GA-Fuzzy Modeling and Classification: Complexity and Performance

Magne Setnes and Hans Roubos

***Abstract*—The use of genetic algorithms (GAs) and other evolutionary optimization methods to design fuzzy rules for systems modeling and data classification have received much attention in recent literature. Authors have focused on various aspects of these randomized techniques, and a whole scale of algorithms have been proposed. We comment on some recent work and describe a new and efficient two-step approach that leads to good results for function approximation, dynamic systems modeling and data classification problems. First fuzzy clustering is applied to obtain a compact initial rule-based model. Then this model is optimized by a real-coded GA subjected to constraints that maintain the semantic properties of the rules. We consider four examples from the literature: a synthetic nonlinear dynamic systems model, the iris data classification problem, the wine data classification problem, and the dynamic modeling of a diesel engine turbocharger. The obtained results are compared to other recently proposed methods.**

***Index Terms*—Classification, dynamic systems, fuzzy clustering, real-coded genetic algorithm (GA), Takagi–Sugeno–Kang (TSK) fuzzy model.**

I. Introduction

We focus on the problem of obtaining a compact and accurate fuzzy rule-based model from observation data. Frequently, in data-driven fuzzy modeling approaches, the Takagi–Sugeno–Kang (TSK)-type fuzzy model is used [1]. The typical identification of the TSK model is done in two steps. First, the fuzzy rule antecedents are determined; then, least squares parameter estimation is applied to determine the consequents. This approach is suboptimal and many methods have been proposed to simultaneously determine all parameters of the model. Genetic algorithms (GAs) is one such technique that has received a lot of attention in recent literature, owing its popularity to the possibility of searching irregular and high-dimensional solution spaces.

GAs have been applied to learn both the antecedent and consequent part of fuzzy rules and models with both fixed and varying number of rules have been considered [2]–[4]. Also, GAs have been combined with other techniques like fuzzy clustering [5], [6], neural networks [7], [8], statistical information criteria [9], Kalman filters [9], hill climbing [8] and even fuzzy expert control of the GAs operators [10], to mention some. This has resulted in many complex algorithms and, as recognized in [11] and [12], often the transparency and compactness of the resulting rule base is not considered to be of importance. In such cases, the fuzzy model becomes a black box and one can question the rationale for applying fuzzy modeling instead of other techniques like, e.g., neural networks.

In the following, we propose a new and efficient two-step approach to the construction of fuzzy rules from data that combines good approximation or classification properties with compactness and transparency. First, fuzzy clustering is applied to obtain an initial rule-based model focusing on compactness and transparency. In the second step, the performance of the initial rule base is optimized by a real-coded GA allowing for simultaneous optimization of both the rule antecedents and the consequents. To maintain the transparency properties of the initial rule base, the GA is subjected to constraints that limits the search space to the neighborhood of the initial rule base. The novelty of the approach is the combination of fuzzy clustering for generating an interpretable initial rule base and the constrained GA for optimizing the performance while maintaining this interpretability. We also show that the GA can be combined with rule base simplification tools. The performance degradation caused by simplification is then to a large extent corrected by the GA.

Section II explains the initial modeling and discusses transparency and accuracy issues. The GA-based optimization is then described in Section III. Section IV considers four examples known from the literature: a synthetic nonlinear dynamic system, the iris data and the wine data classification problems, and the modeling of a real-world truck diesel engine turbocharger. Less complex rule bases than those reported in literature are obtained with comparable or better accuracy. Finally, Section V concludes the paper.

II. Initial Fuzzy Model

In the TSK fuzzy model [1] the rule consequents are usually constant values (singletons) or linear functions of the inputs

$$R_i\text{: } \mathbf{If}\ x_1 \text{ is } A_{i1} \ \mathbf{and}\ \ldots x_n \text{ is } A_{in} \quad \mathbf{then}\ g_i = p_{i1}x_1 + \cdots, p_{\text{in}}x_n + p_{i(n+1)} \qquad i = 1, \ldots, M. \quad (1)$$

Here $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ is the input vector, g_i is the output of the ith rule, and $A_{i1}, \ldots, A_{\text{in}}$ are the antecedent fuzzy sets. The model output is computed by aggregating the individual rules contributions

$$y = \frac{\sum_{i=1}^{M} \beta_i g_i}{\sum_{i=1}^{M} \beta_i} \quad (2)$$

Manuscript received July 26, 1999; revised February 10, 2000. This work was supported in part by the Research Council of Norway.

The authors are with the Delft University of Technology, Faculty of Information Technology and Systems, Control Laboratory, 2600 GA Delft, The Netherlands (e-mail: magne@ieee.org; hans@ieee.org).

Publisher Item Identifier S 1063-6706(00)08463-0.

where β_i is the degree of activation of the ith rule

$$\beta_i = \prod_{j=1}^{n} A_{ij}(x_j), \quad i = 1, 2, \ldots, M. \tag{3}$$

A. Data-Driven Identification

From data, an initial fuzzy rule base is derived in two steps. First, the fuzzy antecedents A_{ij} are determined by means of fuzzy clustering [13]. Then, with the premise fixed, the rule consequents are determined by least squares parameter estimation [14]. For clustering, a regression matrix $\mathbf{X}^T = [\mathbf{x}_1, \ldots, \mathbf{x}_K]$ and an output vector $\mathbf{y}^T = [y_1, \ldots, y_K]$ are constructed from the available data. Note that the number of used inputs (features) is important for the transparency of the resulting model. However, we do not explicitly deal with feature selection in this paper. Assuming that a proper data collection has been done, clustering takes place in the product space of $\mathbf{X}$ and $\mathbf{y}$ to identify regions where the system can be locally approximated by TSK rules. Various cluster algorithms exist, differing mainly in the shape or size of the cluster prototypes applied. To illustrate the proposed modeling approach, we apply the fuzzy c-means algorithm [15].

Given the data $\mathbf{Z}^T = [\mathbf{X}, \mathbf{y}]$ the cluster algorithm computes the fuzzy partition matrix $\mathbf{U}$ whose ikth element $\mu_{ik} \in [0, 1]$ is the membership degree of the data object $\mathbf{z}_k \in \mathbf{Z}$, in cluster i. The rows of $\mathbf{U}$ are thus multidimensional fuzzy sets (clusters) represented pointwise. Univariate fuzzy sets A_{ij} are obtained by projecting the rows of $\mathbf{U}$ onto the input variables x_j and approximate the projections by parametric functions [14]. To illustrate our approach, we apply triangular membership functions

$$\mu(x; a, b, c) = \max\left(0, \min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right)\right). \tag{4}$$

If more smooth membership functions are used, e.g., Gaussian or exponential functions, the resulting model will in general have a higher accuracy in fitting the training data.

Once the antecedent membership functions have been fixed, the consequent parameters p_{iq} of each individual rule are obtained as a local least squares estimate. Let $\theta_i^T = [p_{i1}, \ldots, p_{in}, p_{i(n+1)}]$, let $\mathbf{X}_e$ denote the matrix $[\mathbf{X1}]$ with rows $[\mathbf{x}_k, 1]$, and let $\mathbf{W}_i$ denote a diagonal matrix in $\mathbb{R}^{K \times K}$ having the degree of activation, $\beta_i(\mathbf{x}_k)$ as its kth diagonal element. The consequents of the ith rule is the weighted least squares solution of $\mathbf{y} = \mathbf{X}_e\theta_i + \epsilon$, given by

$$\theta_i = \left[\mathbf{X}_e^T \mathbf{W}_i \mathbf{X}_e\right]^{-1} \mathbf{X}_e^T \mathbf{W}_i \mathbf{y}. \tag{5}$$

B. Interpreting TSK Models

The rule consequents θ_i can also be obtained as a global least squares estimate that considers the contribution of all rules simultaneously. This gives a better approximation of the data, but usually hampers the interpretation of the rules. In [16], it was proposed to combine local and global learning. In our approach, we initialize the GA-based optimization with a local learned TSK model and constrain the global optimization in order to maintain the local interpretability. Unlike in the approach in [16], we consider also the parameters of the antecedent part in the global optimization.

In Section IV we apply TSK models both for modeling dynamic systems and for classification. In systems identification and control engineering, rules with functional consequents are interpreted as local linear models. This offers a means of inspecting and analyzing dynamic systems in terms of, e.g., stability issues and controller design. Few rules and a transparent partitioning helps to make such an analysis more easy.

For classification, we apply singleton consequent TSK models. The rule consequents need not necessarily be the same as the class labels because of the interpolation between the rules. The TSK model output is continuous and an additional step is necessary in order to arrive at a class label. We will show that interpretable rules can still be obtained when the consequent values are close to the actual class labels.

C. Transparency and Accuracy of Fuzzy Models

Fixed membership functions taken to represent predefined linguistic terms are often used to partition the rule base premise. Membership functions derived from the data, however, explain the data-patterns in a better way and typically less sets and less rules result than in the case of a fixed partition approach. If the membership functions derived from data have simple shapes and are well separated, they can still be assigned meaningful linguistic labels by the domain experts.

The initial rule base constructed by fuzzy clustering typically fulfills many criteria for transparency and good semantic properties [11]:

- *Moderate Number of Rules*: Fuzzy clustering helps ensuring a comprehensive sized rule base with rules that describe important regions in the data.
- *Distinguishability*: A low number of clusters leads to distinguishable rules and membership functions.
- *Normality*: By fitting parameterized functions to the projected clusters, normal, and comprehensive membership functions are obtained that can be taken to represent linguistic terms.
- *Coverage*: The deliberate overlap of the clusters (rules) and their position in populated regions of the input–output data space ensure that the model is able to derive an output for all occurring inputs.

The transparency and compactness of the rule base can be further improved by methods like rule reduction [17] or rule-base simplification [18]. (In some examples in Section IV we will apply similarity-driven simplification and this method is briefly described in Section II-D). The approximation capability of the rule base, however, remains suboptimal. The projection of the clusters onto the input variables and their approximation by parametric functions like triangular fuzzy sets, introduces a structural error since the resulting premise partition differs from the cluster partition matrix. Moreover, the separate identification of the rule antecedents and the rule consequents prohibits interactions between them during modeling. To improve the approximation capability of the rule

base, we apply a GA-based optimization method as described in Section III.

D. Rule Base Simplification

The similarity-driven rule base simplification algorithm [18] uses a similarity measure to detect compatible fuzzy sets in a rule base

$$S(A_{lj}, A_{mj}) = \frac{|A_{lj} \cap A_{mj}|}{|A_{lj} \cup A_{mj}|}. \tag{6}$$

Here $|\cdot|$ denotes fuzzy set cardinality and the $\cap$ and $\cup$ operators represent the intersection and union, respectively. $S(l,m) = 1$ when the membership functions A_{lj} and A_{mj} are equal and $S(l,m) = 0$ when the membership functions are nonoverlapping.

The algorithm iteratively evaluates the rule base and the most similar pair of fuzzy sets are merged in order to obtain a more general concept that replaces the occurrences of the similar ones in the rule base. The algorithm terminates when there are no more similar sets in the rule base. This reduces the number of different fuzzy sets (linguistic terms) used in the model. The similarity measure is also used to detect "do not care" terms that can be removed from the rule base. Similarity-driven simplification differ from rule reduction in that the models term set can be reduced without necessarily any rules being removed. Rule reduction occurs when two or more rules get an equal premise.

III. Genetic Algorithms

GAs are gradient free parallel-optimization algorithms that use a performance criterion for evaluation and a population of possible solutions to search for a global optimum. These structured random search techniques are capable of handling complex and irregular solution spaces [19]. GAs are inspired by the biological process of Darwinian evolution where selection, mutation, and crossover play a major role. Good solutions are selected and manipulated to achieve new and possibly better solutions. The manipulation is done by the *genetic operators* that work on the *chromosomes* in which the parameters of possible solutions are encoded. In each generation of the GA, the new solutions replace the solutions in the population that are selected for deletion.

We consider real-coded GAs [19]. Binary coded or classical GAs [20] are less efficient when applied to multidimensional, high-precision or continuous problems. The bitstrings can become very long and the search space blows up. Furthermore, central processing unit (CPU) time is lost to the conversion between the binary and real representation. Other alphabets like the real coding can be favorably applied to variables in the continuous domain. In real-coded GAs, the variables appear directly in the chromosome and are modified by special genetic operators. Various real-coded GAs were recently reviewed in [21]. The main aspects of the proposed GA are discussed below and the implementation is summarized in Section III-E.

A. Fuzzy Model Representation

Chromosomes are used to describe the solutions in a GA. The initialization step by c-means clustering was introduced to allow us to create chromosomes with a similar coding, i.e., each element of each chromosome codes for the same variable. This is necessary for efficient utilization of the crossover operators. The initialization also allows us to limit the search space. This makes the optimization problem more efficient.

The chromosome representation determines the GA structure. With a population size L, we encode the parameters of each fuzzy model (solution) in a chromosome $\mathbf{s}_l, l = 1, \ldots, L$ as a sequence of elements describing the fuzzy sets in the rule antecedents followed by the parameters of the rule consequents. For a model of M fuzzy rules, triangular fuzzy sets (each given by three parameters), a n-dimensional premise and $n+1$ parameters in each consequent function, a chromosome of length $N = M(3n + (n+1))$ is encoded as

$$\mathbf{s}_l = (\mathrm{ant}_1, \ldots, \mathrm{ant}_M, \theta_1, \ldots, \theta_M) \tag{7}$$

where θ_i contains the consequent parameters p_{iq} of rule R_i, and $\mathrm{ant}_i = (a_{i1}, b_{i1}, c_{i1} \ldots, a_{in}, b_{in}, c_{in})$ contains the parameters of the antecedent fuzzy sets $A_{ij}, j = 1, \ldots, n$, according to (4). In the initial population $\mathbf{S}^0 = \{\mathbf{s}_1^0, \ldots, \mathbf{s}_L^0\}$, $\mathbf{s}_l^0$ is the initial model, and $\mathbf{s}_2^0, \ldots, \mathbf{s}_L^0$ are created by random variation (uniform distribution) around $\mathbf{s}_1^0$ within the defined constraints (see Section III-D).

B. Selection Function

The selection function is used to create evolutionary pressure, i.e., well-performing chromosomes have a higher chance to survive. The *roulette wheel* selection method [19] is used to select n_C chromosomes for operation. The chance on the roulette wheel is adaptive and is given as $P_l / \sum_{l'} P_{l'}$, where

$$P_l = \left(\frac{1}{J_l}\right)^2, \quad l, l' \in \{1, \ldots, L\} \tag{8}$$

and J_l is the performance of the model encoded in chromosome $\mathbf{s}_l$ measured in terms of the mean-squared-error (mse)

$$J = \frac{1}{K} \sum_{k=1}^{K} (y_k - \hat{y}_k)^2 \tag{9}$$

where y is the true output and $\hat{y}$ is the model output. The inverse of the selection function is used to select chromosomes for deletion. The best chromosome is always preserved in the population (*elitist* selection).

The chance that a selected chromosome is used in a crossover operation is 95% and the chance for mutation is 5% (in this paper). When a chromosome is selected for crossover (or mutation) one of the used crossover (or mutation) operators are applied with equal probability.

C. Genetic Operators

Two classical operators, *simple arithmetic crossover* and *uniform mutation* and four special real-coded operators are used in the GA. These operators have been successfully applied in [19], [22], [23].[1]

[1]The used operators are rather straightforward, and the interested reader can find other and more sophisticated operators in the specialized literature like, e.g., the IEEE Transaction on Evolutionary Computing and *Proceedings of Annual Events* such as the *Congress on Evolutionary Computing (CEC)* and the *Genetic and Evolutionary Computation Conference (GECCO)*.

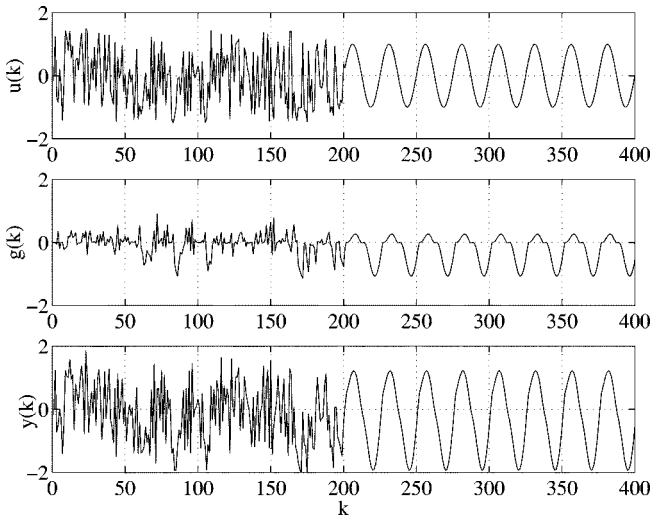

Fig. 1. Input $u(k)$, unforced system $g(k)$, and output $y(k)$ of the plant in (10).

In the following, $r \in [0, 1]$ is a random number (uniform distribution), $t = 0, 1, \ldots, T$ is the generation number, $\mathbf{s}_v$ and $\mathbf{s}_w$ are chromosomes selected for operation, $k \in \{1, 2, \ldots N\}$ is the position of an element in the chromosome, and $v_k^{\min}$ and $v_k^{\max}$ are the lower and upper bounds, respectively, on the parameter encoded by element k.

1) Crossover Operators: For crossover operations, the chromosomes are selected in pairs $(\mathbf{s}_v, \mathbf{s}_w)$.

i) *Simple arithmetic crossover*: $\mathbf{s}_v^t$ and $\mathbf{s}_w^t$ are crossed over at the kth position. The resulting offsprings are: $\mathbf{s}_v^{t+1} = (v_1, \ldots, v_k, w_{k+1}, \ldots, w_N)$ and $\mathbf{s}_w^{t+1} = (w_1, \ldots, w_k, v_{k+1}, \ldots, v_N)$, where k is selected at random from $\{2, \ldots, N-1\}$.

ii) *Whole arithmetic crossover*: a linear combination of $\mathbf{s}_v^t$ and $\mathbf{s}_w^t$ resulting in $\mathbf{s}_v^{t+1} = r(\mathbf{s}_v^t) + (1-r)\mathbf{s}_w^t$ and $\mathbf{s}_w^{t+1} = r(\mathbf{s}_w^t) + (1-r)\mathbf{s}_v^t$.

iii) *Heuristic crossover*: $\mathbf{s}_v^t$ and $\mathbf{s}_w^t$ are combined such that $\mathbf{s}_v^{t+1} = \mathbf{s}_v^t + r(\mathbf{s}_w^t - \mathbf{s}_v^t)$ and $\mathbf{s}_w^{t+1} = \mathbf{s}_w^t + r(\mathbf{s}_v^t - \mathbf{s}_w^t)$.

2) Mutation Operators: For mutation operations, single chromosomes are selected.

i) *Uniform mutation*: a random selected element v_k, $k \in \{1, 2, \ldots, N\}$ is replaced by v_k', which is a random number in the range $[v_k^{\min}, v_k^{\max}]$. The resulting chromosome is $\mathbf{s}_v^{t+1} = (v_1, \ldots, v_k', \ldots, v_m)$.

ii) *Multiple uniform mutation*: uniform mutation of n randomly selected elements, where n is also selected at random from $\{1, \ldots, N\}$.

iii) *Gaussian mutation*: all elements of a chromosome are mutated such that $\mathbf{s}_v^{t+1} = (v_1', \ldots, v_k', \ldots, v_m')$, where $v_k' = v_k + f_k$, $k = 1, 2, \ldots, N$. Here f_k is a random number drawn from a *Gaussian* distribution with zero mean and an adaptive variance $\sigma_k = ((T-t)/T)((v_k^{\max} - v_k^{\min})/3)$. The parameter tuning performed by this operator becomes finer and finer as the generation counter t increases.

D. Constraints

To maintain the transparency properties of the initial rule base as discussed in Section II-C, the optimization performed by the GA is subjected to two types of constraints: *partition* and *search space* constraints.

The partition constraint prohibits gaps in the partitioning of each input (antecedent) variable. The coding of a fuzzy set must comply with (4), i.e., $a \le b \le c$. To avoid gaps in the partition, pairs of neighboring fuzzy sets are constrained by $a_R \le c_L$, where L and R denote left and right set, respectively. After initialization of the initial population and after each generation of the GA, these conditions are forced, i.e., if for some fuzzy set $a > b$, then a and b are swapped and if $a_R > c_L$, then a_R and c_L are swapped.

Given the initial model, an initial chromosome $\mathbf{s}_1^0$ is generated. A chromosome consist of a vector of N elements $[v_1, v_2, \ldots, v_k, \ldots, v_N]$. The upper and lower limits of the search space are determined by the two user defined bounds α_1 and α_2, that determine the maximum allowed variation around the initial chromosome $\mathbf{s}_1^0$ for the antecedent and the consequent parameters, respectively. Following (7), the bound α_1 regards the allowed variation of the $M(3n)$ first elements of the chromosome vector, while the bound α_2 regards the allowed variation of the $M(n+1)$ last elements of the chromosome vector.

The first bound α_1 is intended to maintain the distinguishability of the models term set (the fuzzy sets) by allowing the parameters describing the fuzzy sets A_{ij} to vary only within a bound of $\pm\alpha_1|\mathcal{X}_j|$ around their initial values, where $|\mathcal{X}_j|$ is the length (range) of the domain on which the fuzzy sets A_{ij} are defined. By a low value of α_1, one can avoid the generation of domain-wide and multiple overlapping fuzzy sets, which is a typical feature of unconstrained optimization. The second bound, α_2, is intended to maintain the local-model interpretation of the rules by allowing the qth consequent parameter of the ith rule, p_{iq}, to vary within a bound of $\pm\alpha_2(\max_i(p_{iq}) - \min_i(p_{iq}))$ around its initial value.

The search space constraints are coded in the two vectors, $\mathbf{v}^{\max} = [v_1^{\max}, \ldots, v_N^{\max}]$ and $\mathbf{v}^{\min} = [v_1^{\min}, \ldots, v_N^{\min}]$, giving the upper and lower bounds on each of the N elements in a chromosome. During generation of the initial partition and in the case of a uniform mutation, elements are generated at random within these bounds. Only the heuristic crossover and the Gaussian mutation can produce solutions that violate the bounds. After these operations the constraints are forced, i.e., all elements v_k of the operated chromosomes are subjected to $v_k := \max(v_k^{\min}, \min(v_k, v_k^{\max}))$.

E. Proposed GA with Constrained Search Space

Given the pattern matrix $\mathbf{Z}$ and a fuzzy rule base, select the number of generations T, the population size L, the number of operations n_C and the constraints α_1 and α_2. Let $\mathbf{S}^t$ be the current population of solutions $\mathbf{s}_l^t, l = 1, \ldots, L$ and let $\mathbf{J}^t$ be the vector of corresponding values of the evaluation function.

1) Make an initial chromosome $\mathbf{s}_1^0$ from the initial fuzzy rule base.
2) Calculate the constraint vectors $\mathbf{v}^{\min}$ and $\mathbf{v}^{\max}$ using $\mathbf{s}_1^0$ and α_1 and α_2.
3) Create the initial population $\mathbf{S}^0 = \{\mathbf{s}_1^0, \ldots, \mathbf{s}_L^0\}$ where $\mathbf{s}_l^0, l = 2, \ldots, L$ are created by constrained uniform

random variations around $\mathbf{s}_1^0$ and the partition constraints apply.

4) *Repeat genetic optimization for* $t = 0, 1, 2, \ldots, T-1$:
   a) evaluate $\mathbf{S}^t$ by simulation and obtain $\mathbf{J}^t$;
   b) select n_C chromosomes for operation;
   c) select n_C chromosomes for deletion;
   d) operate on chromosomes acknowledging the search space constraints;
   e) exercise partition constraints;
   f) create new population $\mathbf{S}^{t+1}$ by substituting the operated chromosomes for those selected for deletion.
5) Select best solution from $\mathbf{S}^T$ by evaluating $\mathbf{J}^T$.

IV. Examples

In the following four subsections, we consider four different modeling problems. The first is a synthetic example regarding the modeling of a nonlinear dynamic plant. Then, the iris data and the wine data classification problems are studied, and finally the modeling of a real-world truck diesel engine turbocharger is considered. In all examples, we apply a GA with the population size $L = 40$ from which $n_C = 10$ are selected for operation in each iteration (25% replacement).

A. Nonlinear Dynamic Plant

We consider the second-order nonlinear plant studied by Wang and Yen in [9], [24], and [17]

$$y(k) = g(y(k-1), y(k-2)) + u(k) \tag{10}$$

where

$$g(y(k-1), y(k-2)) = \frac{y(k-1)y(k-2)(y(k-1)-0.5)}{1+y^2(k-1)y^2(k-2)}. \tag{11}$$

The goal is to approximate the nonlinear component $g(y(k-1), y(k-2))$ of the plant with a fuzzy model. As in [9], 400 simulated data points were generated from the plant model (10). Starting from the equilibrium state $(0, 0)$, 200 samples of identification data were obtained with a random input signal $u(k)$ uniformly distributed in $[-1.5, 1.5]$, followed by 200 samples of evaluation data obtained using a sinusoid input signal $u(k) = \sin(2\pi k/25)$. The resulting signals are shown in Fig. 1.

1) Solutions in the Literature: We compare our results with those obtained by the three different approaches proposed in [9], [24] and [17]. These approaches are described below, and the best results obtained in each case are summarized in Table II.

In [9] a GA was combined with a Kalman filter to obtain a fuzzy model of the plant. The antecedent fuzzy sets of 40 rules, encoded by Gaussian membership functions, were determined initially by clustering and kept fixed. A binary GA was used to select a subset of the initial 40 rules in order to produce a more compact rule base with better generalization properties. The consequents of the various models in the GA population were estimated after each generation by the Kalman filter, and an information criterion was used as evaluation function to balance the tradeoff between the number of rules and the model accuracy.

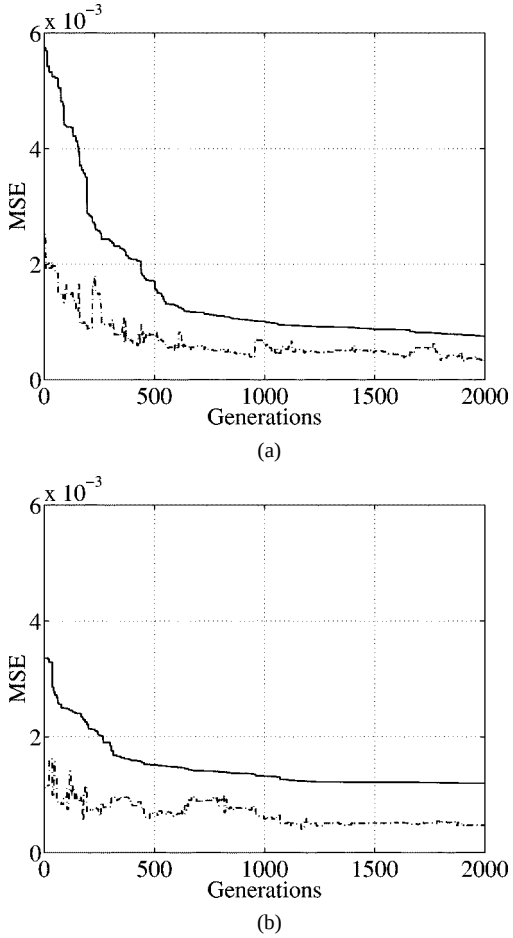

Fig. 2. The convergence (mse) of the GA applied to the optimization of the TSK model with linear consequents. (a) Five-rule model with eight fuzzy sets. (b) Four-rule model with four fuzzy sets. Training data (solid lines) and evaluation data (dash-dot lines) are shown.

In [24] various information criteria were used to successively pick rules from a set of 36 rules in order to obtain a compact and accurate model. The initial rule base was obtained by partitioning each of the two inputs $y(k-1)$ and $y(k-2)$ by six equidistant fuzzy sets. The rules were picked in an order determined by an orthogonal transform.

In [17] various orthogonal transforms for rule selection were studied using an initial model with 25 rules. In this initial model, 20 rules were obtained by clustering, while five redundant rules were added to evaluate the selection performance of the studied techniques.

2) Proposed Approach: We applied the modeling approach proposed in this paper. The real-coded GA described in Section III was applied with the constraints $\alpha_1 = 25\%$, $\alpha_2 = 25\%$, and $T = 2000$ generations.

First, we considered a *singleton* TSK model consisting of seven rules obtained by fuzzy c-means clustering as described in Section II. The mse for both training and validation data were comparable, indicating that the initial model is not overtrained. By GA optimization, the mse was reduced by 81% from $1.6e^{-2}$ to $3.0e^{-3}$ on the training data and by 96% from $1.2e^{-2}$ to $4.9e^{-4}$ on the evaluation data.

Considering the good results obtained with the singleton model, we decided to look for a smaller rule base. This time a

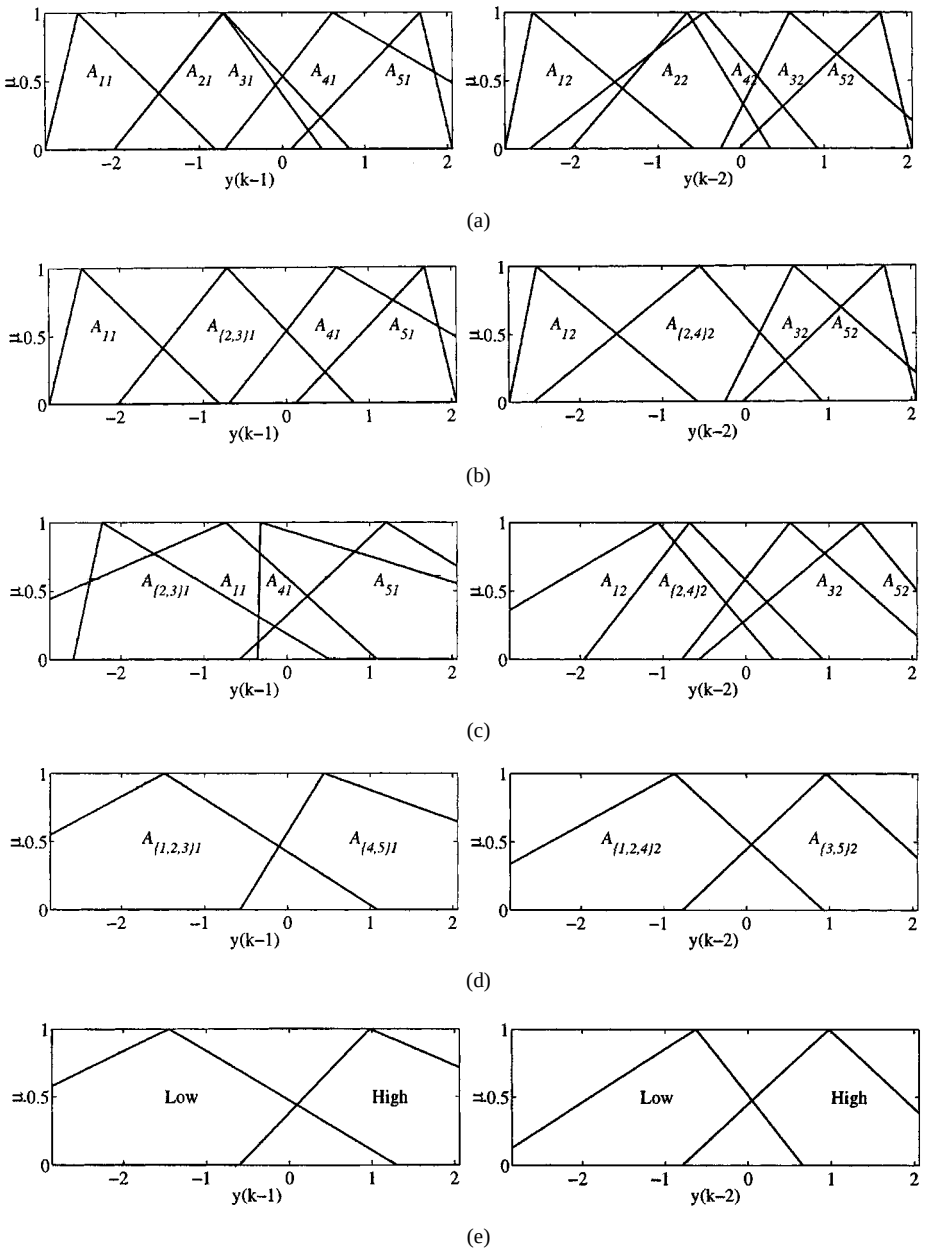

Fig. 3. The fuzzy sets in the antecedent of the linear consequent TSK model of the plant. (a) Initial model (five rules, ten fuzzy sets) obtained by c-means clustering. (b) Simplified model (five rules, eight fuzzy sets). (c) Model after GA optimization. (d) Model after second simplification (four rules and four fuzzy sets). (e) Final model after second GA optimization.

TSK model with *linear consequent functions* was used since the more powerful approximation capabilities of the functional consequents allows for less rules. An initial model of five rules was constructed by clustering, see Fig. 3(a). To further simplify the initial model, we applied the similarity-driven simplification described in Section II-D. Two sets were aggregated in each premise as shown in Fig. 3(b). The GA was applied to this model (five rules, eight fuzzy sets), and the convergence is shown in Fig. 2(a). The mse is lower for the evaluation data than for the training data, indicating that the low-frequency information in the output signal is well learned and we are not overtraining. The mse reduced from $3.8e^{-3}$ to $7.5e^{-4}$ for training data, and from $2.5e^{-3}$ to $3.5e^{-4}$ for evaluation data. The fuzzy sets in the optimized model are shown in Fig. 3(c).

In search of further simplification of the optimized model, once again, we applied similarity-driven simplification. This gave a reduced model of four rules, with the four fuzzy sets as shown in Fig. 3(d). As a result of this, the mse increased. The GA, however, was able to bring the performance back to an acceptable level, mse $1.2e^{-3}$ and $4.7e^{-4}$, for training and evaluation data, respectively. The convergence is shown in Fig. 2(b). Only small modifications of the model parameters were done by the GA, as seen when comparing Fig. 3(d) with Fig. 3(e). The final rule base is given in Table I, and Fig. 4 shows the corresponding model surface and the individual rule consequents.

From the results summarized in Table II, we see that the proposed modeling approach is capable of obtaining good results using fewer rules than other approaches reported in the litera-

TABLE I
FUZZY MODEL OF THE NONLINEAR DYNAMIC PLANT

R_1	If y_{k-1} is Low **and** y_{k-2} is Low	**then** $g = 0.4502y_{k-1} + 0.1686y_{k-2} + 0.1413$
R_2	If y_{k-1} is Low **and** y_{k-2} is High	**then** $g = -0.4193y_{k-1} + 0.1575y_{k-2} - 0.0937$
R_3	If y_{k-1} is High **and** y_{k-2} is Low	**then** $g = -0.2699y_{k-1} + 0.0890y_{k-2} + 0.1327$
R_4	If y_{k-1} is High **and** y_{k-2} is High	**then** $g = 0.3213y_{k-1} + 0.0584y_{k-2} - 0.1716$
	Fuzzy set parameters	
y_{k-1}	Low $= (-4.7899, -1.4475, 1.2972)$	High $= (-0.6048, 0.9765, 4.7889)$
y_{k-2}	Low $= (-3.1795, -0.6248, 0.6600)$	High $= (-0.7942, 0.9789, 2.7312)$

TABLE II
FUZZY MODELS OF THE NONLINEAR DYNAMIC PLANT. ALL MODELS ARE OF THE TSK TYPE

Ref.	No. of rules	No. of Sets	Consequent	MSE train	MSE eval
[9]	40 rules (initial)	40 Gauss. (2D)	Singleton	$3.3e^{-4}$	$6.9e^{-4}$
	28 rules (optimized)	28 Gauss. (2D)	Singleton	$3.3e^{-4}$	$6.0e^{-4}$
[24] [2]	36 rules (initial)	12 B-splines	Singleton	$2.8e^{-5}$	$5.1e^{-3}$
	23 rules (optimized)	12 B-splines	Singleton	$3.2e^{-5}$	$1.9e^{-3}$
	36 rules (initial)	12 B-splines	Linear	$1.9e^{-6}$	$2.9e^{-3}$
	24 rules (optimized)	12 B-splines	Linear	$2.0e^{-6}$	$6.4e^{-4}$
[17]	25 rules (initial)	25 Gauss. (2D)	Singleton	$2.3e^{-4}$	$4.1e^{-4}$
	20 rules (optimized)	20 Gauss. (2D)	Singleton	$6.8e^{-4}$	$2.4e^{-4}$
This	7 rules (initial)	14 triangular	Singleton	$1.6e^{-2}$	$1.2e^{-3}$
paper	7 rules (optimized)	14 triangular	Singleton	$3.0e^{-3}$	$4.9e^{-4}$
Fig. 3a	5 rules (initial)	10 triangular	Linear	$5.8e^{-3}$	$2.5e^{-3}$
Fig. 3c	5 rules (optimized)	8 triangular	Linear	$7.5e^{-4}$	$3.5e^{-4}$
Fig. 3e	4 rules (optimized)	4 triangular	Linear	$1.2e^{-3}$	$4.7e^{-4}$

[2] The low MSE on the training data in contrast to the MSE on the evaluation data may indicate overtraining.

ture. Moreover, simple triangular membership functions were used as opposed to cubic B-splines in [24] and Gaussian-type basis functions in [9], [17]. By applying the GA after a rule-base simplification step, not only an accurate, but also a compact and transparent rule base was obtained.

B. Iris Data Classification

The iris data is a common benchmark in classification and pattern recognition studies [2], [10], [25], [26]. It contains 50 measurements of four features from each of the three species *Iris setosa*, *Iris versicolor*, and *Iris virginica* [27].[2] We label the species 1, 2, and 3, respectively, which gives a 5×150 pattern matrix Z of observation vectors

$$\mathbf{z}_k^T = [x_{k1}, x_{k2}, x_{k3}, x_{k4}, c_k], \quad c_k \in \{1, 2, 3\}, \quad k = 1, 2, \ldots, 150 \quad (12)$$

where x_{k1}, x_{k2}, x_{k3}, and x_{k4} are the sepal length, sepal width, petal length, and petal width, respectively. The measurements are shown in Fig. 5.

1) Solutions in the Literature: Ishibuchi *et al.* [26] reviewed nine fuzzy classifiers and ten nonfuzzy classifiers from the literature, giving between three and 24 misclassifications for the iris classification problem. They also proposed various approaches using a fixed grid fuzzy partitioning of the antecedent space. The best result, four misclassifications with leave-one-out evaluation (resubstitution), was obtained with weighted voting with multiple fuzzy rule bases and a total of 864 rules.[3] Using 1296 rules and a *single winner* approach, the complete data set could be learned with two training errors [26].

Bezdek *et al.* [28] compared various multiple prototype *generation* schemes. With the so-called *dog-rabbit* model, five prototypes were obtained which gave three resubstitution errors. In [29] a nearest prototype approach, with three prototypes *selected* by either a binary coded GA or random search, gave also three resubstitution errors.

Shi *et al.* [10] used a GA with *integer coding* to learn a Mamdani type fuzzy model. Starting with three fuzzy sets associated with each feature, the membership function shapes and types, and the fuzzy rule set, including the number of rules, were evolved using a GA. Furthermore, a fuzzy expert system was used to adapt the GAs learning parameters. After several trials with varying learning options, a four rule model was obtained, which gave three errors in learning the data.

[2]The original Iris data was recently republished in [25].

[3]The reader is reminded that there are only 150 data samples available in the Iris data set . . ..

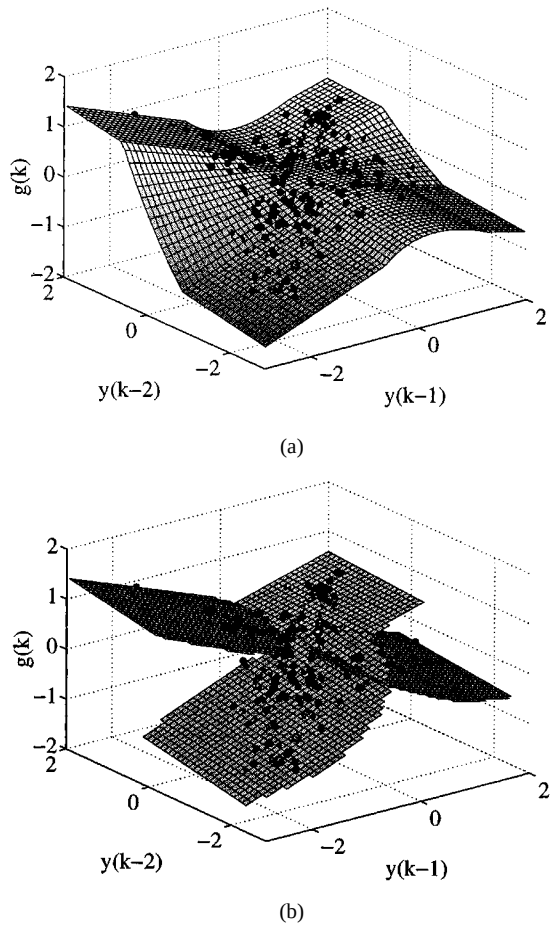

Fig. 4. Nonlinear dynamic plant. (a) Surface plot of the TSK model in Table I and (b) the individual rule consequent functions. Training data are shown as black dots.

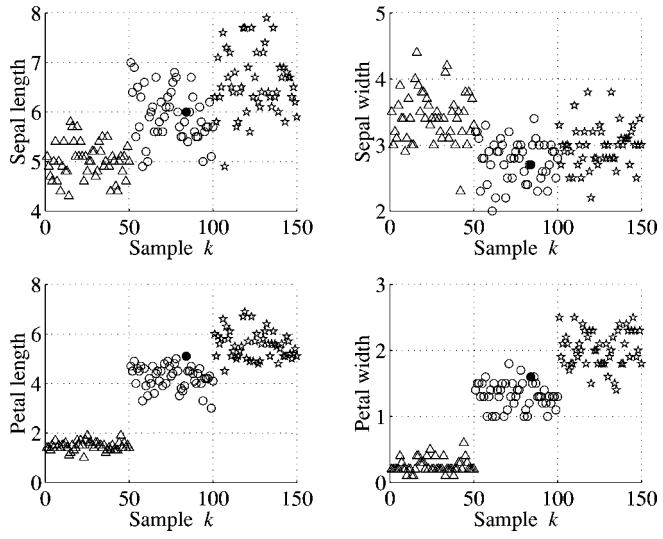

Fig. 5. Iris data: *iris setosa* ($\triangle$), *iris versicolor* ($\circ$), and *iris virginica* ($\star$). Sample no. 84, indicated by the $\bullet$ is misclassified by our approach.

2) Proposed Approach: We applied the fuzzy c-means clustering as described in Section II to obtain an initial TSK model with singleton consequents. In order to perform classification,

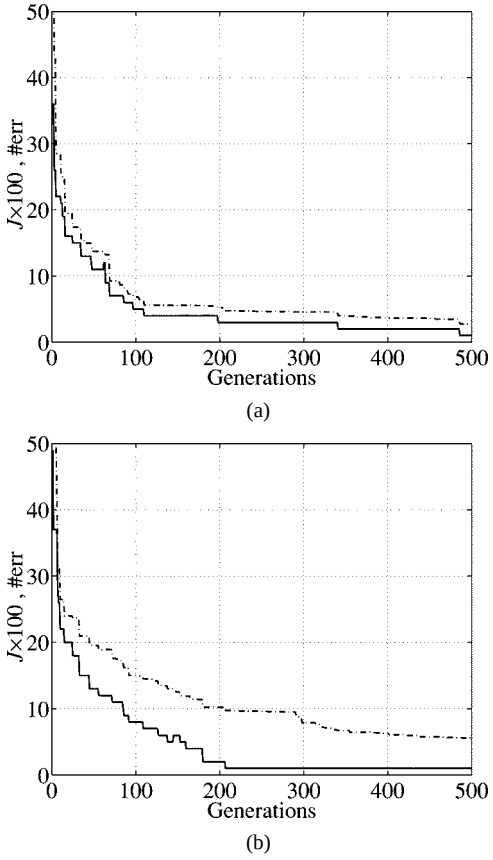

Fig. 6. The convergence of the GA in learning the iris data (a) with three rules and (b) with two rules. Both the number of classification errors (solid lines) and the GA objective function (dash-dotted lines) given in (14) are shown.

the output y_k of the TSK model was used with the following classification rule:

$$c_k = \begin{cases} 1, & \text{if } y_k < 1.5 \\ 2, & \text{if } 1.5 \le y_k < 2.5 \\ 3, & \text{if } 2.5 \le y_k. \end{cases} \tag{13}$$

First, an initial model with three rules was constructed from clustering where each rule described a class (singleton consequents). The classification accuracy of the initial model was rather discouraging, giving 36 misclassifications on the training data. We then optimized this model with the GA described in Section III, using an objective function that combines the mse with the classification error

$$J = \frac{1}{N}\left(\sum_{k=1}^{N}(y_k - \hat{y}_k)^2 + \sum_{k=1}^{N}(c_k \neq \hat{c}_k)\right). \tag{14}$$

The mse is needed to differentiate between various solutions with the same number of classification errors, and it was found to speed up the convergence of the GA for the used model type (continuous output). The GA was applied with the constraints $\alpha_1 = 50\%$, $\alpha_2 = 25\%$, and $T = 500$ generations. The convergence is illustrated in Fig. 6(a), and the resulting model is capable of classifying the data with one error only. The model is suitable for interpretation since the rules consequents are so close to the actual class labels that each rule can be taken to describe a class. The fuzzy sets of the optimized model are shown in Fig. 7, while the corresponding rules are given in Table III.

TABLE III
THREE RULE FUZZY CLASSIFIER FOR THE IRIS DATA. (FUZZY SETS IN FIG. 7)

R_1	**If** x_1 is Short	**and** x_2 is Wide	**and** x_3 is Short **and** x_4 is Narrow **then** $g = 0.9763$
R_2	**If** x_1 is Long	**and** x_2 is Medium	**and** x_3 is Medium **and** x_4 is Medium **then** $g = 2.0054$
R_3	**If** x_1 is Medium	**and** x_2 is Narrow	**and** x_3 is Long **and** x_4 is Wide **then** $g = 3.0185$
	Fuzzy set parameters		
x_1	S=(4.2046, 5.3171, 5.9475)	M=(3.7522, 5.9779, 8.2048)	L=(4.3795, 6.5593, 8.2816)
x_2	N=(1.2431, 2.5759, 4.4562)	M=(1.7047, 3.5304, 3.8480)	W=(1.8674, 4.143, 4.935)
x_3	S=(0.3542, 1.061, 4.1049)	M=(0.5027, 3.9518, 5.4715)	L=(4.1304, 6.8883, 7.808)
x_4	N=(−0.3903, −0.0909, 1.097)	M=(0.4008, 1.130, 2.3095)	W=(1.1088, 2.1603, 3.0233)

TABLE IV
TWO RULE FUZZY CLASSIFIER FOR THE IRIS DATA. (FUZZY SETS IN FIG. 8)

R_1	**If** x_1 is A	**and** x_2 is Wide	**and** x_3 is Short **and** x_4 is Narrow **then** $g = 1.0453$
R_2	**If** x_1 is B	**and** x_2 is Narrow	**and** x_3 is Long **and** x_4 is Wide **then** $g = 2.8034$
	Fuzzy set parameters		
x_1	A=(3.2623, 5.8125, 9.4886)	B=(4.2430, 6.5271, 7.570)	
x_2	N=(1.5607, 2.6806, 4.2887)	W=(1.7444, 3.7105, 5.5704)	
x_3	S=(−0.3239, 0.9202, 5.7306)	L=(1.4735, 6.2559, 6.9432)	
x_4	N=(−0.4926, 0.0813, 2.9029)	W=(0.7133, 3.5986, 4.5500)	

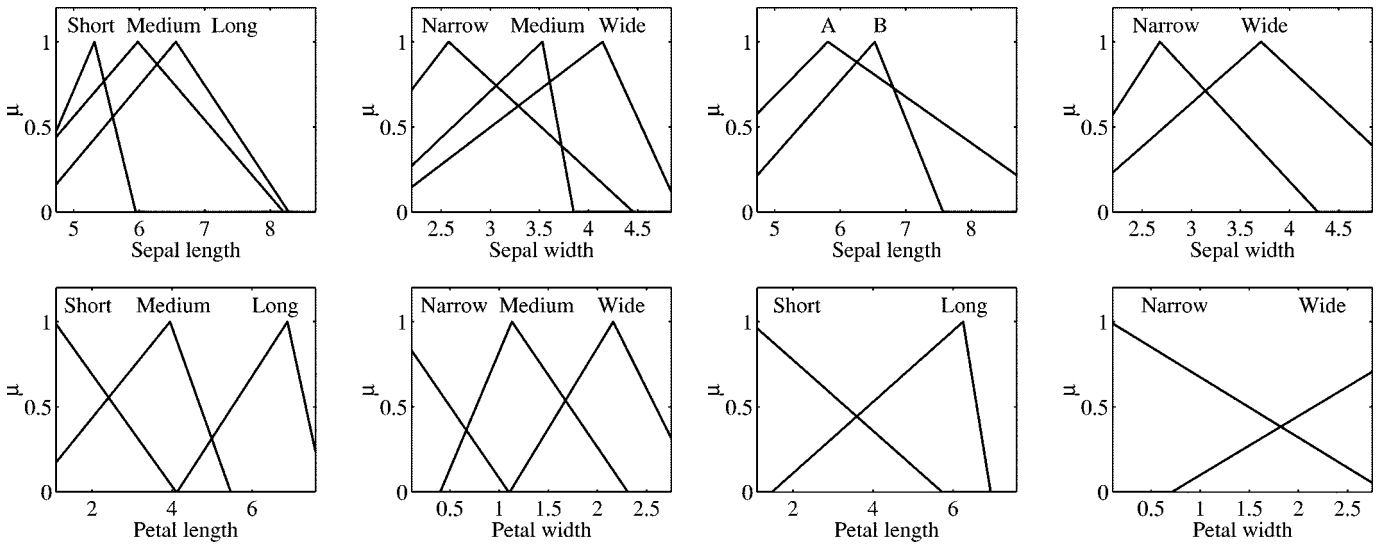

Fig. 7. The fuzzy sets in the antecedent of the optimized three rule classifier for the iris data.

Fig. 8. The fuzzy sets in the antecedent of the optimized two rule classifier for the iris data.

Motivated by the good results obtained with three rules, we decided to constructed a two rule model. Such a model is less suited for interpretation, as it will have to interpolate between two classification rules in order to generate the three class outputs. The initial model constructed by clustering gave 49 classification errors. When we optimized this model using the proposed GA, the misclassifications reduced to one error only [see Fig. 6(b)]. The resulting two fuzzy classification rules are given in Table IV and the optimized membership functions are shown in Fig. 8. Comparing the rules in Tables III and IV, it looks as if the two rule classifier interpolates between R_1 and R_3 of the three rule classifier. In the two rule classifier it is difficult to assign sensible labels to the partition of *sepal length* (x_1) and no class label can be attached to the rule consequents.

The results obtained with the proposed modeling approach for the iris data case illustrates the power of the GA for optimizing fuzzy models. By simultaneously optimizing the antecedent and consequent parts of the rules, the GA found an optimum for the model parameters in the neighborhood of the initializations, which gave drastic improvements in classification performance.

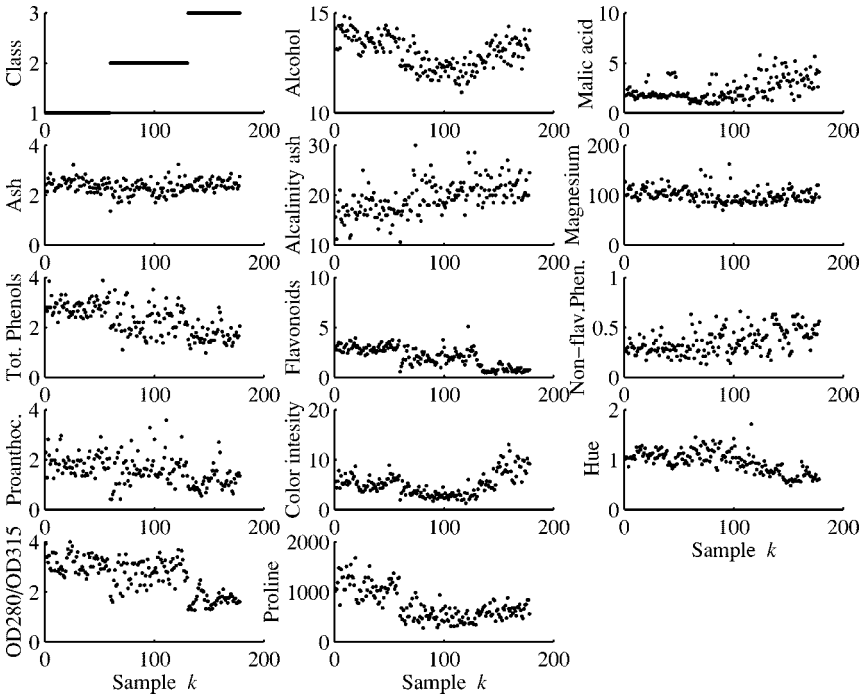

Fig. 9. Wine data: 3 classes and 13 attributes.

C. Wine Classification Data

The wine data[4] contains the chemical analysis of 178 wines grown in the same region in Italy but derived from three different cultivars. The 13 continuous attributes are available for classification: alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoids phenols, proanthocyaninsm color intensity, hue, OD280/OD315 of dilluted wines and proline (Fig. 9).

Corcoran and Sen [30] applied all the 178 samples for learning 60 nonfuzzy IF-THEN rules in a real-coded genetic-based machine learning approach. They used a population of 1500 individuals and applied 300 generations, with full replacement, to come up with the following result for ten independent trials: best classification rate 100%, average classification rate 99.5% and worst classification rate 98.3%, which is three misclassifications. Ishibuchi *et al.* [4] applied all the 178 samples designing a fuzzy classifier with 60 fuzzy rules by means of an integer-coded GA and grid partitioning. Their population contained 100 individuals and they applied 1000 generations, with full replacement, to come up with the following result for ten independent trials: best classification rate 99.4% (one misclassifications), average classification rate 98.5% and worst classification rate 97.8% (four misclassifications).

In both approaches the final rule base contains 60 rules. The main difference is the number of model evaluations that was necessary to come to the final result. In [30] the Pittsburgh approach of genetic-based learning is used where each individual in the population contains a complete fuzzy model, resulting in 150 000 model evaluations. In [4], the Michigan approach is followed were each individual contains one rule and the complete population consists of one fuzzy model and thus only 1000 model evaluations were performed.

1) Proposed Approach: An initial TSK singleton model with three rules was constructed from c-means clustering an all the available 178 samples as described in Section II. In order to perform classification, the continuous output of the TSK model was used with the same classification rule as in the iris example (13).

The initial model gave 60 misclassifications. We then applied similarity-driven simplification and the number of fuzzy sets was reduced with 11. This model was optimized with the GA described in Section III, with $\alpha_1 = 50\%$ and $\alpha_2 = 20\%$, using the same objective function (14) as for the iris data. The convergence is shown in Fig. 10(a), and after 200 iterations the model was capable of classifying the data with nine misclassifications. The obtained model was again subjected to similarity-driven simplification and the reduced model, with an additional seven sets less, was again optimized in 400 iterations by the GA Fig. 10(b). The final model improved to three misclassifications. The fuzzy sets of the optimized model are shown in Fig. 11, while the corresponding rules are given in Table V.

In this example, the repetitive simplification and optimization left four features without antecedent terms. Thus, feature reduction is obtained, and the resulting three rule classifier uses only nine of the initial 13 features. Comparing the fuzzy sets in Fig. 11 with the data in Fig. 9 shows that the obtained rules are highly interpretable. For example, the Flavonoids are divided in low, medium and high, which is clearly visible in the data. Visual inspection of the data also shows that "do not care" elements were obtained for features that contain little variation over the three classes (Ash, Mag, nFlav and Hue). The number of model evaluations was in total $(200+400)\cdot 25\%\cdot 100 = 6000$

[4]The wine data is available from the University of California, Irvine, via anonymous ftp ftp.ics.uci.edu/pub/machine-learning-databases.

TABLE V
THREE RULE FUZZY CLASSIFIER FOR THE WINE DATA. THE LABELS L, M, AND H, CORRESPONDS TO LOW, MEDIUM AND HIGH, RESPECTIVELY

	Alc	Mal	Ash	aAsh	Mag	Tot	Fla	nFlav	Pro	Col	Hue	OD2	Pro	Class
R_1	H	L	-	L	-	H	H	-	H	M	-	H	H	0.94
R_2	L	L	-	L	-	H	M	-	H	L	-	M	L	2.04
R_3	L	H	-	H	-	L	L	-	L	H	-	L	L	3.01

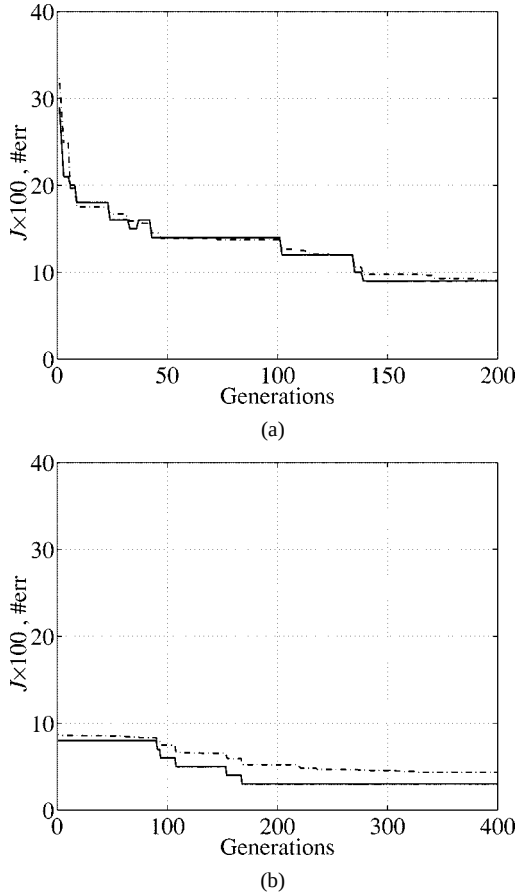

Fig. 10. The convergence of the GA in learning the wine data in (a) the first optimization and (b) final optimization. Both the number of classification errors (solid lines) and the GA objective function (dash-dotted lines) given in (14) are shown.

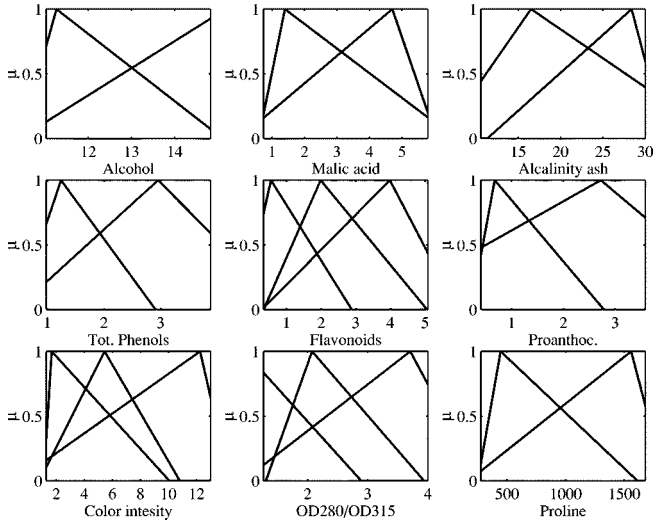

Fig. 11. The fuzzy sets in the antecedent of the optimized three rule classifier for the wine data.

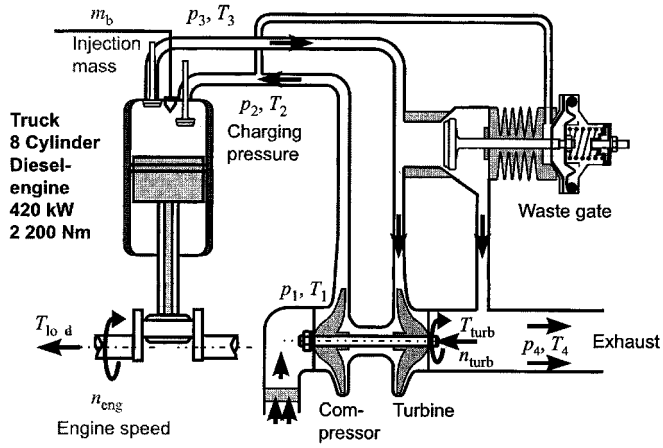

Fig. 12. Scheme of the diesel engine turbocharger.

since we apply the Pittsburgh approach with 25% replacement in each iteration. The obtained classification result is comparable to those in [4] and [30], but our model uses far less rules (three compared to 60) and less features.

D. Identification of a Turbocharger

The dynamic modeling of a truck Diesel engine turbocharger [31] is considered. The goal is to obtain an accurate simulation model, but at the same time it should be transparent and of low complexity in order to support applications like controller design, hardware-in-the-loop simulations and fault diagnosis.

The charging process of the diesel engine by an exhaust turbocharger is schematically presented in Fig. 12. A physical model of the turbocharger is hard to obtain and it is to complex for controller design. The identification method as proposed in this paper delivers an appropriate alternative and is only based on recorded input–output data. For a *hardware-in-the-loop* simulation, the fuel injection, $m_b(mg)$ per injection and the engine speed $n_{\rm eng}$ in (rpm) are chosen as the inputs and the charging pressure p_2 in (bar) is the models output. Three data sets with input–output data were obtained from the engine with different driving cycles. The data has a sampling time of 0.2 s and is shown in Fig. 13. The learning data *Set 1* is obtained by driving on a flat test track with the biggest possible load. For validation, *Set 2* was obtained in a similar way and *Set 3* was obtained in a driving cycle meant to reproduce realistic conditions in urban traffic.

In [31] the performance of two TSK fuzzy modeling approaches were compared on this example: the LOLIMOT local linear model tree algorithm [32] and Gustafson–Kessel

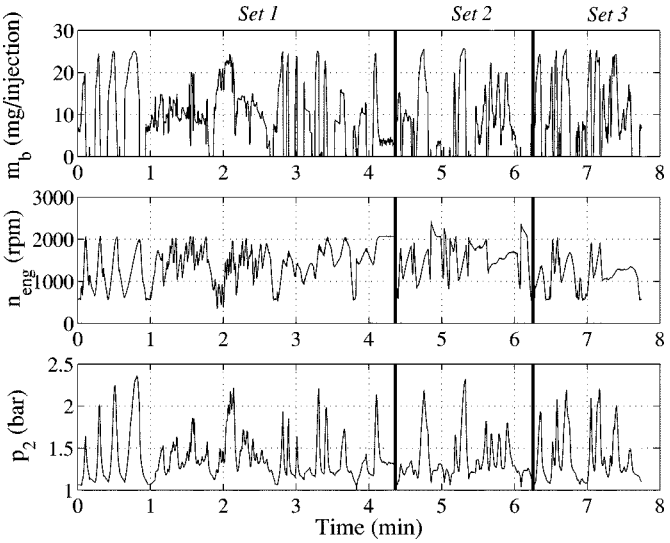

Fig. 13. Training data: *Set 1*; validation data: *Sets 2* and *3*.

product space clustering for rule construction [14]. The latter is comparable with our initial model step described in Section II. The charging pressure was modeled by

$$p2(k+1) = f(m_b(k), m_b(k-1), m_b(k-2), n_{\text{eng}}(k), n_{\text{eng}}(k-1), n_{\text{eng}}(k-2), p_2(k), p_2(k-1)). \tag{15}$$

The LOLIMOT algorithm used ten rules with multivariate Gaussian membership functions while the rule base constructed from product space clustering used three rules and univariate piecewise exponential membership functions. The root-mean-squared-error (rmse) for simulation on both the training set (*Set 1*) and the validation sets (*Set 2* and *Set 3*) was 0.021, 0.023, and 0.032 for the LOLIMOT model and 0.024, 0.023, 0.036 with the clustering approach. It must be noted that the model performance in this example is based on simulation and not *one-step-ahead* prediction as in the dynamic model example in Section IV-A, i.e., the state variables are only given at the initial time, and for the remaining time the simulation is recursive.

1) Proposed Approach: Based on the good experience with the proposed method we decided to start with a simple nonlinear model to predict the charging pressure:

$$p_2(k+1) = f(p_2(k), m_b(k), n_{\text{eng}}(k)). \tag{16}$$

An initial model was made with only two clusters. We then applied similarity-driven simplification and the conditions for fuel injection m_b were removed from antecedent part of the model (replaced by "do not care"). The simulation of the initial model gives an rmse of 0.152 for the training data and 0.227 and 0.151 for the two validation sets.

We then optimized this model with the GA described in Section III ($\alpha_1 = 50\%$ and $\alpha_2 = 20\%$), using the rmse as the objective function and $T = 200$ iterations. The rmse of the optimized model is 0.046 for the training data and 0.048 and 0.053 for the two validation sets. The convergence of the GA is shown in Fig. 14(a). It should be noted that this optimization has a high computational load due to the recursive simulation that needs to

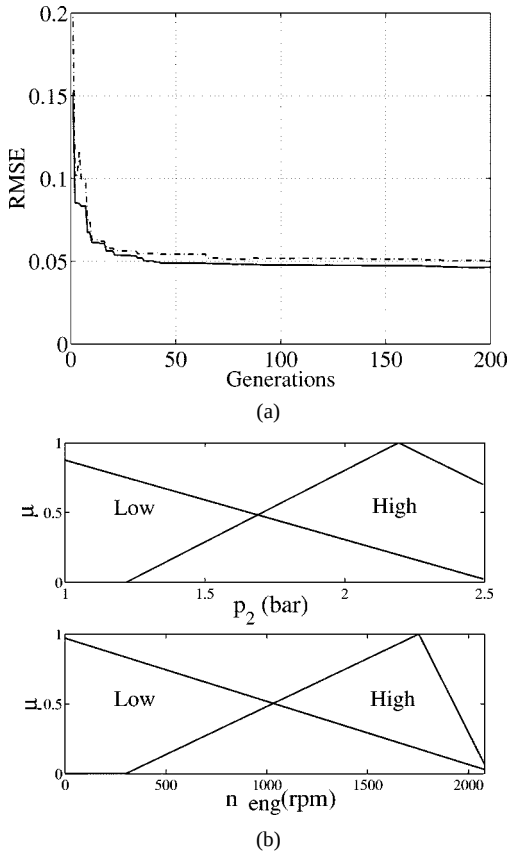

Fig. 14. (a) The GA convergence for the turbocharger model and (b) the fuzzy sets of the obtained model. In (a) both the rmse on the learning data *Set 1* (solid lines) and the mse of the validation data (*Set 2*+*Set 3*)(dash-dotted lines) are shown.

be performed for each new individual in the GA. However, this only needs to be done a few times during the model development. The obtained fuzzy sets in the two rule model are shown in Fig. 14(b) and the two rules are given in Table VI. The simulation results are shown in Fig. 15.

V. Conclusion

Many approaches to fuzzy rule-based modeling by means of evolutionary methods or other data-driven techniques have been published. The variety of possibilities and automated algorithms has often resulted in unnecessarily complex models that more resembles black-box approaches.

We have described a method to construct compact and transparent, yet accurate fuzzy rule-based models of the TSK type. Fuzzy clustering in the product space of the models inputs and output is applied to obtain an initial fuzzy rule base. This helps to ensure a qualitatively tractable rule base where the rules are local models that describe important regions in the systems input–output space. To increase the quantitative properties of the initial model we have proposed a real-coded GA that simultaneously optimizes the parameters of the antecedent membership functions and the rule consequents. The tractable structure of the initial model is maintained by constraining the search space of the GA. In cases where transparency issues are less important, the constraints on the GA search space can be relaxed. We also showed that successive application

TABLE VI
FUZZY MODEL OF THE DIESEL TURBOCHARGER

R_1	**If** $m_b(k)$ is Low **and** $p_2(k)$ is Low
	then $p_2(k+1) = 0.8716p_2(k) + 0.0031m_b(k) + 2.053 \cdot 10^{-3}n_{eng}(k) + 0.1144$
R_2	**If** $m_b(k)$ is High **and** $p_2(k)$ is High
	then $p_2(k+1) = 0.8077p_2(k) + 0.0122m_b(k) + 9.188 \cdot 10^{-3}n_{eng}(k) + 0.0013$
	Fuzzy set parameters
$p_2(k)$	Low = $(0.5033, 0.7809, 2.5386)$ High = $(1.2180, 2.1925, 3.2040)$
$m_b(k)$	Don't care = Universal set
$n_{eng}(k)$	Low = $(-83.1, -64.9, 2146.3)$ High = $(300.3, 1753.9, 2103.3)$

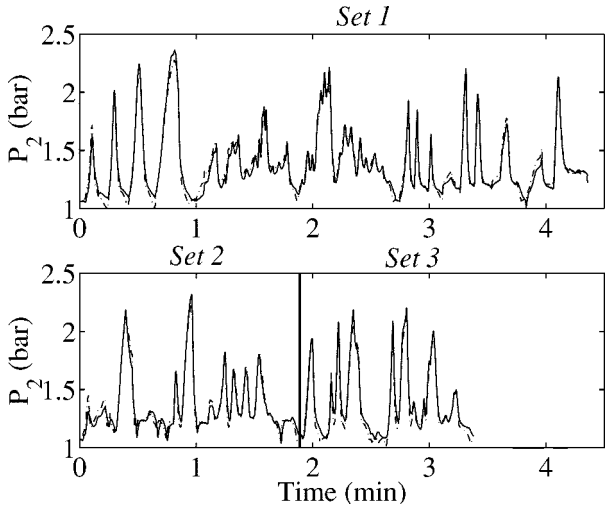

Fig. 15. Simulation of the optimized Turbocharger model. Both training data (top) and validation data (bottom) are shown (solid line) together with the model output (dash-dotted line).

of rule-base simplification and GA-based optimization can be considered to further reduce the model complexity. In this case, the proposed constrained GA was capable of efficiently correcting for the decrease in model accuracy introduced by the simplification.

The proposed modeling approach was successfully applied to four problems known from the literature: the modeling of a synthetic nonlinear dynamic plant, the iris data classification problem, the wine data classification problem, and the modeling of a diesel engine turbocharger. Quantitatively, the obtained models were comparable to the best results reported in the literature. However, the obtained models used fewer rules than other fuzzy models reported in the literature. Moreover, these results were obtained with simple triangular membership function constructs in order to allowed for linguistic interpretation of the models term set. Even better model accuracy can be obtained by using smooth membership functions and by relaxing the constraints on the GA search space.

ACKNOWLEDGMENT

The authors would like to thank the Laboratory of Control Systems and Process Automation, Institute of Automatic Control, Darmstadt University of Technology, Darmstadt, Germany, for providing the data of the diesel engine turbocharger. They would also like to thank the anonymous reviewers for assisting in improving this manuscript.

REFERENCES

[1] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. 15, pp. 116–132, 1985.
[2] H. Ishibuchi, T. Murata, and I. B. Türkşen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets Syst.*, vol. 89, pp. 135–150, 1997.
[3] C.-H. Wang, T.-P. Hong, and S.-S. Tseng, "Integrating fuzzy knowledge by genetic algorithms," *Fuzzy Sets Syst.*, vol. 2, no. 4, pp. 138–149, 1998.
[4] H. Ishibuchi, T. Nakashima, and T. Murata, "Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 29, pp. 601–618, Oct. 1999.
[5] L. O. Hall, I. B. Özyurt, and J. C. Bezdek, "Clustering with genetically optimized approach," *IEEE Trans. Evolut. Computing*, vol. 3, pp. 103–112, July 1999.
[6] H.-S. Hwang, "Control strategy for optimal compromise between trip time and energy consumption in a high-speed railway," *IEEE Trans. Syst., Man, Cybern.*, pt. A, vol. 28, pp. 791–802, Nov. 1998.
[7] I. Jagielska, C. Matthews, and T. Whitfort, "An investigation into the application of neural networks, fuzzy logic, genetic algorithms, and rough sets to automated knowledge acquisition for classification problems," *Neurocomputing*, vol. 24, pp. 37–54, 1999.
[8] M. Russo, "FuGeNeSys—A fuzzy genetic neural system for fuzzy modeling," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 373–388, Aug. 1998.
[9] L. Wang and J. Yen, "Extracting fuzzy rules for system modeling using a hybrid of genetic algorithms and Kalman filter," *Fuzzy Sets Syst.*, vol. 101, pp. 353–362, 1999.
[10] Y. Shi, R. Eberhart, and Y. Chen, "Implementation of evolutionary fuzzy systems," *IEEE Trans. Fuzzy Syt.*, vol. 7, pp. 109–119, Apr. 1999.
[11] J. Valente de Oliveira, "Semantic constraints for membership function optimization," *IEEE Trans. Systems, Man, Cybern.*, pt. A, vol. 29, pp. 128–138, Jan. 1999.
[12] M. Setnes, R. Babuška, and H. B. Verbruggen, "Rule-based modeling: Precision and transparency," *IEEE Trans. Syst., Man, Cybern.*, pt. C, vol. 28, pp. 165–169, Feb. 1998.
[13] F. Höppner, F. Klawonn, R. Kruse, and T. Runkler, *Fuzzy Cluster Analysis*. New York: Wiley, 1999.
[14] R. Babuška, *Fuzzy Modeling for Control*. Boston, MA: Kluwer, 1998.
[15] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Functions*. New York: Plenum, 1981.
[16] J. Yen, L. Wang, and C. W. Gillespie, "Improving the interpretability of TSK fuzzy models by combining global learning with local learning," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 530–537, Nov. 1998.
[17] J. Yen and L. Wang, "Simplifying fuzzy rule-based models using orthogonal transformation methods," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 29, pp. 13–24, Feb. 1999.
[18] M. Setnes, R. Babuška, U. Kaymak, and H. R. van Nauta Lemke, "Similarity measures in fuzzy rule base simplification," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 28, pp. 376–386, June 1998.
[19] Z. Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*, 2nd ed. New York: Springer-Verlag, 1994.
[20] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[21] F. Herrera, M. Lozano, and J. L. Verdegay, "Tackling real-coded genetic algorithms: Operators and tools for behavioral analysis," *Artificial Intell. Rev.*, vol. 12, pp. 265–319, 1998.
[22] M. Setnes and J. A. Roubos, "Transparent fuzzy modeling using fuzzy clustering and GA's," in *18th Int. Conf. North Amer. Fuzzy Inform. Processing Soc.*. New York, June 1999, pp. 198–202.
[23] J. A. Roubos and M. Setnes, "Compact fuzzy models through complexity reduction and evolutionary optimization," in *Proc. 9th IEEE Conf. Fuzzy Syst.*, San Antonio, TX, May 2000, pp. 762–767.
[24] J. Yen and L. Wang, "Application of statistical information criteria for optimal fuzzy model construction," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 362–371, Aug. 1998.
[25] J. C. Bezdek, J. M. Keller, R. Krishnapuram, L. I. Kuncheva, and N. R. Pal, "Will the *real* iris data please stand up?," *IEEE Trans. Fuzzy Syst.*, vol. 7, pp. 368–369, June 1999.
[26] H. Ishibuchi and T. Nakashima, "Voting in fuzzy rule-based systems for pattern classification problems," *Fuzzy Sets Syst.*, vol. 103, pp. 223–238, 1999.
[27] E. Anderson, "The Irises of the Gaspe peninsula," *Bull. Amer. Iris Soc.*, vol. 59, pp. 2–5, 1935.
[28] J. C. Bezdek, T. R. Reichherzer, G. S. Lim, and Y. Attikiouzel, "Multiple-prototype classifier design," *IEEE Trans. Syst., Man, Cybern.*, pt. C, vol. 28, pp. 67–79, Feb. 1998.
[29] L. I. Kuncheva and J. C. Bezdek, "Nearest prototype classification: Clustering, genetic algorithms, or random search?," *IEEE Trans. Syst., Man, Cybern.*, pt. C, vol. 28, pp. 160–164, Feb. 1998.
[30] A. L. Corcoran and S. Sen, "Using real-valued genetic algorithms to evolve rule sets for classification," in *Proc. 1st IEEE Conf. Evolut. Computat.*, Orlando, FL, June 1994, pp. 120–124.
[31] O. Nelles, A. Fink, R. Babuška, and M. Setnes, "Comparison of two construction algorithms for Takagi-Sugeno fuzzy models," in *Proc. EUFIT (CD-Rom)*, Aachen, Germany, Sept. 1999.
[32] O. Nelles and R. Isermann, "Basis function networks for interpolation of local linear models," in *Proc. IEEE Conf. Decision Contr.*, Kobe, Japan, Dec. 1996, pp. 470–475.

Magne Setnes was born in Bergen, Norway, in 1970. He received the B.Sc. degree in robotics from the Kongsberg College of Engineering, Norway, in 1992, the M.Sc. (electrical engineering) and the Chartered Designer (information technology) degrees from the Delft University of Technology, The Netherlands, in 1995 and 1997, respectively.

Currently, he is with Heineken Technical Services, Research and Development, Zoeterwoude, The Netherlands. His interests include fuzzy systems, beer drinking, and computational intelligence techniques for modeling, control, and decision making.

Hans Roubos was born in Sprang-Capelle, The Netherlands, in 1973. He received the M.Sc. degree in bioprocess engineering from the Wageningen University, The Netherlands, in 1997. He is currently working toward the Ph.D. degree on model-based process development of fed-batch fermentation processes at both the Control Laboratory and Kluyver Laboratory for Biotechnology, Delft University of Technology, The Netherlands.

For eight months he was with the European FAMIMO Project at the Control Laboratory, Delft University of Technology, The Netherlands, working on MIMO fuzzy model-based predictive control. His research interests include GAs and fuzzy systems for modeling, identification and control, hybrid models, and biotechnological processes.