A Two-Stage Evolutionary Process for Designing TSK Fuzzy Rule-Based Systems

Oscar Cordón and Francisco Herrera

***Abstract*—Nowadays, fuzzy rule-based systems are successfully applied to many different real-world problems. Unfortunately, relatively few well-structured methodologies exist for designing them and, in many cases, human experts are not able to express the knowledge needed to solve the problem in the form of fuzzy rules. Takagi–Sugeno –Kang (TSK) fuzzy rule-based systems were enunciated in order to solve this design problem because they are usually identified using numerical data. In this paper we present a two-stage evolutionary process for designing TSK fuzzy rule-based systems from examples combining a generation stage based on a (μ, λ)-evolution strategy, in which the fuzzy rules with different consequents compete among themselves to form part of a preliminary knowledge base, and a refinement stage, in which both the antecedent and consequent parts of the fuzzy rules in this previous knowledge base are adapted by a *hybrid evolutionary process* composed of a genetic algorithm and an evolution strategy to obtain the final Knowledge Base whose rules cooperate in the best possible way.**

Some aspects make this process different from others proposed until now: the design problem is addressed in two different stages, the use of an *angular coding* of the consequent parameters that allows us to search across the whole space of possible solutions, and the use of the available knowledge about the system under identification to generate the initial populations of the Evolutionary Algorithms that causes the search process to obtain good solutions more quickly. The performance of the method proposed is shown by solving two different problems: the fuzzy modeling of some three-dimensional surfaces and the computing of the maintenance costs of electrical medium line in Spanish towns. Results obtained are compared with other kind of techniques, evolutionary learning processes to design TSK and Mamdani-type fuzzy rule-based systems in the first case, and classical regression and neural modeling in the second.

***Index Terms*— Evolution strategies, evolutionary algorithms, genetic algorithms, learning, Takagi–Sugeno –Kang (TSK) fuzzy rule-based systems, TSK knowledge base.**

I. Introduction

FUZZY rule-based systems (FRBS's) are now considered as one of the most important applications of fuzzy set theory suggested by Zadeh in 1965 [1]. These kinds of systems constitute an extension of the classical rule-based systems because they deal with fuzzy rules instead of classical logic rules. Thanks to this, they have been successfully applied to a wide range of problems presenting uncertainty and vagueness in different ways [2]. In particular, the most promising results have been obtained by the fuzzy logic controllers [3], the FRBS's for control problems.

Several tasks have to be performed in order to design an intelligent system of this kind for a concrete application. They can be grouped into two main tasks: to design the FRBS Inference System, i.e., to select the fuzzy operators considered to make inference, and to obtain an accurate knowledge base (KB) comprising the known knowledge about the problem being solved. The latter used to be the most important and difficult, due to the fact that human experts are not sometimes able to express their knowledge in the form of fuzzy if-then rules. This has forced researchers to develop automatic techniques for performing this task.

Over the last few years, many different approaches have been presented taking genetic algorithms (GA's) [4] as their base, obtaining the so called genetic fuzzy systems (GFS's) [5], [6] or, more generically, *evolutionary fuzzy systems* (EFS's) when an Evolutionary Algorithm (EA) [7] is used instead of a GA. For a wider description of some of these approaches refer to [5], [6], [8], and for an extensive bibliography see [9].

In this paper, we present a two-stage evolutionary process to automatically learn Takagi–Sugeno –Kang (TSK) KB's from examples. The learning process is divided into two stages: the *generation* and *refinement* stages. The first one, based on the combination of an inductive algorithm and a (μ, λ)-evolution strategy ((μ, λ)-ES) [7], will allow us to automatically generate a preliminary TSK-type KB for a concrete problem when a training data set representing its behavior is available. It is able to decide the number of rules composing the KB and to determine their consequent parameters generating a locally optimal KB. The second stage is addressed by means of a *hybrid GA-ES process* (a genetic local search process) that works with a population of KB's, taking the preliminary definition obtained in the previous stage as a base, to obtain another one presenting an optimal global behavior.

The performance of the EFS proposed is analyzed in the solving of two different problems: the fuzzy modeling of some three-dimensional surfaces and the computing of the maintenance costs of electrical medium line in Spanish towns. In the first case, results obtained are compared with other Mamdani and TSK-type FRBS evolutionary design processes (a Mamdani-type two-stage EFS based on the Wang and Mendel fuzzy rule generation method [10], a three-stage Mamdani-type EFS [11], and a TSK-type EFS [12], [13]). In the second application, the same EFS's are considered, and

Manuscript received March 1, 1998; revised May 10, 1999. This work was supported by CICYT under Grant TIC96-0778. This paper was recommended by Associate Editor T. H. Lee.

The authors are with the Department of Computer Science and Artificial Intelligence, E.T.S. Ingeniería Informática, University of Granada, 18071 Granada, Spain (e-mail: ocordon@decsai.ugr.es).

Publisher Item Identifier S 1083-4419(99)08063-2.

other kind of techniques such as classical regression and neural modeling as well.

In order to put this into effect, this paper is set up as follows. The next section presents some guidelines about the TSK FRBS and its design. In Section III, a new coding to represent TSK fuzzy model rule consequents is introduced, allowing us to explore the whole possible solution space when using EA's. Section IV introduces the structure of the EFS proposed. Both stages composing it are described in Sections V and VI, respectively, while Section VII shows the experiments developed and the different results obtained. In Section 8, some concluding remarks are pointed out. Finally, two Appendices briefly describing ES's and collecting different tables of the results obtained in the fuzzy modeling of the three functions, respectively, are presented.

II. TSK Fuzzy Rule-Based Systems

The TSK fuzzy model was first presented in [14]. It is based on rules in which the consequent is not a linguistic variable, as in the Mamdani-type fuzzy model, but a function of the input variables. This kind of rule usually presents the following form:

$$\begin{aligned}&\text{If } X_1 \text{ is } A_1 \text{ and } \cdots \text{ and } X_n \text{ is } A_n\\ &\quad \text{then } Y = p_1 \cdot X_1 + \cdots + p_n \cdot X_n + p_0\end{aligned}$$

where X_i are the system input variables, A_i are fuzzy sets specifying their meaning, and Y is the output variable.

The output of a FRBS using a KB composed of m TSK rules is computed as the weighted average of the individual rule outputs, Y_i, $i = 1 \cdots m$, in the following way:

$$\frac{\sum_{i=1}^{m} h_i \cdot Y_i}{\sum_{i=1}^{m} Y_i}$$

where $h_i = T(A_1(x_1), \cdots, A_n(x_n))$ is the matching between the antecedent part of the rule i and the current system inputs, $x = (x_1, \cdots, x_n)$, with T being a t-norm.

The design process of these kinds of FRBS's is easier than others due to two main reasons. On the one hand, the only design decision that has to be made to set up the Inference System is to choose the t-norm T considered to compute the matching for the rule antecedents. On the other hand, TSK FRBS's were originally designed to be identified from numerical examples, as stated by their creators: "is quite useful to give a way to model control actions using numerical data about the system behavior" [14].

Many different techniques have been employed until now to derive the TSK KB from examples since Takagi and Sugeno first presented a process based on the least squares method [14]. For example, Neural Networks [15], [16] and gradient descent methods [17] have been considered. The use of EA's, either specific, GA's [12], [18] and EE's [19]; or hybrid [19], [20], has increased over the last few years.

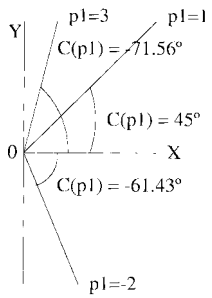


Fig. 1. Examples of angular coding.

III. A New Coding Scheme to Represent TSK Rule Consequents

There is a problem when designing TSK FRBS's using EA's. Usually, an EA needs to know the intervals in which each problem variable is defined to solve a specific problem. This information is necessary to define the genetic coding of the possible solutions and to perform evolution on them using the genetic operators. Unfortunately, this information is not available in the problem of learning the TSK rule consequent parameters.

This problem has usually been solved by the authors [12], [13], [18]–[20] by fixing sufficiently large values for the low and high interval extremes. This is not a bad idea because the powerful search of the EA allows us to obtain good solutions working in this way but presents the drawback that not all the solution space is considered, so it may not be possible to find the global problem solution since the value of some of the parameters may lie outside the intervals considered.

In this section, we propose a new coding scheme, called *angular coding*, which was first presented in [21]. It is based on encoding the values of the angles instead of the tangent values for each TSK rule consequent parameter, thus allowing us to have all the variables lying in the same fixed interval and to represent the whole space of possible solutions.

As can be seen, the partial linear relation defined by the consequent of a TSK rule determines a geometrical figure in the corresponding hyperspace. For example, when working with a system with a single input variable, each TSK rule output, $Y = p_1 \cdot X + p_0$, represents a straight line in a part of the plane determined by the rule fuzzy input subspace. When a greater number of inputs iv is considered, each output relation corresponds to a hyperplane of dimension iv in a part of the $(iv + 1)$-dimensional space $\mathbb{R}^{iv+1}$.

Bearing this in mind, and focusing on the case of one single variable for simplicity, we know that the real value p_1 in the consequent $Y = p_1 \cdot X + p_0$ is simply the tangent of the angle existing between this line and the X axis. Thus, if we code the angle value instead of the tangent one by means of the function

$$C\colon \mathbb{R} \to \left(-\frac{\pi}{2}, \frac{\pi}{2}\right); \qquad C(x) = \arctan(x)$$

all the possible values of the parameter p_1 lie inside the interval $(-\pi/2, \pi/2)$. Fig. 1 shows some examples.

As may be observed graphically in the figure, using very short intervals, a very large part of the possible solution space may be represented. For example, when considering

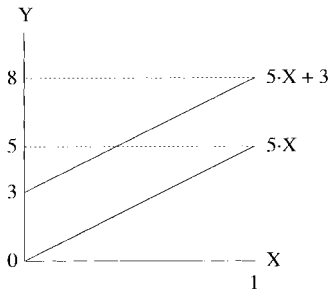


Fig. 2. Geometrical interpretation of the parameter p_0.

the interval $[-20, 20]$, we work with the angular interval $[-87.13°, 87.13°]$. Thus, approximately only 3.3% of the search space (more or less 5.74°) is not taken into account. This justifies the fact that the EA-based design processes which consider fixed intervals allow us to obtain good results. Anyway, it seems more appropriate to represent the whole possible solution space when performing a search toward a global solution.

As regards the parameter p_0, when working in the plane it determines the movement of the straight line from the origin along the Y axis, as shown in Fig. 2. Since the values of the parameter p_0 may be very different from one TSK rule to another, the consideration of a fixed interval is not a good solution for their evolutionary learning, and angular coding becomes a powerful tool to solve the problem. In this case, there is no geometric interpretation in the coding (remember that p_0 does not correspond to the tangent of any angle in the concrete hyperspace), we only use angular coding to translate an interval with undefined extents, $\mathbb{R}$, to another with defined ones, $(-\pi/2, \pi/2)$. Therefore, with this transformation we can use the EA to adequately search in the solution space to learn the values of these parameters.

IV. Structure of the Proposed EFS

The EFS presented in this paper is composed of the following two stages:

1) An *evolutionary generation process* for learning a preliminary TSK KB from examples. This first process is based on an iterative algorithm that studies the existence of data in the different fuzzy input subspaces. Each time data are located in one of them, the process applies a *TSK rule consequent learning method* to determine the existing partial linear input-output relation, taking the data located in this input subspace, a subset of the global data set, as a base. The latter method is based on a (μ, λ)-ES [7] using the angular coding proposed in the previous Section and a local measure of error, and takes into account the knowledge contained in the said training data subset to improve the search process.
2) An *evolutionary refinement process* for adjusting both the consequent and the antecedent parts of the fuzzy rules in the preliminary KB obtained from the first stage. The second process is composed of a special real-coded GA (a genetic local search algorithm [22]) which includes an $(1 + 1)$-ES [7] as a genetic operator to improve the search process. The algorithm works on chromosomes encoding the whole preliminary definition of the KB obtained and globally adjusts this definition. The fitness function considered in this case is based on a global error measure—the mean square error computed on the training data set—more adequate for the purpose followed. The available knowledge is again considered to generate the initial population of the GA. In this case, the preliminary definition of the TSK KB is taken into account for this generation.

The following two sections will present each one of the EFS stages, respectively.

V. Evolutionary Generation Process

In this section, we introduce the evolutionary generation process that was first presented as a single design process in [21]. First of all, the TSK rule consequent learning method is introduced. Then we propose the use of the knowledge contained in the training data set to improve the search process. Finally we present the algorithm of the generation process, which makes use of the two previous aspects.

A. TSK Rule Consequent Learning Method

In this method, the (μ, λ)-ES (see Appendix I) is considered to define TSK rule consequent parameters. The dimension n of the object variable vector $\vec{x}$ is determined by the number of input variables in the problem under control. When there are iv input variables, there are $n = iv+1$ parameters to learn in the TSK rule consequent. The $\vec{x}$ part of the individuals forming the (μ, λ)-ES population is built by encoding the possible values using *angular coding.*

The evolutionary learning is guided by a fitness function composed of a local error measure. This will allow us to obtain an optimal TSK rule consequent in the fuzzy subspace defined by the rule antecedents. The expression of the measure used is the following [18]:

$$\sum_{e_l \in E} h_l \cdot (ey^l - S(ex^l))^2$$

where E is the set of input-output data pairs $e_l = (ex_1^l, \cdots, ex_{iv}^l, ey^l)$ located in the fuzzy input subspace defined by the rule antecedent, $h_l = T(A_1(ex_1^l), \cdots, A_{iv}(ex_{iv}^l))$ is the matching between the antecedent part of the rule and the input part of the current data pair ex^l, and $S(ex^l)$ is the output provided by the TSK fuzzy rule when it receives ex^l as input.

The object variables of the individuals in the first population are generated in the way shown in the next subsection, taking into account the knowledge contained in the input-output data set. As regards the initialization of the remaining vectors, the components of $\vec{\sigma}$ are set to 0.001, and the ones in $\vec{\alpha}$, when considered, are set to arctan (1).

B. Using Available Knowledge in the Design Process

To develop the knowledge-based generation of the initial population, we compute the following indices and obtain the

following set from the input-output data set E:

$$y_{med} = \frac{\sum_{e_l \in E} ey^l}{|E|}; \qquad y_{\min} = \min_{e_l \in E}\{ey^l\}$$
$$y_{\max} = \max_{e_l \in E}\{ey^l\}$$
$$h_{\max} = \max_{e_l \in E}\{h_l\}; \qquad E_\theta = \{e_l \in E / h_l \geq \theta \cdot h_{\max}\}.$$

Therefore, we generate the initial population of the proposed ES in three steps as follows:

1) Generate 1 individual initiating parameters p_i, $i = 1, \cdots, iv$, to zero, and parameter p_0 to the angular coding of y_{med}.
2) Generate γ individuals, with $\gamma \in \{0, \cdots, \mu - 1\}$ defined by the EFS designer, initiating parameters p_i, $i = 1, \cdots, iv$, to zero, and p_0 to the angular coding of values computed at random in the interval $[y_{\min}, y_{\max}]$.
3) Generate the remaining $\mu - (\gamma + 1)$ individual initiating parameters p_i, $i = 1, \cdots, iv$, to the angular coding of values computed at random in the interval $(-\pi/2, \pi/2)$, and p_0 to the angular coding of a value computed from a randomly selected element e in E_θ ($\theta \in [0.5, 1]$ is provided by the EFS designer as well) in such a way that e belongs to the hyperplane defined by the TSK rule consequent generated. Thus, we shall ensure that this hyperplane intersects with the swarm of points contained in E_θ, the most significative ones from E.

Since with small angular values, large search space zones are covered, it seems interesting to generate small values for the parameters p_i in this third step. To do so, we make use of a modifier function that assigns greater probability of appearance to smaller angles according to a parameter q, also provided by the EFS designer. We use the following function:

$$f\colon [0, 1] \times \{-1, 1\} \rightarrow \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$$
$$f(x, z) = z \cdot \frac{\pi}{2} \cdot x^q.$$

Hence, the generation of the individuals is performed in this third step as follows:

For $j = 1, \cdots, \mu - (\gamma + 1)$ do

a) For $i = 1, \cdots, iv$ do
   1) Generate x at random in [0, 1].
   2) Generate z at random in $\{-1, 1\}$.
   3) Set p_i to $f(x, z)$.

b) Generate the value of p_0:
   1) Select e at random from E_θ.
   2) Set p_0 to $ey - \sum_{k=1}^{iv} C^{-1}(p_k) \cdot ex_k$, where $C^{-1}(\beta) = \tan(\beta)$ is the inverse of C.

C. Algorithm of the Evolutionary Generation Process

The generation process proposed is developed by means of the following steps:

1) Consider a fuzzy partition of the input variable spaces obtained from the expert information (if it is available) or equally partitioning them in a number of linguistic terms, each one with an associated fuzzy set defining its meaning. In this paper, we shall work in the latter way using symmetrical fuzzy partitions with triangular membership functions (see Fig. 3).
2) For each multidimensional fuzzy subspace obtained by combining the individual input variable subspaces using the *and* conjunction do:
   a) Build the set E' composed of the input-output data pairs $e \in E$ that are located in this subspace.
   b) If $|E'| \neq 0$, i.e., if there is any data in this space zone, apply the TSK rule consequent learning method over the data set E' to determine the partial linear input-output relation existing in this subspace. Therefore, no rules are considered in the fuzzy subspaces in which no data are located.
   c) Add the generated rule to the preliminary KB.

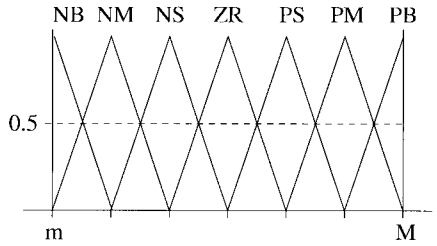


Fig. 3. Fuzzy partition used.

VI. Evolutionary Refinement Process

The evolutionary refinement process is a tuning process that takes a TSK KB as input and adjusts the preliminary definitions of the antecedent membership functions and consequent parameters according to the global behavior of the KB evolved in the problem being solved, represented as a training data set. It is composed of a special real-coded GA including an $(1 + 1)$-ES as another genetic operator to improve the search process, guided by a global error measure over the training data set. We describe the hybrid EA components below.

A. Representation

A chromosome C encoding a TSK KB definition is composed of two different parts, C^1 and C^2, the first one corresponding to the definition of the fuzzy membership functions considered in the antecedent part of the different fuzzy rules in the KB, and the other to the consequent parameters.

A computational way to characterize a triangular membership function is by using a parametric representation achieved by means of the 3-tuple (a, b, c). Therefore, a primary fuzzy partition as the one shown in Fig. 3 can be represented by an array composed by N 3-tuples ($3 \cdot N$ real values) (a_l, b_l, c_l), $l = 1, \cdots, N$, with N being the number of terms forming the linguistic variable term set. The complete definition of all the input variable fuzzy partitions for a problem in which iv input variables are involved is encoded into the first part C^1 of each chromosome C_j in the population. C^1 is built by joining the partial representations of each one of the iv input variable

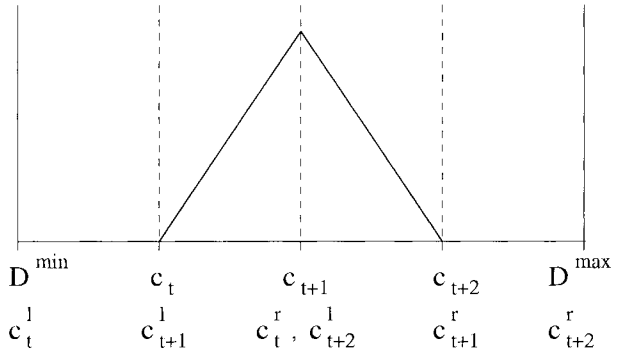


Fig. 4. Example of triangular membership function and intervals of performance for the refinement process.

fuzzy partitions as is shown below:

$$C_i^1 = (a_{i1}, b_{i1}, c_{i1}, \cdots, a_{iN_i}, b_{iN_i}, c_{iN_i}),$$
$$C^1 = C_1^1 C_1^2 \cdots C_1^{iv}.$$

Each one of the triangular fuzzy sets $D_{ij} = (a_{ij}, b_{ij}, c_{ij})$, $i = 1, \cdots, iv$, $j = 1, \cdots, N_i$, defining these preliminary fuzzy partitions are allowed to vary freely in any meaningful way in an interval of performance $[D_{ij}^{\min}, D_{ij}^{\max}]$. The extremes of these intervals are computed before running the refinement process according to the preliminary fuzzy partition definitions provided by the FRBS designer, in the following way:

$$[D_{ij}^{\min}, D_{ij}^{\max}] = \left[a_{ij} - \frac{b_{ij} - a_{ij}}{2}, c_{ij} + \frac{c_{ij} - b_{ij}}{2}\right].$$

Therefore, the interval of performance of each gene in C^1 will depend on the fuzzy membership function to which it is associated. Each one of these intervals of performance will be the interval of adjustment for the corresponding gene, $c_t \in [c_t^l, c_t^r]$. If $(t \bmod 3) = 1$ then c_t is the left value of the support of a fuzzy set, which is defined by the three parameters (c_t, c_{t+1}, c_{t+2}) and the intervals of performance are the following:

$$c_t \in [c_t^l, c_t^r] = [D^{\min}, c_{t+1}]$$
$$c_{t+1} \in [c_{t+1}^l, c_{t+1}^r] = [c_t, c_{t+2}]$$
$$c_{t+2} \in [c_{t+2}^l, c_{t+2}^r] = [c_{t+1}, D^{\max}]$$

with $D^{\min}$ and $D^{\max}$ being the extremes of the interval of performance in the fuzzy set defined by the 3-tuple (c_t, c_{t+1}, c_{t+2}). These values are the only ones defining the intervals of adjustment of the c_t's that remain constant during the GA run. Fig. 4 shows an example of these intervals.

As regards the second part of the chromosome, C^2, it encodes the consequent parameters of each fuzzy rule in the preliminary definition of the TSK KB. Thus, it is composed of $m \cdot (iv+1)$ genes, where m stands for the number of rules in the KB and $iv+1$ for the number of consequent parameters for TSK fuzzy rule:

$$C_i^2 = (p_{i0}, p_{i1}, \cdots, p_{i\,iv}), \qquad i = 1, \cdots, m,$$
$$C^2 = C_1^2 C_2^2 \cdots C_m^2.$$

Since all of these parameters are encoded by using the proposed angular coding, the interval of performance of all the genes in C^2 is the same, $(-\pi/2, \pi/2)$.

Now, the fundamental underlying mechanisms of a GA, formation of an initial gene pool, fitness function, and genetic operators are developed.

B. Initial Gene Pool

The second stage uses the available knowledge to initialize the first population as well. In this case, we make use of the preliminary definition of the KB being optimized in order to perform this task. With M being the GA population size, the initial population generation process is performed in three steps as follows:

1) The preliminary definition of the KB taken as process input is encoded directly into a chromosome, denoted as C_1.
2) The following $M/2 - 1$ chromosomes are initiated by generating, at random, the first part, C^1, with each gene being in its respective interval of performance, and by encoding the preliminary definition of the rule consequent parameters in C^2.
3) The remaining $M/2$ are set up by generating C^1 in the same way followed in the previous step, and by generating the values for C^2 by adding a random value distributed following a normal distribution $N(0, d)$ to the values in the C^2 part of the previous chromosomes.

C. Evaluation of Individual Fitness

The fitness function is based on a training input-output data set, E, and a global error measure, the mean square error (SE). In this way, the adaptation value associated to an individual C_j is obtained by computing the error between the outputs given by the TSK FRBS using the KB encoded in the chromosome and those contained in the training data set. The fitness function is thus represented by the following expression:

$$F(C_j) = \frac{1}{2|E|} \sum_{e_l \in E} (ey^l - S(ex^l))^2$$

with the same equivalences presented in Section V.

D. Genetic Operators

The *selection procedure* considered is Baker's stochastic universal sampling [23], in which the number of any structure offspring is limited by the floor and ceiling of the expected number of offspring, together with the elitist selection. As regards the genetic operators, the ones described in the following subsections are going to be considered.

1) Mutation: We shall use Michalewicz's nonuniform mutation operator [4], which has demonstrated an accurate behavior when working with real coding schemes. It works as follows.

If $C_v^t = (c_1, \cdots, c_k, \cdots, c_H)$ is a chromosome and the gene c_k was selected for this mutation ($c_k \in [c_{kl}, c_{kr}]$), the

new mutated value, c'_k, is

$$c'_k = \begin{cases} c_k + \Delta(t, c_{kr} - c_k), & \text{if } a = 0 \\ c_k - \Delta(t, c_k - c_{kl}), & \text{if } a = 1 \end{cases}$$

where a is a random number that may have a value of zero or one, and the function $\Delta(t, y)$ returns a value in the range $[0, y]$ such that the probability of $\Delta(t, y)$ being close to zero increases as t increases:

$$\Delta(t, y) = y(1 - r^{(1-t/T)^b})$$

where

- r random number in the interval $[0, 1]$;
- T maximum number of generations;
- b parameter chosen by the user, which determines the degree of dependency with the number of iterations.

This property causes this operator to make a uniform search in the initial space when t is small, and a very local one in later stages.

2) Crossover: We shall work with another genetic operator which has shown good behavior for real-coded GA's, the max-min-arithmetical crossover. This crossover operator was proposed in [24] and has been widely used in the field of EFS's [11], [25]–[27]. It works as follows.

If $C_v^t = (c_1, \cdots, c_k, \cdots, c_H)$ and $C_w^t = (c'_1, \cdots, c'_k, \cdots, c'_H)$ are to be crossed, the following four offspring are generated:

$$\begin{aligned} &C_1^{t+1} = aC_w^t + (1-a)C_v^t \\ &C_2^{t+1} = aC_v^t + (1-a)C_w^t \\ &C_3^{t+1} \text{ with } c_{3k}^{t+1} = \min\{c_k, c'_k\} \\ &C_4^{t+1} \text{ with } c_{4k}^{t+1} = \max\{c_k, c'_k\}. \end{aligned}$$

This operator can use a parameter a which is either a constant, or a variable whose value depends on the age of the population. The resulting descendants are the two best of the four aforementioned offspring.

3) Evolution Strategy: The last genetic operator to be applied consists of an (1 + 1)-ES. This optimization technique has been selected and integrated into the genetic recombination process in order to locally refine the best individuals in each generation, following the assumptions of the so-called *genetic local search* [22]. The ES will be applied over a percentage δ of the best different population individuals existing in the current genetic population. This idea has already been applied in the field of EFS's [25].

The basis of the (1 + 1)-ES employed are to be briefly presented in Appendix I. The coding scheme and the fitness function considered are the same as those used in the GA. Thus, the only changes to be performed have to be done in the generic ES mutation scheme, and the great majority of them only when working with the first part of the individual, C^1. In this case, the following two changes have to be put into effect.

- *Definition of Multiple Step Sizes:* The mutation strength depends directly on the value of the parameter σ, which determines the standard deviation of the normally distributed random variable z_i (see Appendix I). In our case, the step size σ cannot be a single value because the membership functions encoded in the first part of the chromosome are defined over different universes and so require different order mutations. Therefore, a step size $\sigma_i = \sigma \cdot s_i$ for each component in C^1 is going to be used in the (1 + 1)-ES. Anyway the relations of all σ_i were fixed by the values s_i and only the common factor σ is adapted following the assumptions presented in [7].
- *Incremental Optimization of the Individual Parameters:* Usually, the different parent components are not related and the ES adapts all of them at the same time. Unfortunately, in our problem each three correlative parameters (x_0, x_1, x_2) in C^1 define a triangular-shaped membership function, and the property $x_0 \le x_1 \le x_2$ must be verified in order to obtain meaningful fuzzy sets. Therefore, there is a need to develop an incremental optimization of the individual parameters because the intervals of performance for each one of them will depend on the value of any of the others.

  As we have commented in the description of the coding scheme, a global interval of performance (in which the three parameters defining the membership function may vary freely) is defined for each fuzzy set involved in the optimization process. With $C_{ij} = (x_0, x_1, x_2)$ being the membership function currently adapted, the associated interval of performance is $[C_{ij}^{\min}, C_{ij}^{\max}] = [x_0 - (x_1 - x_0)/2, x_2 + (x_2 - x_1)/2]$. The incremental adaptation is based on generating the mutated fuzzy set $C'_{ij} = (x'_0, x'_1, x'_2)$ by first adapting the modal point x_1 obtaining the mutated value x'_1 defined in the interval $[x_0, x_2]$, and then adapting the left and right points x_0 and x_2 obtaining the values x'_0 and x'_2 defined, respectively, in the intervals $[C_{ij}^{\min}, x'_1]$ and $[x'_1, C_{ij}^{\max}]$. It may be clearly observed that the progressive application of this process allows us to obtain fuzzy sets freely defined in the said interval of performance.

The value of the parameter $s(x_i)$ determining the particular step sizes, $\sigma_i = \sigma \cdot s(x_i)$, is computed each time the component x_i is going to be mutated. When $i = 1$, the modal point is being adapted, and then $s(x_1)$ is equal to $\text{Min}(x_1 - x_0, x_2 - x_1)/2$. In the other two cases, $i = 0$ and $i = 2$, $s(x_0) = \text{Min}(x_0 - C_{ij}^{\min}, x'_1 - x_0)/2$ and $s(x_2) = \text{Min}(x_2 - x'_1, C_{ij}^{\max} - x_2)/2$. Hence, when σ takes value 1 at the first ES generation, the obtaining of a large quantity of z_i normal values performing a successful x_i mutation (i.e., the corresponding $x'_i = x_i + z_i$ with $z_i \sim N_i(0, \sigma'^2_i)$ lying in the expected interval for x_i) is ensured. If the mutated value lies outside, it is assigned the value of the interval extent closest to $x_i + z_i$.

However, when working with the second part of the chromosome, C^2, the latter problem does not appear. In this case, the different components are not related and the mutation can be performed in its usual way. The only change that has to be made is to adapt the step size to the components in C^2. As all of them are defined over the same interval of performance, $(-\pi/2, \pi/2)$, they all will use the same step size $\sigma_i = \sigma \cdot s_i$ with $s_i = 0.000\,01$.

VII. Experiments and Results Obtained

A. Fuzzy Modeling of Some 3-D Surfaces

The first application selected to analyze the accuracy of the method proposed is the fuzzy modeling of three three-dimensional surfaces presenting different characteristics. The associated functions and the variable universes of discourse considered are as follows, while their graphical representations are collected in Fig. 5:

$$F_1(x_1, x_2) = x_1^2 + x_2^2, \quad x_1, x_2 \in [-5, 5],\ F_1(x_1, x_2) \in [0, 50]$$

$$F_2(x_1, x_2) = 10 \cdot \frac{x_1 - x_1 x_2}{x_1 - 2x_1 x_2 + x_2}, \quad x_1, x_2 \in [0, 1],\ F_2(x_1, x_2) \in [0, 10]$$

$$F_3(x_1, x_2) = e^{x_1} \cdot \sin^2 x_2 + e^{x_2} \cdot \sin^2 x_1, \quad x_1, x_2 \in [-8, 8],\ F_3(x_1, x_2) \in [0, 5836].$$

We consider four EFS's to model the described surfaces:

1) **M1**: two-stage Mamdani EFS based on obtaining a complete KB by first deriving the RB by means of Wang and Mendel (WM) method [10], and then defining the DB using the descriptive genetic tuning process presented in [11];
2) **M2**: three-stage Mamdani EFS presented in [11];
3) **T1**: single-stage TSK EFS presented in [12], [13] (improved by considering a real coding scheme);
4) **T2**: two-stage TSK EFS proposed in this paper.

For each function, a training data set uniformly distributed in the three-dimensional definition space has been obtained experimentally (the three data sets are composed of 1681, 674, and 1089 pieces of data, respectively). Three other data sets (of size 168, 67, and 108, i.e., 10% of the corresponding training set one) have been randomly generated for their use as test sets to evaluate the generalization capability of the TSK FRBS's generated.

The initial DB used in processes **M1**, **M2**, and **T2** is constituted by three primary, equally partitioned fuzzy partitions (two corresponding to the input variables and one associated to the output one, the latter in the case of both Mamdani-type EFS's) formed by *seven linguistic terms* with triangular-shaped fuzzy sets giving meaning to them (as shown in Fig. 3), and the adequate scaling factors to translate the generic universe of discourse into the one associated with each problem variable.

To design the inference system in the Mamdani-type FRBS's generated by means of the first two processes, we have selected the *minimum t-norm* playing the role of the implication and conjunctive operators, and the *center of gravity weighted by the matching* strategy acting as the defuzzification operator [28]. In the TSK-type ones obtained from processes **T1** and **T2**, the role of conjunctive operator is played by the minimum t-norm as well.

We have performed different runs of the proposed process, **T2,** using two of the four usual combinations of dimensions of vectors $\vec{\sigma}$ and $\vec{\alpha}$ (see Appendix I), $(n_\sigma, n_\alpha) = \{(n, 0), (n, (n \cdot (n-1)/2))\}$, and three different values for the parameter

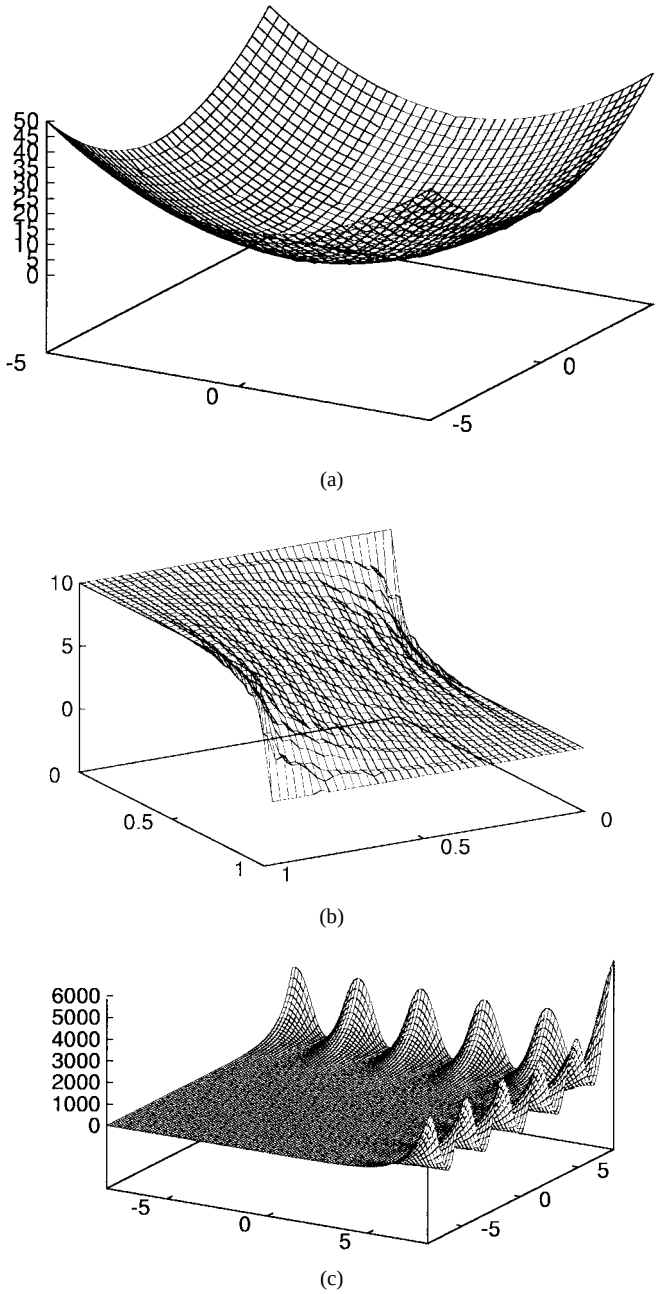


Fig. 5. Graphical representations of (a) F_1, (b) F_2, and (c) F_3.

δ, $\delta = \{0, 0.1, 0.2\}$, defining the percentage of population individuals to which the ES is applied in the refinement stage.

The remaining parameter values for EFS **T2** are as follows.

1) *Evolutionary Generation Process:* 500 iterations, $\mu = 15$, $\lambda = 100$, $\gamma = 0.2 \cdot \mu = 3$, $\theta = 0.7$, $q = 5$, $\vec{r} = (r_{\vec{x}}, r_{\vec{\sigma}}, r_{\vec{\alpha}}) = (3, 2, 0)$, and $\vec{\zeta} = (\zeta_{\vec{x}}, \zeta_{\vec{\sigma}}, \zeta_{\vec{\alpha}}) = (\mu, \mu, 1)$.
2) *Evolutionary Refinement Process:* 1000 GA iterations, $N = 61$, $P_c = 0.6$, $P_m = 0.1$ (per individual), $a = 0.35$, $b = 5$, 25 $(1+1)$-ES iterations, $c = 0.9$, $d = 0.001$.

The results obtained in the different experiments developed with design process **T2** are shown in Tables III–V, collected in Appendix II, where SE^x_{tra} and SE^x_{tst} stand for values obtained by the specific TSK FRBS designed in the SE measure

TABLE I
FUZZY MODELING OF THE THREE FUNCTIONS USING THE FOUR EFS'S

	F_1			F_2			F_3		
EFS	$\#R^{F_1}$	TRA^{F_1}	TST^{F_1}	$\#R^{F_2}$	TRA^{F_2}	TST^{F_2}	$\#R^{F_3}$	TRA^{F_3}	TST^{F_3}
M1	49	0.358522	0.377134	49	0.060296	0.028621	49	52705.00	40999.98
		2.836611	2.465526		4.820671	0.665405		1817.60	1591.62
M2	62	0.335871	0.262573	98	0.027351	0.016716	100	44523.62	34242.32
		3.171968	2.263699		2.260658	0.671948		1778.55	**1088.94**
T1	56	0.579574	0.233781	56	0.067474	0.025405	56	68971.09	56671.50
		5.268021	3.169657		10.000000	2.070384		5113.01	2549.98
T2	49	**0.006592**	**0.006568**	49	**0.015290**	**0.001763**	49	**28691.20**	**19838.80**
		0.351849	**0.332455**		**2.559241**	**0.189650**		**1455.16**	1103.95

computed over the training and test data sets, respectively (x can be equal to g and r standing for the generation and refinement stages). All the KB's learned are composed of 49 fuzzy rules. The final values included in the last two columns of each table, noted as SE_{tra}^{avg} and SE_{tst}^{avg}, respectively, are computed as an average of three EFS runs with different values for the random seed in order to give us more information about the process accuracy.

The results collected in these tables may help us to decide good values for the different EFS parameters. In view of them, the best results are obtained when not considering the angle vector in the evolutionary generation process (μ, λ)-ES ((n_σ, n_α) $= (n, 0)$), and the best value for δ parameter in the second stage depends on the specific application. The use of the $(1 + 1)$-ES in the refinement stage leads sometimes to better TSK KB definitions, while causing an undesirable overlearning (better approximation but worse generalization) in other cases.

To analyze the performance of the proposed process, we compare it with the other three mentioned earlier. The best results obtained in the three applications are collected in Table I, where $\#R$ stands for the number of rules in the corresponding KB and TRA and TST for the results obtained over the training and test data sets, respectively. In each cell, the number at the top corresponds to the SE and the one at the bottom to the maximum linear error. The best result for each application and measure appears in boldface.

From an analysis of these results, the good behavior presented by the proposed EFS may be observed. The FRBS's designed using it are more accurate to a high degree than the ones based on the other three EFS's in the fuzzy modeling of the three functions. Moreover, they have demonstrated the robustness of our evolutionary learning process. The optimization of the local and global error measures in two stages leads us to obtain fuzzy models with the best results in the SE and the maximum linear error measures over the training and test data sets.

To illustrate the behavior of the proposed EFS with respect to the remaining ones, the graphical representation of the best fuzzy modeling obtained for each function using them is shown, respectively, in Figs. 6–8.

B. Real-World Electrical Application

In Spain, electrical industries do not charge the energy bill directly to the final user, but they share the ownership of an enterprise called "Red Eléctrica Española" (R.E.E.) which gets all payments and then distributes them according to some complex criteria (amount of power generation of every company, number of customers, etc.).

Recently, some of these companies asked to revise the rules. One of the proposed modifications involved a redistribution of the maintenance costs of the network. Since maintenance costs depend on the total length of electrical line each company owns, and on their kind (high, medium, urban low and rural low voltage) it was necessary to know the exact length of every kind of line each company was maintaining.

To compute the maintenance costs of town medium voltage lines, there is a need to know which would be the total line length if the installation made would have been the optimal one. Clearly, it is impossible to obtain this value by directly measuring it, since the medium voltage lines existing in a town have been installed incrementally, according to its own electrical needs in each moment.

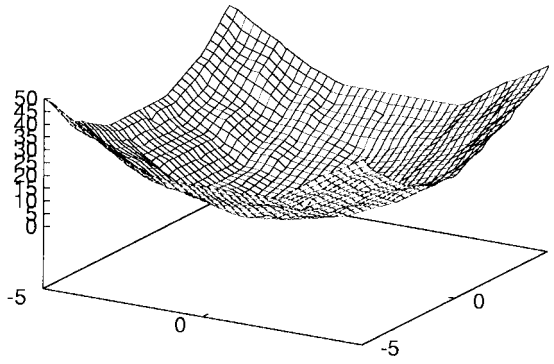
Therefore, we need to solve the problem using other kinds of techniques, which are able to relate some characteristics of a certain town with its maintenance cost [29]. In this paper, we consider evolutionary fuzzy modeling techniques and compare its behavior with classical regression and neural techniques.

To solve the problem, we were provided with data related to four different characteristics of the towns:

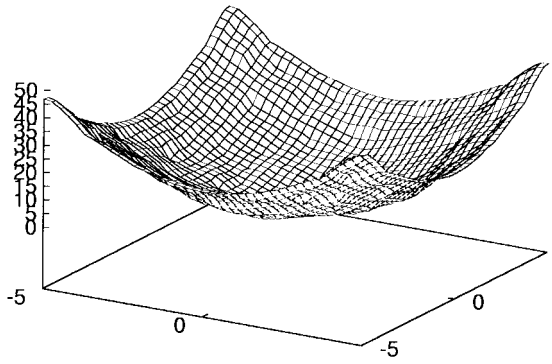
x_1 sum of the lengths of all streets in the town;
x_2 total area of the town;
x_3 area that is occupied by buildings;
x_4 energy supply to the town;

and to the maintenance costs of line (y) in each one of them in a sample of 1059 simulated towns. Our objective was to relate the last variable (maintenance costs) with the other four ones by the different said techniques.

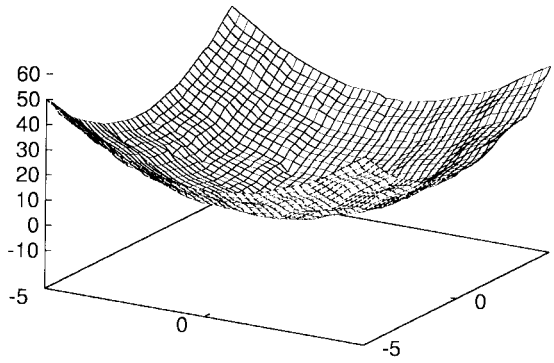
As regards classical methods, we have considered linear and polynomial regression, and neural network models. The parameters of the polynomial models were fitted by Levenberg–Marquardt method and the neural model (a three-layer perceptron) was trained with the QuickPropagation Algorithm [30]. The number of neurons in the hidden layer was chosen to minimize the test error; note that the training error could be made much lower than the shown, but not without making the test error higher. We used four input, five hidden, and one output nodes.



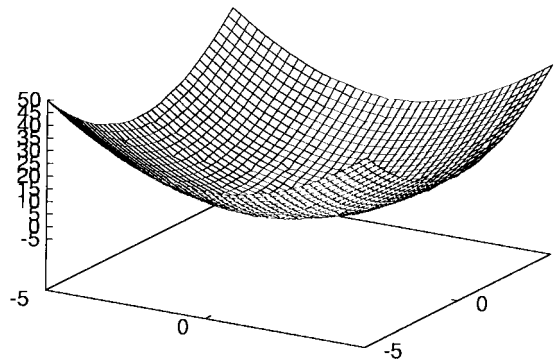
(a)



(b)

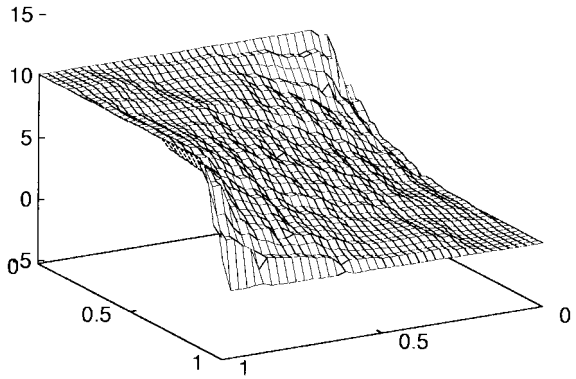


(c)

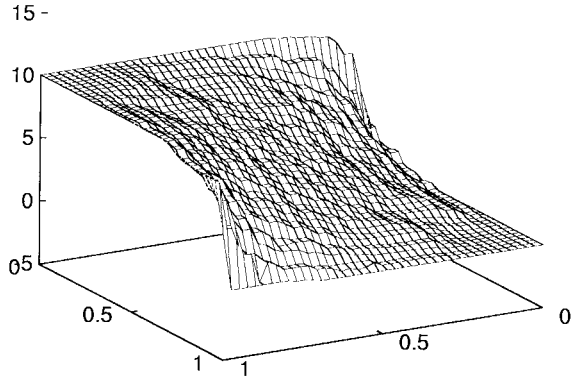


(d)

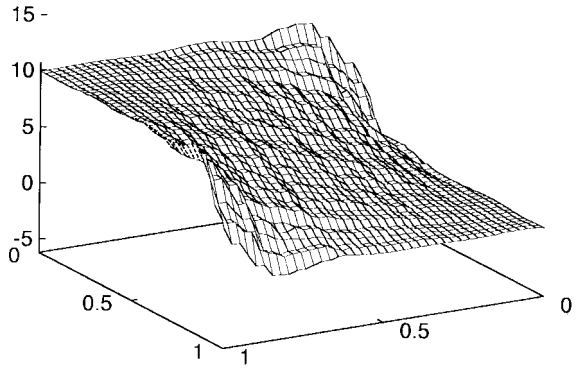
Fig. 6. Fuzzy modeling obtained for F_1 using EFS's (a) **M1** (top left), (b) **M2**, (c) **T1**, and (d) **T2**.



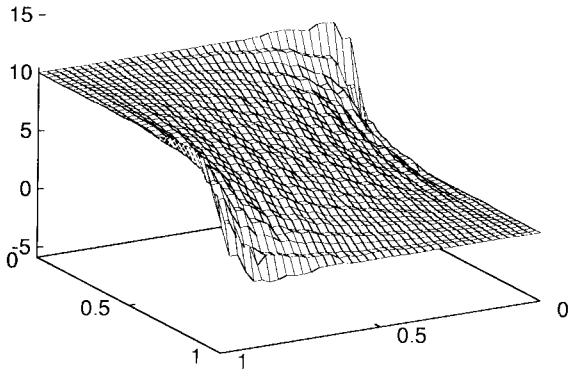
(a)



(b)



(c)



(d)

Fig. 7. Fuzzy modeling obtained for F_2 using EFS's (a) **M1**, (b) **M2**, (c) **T1**, and (d) **T2**.

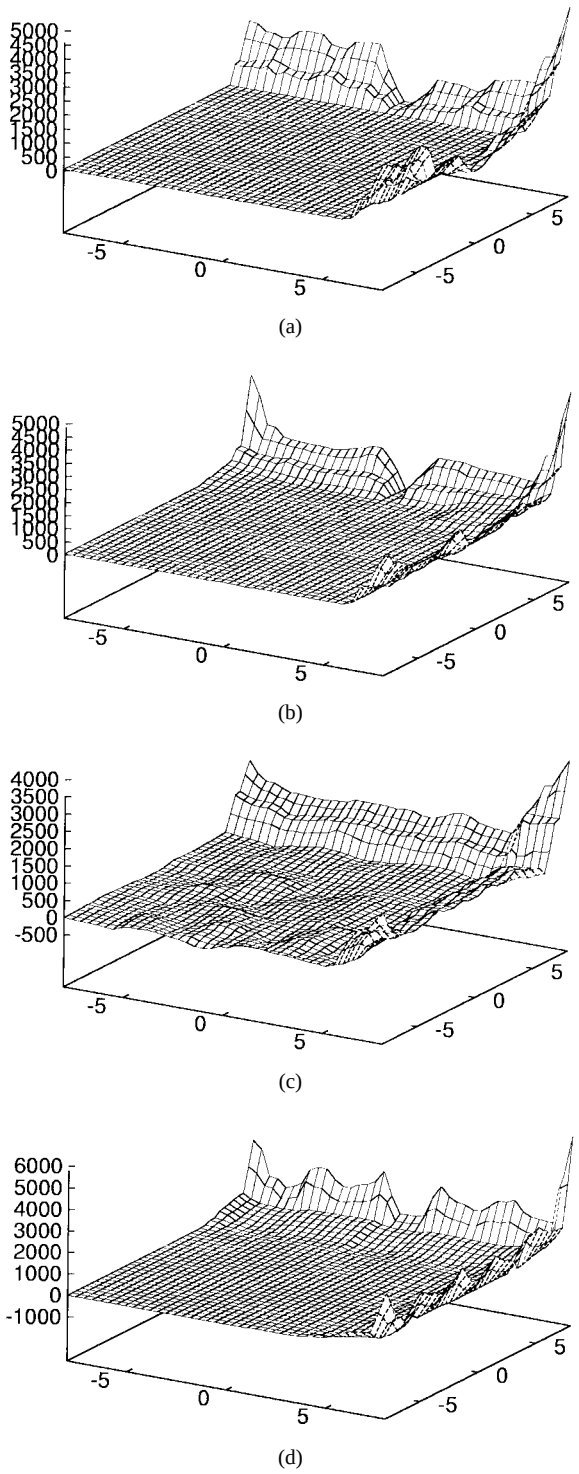


Fig. 8. Fuzzy modeling obtained for F_3 using EFS's (a) **M1**, (b) **M2**, (c) **T1**, and (d) **T2**.

For the fuzzy models, we have considered an initial DB constituted by five primary equally partitioned fuzzy partitions formed by *five linguistic terms* with triangular-shaped fuzzy sets associated.

To compare the mentioned techniques, we have divided the sample into two sets comprising 847 and 212 examples. SE values over these two sets are labeled SE_{tra} and SE_{tst}. Results obtained in the different experiments developed are shown in Table II, where column *COMPLEXITY* contains the number of parameters and the number of nodes in the parse tree of the expression, as well as the number of rules in the KB of the generated fuzzy model. The best fuzzy model generated by the EFS proposed has been obtained using the parameter values $(n_\sigma, n_\alpha) = (n, 0)$, and $\delta = 0.2$.

TABLE II
RESULTS OBTAINED IN THE ELECTRICAL PROBLEM SOLVING

METHOD	SE_{tra}	SE_{tst}	COMPLEXITY
Classical Regression:			
Linear	164662	36819	17 nodes, 5 par.
2th order polynomial	103032	45332	77 nodes, 15 par.
Neural Modeling:			
3 layer percept. 4-5-1	86469	33105	35 par.
Fuzzy Modeling:			
M1	20318	27615	66 rules
M2	19679	22590	63 rules
T1	149144	128942	6 rules
T2	**11073**	**11836**	268 rules

In view of them, we can conclude that the best result is obtained by the EFS proposed in this paper, although it is important to note that the TSK fuzzy model obtained from it is the most complex one. Moreover, three of the four fuzzy models clearly outperform classical non linear regression methods, being superior to the neural model. This result has great significance, because it means that neural networks performance can be achieved with a model with a high descriptive power. Even the TSK fuzzy models, the less interpretable ones, have associated a higher level of description than neural models, because of the possibility of locally analyzing the model and of interpreting the antecedent part of the fuzzy rules.

VIII. CONCLUDING REMARKS

A two-stage EFS to design TSK FRBS's has been presented. The evolutionary process is based on a first stage which generates a preliminary definition of a KB rule by rule according to a local error measure and a second stage—guided by a global error measure—which globally refines the latter by tuning the antecedent membership function and consequent parameter definitions.

In order to put this into effect, the design process makes use of three well-known EA's—GA's, and (μ, λ) and $(1 + 1)$-ES's—in the two stages composing it, and considers two new concepts, the *angular coding* of the rule consequent parameters, and the use of the available knowledge about the problem being solved to improve the search process.

The performance of the proposed EFS has been analyzed in two different problems—one of them an electrical real-world application—and it has been compared with other EFS's for designing Mamdani and TSK-type FRBS's and with classical techniques. It has shown very good results.

TABLE III
FUZZY MODELING OF F_1 USING EFS T2

PARAMETERS				GENERATION		REFINEMENT		AVERAGE	
n_σ	n_α	δ	Run	SE^g_{tra}	SE^g_{tst}	SE^r_{tra}	SE^r_{tst}	SE^{avg}_{tra}	SE^{avg}_{tst}
3	0	0	1			0.006592	0.006568		
3	0	0	2	0.050862	0.051457	0.008108	0.009008	0.006980	0.007504
3	0	0	3			0.006240	0.006938		
3	0	0.1	1			0.007828	0.007737		
3	0	0.1	2	0.050862	0.051457	0.006627	0.007717	0.006921	0.007498
3	0	0.1	3			0.006310	0.007042		
3	0	0.2	1			0.007268	0.008018		
3	0	0.2	2	0.050862	0.051457	0.006566	0.007711	0.007150	0.007805
3	0	0.2	3			0.007618	0.007687		
3	3	0	1			0.010610	0.011168		
3	3	0	2	0.073856	0.056382	0.012539	0.013282	0.011431	0.011646
3	3	0	3			0.011145	0.010490		
3	3	0.1	1			0.010338	0.010349		
3	3	0.1	2	0.073856	0.056382	0.011458	0.013186	0.011173	0.011928
3	3	0.1	3			0.011725	0.012251		
3	3	0.2	1			0.012412	0.015150		
3	3	0.2	2	0.073856	0.056382	0.010139	0.010494	0.010868	0.012276
3	3	0.2	3			0.010055	0.011185		

APPENDIX I
EVOLUTIONARY STRATEGIES

ES's [7] were initially developed by Rechenberg and Schwefel in 1964 with a strong focus on building systems capable of solving difficult real-valued parameter optimization problems. The natural representation was a vector of real-valued parameters primarily manipulated by mutation operators designed to perturb them in useful ways.

There are different kinds of ES's. Next, we shall introduce the two of them considered in this paper.

A. The (1 + 1)-Evolution Strategy

The first ES algorithm, the so-called (1 + 1)-ES, was based on only two individuals per generation, one parent and one descendent. The parent string is evolved by applying a mutation operator to each one of its components. The mutation strength is determined by a value σ, a standard deviation of a normally distributed random variable. This parameter is associated to the parent and it is evolved in each process step as well. If the evolution has been performed successfully, then the descendent substitutes the parent in the next generation. The process is iterated until a certain finishing condition is satisfied.

The mutation operator **mut** has two components. The first one, $\mathbf{mu}_\sigma$, evolves the value of the standard deviation σ using Rechenberg's 1/5-success rule

$$\sigma' = \mathbf{mu}_\sigma(\sigma) = \begin{cases} \dfrac{\sigma}{\sqrt[n]{c}}, & \text{if } p > \dfrac{1}{5} \\ \sigma \cdot \sqrt[n]{c}, & \text{if } p < \dfrac{1}{5} \\ \sigma, & \text{if } p = \dfrac{1}{5} \end{cases}$$

where p is the relative frequency of successful mutations and c is a constant determining the updating amount of σ.

The second one, $\mathbf{mu}_x$, mutates each component in the real coded string by adding normally distributed variations with standard deviation σ' ($z_i \sim N_i(0, \sigma'^2)$) to it:

$$x' = \mathbf{mu}_x(x) = (x_1 + z_1, \cdots, x_n + z_n).$$

B. (μ, λ)-Evolution Strategy

This second kind of ES is based on performing evolution on a population of μ possible n-dimensional solutions, obtaining λ offspring and selecting the best μ from them to form the new population. The offspring are obtained by first recombining a single or some parents in a single n-dimensional vector of object variables, and then creating a new one from this by applying mutations with identical or different standard deviations to each object variable. The main quality of the algorithm is its ability to incorporate the most important parameters from the strategy (standard deviations and correlation coefficients of normally distributed mutations) into the search process, such that adaptation also takes place in the strategy parameters according to the current local topology of the search space. This property is called *self-adaptation* [7].

Therefore, each population individual consists of three vectors, $\vec{a} = (\vec{x}, \vec{\sigma}, \vec{\alpha})$, representing, respectively, the object variable, the standard deviation and the rotation angle values. The vector $\vec{x}$ has n dimensions, equal to the number of problem variables. The n_σ dimensions of a vector $\vec{\sigma}$ can be up to n (in this case, each object variable x_i, $i = 1, \cdots, n$, has associated a different step size σ_i), and n_α can be up to $(2 \cdot n - n_\sigma) \cdot (n_\sigma - 1)/2$. The set of strategy parameters consisting of standard deviations and rotation angles provides a complete description of the generalized n-dimensional distribution with an expectation value vector $\vec{0}$. Anyway, n_α may be set to zero, indicating that the rotation angles are not considered. The more usual values for n_σ and n_α are the following [7]: $(n_\sigma, n_\alpha) = \{(1, 0), (n, 0), (n, n \cdot (n-1)/2), (2, n-1)\}$.

TABLE IV
FUZZY MODELING OF F_2 USING EFS T2

PARAMETERS				GENERATION		REFINEMENT		AVERAGE	
n_σ	n_α	δ	Run	SE^g_{tra}	SE^g_{tst}	SE^r_{tra}	SE^r_{tst}	SE^{avg}_{tra}	SE^{avg}_{tst}
3	0	0	1			0.016275	0.004175		
3	0	0	2	0.029530	0.004223	0.015290	0.001763	0.015954	0.003002
3	0	0	3			0.016298	0.003069		
3	0	0.1	1			0.009134	0.011710		
3	0	0.1	2	0.029530	0.004223	0.007656	0.024311	0.007453	0.018500
3	0	0.1	3			0.005569	0.019480		
3	0	0.2	1			0.007190	0.027646		
3	0	0.2	2	0.029530	0.004223	0.004297	0.023401	0.005142	0.025348
3	0	0.2	3			0.003940	0.024999		
3	3	0	1			0.014823	0.006048		
3	3	0	2	0.044170	0.058803	0.014322	0.004197	0.012686	0.004308
3	3	0	3			0.008913	0.002680		
3	3	0.1	1			0.015229	0.009054		
3	3	0.1	2	0.044170	0.058803	0.009706	0.003222	0.011458	0.005823
3	3	0.1	3			0.009441	0.005193		
3	3	0.2	1			0.013696	0.006228		
3	3	0.2	2	0.044170	0.058803	0.008015	0.008940	0.009331	0.006155
3	3	0.2	3			0.006284	0.003299		

TABLE V
FUZZY MODELING OF F_3 USING EFS T2

PARAMETERS				GENERATION		REFINEMENT		AVERAGE	
n_σ	n_α	δ	Run	SE^g_{tra}	SE^g_{tst}	SE^r_{tra}	SE^r_{tst}	SE^{avg}_{tra}	SE^{avg}_{tst}
3	0	0	1			27875.615	28289.871		
3	0	0	2	46104.042	34909.113	28239.669	24423.539	28666.989	26569.314
3	0	0	3			29885.685	26994.533		
3	0	0.1	1			29403.921	30013.451		
3	0	0.1	2	46104.042	34909.113	30676.896	25773.236	29590.672	25.208.497
3	0	0.1	3			28691.201	19838.804		
3	0	0.2	1			27798.847	22102.109		
3	0	0.2	2	46104.042	34909.113	30608.802	23747.402	28246.468	23299.519
3	0	0.2	3			26331.755	24049.048		
3	3	0	1			42419.597	35670.113		
3	3	0	2	82619.757	56026.214	46582.660	24168.587	41613.676	27155.026
3	3	0	3			35838.773	21626.378		
3	3	0.1	1			43105.207	44799.027		
3	3	0.1	2	82619.757	56026.214	43585.875	32632.888	40557.106	35086.336
3	3	0.1	3			34980.242	27827.101		
3	3	0.2	1			35922.062	28787.998		
3	3	0.2	2	82619.757	56026.214	39166.980	21030.007	37788.360	24416.094
3	3	0.2	3			38276.042	23430.279		

The following algorithm generically describes the behavior of the (μ, λ)-ES. The parameter t stands for the number of the current generation and $P(t)$ for the population in it.

1) Initialize and evaluate $P(0)$. Initialize $t \leftarrow 0$.
2) Recombine ζ of the μ individuals of $P(t)$ λ times, by using one of the following gene recombination mechanisms: $r \in \{0, 1, 2, 3\}$, $i = 1, \cdots, n + n_\sigma + n_\alpha$

$$a'_i = \begin{cases} a_{S,i}; \quad S \sim U(\{1, \cdots, \zeta\}) \text{ equal } \forall i \\ \quad r = 0, \text{ no recombination} \\ \dfrac{\sum_{j=1}^{\zeta} a_{j,i}}{\zeta} \quad r = 1, \text{ global intermediary} \\ u \cdot a_{S,i} + (1-u) \cdot a_{T,i}; \quad S, T \sim U(\{1, \cdots, \zeta\}) \\ \quad r = 2, \text{ local intermediary} \\ a_{S,i}; \quad S \sim U(\{1, \cdots, \zeta\}) \quad r = 3 \text{ discrete.} \end{cases}$$

This operation generates λ individuals forming $P'(t)$.

3) Mutate $P'(t)$ by adapting the λ individuals to obtain λ offspring forming $P''(t)$ in the way.
   a) Mutate the values of $\vec{\sigma}'$ to obtain the array $\vec{\sigma}''$

$$\vec{\sigma}'' = (\sigma'_1 \cdot \exp(z_1 + z_0), \cdots, \sigma'_{n_\sigma} \cdot \exp(z_{n_\sigma} + z_0))$$

   where $z_i \sim N(0, 1/\sqrt{2 \cdot \sqrt{n}}^2)$, $i = 1, \cdots, n_\sigma$, and $z_0 \sim N(0, 1/\sqrt{2 \cdot n}^2)$.
   b) Mutate the values of $\vec{\alpha}'$ to obtain the vector $\vec{\alpha}''$

$$\vec{\alpha}'' = (\alpha'_1 + z_1, \cdots, \alpha'_{n_\alpha} + z_{n_\alpha})$$

   where $z_i \sim N(0, 0.0873^2)$, $i = 1, \cdots, n_\alpha$.
   c) Mutate the values of $\vec{x}'$ to obtain the vector $\vec{x}''$

$$\vec{x}'' = (x'_1 + cor_1(\vec{\sigma}'', \vec{\alpha}''), \cdots, x'_n + cor_n(\vec{\sigma}'', \vec{\alpha}''))$$

   where $cor(\vec{\sigma}'', \vec{\alpha}'')$ is a normally distributed random vector of correlated values.

4) Evaluate $P''(t)$ and select the best μ individuals to form $P(t+1)$.
5) Set the counter of generations $t \leftarrow t+1$
6) If not (termination condition), then go to 2, else Stop.

For more information about (μ, λ)-ES refer to [7].

APPENDIX II
RESULTS OBTAINED IN THE FUZZY MODELING OF THE FUNCTIONS

This Appendix contains the results obtained in the different experiments developed with the proposed TSK EFS in the fuzzy modeling of the three-dimensional surfaces F_1, F_2, and F_3 presented in Section VII-A. These results are collected in Tables III–V, respectively.

ACKNOWLEDGMENT

The authors would like to thank to L. Sánchez, Oviedo University, for the electrical engineering application from Hidroeléctrica del Cantábrico and for solving it by means of classical and neural techniques.

REFERENCES

[1] L. A. Zadeh, "Fuzzy sets," *Inf. Contr.*, vol. 8, pp. 338–353, Feb. 1965.
[2] W. Pedrycz, Ed., *Fuzzy Modeling. Paradigms and Practice.* Norwell, MA: Kluwer, 1996.
[3] D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control.* Berlin, Germany: Springer-Verlag, 1993.
[4] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs.* Berlin, Germany: Springer-Verlag, 1996.
[5] O. Cordón and F. Herrera, "A general study on genetic fuzzy systems," in *Genetic Algorithms in Engineering and Computer Science*, J. Periaux, G. Winter, M. Galán, and P. Cuesta, Eds. New York: Wiley, 1995, pp. 33–57.
[6] O. Cordón, F. Herrera, F. Hoffmann, and L. Magdalena, *Genetic Fuzzy Systems. Evolutionary Tuning and Learning of Fuzzy Knowledge Bases.* Singapore: World Scientific, in press.
[7] T. Bäck, *Evolutionary Algorithms in Theory and Practice.* Oxford, U.K.: Oxford Univ. Press, 1996.
[8] F. Herrera and J. L. Verdegay, Eds., *Genetic Algorithms and Soft Computing.* Berlin, Germany: Physica-Verlag, 1996.
[9] O. Cordón, F. Herrera, and M. Lozano, "A classified review on the combination fuzzy logic-genetic algorithms bibliography: 1989–1995," in *Genetic Algorithms and Fuzzy Logic Systems. Soft Computing Perspectives*, E. Sanchez, T. Shibata, and L. Zadeh, Eds. Singapore: World Scientific, 1997, pp. 209–241.
[10] L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 1414–1427, Nov. 1992.
[11] O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples," *Int. J. Approx. Reas.*, vol. 17, pp. 369–407, Dec. 1997.
[12] M. A. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithms," in *Proc. 2nd IEEE Int. Conf. Fuzzy Systems*, San Francisco, CA, Mar. 1993, pp. 613–617.
[13] ——, "Embedding apriori knowledge into an integrated fuzzy system design method based on genetic algorithms," in *Proc. 5th Int. Fuzzy Systems Assoc. World Congr.*, Seoul, Korea, July 1993, pp. 1293–1296.
[14] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 116–132, Feb. 1985.
[15] H. Takagi, N. Suzuki, T. Koda, and Y. Kojima, "Neural networks designed on approximate reasoning architecture and their applications," *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 752–760, 1992.
[16] J. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 665–684, May 1993.
[17] H. Nomura, L. Hayashi, and N. Wakami, "A self-tuning method of fuzzy control by descent method," in *Proc. 4th Int. Fuzzy Systems Assoc. World Congr.*, Brussels, Belgium, July 1991, pp. 155–158.
[18] J. Yen and W. Gillespie, "Integrating global and local evaluations for fuzzy model identification using genetic algorithms," in *Proc. 6th Int. Fuzzy Systems Assoc. World Congr.*, Sao Paulo, Brazil, July 1995, pp. 121–124.
[19] M. A. Lee and R. Saloman, "Hybrid evolutionary algorithms for fuzzy system design," in *Proc. 6th Int. Fuzzy Systems Assoc. World Congr.*, Sao Paulo, Brazil, July 1995, pp. 269–272.
[20] W. Wienholt, "Improving a fuzzy inference system by means of evolution strategy," in *Proc. 4th Fuzzy Days*, Dortmund, Germany, June 1994, pp. 163–172.
[21] O. Cordón and F. Herrera, "Evolutionary design of TSK fuzzy rule-based systems using (μ, λ)-evolution strategies," in *Proc. 6th IEEE Int. Conf. Fuzzy Systems*, Barcelona, Spain, July 1997, vol. 1, pp. 509–514.
[22] J. Y. Suh and D. V. Gutch, "Incorporating heuristic information on genetic search," in *Proc. 2nd Int. Conf. Genetic Algorithms*, 1987, pp. 100–107.
[23] J. E. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proc. 2nd Int. Conf. Genetic Algorithms*, 1987, pp. 14–21.
[24] F. Herrera, M. Lozano, and J. L. Verdegay, "Fuzzy connectives based crossover operators to model genetic algorithms population diversity," *Fuzzy Sets Syst.*, vol. 92, no. 1, pp. 21–30, 1997.
[25] O. Cordón and F. Herrera, "Hybridizing genetic algorithms with sharing scheme and evolution strategies for designing approximative fuzzy logic controllers," *Fuzzy Sets Syst.*, to be published.
[26] F. Herrera, M. Lozano, and J. L. Verdegay, "Tuning fuzzy controllers by genetic algorithms," *Int. J. Approx. Reas.*, vol. 12, pp. 299–315, 1995.
[27] ——, "A learning process for fuzzy control rules using genetic algorithms," *Fuzzy Sets Syst.*, vol. 100, pp. 143–158, 1998.
[28] O. Cordón, F. Herrera, and A. Peregrín, "Applicability of the fuzzy operators in the design of fuzzy logic controllers," *Fuzzy Sets Syst.*, vol. 86, pp. 15–41, Feb. 1997.
[29] O. Cordón, F. Herrera, and L. Sánchez, "Solving electrical distribution problems using hybrid evolutionary data analysis techniques," *Appl. Intell.*, vol. 10, no. 1, pp. 5–24, 1999.
[30] S. E. Fahlman, "Fast learning variations on back-propagation: an empirical study," in *Proc. 1998 Connectionist Models Summer School*, Pittsburgh, PA, 1988, pp. 38–51.

Oscar Cordón was born in Cadiz, Spain, in 1972. He received the M.S. degree in computer science in 1994 and the Ph.D. degree in computer science in 1997, both from the University of Granada, Granada, Spain.

He is an Assistant Professor with the Department of Computer Science and Artificial Intelligence, University of Granada, where he is a member of the Approximate Reasoning and Artificial Intelligence Research Group. His current main research interests are in the fields of fuzzy rule-based systems, fuzzy and linguistic modeling, fuzzy logic controllers, fuzzy classification, genetic fuzzy systems, and combination of fuzzy logic and genetic algorithms.

Francisco Herrera received the M.S. degree in mathematics in 1988 and the Ph.D. degree in mathematics in 1991, both from the University of Granada, Granada, Spain. He is an Associate Professor with the Department of Computer Science and Artificial Intelligence, University of Granada.

He coedited (with J. L. Verdegay) the book *Genetic Algorithms and Soft Computing* (Berlin, Germany: Physica Verlag) and two journal special issues, one on genetic fuzzy systems for control and robotics for the *International Journal of Approximate Reasoning* and another on genetic fuzzy systems for the *International Journal of Intelligent Systems* (with L. Magdalena). His research interests include decision making problems in fuzzy environment, fuzzy rule-based systems, machine learning, genetic algorithms, genetic fuzzy systems, and combination of fuzzy logic and genetic algorithms.