variable. We shall use the vector representation of fuzzy sets, which assigns a vector of its characteristic points to every fuzzy set. The representation of the piecewise linear fuzzy set A will be denoted by vector $\mathrm{a} = [\mathrm{a}_{-\mathrm{m}}, \ldots, \mathrm{a}_0, \ldots, \mathrm{a}_\mathrm{n}]$, where $\mathrm{a_k}$ ($\mathrm{k} \in [-\mathrm{m}, \mathrm{n}]$) are the characteristic points of A and a_0 is the reference point of A having membership degree one. A partial ordering among CNF fuzzy sets (convex and normal fuzzy sets) is defined as: $A \prec B$ if $\mathrm{a_k} \leq \mathrm{b_k}$ ($\mathrm{k} \in [-\mathrm{m}, \mathrm{n}]$).

The basic idea of fuzzy rule interpolation (KH-interpolation) is formulated in the Fundamental Equation of Rule Interpolation (FERI):

$$D(A^*, A_1) : D(A^*, A_2) = D(B^*, B_1) : D(B^*, B_2) \tag{2}$$

In this equation A^* and B^* denote the observation and the corresponding conclusion, while $\mathrm{R}_1{=}\mathrm{A}_1 \rightarrow \mathrm{B}_1$, $\mathrm{R}_2{=}\mathrm{A}_2 \rightarrow \mathrm{B}_2$ are the rules to be interpolated, such that $A_1 \prec A^* \prec A_2$ and $B_1 \prec B_2$. If in some sense D denotes the Euclidean distance between two symbols, the solution for B^* results in simple linear interpolation. If $D = \tilde{d}$ (the fuzzy distance family), linear interpolation between corresponding α-cuts is performed and the generated conclusion can be computed as below, (as it is first described in (Kóczy and Hirota, 1993c)):

$$b_k^* = \frac{\dfrac{b_{1k}}{d(a_{1k}, a_k^*)} + \dfrac{b_{2k}}{d(a_{2k}, a_k^*)}}{\dfrac{1}{d(a_{1k}, a_k^*)} + \dfrac{1}{d(a_{2k}, a_k^*)}} \tag{3}$$

where the first index (*1* or *2*) represents the number of the rule, while the second (k) the corresponding α-cut. From now on $d(x, y) = |x - y|$, so that (3) becomes: ${}^{KH}b_k^* = (1 - \lambda_k)b_{1k} + \lambda_k b_{2k}$, where $\lambda_k = (a_k^* - a_{1k})/(a_{2k} - a_{1k})$ (for the left and right side, respectively).

This family of interpolation techniques has various advantageous properties, mainly simplicity, convenient practical applicability, stability, however, the direct applicability of this result is limited as the points defined by these equations often describe an abnormal membership function for B^* that needs further transformation to obtain a regular fuzzy set. Even then, the conclusion may need further transformation, since it may not be normal. Furthermore, the method does not conserve piecewise linearity of the rules and the observation. In order to avoid this difficulty, some alternative or modified interpolation algorithms were proposed, which solve the problem of abnormal solutions.

Some of these algorithms rely on the same idea of applying the Fundamental Equation of rule interpolation for some kind of metric, or in some cases non-metric dissimilarity degree. However, the way of measuring this distance or dissimilarity is varied. In (Gedeon and Kóczy, 1996), the distance is measured only for $\alpha=1$ and so the position and length of the core in each of the k input dimensions determine the position and the core of conclusion in the output space Y. The remaining information is contained in the shape of the flanks of the fuzzy sets describing the rules, (the parts where $0<\mu<1$). Depending on the type of the membership functions (trapezoidal or more general) a single value of fuzziness, F, is the basis for the calculation of the points of the flanks of B^*, where $F = |S_0\text{-}S_1|$, S is an arbitrary fuzzy set, the subscripts refer to the corresponding minimal or maximal points of the core and subscript, "core and support points", respectively. Alternatively, a function of α-dependent fuzziness values based on either the core or the support points is the base for this calculation. Here however a logarithmic type of dissimilarity is applied. These latter methods guarantee that the resulting conclusion is always "normal" (in the sense of not being abnormal).

In (Baranyi et al., 1999) a coordinate transformation of the data is used before applying the KH interpolation, which guarantees the normality of the conclusion obtained after interpolation.

3.2 Hierarchical Fuzzy Rule Bases

The basic idea of using hierarchical fuzzy rule bases is the following: Often the multi-dimensional input space $X = X_1 \times X_2 \times \ldots \times X_m$ can be decomposed, so that some of its components, e.g. $Z_0 = X_1 \times X_2 \times \ldots \times X_p$ determine a subspace of $X (p < m)$, so that in Z_0 a partition $\Pi=\{D_1, D_2, \ldots, D_n\}$ can be determined: $\bigcup_{i=1}^{n} D_i = Z_0$.

In each element of Π, i.e. D_i, a sub-rule base R_i can be constructed with local validity. In the worst case, each sub-rule base refers to exactly $X/Z_0 = X_{p+1} \times \ldots \times X_m$. The complexity of the whole rule base $O(T^m)$ is not decreased, as the size of R_0 is $O(T^p)$, and each $R_i, i > 0$, is of order $O(T^{m-p})$, $O(T^p) \times O(T^{m-p}) = O(T^m)$.

A way to decrease the complexity would be finding in each D_i a proper subset of $\{X_{p+1} \times \ldots \times X_m\}$, so that each R_i contains only less than $m - p$ input variables. In some concrete applications in each D_i

a proper subset of $\{X_{p+1}, \ldots, X_m\}$ can be found so that each R_i contains only less than $m-p$ input variables, and the rule base has the following structure:

R_0: **If** z_0 is D_1 **then** use R_1
If z_0 is D_2 **then** use R_2
...
If z_0 is D_n **then** use R_n

R_1: **If** z_1 is A_{11} **then** y is B_{11}
If z_1 is A_{12} **then** y is B_{12}
...
If z_1 is A_{1r1} **then** y is B_{1r1}

R_2: **If** z_2 is A_{21} **then** y is B_{21}
If z_2 is A_{22} **then** y is B_{22}
...
If z_2 is A_{2r2} **then** y is B_{2r2}

R_n: **If** z_n is A_{n1} **then** y is B_{n1}
If z_n is A_{n2} **then** y is B_{n2}
...
If z_n is A_{nrn} **then** y is B_{nrn}

where $z_i \in Z_i$, $Z_0 \times Z_i$ being a proper subspace of X for $i=1, .., n$.

If the number of variables in each Z_i is $k_i < m - p$ and $\max_{i=1}^{n} k_i = K < m - p$, then the resulting complexity will be $O(T^{p+K}) < O(T^m)$, so the structured rule base leads to a reduction of the complexity.

The task of finding such a partition is often difficult, if not impossible, (sometimes such a partition does not even exist). There are cases when, locally, some variables unambiguously dominate the behavior of the system, and consequently the omission of the other variables allows an acceptably accurate approximation. The bordering regions of the local domains might not be crisp or even worse, these domains overlap. For example, there can be a region D_1, where the proper subspace Z_1 dominates, and another region D_2, where another proper subspace Z_2 is sufficient for the description of the system, however, in the region between D_1 and D_2 all variables in $[Z_1 \times Z_2]$ play a significant role ($[.\times.]$ denoting the space that contains all variable that occur in either argument within the brackets). In this case, sparse fuzzy partitions can be used, so that in each element of the partition a proper subset of the remaining input state variables is identified as exclusively dominant. Such a sparse fuzzy partition can be described as follows: $\hat{\Pi}=\{D_1, D_2, \ldots, D_n\}$ and $\bigcup_{i=1}^{n} Core(D_i) \subset Z_0$ in the proper sense (fuzzy partition). Even $\bigcup_{i=1}^{n} Supp(D_i) \subset Z_0$ is possible (sparse partition). If the fuzzy partition chosen is informative enough concerning the behavior of the system, it is possible to interpolate its model among the elements of $\hat{\Pi}$.

Each element D_i will determine a sub-rule base R_i referring to another subset of variables. The technical difficulty is how to combine the "sub-conclusions" B_i^* with the help of R_0 into the final conclusion.

3.3 Fuzzy Rule Interpolation in Hierarchical Rule Bases

This section deals with reduction of both the maximum T linguistic terms, and the k variables.

Let us assume that the observation on X is A^* and its projections are:

$$A_0^* = A^*/Z_0,\ A_1^* = A^*/Z_1,\ A_2^* = A^*/Z_2.$$

Using the Fundamental Equation, the two sub-conclusions, obtained from the two sub-rule bases R_1 and R_2 are:

$$^{KH}b_k^{1*} = (1 - \lambda_k^1)b_{1k}^1 + \lambda_k^1 b_{2k}^1, \text{ and}$$
$$^{KH}b_k^{2*} = (1 - \lambda_k^2)b_{1k}^2 + \lambda_k^2 b_{2k}^2 \text{ respectively.}$$

(The superscript shows the reference to the rule base R_1 and R_2.)

Finally, by substituting the sub-conclusions into the meta-rule base we get:

$$^{KH}b_k^* = (1 - \lambda_k^0)b_k^{1*} + \lambda_k^0 b_k^{2*} \tag{4}$$

The steps of the algorithm are as follows:

1. Determine the projection $A_0{}^*$ of the observation A^* to the subspace of the fuzzy partition $\hat{\Pi}$.
2. Find the interpolating rules.
3. Determine λ_k^0.
4. For each R_i determine $A_i{}^*$ the projection of A^* to Z_i. Find the interpolating rules in each R_i.
5. Determine the sub-conclusions for each sub-rule base R_i.
6. Using the sub-conclusions from step 5, compute the final conclusion according to (4).

Figure 1 shows an example of the algorithm. In step 4, in each sub-rule base, a different inference engine can be applied, e.g., if the sub-rule base itself is dense, the Mamdani-algorithm (or one of its variations), or in any case one of the interpolation algorithms can be used. It might be reasonable to apply the KH interpolation or one of the methods summarized above in step 4, while in step 5 usually the best recommendation is the method in (Gedeon and Kóczy, 1996) as it is directly applicable also for flanking domains D_i with wide areas and projected observations $A_0{}^*$ with narrow supports, e.g. crisp singletons.

4 Fuzzy Model Identification

One of the crucial problems of fuzzy rule based modeling is how to find an optimal or at least a quasi-optimal rule base for a certain system. In most applications there is

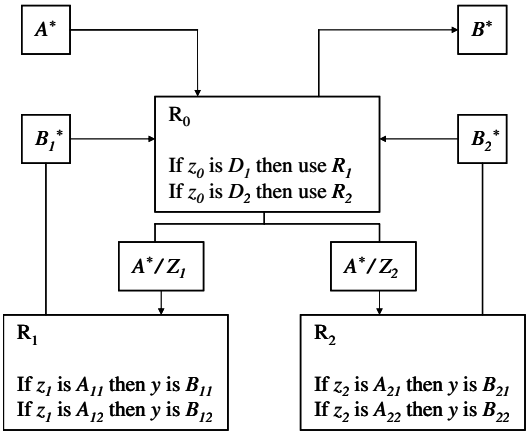

Fig. 1 Interpolation in a hierarchical model

no human expert available, thus some automatic method to determine the fuzzy rule base must be employed. In this section and in the next one some of these methods are introduced.

4.1 Clustering-based Rule Extraction Technique

Recently, clustering-based approaches have been proposed for rule extraction (Wong et al., 1997; Wang and Mendel, 1992). Most of the techniques use the idea of partitioning the input space into fixed regions to form the antecedents of the fuzzy rules. Although these techniques have the advantage of efficiency, they may lead to the creation of a dense rule base that suffers from rule explosion. In general, the number of rules generated is T^k where k is the number of input dimensions and T is the terms per input. In this case, the number of rules generated grows exponentially with the increase of input dimensions. Due to this reason, the techniques are not suited for generating fuzzy rule bases that have a large number of input dimensions.

Among the rule extraction techniques proposed in the literature, Sugeno and Yasukawa's (Sugeno and Yasukawa, 1993) technique (referred to as "SY method" hereafter) is one of the earliest work that emphasize the generation of a sparse rule base. The SY approach clusters only the output data and induces the rules by computing the projections to the input domains of the cylindrical extensions of the fuzzy clusters. This way, the method produces only the necessary number of rules for the input-output sample data (more details later). The paper (Sugeno and Yasukawa, 1993) discusses the proposed technique at the methodological level leaving out some implementation details. The SY technique was further examined in (Tikk et al., 2002) where additional readily implementable techniques are proposed to complete the modeling methodology.

In the first step of SY modeling, the Regularity criterion (Ihara, 1980) is used to assist in the identification of "true" input variables that have significant influence on the output. The input variables that have little or no influence on the output are ignored for the rest of the process. The true input variables are then used in the actual rule extraction process. The rule extraction process starts with the determination of the partition of the output space. This is done by using fuzzy c-means clustering (Bezdek, 1981) (see Sect. 4.2).

For each output fuzzy cluster B_i resulting from the fuzzy c-means clustering, a cluster in the input space A_i can be induced. The input cluster can be projected onto the various input dimensions to produce rules of the form:

If x_1 *is* A_{i1} *and* x_2 *is* A_{i2} *and* ... *and* x_n *is* A_{in} *then y is* B_i

4.2 Fuzzy c-Means Clustering

Given a set of data, Fuzzy c-Means clustering (FCMC) performs clustering by iteratively searching for a set of fuzzy partitions and the associated cluster centers that represent the structure of the data as best as possible. The FCMC algorithm relies on the user to specify the number of clusters present in the set of data to be clustered. Given the number of cluster c, FCMC partitions the data $X = \{x_1, x_2, \ldots, x_n\}$ into c fuzzy partitions by minimizing the within group sum of squared error objective function as follows (5).

$$J_m(U, V) = \sum_{k=1}^{n} \sum_{i=1}^{c} (U_{ik})^m \|x_k - v_i\|^2, \tag{5}$$
$$1 \leq m \leq \infty$$

where $J_m(U, V)$ is the sum of squared error for the set of fuzzy clusters represented by the membership matrix U, and the associated set of cluster centers V. $||.||$ is some inner product induced norm. In the formula, $||x_k - v_i||^2$ represents the distance between the data x_k and the cluster center v_i. The squared error is used as a performance index that measures the weighted sum of distances between cluster centers and elements in the corresponding fuzzy clusters. The number m governs the influence of membership grades in the performance index. The partition becomes fuzzier with increasing m and it has been shown that the FCMC algorithm converges for any $m \in (1, \infty)$. The necessary conditions for (5) to reach its minimum are

$$U_{ik} = \left(\sum_{j=1}^{c} \left(\frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^{2/(m-1)} \right)^{-1} \quad \forall i, \forall k \tag{6}$$

and

$$v_i = \frac{\sum_{k=1}^{n} (U_{ik})^m x_k}{\sum_{k=1}^{n} (U_{ik})^m} \tag{7}$$

In each iteration of the FCMC algorithm, matrix U is computed using (6) and the associated cluster centers are computed as (7). This is followed by computing the square error in (5). The algorithm stops when either the error is below a certain tolerance value or its improvement over the previous iteration is below a certain threshold.

The FCMC algorithm cannot be used in situations where the number of clusters in a set of data is not known in advance. Since the introduction of FCMC, a reasonable amount of work has been done on finding the optimal number of clusters in a set of data. This is referred to as the cluster validity problem. Among numerous alternative cluster validity indices proposed in the literature the most suitable proved to be the one proposed by Fukuyama and Sugeno (1989).

$$S(c) = \sum_{k=1}^{n} \sum_{i=1}^{c} (U_{ik})^m (\|x_k - v_i\|^2 - \|v_i - \bar{x}\|^2) \tag{8}$$
$$2 < c < n,$$

where n is the number of data points to be clustered; c is the number of clusters; x_k is the k^{th} data, $\overline{x}$ is the average of data; v_i is the i^{th} cluster center; U_{ik} is the membership degree of the k^{th} data with respect to the i^{th} cluster and m is the fuzzy exponent. The number of clusters, c, is determined so that $S(c)$ reaches a local minimum as c increases. The terms $||x_k - v_i||$ and $||v_i - \overline{x}||$ represent the variance in each cluster and variance between clusters respectively. Therefore, the optimal number of clusters is found by minimizing the distance among data and the corresponding cluster center and maximizes the distance among data in different clusters. Other cluster validity indices can be found in (Yang and Wu, 2001).

4.3 Hierarchical Fuzzy Modeling

Fuzzy clustering plays an important role in feature selection. The idea is that given a set of clusters, the most important subset of features (true inputs) can be selected by considering its capability to separate the clusters (Tikk and Gedeon, 2000; Pal, 1992). We use the interclass separability criterion for this purpose. Consider a set of N input-output pairs $F = \{X; y\}$, $X = \{x_i | i \in I\}$ where I is the index set, x_i and y are column vectors. By deleting some features (input variables), we obtain a subspace $X' = \{x_i | i \subset I\}$. Suppose that the input X is clustered into clusters $C_i (i = 1, \ldots, N_c)$ then the criterion function for feature ranking based on the interclass separability is formulated by means of the following fuzzy between-class (9) and within-class (11) scatter (covariance) matrices.

$$Q_b = \sum_{i=1}^{N_c} \sum_{j=1}^{N} \mu_{ij}^m (v_i - \bar{v})(v_i - \bar{v})^T \tag{9}$$

$$Q_i = \frac{1}{\sum_{j=1}^{N} \mu_{ij}^m} \sum_{j=1}^{N} \mu_{ij}^m (x_j - v_i)(x_j - v_i)^T \tag{10}$$

$$Q_w = \sum_{i=1}^{N_c} Q_i \tag{11}$$

where

$$\bar{v} = \frac{1}{N_c} \sum_{i=1}^{N_c} v_i \tag{12}$$

Here, v_i is given by (7). The criterion is a trade off between Q_b (9) and Q_w (11), often expressed as:

$$J(X') = \frac{tr\,(Q_b)}{tr\,(Q_w)} \tag{13}$$

where '*tr*' denotes the trace of a matrix. In Tikk and Gedeon (2000), the set of classes C is determined by clustering the output space using fuzzy clustering algorithms such as fuzzy c-means (Bezdek, 1981). The elements of the resulting partition matrix $U = \{\mu_{ik} | i = 1 \ldots N_c, k = 1 \ldots N\}$ are then used as weights (9–12). Each feature can be ranked using the sequential backwards algorithm. Firstly, different subsets of data are obtained by temporary deleting each feature. This is followed by deleting permanently the feature whose removal resulted in the largest value. This process is repeated until all features are deleted and the order of the deleted variables gives their rank of importance.

From here, we obtain a set of features ordered (ascending) by their importance. The set of true inputs can then be determined by selecting the n most important features that minimizes:

$$\sum_i \left\{ y_i - defuzz \left(\frac{\sum_c \mu_{ci}}{\left(\sum_{j=1}^{c} \left(\frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^{2/(m-1)} \right)} \right) \right\}^2 \tag{14}$$

where v_i is given by (7). Here, *defuzz*(.) denotes a defuzzification method, such as the center of area (COA), that is used in the fuzzy inference.

4.3.1 Finding Π and Z_0

The main requirement of a reasonable Π is that each of its elements D_i can be modelled by a rule base with local validity. In this case, it is reasonable to expect D_i to contain homogeneous data. The problem of finding Π can thus be reduced to finding homogeneous structures within the data. This can be achieved by clustering algorithms.

The subspace Z_0 is used by meta-rules to select the most appropriate sub-rule base to infer the output for a given observation (i.e. system input). In general, the more separable are the elements in Π, the easier the sub-rule base selection becomes. Therefore, the proper subspace can be determined by means of some criterion that ranks the importance of different subspaces (combination of variables) based on their capability in separating the components D_i.

Unfortunately, the problems of finding Π and Z_0 are not independent of each other. Consider the following helicopter control fuzzy rules extracted from the hierarchical fuzzy system in (Sugeno, 1991):

If *distance* (from obstacle) is *small* then *hover*
Hover:
    if (helicopter) *body* rolls *right* then
        move *lateral* stick *leftward*
    if (helicopter) *body* pitches *forward* then
        move *longitudinal* stick *backward*

On one hand, we need to rely on the feature selection technique to find a "good" subspace (Z_0) for the clustering algorithm to be effective. On the other hand, most of the feature selection criteria assume the existence of the clusters beforehand.

One way to tackle this problem is to adopt a projection based approach. The data are first projected to each individual dimension and fuzzy clustering is performed on the individual dimensions. The number of clusters C for clustering can be chosen arbitrarily large. With the set of one dimensional fuzzy clusters obtained, the separability (importance) of each input dimension can be determined separately by computing (13). With the ranking, we can then select the n most important features to form the n-dimensional subspace Z_0.

From here, a hierarchical fuzzy system can be obtained using the following steps:

1. Perform fuzzy c-means clustering on the data along the subspace Z_0. The optimal number of clusters, C, within the set of data is determined by means of the FS index (8).
2. The previous step results in a fuzzy partition $\Pi = \{D_1, \ldots D_C\}$. For each component in the partition, a meta rule is formed as:
   If Z_0 *is* D_i *then use* R_i
3. From the fuzzy partition Π, a crisp partition of the data points is constructed. I.e. for each fuzzy cluster D_i, the corresponding crisp cluster of points is determined as $P_i = \{p | \mu_i(p) > \mu_j(p) \forall j \neq i\}$.

4. Construct the sub-rule bases. For each crisp partition P_i, apply a feature extraction algorithm to eliminate unimportant features. The remaining features (know as true inputs) are then used by a fuzzy rule extraction algorithm to create the fuzzy rule base R_i. Here, we suggest the use of the projection-based fuzzy modeling approach (Chong et al., 2002). We remark however, that the hierarchical fuzzy modeling scheme does not place any restriction on the technique used. If more hierarchical levels are desired, repeat steps 1 – 4 with the data points in P_i using another subspace.

4.3.2 Modeling

Now, the proposed fuzzy modeling technique is described. The algorithm is as follows:

1. Rank the importance of input variables, using fuzzy clustering and the Interclass Separability Criterion and let F be the set of features ordered (ascending) by their importance
2. For $i = 1 \ldots |F|$

   a. Construct a hierarchical fuzzy system using the subspace $Z_0 = X_1 \times \ldots \times X_i$.
   b. Compute ε_i to be the re-substitution error of the fuzzy system constructed during the i^{th} iteration.
   c. If $i > 1$ and $\varepsilon_i > \varepsilon_{i-1}$, stop

   end for
3. Perform parameter tuning. The completed hierarchical fuzzy rule base then goes through a parameter identification process where the parameters of the membership function used in the fuzzy rules are adjusted on a trial and error basis to improve the overall performance. The method proposed in (Sugeno and Yasukawa, 1993) adjusts each of the trapezoidal fuzzy set parameters in both directions and chooses the one that gives the best system performance.

5 Bacterial Memetic Algorithm

There are various successful evolutionary optimization algorithms known from the literature. The advantage of these algorithms is their ability to solve and quasi-optimize problems with non-linear, high-dimensional, multi-modal, and discontinuous character. The original genetic algorithm was based on the process of evolution of biological organisms. These processes can be easily applied in optimization problems where one individual corresponds to one solution of the problem. A more recent evolutionary technique is called the bacterial evolutionary algorithm (Nawa and Furuhashi, 1999; Botzheim et al., 2002). This mimics microbial rather than eukaryotic evolution. Bacteria share chunks of their genes rather than perform neat crossover in chromosomes. This mechanism is used in the bacterial mutation

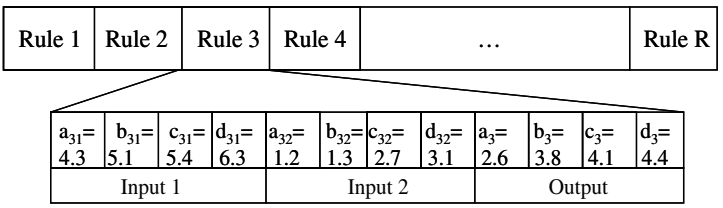

Fig. 2 Encoding of the fuzzy rules

and in the gene transfer operations. For the bacterial algorithm, the first step is to determine how the problem can be encoded in a bacterium (chromosome). Our task is to find the optimal fuzzy rule base for a pattern set. Thus, the parameters of the fuzzy rules must be encoded in the bacterium. The parameters of the rules are the breakpoints of the trapezoids, thus, a bacterium will contain these breakpoints. For example, the encoding method of a fuzzy system with two inputs and one output can be seen in Fig. 2.

Neural networks training algorithms can also be used for fuzzy rule optimization (Botzheim et al., 2004). They result in an accurate local optimum. Incorporating the neural network training algorithm with the bacterial approach, the advantages of both methods can be utilized in the optimization process. The hybridization of these two methods leads to a new kind of memetic algorithm, because the bacterial technique is used instead of the classical genetic algorithm, and the Levenberg-Marquardt as the local searcher. This method is the Bacterial Memetic Algorithm (BMA) (Botzheim et al., 2005). The flowchart of the algorithm can be seen in Fig. 3. The difference between the BEA and the BMA is that the latter contains a local searcher step, the Levenberg-Marquardt procedure.

5.1 Initial Population Creation

First the initial (random) bacteria population is created. The population consists of N_{ind} bacteria. This means that all membership functions in the bacteria must be randomly initialized. The initial number of rules in a single bacterium is initially constant N_{rule}. So, $N_{ind}(k+1)N_{rule}$ membership functions are created, where k is the number of input variables in the given problem and each membership function has four parameters.

5.2 Bacterial Mutation

The bacterial mutation is applied to each bacterium one by one (Nawa and Furuhashi, 1999). First, N_{clones} copies (clones) of the rule base are generated. Then a certain part of the bacterium (e.g. a rule) is randomly selected and the parameters

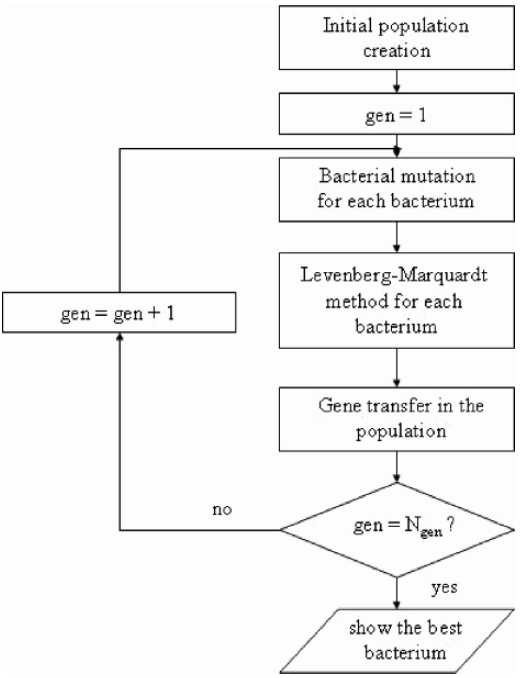

Fig. 3 Flowchart of the Bacterial Memetic Algorithm

of this selected part are randomly changed in each clone (mutation). Next all the clones and the original bacterium are evaluated by an error criterion. The best individual transfers the mutated part into the other individuals. This cycle is repeated for the remaining parts, until all parts of the bacterium have been mutated and tested. At the end the best rule base is kept and the remaining N_{clones} are discharged.

5.3 Levenberg-Marquardt Method

After the bacterial mutation step, the Levenberg-Marquardt algorithm (Marquardt, 1963) is applied for each bacterium. In this step a minimization criterion has to be employed also, which is related to the quality of the fitting. The training criterion that will be employed is the usual Sum-of-Square-of-Errors (SSE):

$$\Omega = \frac{\left\| \underline{t} - \underline{y} \right\|^2}{2} = \frac{\left\| \underline{e}\,[k] \right\|^2}{2} \tag{15}$$

where t stands for the target vector, y for the output vector, e for the error vector, and $\| \|$ denotes the 2-norm. It will be assumed that there are m patterns in the training set.

The most commonly used method to minimize (15) is the Error-Back-Propagation (BP) algorithm, which is a steepest descent algorithm. The BP algorithm is a first-order method as it only uses derivatives of the first order. If no line-search is used, then it has no guarantee of convergence and the convergence rate obtained is usually very slow. If a second-order method is to be employed, the best to use is the Levenberg-Marquardt (LM) algorithm (Marquardt, 1963), which explicitly exploits the underlying structure (sum-of-squares) of the optimization problem on hand.

Denoting by J the Jacobian matrix:

$$\underline{\underline{J}}[k] = \left[\frac{\partial y(\underline{x}^{(p)})[k]}{\partial \underline{par}[k]} \right] \tag{16}$$

where the vector $\underline{par}$ contains all membership functions' parameters (all breakpoints in the membership functions), and k is the iteration variable. The new parameter values can be obtained by the update rule of the LM method:

$$\underline{par}[k+1] = \underline{par}[k] - \left[\underline{\underline{J}}^T[k]\, \underline{\underline{J}}[k] + \alpha \underline{\underline{I}} \right]^{-1} \underline{\underline{J}}^T[k]\, \underline{e}[k] \tag{17}$$

In (17), α is a regularization parameter, which controls the both the search direction and the magnitude of the update. The search direction varies between the Gauss-Newton direction and the steepest direction, according to the value of α. This is dependent on how well the actual criterion agrees with a quadratic function, in a particular neighborhood. The good results presented by the LM method (compared with other second-order methods such as the quasi-Newton and conjugate gradient methods) are due to the explicit exploitation of the underlying characteristics of the optimization problem (a sum-of-square of errors) by the training algorithm. The Jacobian matrix with respect to the parameters in the bacterium must be computed. This can be done on a pattern by pattern basis (Botzheim et al., 2004; Ruano et al., 2001).

Jacobian computation

Because trapezoidal membership functions are used, and each trapezoid has four parameters, thus the relative importance of the j^{th} fuzzy variable in the i^{th} rule is:

$$\mu_{ij}\left(x_j\right) = \frac{x_j - a_{ij}}{b_{ij} - a_{ij}} N_{i,j,1}(x_j) + N_{i,j,2}(x_j) + \frac{d_{ij} - x_j}{d_{ij} - c_{ij}} N_{i,j,3}(x_j) \tag{18}$$

where $a_{ij} \leq b_{ij} \leq c_{ij} \leq d_{ij}$ must hold and:

$$N_{i,j,1}(x_j) = \begin{cases} 1, & if\ x_j \in [a_{ij},\ b_{ij}] \\ 0, & if\ x_j \notin [a_{ij},\ b_{ij}] \end{cases}$$
$$N_{i,j,2}(x_j) = \begin{cases} 1, & if\ x_j \in (b_{ij},\ c_{ij}) \\ 0, & if\ x_j \notin (b_{ij},\ c_{ij}) \end{cases} \tag{19}$$
$$N_{i,j,3}(x_j) = \begin{cases} 1, & if\ x_j \in [c_{ij}, d_{ij}] \\ 0, & if\ x_j \notin [c_{ij}, d_{ij}] \end{cases}$$

The activation degree of the i^{th} rule (the t-norm is the minimum):

$$w_i = \min_{j=1}^{n} \mu_{ij}(x_j) \tag{20}$$

where n is the number of input dimensions. w_i is the importance of the i^{th} rule if the input vector is $\mathbf{x}$ and $\mu_{i,j}(x_j)$ is the j^{th} membership function in the i^{th} rule. The i^{th} output is being cut in the height w_i. Then defuzzification method is calculated with the COG output:

$$y(\underline{x}) = \frac{\sum_{i=1}^{R} \int_{y \in \text{supp}\mu_i(y)} y\mu_i(y)dy}{\sum_{i=1}^{R} \int_{y \in \text{supp}\mu_i(y)} \mu_i(y)dy} \tag{21}$$

where R is the number of rules. If this defuzzification method is used, the integrals can be easily computed. In (21) $y(\underline{x})$ will be the following:

$$\begin{aligned} y(\underline{x}) &= \frac{1}{3} \frac{\sum_{i=1}^{R} (C_i + D_i + E_i)}{\sum_{i=1}^{R} 2w_i(d_i - a_i) + w_i^2(c_i + a_i - d_i - b_i)} \\ C_i &= 3w_i(d_i^2 - a_i^2)(1 - w_i) \\ D_i &= 3w_i^2(c_i d_i - a_i b_i) \\ E_i &= w_i^3(c_i - d_i + a_i - b_i)(c_i - d_i - a_i + b_i) \end{aligned} \tag{22}$$

Then, the Jacobian matrix can be written as:

$$\underline{\underline{J}} = \left[\frac{\partial y(\underline{x}^{(p)})}{\partial a_{11}} \quad \frac{\partial y(\underline{x}^{(p)})}{\partial b_{11}} \cdots \frac{\partial y(\underline{x}^{(p)})}{\partial a_{12}} \cdots \frac{\partial y(\underline{x}^{(p)})}{\partial d_1} \cdots \frac{\partial y(\underline{x}^{(p)})}{\partial d_R} \right] \tag{23}$$

where

$$\begin{aligned}
\frac{\partial y(\underline{x}^{(p)})}{\partial a_{ij}} &= \frac{\partial y}{\partial w_i}\frac{\partial w_i}{\partial \mu_{ij}}\frac{\partial \mu_{ij}}{\partial a_{ij}} \\
\frac{\partial y(\underline{x}^{(p)})}{\partial b_{ij}} &= \frac{\partial y}{\partial w_i}\frac{\partial w_i}{\partial \mu_{ij}}\frac{\partial \mu_{ij}}{\partial b_{ij}} \\
\frac{\partial y(\underline{x}^{(p)})}{\partial c_{ij}} &= \frac{\partial y}{\partial w_i}\frac{\partial w_i}{\partial \mu_{ij}}\frac{\partial \mu_{ij}}{\partial c_{ij}} \\
\frac{\partial y(\underline{x}^{(p)})}{\partial d_{ij}} &= \frac{\partial y}{\partial w_i}\frac{\partial w_i}{\partial \mu_{ij}}\frac{\partial \mu_{ij}}{\partial d_{ij}}
\end{aligned} \tag{24}$$

From (20), w_i depends on the membership functions, and each membership function depends only on four parameters (breakpoints). So, the derivatives of w_i are

$$\frac{\partial w_i}{\partial \mu_{ij}} = \begin{cases} 1, & if\ \mu_{ij} = \min\limits_{k=1}^{n} \mu_{ik} \\ 0, & otherwise \end{cases} \tag{25}$$

and the derivatives of the membership functions will be

$$\begin{aligned}
\frac{\partial \mu_{ij}}{\partial a_{ij}} &= \frac{x_j^{(p)} - b_{ij}}{(b_{ij} - a_{ij})^2} N_{i,j,1}(x_j^{(p)}) \\
\frac{\partial \mu_{ij}}{\partial b_{ij}} &= \frac{a_{ij} - x_j^{(p)}}{(b_{ij} - a_{ij})^2} N_{i,j,1}(x_j^{(p)}) \\
\frac{\partial \mu_{ij}}{\partial c_{ij}} &= \frac{d_{ij} - x_j^{(p)}}{(d_{ij} - c_{ij})^2} N_{i,j,3}(x_j^{(p)}) \\
\frac{\partial \mu_{ij}}{\partial d_{ij}} &= \frac{x_j^{(p)} - c_{ij}}{(d_{ij} - c_{ij})^2} N_{i,j,3}(x_j^{(p)})
\end{aligned} \tag{26}$$

$\frac{\partial y}{\partial w_i}$ and the derivatives of the output membership functions parameters have to be also computed. From (22):

$$\frac{\partial y}{\partial *_{i^*}} = \frac{1}{3}\frac{den\frac{\partial F_{i^*}}{\partial *_{i^*}} - num\frac{\partial G_{i^*}}{\partial *_{i^*}}}{(den)^2} \tag{27}$$

where $*_i = w_i, a_i, b_i, c_i, d_i$, den and num are the denominator and the numerator of (22), respectively F_{i^*} and G_{i^*} are the i^* member of the sum in the numerator and the denominator. The derivatives will be as follows:

$$\frac{\partial F_i}{\partial w_i} = 3(d_i^2 - a_i^2)(1 - 2w_i) + 6w_i(c_i d_i - a_i b_i) + 3w_i^2 [\ (c_i - d_i)^2 - (a_i - b_i)^2\ ] \tag{28}$$

$$\frac{\partial G_i}{\partial w_i} = 2(d_i - a_i) + 2w_i(c_i + a_i - d_i - b_i)$$

$$\frac{\partial F_i}{\partial a_i} = -6w_i a_i + 6w_i^2 a_i - 3w_i^2 b_i - 2w_i^3(a_i - b_i) \tag{29}$$

$$\frac{\partial G_i}{\partial a_i} = -2w_i + w_i^2$$

$$\frac{\partial F_i}{\partial b_i} = -3w_i^2 a_i + 2w_i^3(a_i - b_i)$$

$$\frac{\partial G_i}{\partial b_i} = -w_i^2$$

$$\frac{\partial F_i}{\partial c_i} = 3w_i^2 d_i - 2w_i^3(d_i - c_i)$$

$$\frac{\partial G_i}{\partial c_i} = w_i^2$$

$$\frac{\partial F_i}{\partial d_i} = 6w_i d_i - 6w_i^2 d_i + 3w_i^2 c_i + 2w_i^3(d_i - c_i)$$

$$\frac{\partial G_i}{\partial d_i} = 2w_i - w_i^2$$

5.4 Gene Transfer

The gene transfer operation allows the recombination of genetic information between two bacteria. First the population must be divided into two halves. The better bacteria are called the superior half, the other bacteria are called the inferior half. One bacterium is randomly chosen from the superior half, this will be the source bacterium, and another is randomly chosen from the inferior half, this will be the destination bacterium. A part (e.g. a rule) from the source bacterium is chosen and this part will overwrite a rule of the destination bacterium. This cycle is repeated for N_{inf} times, where N_{inf} is the number of "infections" per generation.

5.5 Stop Condition

If the population satisfies a stop condition or the maximum number of generation N_{gen} is reached then the algorithm ends, otherwise it returns to the bacterial mutation step.

6 Conclusion

Fuzzy rule base reduction methods and fuzzy modeling techniques were introduced in this paper. We discussed how the complexity of a fuzzy rule base can be reduced. The classical fuzzy models deal with dense rule bases where the universe of discourse is fully covered. The reduction of the number of linguistic terms in each dimension leads to sparse rule bases and rule interpolation. By decreasing the dimension of the sub-rule bases by using meta-levels or hierarchical fuzzy rule bases the complexity can also be reduced. It is also possible to apply interpolation in hierarchical rule bases.

The paper discussed automatic methods to determine the fuzzy rule base. After an overview of some clustering methods, the bacterial memetic algorithm was introduced. This approach combines the bacterial evolutionary algorithm and a gradient based learning method, namely the Levenberg-Marquardt procedure. By this combination the advantages of both methods can be exploited.

Acknowledgments Supported by the Széchenyi University Main Research Direction Grant 2005, a National Scientific Research Fund Grant OTKA T048832 and the Australian Research Council.

Special acknowledgements to Prof. Antonio Ruano and Cristiano Cabrita (University of Algarve, Faro, Portugal) and to Dr. Alex Chong (formerly with Murdoch University, Perth, Australia).

References

P. Baranyi, D. Tikk, Y. Yam, and L. T. Kóczy: "lnvestigation of a new alpha-cut Based Fuzzy lnterpolation Method", Tech. Rep., Dept. Of Mechanical and Automation Engineering, The Chinese University of Hong Kong, 1999.

J. C. Bezdek, Recognition with Fuzzy Objective Function Algorithms, New York: Plenum Press, 1981.

J. Botzheim, B. Hámori, L. T. Kóczy, and A. E. Ruano, "Bacterial algorithm applied for fuzzy rule extraction", in Proceedings of the International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems, IPMU 2002, pp. 1021–1026, Annecy, France, 2002.

J. Botzheim, C. Cabrita, L. T. Kóczy, and A. E. Ruano, "Estimating Fuzzy Membership Functions Parameters by the Levenberg-Marquardt Algorithm", in Proceedings of the IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2004, pp. 1667–1672, Budapest, Hungary, 2004.

J. Botzheim, C. Cabrita, L. T. Kóczy, and A. E. Ruano, "Fuzzy rule extraction by bacterial memetic algorithm", in Proceedings of the 11th World Congress of International Fuzzy Systems Association, IFSA 2005, pp. 1563–1568, Beijing, China, 2005.

A. Chong, T. D. Gedeon, and L. T. Kóczy, "Projection Based Method for Sparse Fuzzy System Generation", in Proceedings of 2nd WSEAS International Conference on Scientific Computation and Soft Computing, pp. 321–325, Crete, 2002.

Y. Fukuyama and M. Sugeno, "A new method of choosing the number of clusters for fuzzy c-means method" in Proceedings of the 5th Fuzzy System Symposium, 1989.

T. D. Gedeon and L. T. Kóczy, "Conservation of fuzziness in rule interpolation", Intelligent Technologies, vol. 1 International Symposium on New Trends in Control of Large Scale Systems, Herlany, 13–19, 1996.

T. D. Gedeon, P. M. Wong, Y. Huang, and C. Chan, "Two Dimensional Fuzzy-Neural Interpolation for Spatial Data" in Proceedings Geoinformatics'97: Mapping the Future of the Asia-Pacific, Taiwan, vol. 1: 159–166, 1997.

T. D. Gedeon, K. W. Wong, P. M. Wong, and Y. Huang, "Spatial Interpolation Using Fuzzy Reasoning", Transactions on Geographic Information Systems, 7(1), pp. 55–66, 2003.
J. Ihara, "Group method of data handling towards a modelling of complex systems – IV" Systems and Control (in Japanese), 24, pp. 158–168, 1980.
L. T. Kóczy and K. Hirota, "Approximate reasoning by linear rule interpolation and general approximation" International Journal of Approximate Reasoning, 9, pp. 197–225 (1993a).
L. T. Kóczy and K. Hirota, "Interpolative reasoning with insufficient evidence in sparse fuzzy rule bases" Information Sciences, 71, pp. 169–201 (1993b).
L. T. Kóczy and K. Hirota, "Interpolation in structured fuzzy rule bases" in Proceedings of the IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'93, pp. 803-808, San Francisco (1993c).
E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," Int. J. Man-Mach. Stud., vol. 7, pp. 1–13, 1975.
D. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters", J. Soc. Indust. Appl. Math., 11, pp. 431–441, 1963.
N. E. Nawa and T. Furuhashi, "Fuzzy System Parameters Discovery by Bacterial Evolutionary Algorithm", IEEE Tr. Fuzzy Systems vol. 7, pp. 608–616, 1999.
S. K. Pal, "Fuzzy set theoretic measure for automatic feature evaluation – II" Information Sciences, pp. 165–179, 1992.
A. E. Ruano, C. Cabrita, J. V. Oliveira, L. T. Kóczy, and D. Tikk, "Supervised Training Algorithms for B-Spline Neural Networks and Fuzzy Systems", in Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference, Vancouver, Canada, 2001.
M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modelling", IEEE Transactions on Fuzzy Systems, 1(1): pp. 7–31, 1993.
M. Sugeno, T. Murofushi, J. Nishino, and H. Miwa, "Helicopter flight control based on fuzzy logic", in Proceedings of IFES'91, Yokohama, 1991.
T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control", IEEE Trans. Syst., Man, Cybern., vol. SMC-15, pp. 116–132, 1985.
D. Tikk and T. D. Gedeon, "Feature ranking based on interclass separability for fuzzy control application", in Proceedings of the International Conference on Artificial Intelligence in Science and Technology (AISAT'2000) pp. 29–32, Hobart, 2000.
D. Tikk, Gy. Biró, T. D. Gedeon, L. T. Kóczy, and J. D. Yang, "Improvements and Critique on Sugeno's and Yasukawa's Qualitative Modeling", IEEE Tr. on Fuzzy Systems, vol.10. no.5., October 2002.
L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples" IEEE Transactions on Systems, Man, and Cybernetics, 22(6): pp. 1414–1427, 1992.
K. W. Wong, C. C. Fung, and P. M. Wong, "A self-generating fuzzy rules inference systems for petrophysical properties prediction" in Proceedings of IEEE International Conference on Intelligent Processing Systems, Beijing, 1997.
M. S. Yang and K. L. Wu, "A New Validity Index For Fuzzy Clustering" in Proceedings of the IEEE International Conference on Fuzzy Systems, pp. 89–92, Melbourne, 2001.
L. A. Zadeh, "Outline of a new approach to the analysis of complex systems," IEEE Trans. Syst., Man, Cybern., vol. SMC-1, pp. 28–44, 1973.