Applied Intelligence 10, 5–24 (1999)

Solving Electrical Distribution Problems Using Hybrid Evolutionary Data Analysis Techniques

OSCAR CORDÓN AND FRANCISCO HERRERA
Department of Computer Science and Artificial Intelligence, University of Granada, 18071, Granada, Spain
ocordon@decsai.ugr.es
herrera@decsai.ugr.es

LUCIANO SÁNCHEZ
Department of Computer Science, University of Oviedo, Oviedo, Spain
luciano@lsi.uniovi.es

Abstract. Real-world electrical engineering problems can take advantage of the last Data Analysis methodologies. In this paper we will show that Genetic Fuzzy Rule-Based Systems and Genetic Programming techniques are good choices for tackling with some practical modeling problems. We claim that both evolutionary processes may produce good numerical results while providing us with a model that can be interpreted by a human being. We will analyze in detail the characteristics of these two methods and we will compare them to the some of the most popular classical statistical modeling methods and neural networks.

Keywords: electrical engineering, data analysis, evolutionary algorithms, genetic algorithm program, genetic fuzzy rule-based systems

1. Introduction

Spanish electrical market is evolving towards competence, but it is not completely deregulated. There are four major agents (Iberdrola, Endesa, Unión Fenosa, Hidroeléctrica del Cantábrico) that simultaneously own almost all power generation plants and distribution networks in Spain. Due to the characteristics of the Spanish market, any of these companies could have a great influence over the price of the electrical energy, which has an obvious strategic importance in the economical development of the country.

Hence, the Spanish government decided some years ago to nationalize high voltage lines and to separate distribution and generation markets, thus forcing the mentioned companies to act as two different entities each. This way, generation plants sell the energy they produce in a partially regulated market and the distribution companies buy the energy in this same market. (Bilateral contracts between suppliers of energy and consumers are also allowed in certain cases.) The energy bill that users pay is not completely received by the companies, but the payments are redistributed according to some complex criteria (amount of power generation of every company, number of customers, etc.) with the aim of equilibrating the spanish market.

Recently, some of these companies have asked to revise these rules. We will discuss some models originated in the new proposals of redistribution of the maintenance costs of the network. Maintenance costs depend (among other factors) on the total length of electrical line each company owns, and on its kind (high, medium, urban low and rural low voltage). To justify the expenses of the companies, models of the length of line are used. This is so because, besides high voltage lines can be easily measured, a problem comes when trying to estimate the maintenance costs of medium and low voltage lines. Specially in the latter case, low voltage line is contained in cities and villages, and it is very difficult and expensive to measure it. This kind of line uses to be very convoluted and companies can serve as much as 10000 small nuclei.

An indirect method for determining the length of line is needed.

Moreover, some of the modifications of the payment structure were based on the optimal network and not on the actual network. It is argued that low and medium voltage lines existing in a town have been installed incrementally and thus the actual distribution is far from the optimal one. If lengthy networks are rewarded, there are not incentives to modernize obsolete distribution networks.

The reasons that make models of the length of line important in the spanish electrical market are clear. All companies develop their own models and the government also uses some of these models to decide the part of the payment that the companies receive. In this context, we were asked to solve two problems: to relate some characteristics of a certain village with the actual length of low voltage line contained in it, and to relate the maintenance cost of the network installed in certain towns with some other characteristics of these towns [1, 2]. In both cases, it would be preferable that the solutions obtained verify another requirement: they have not only to be numerically accurate in the problem-solving, but must be able to explain how a specific value is computed for a certain village or town. That is, it is interesting that *these solutions are interpretable by human beings to some degree*.

In this paper, we propose different solutions to both distribution problems in the field of Data Analysis (DA). DA can be considered as a process in which, starting from some given data sets, information about the respective application is generated. In this sense, DA can be defined as a search for structure in data. Since in our problems there is a need to find relationships between some variables (the village characteristics and the length of low voltage line in it, in the first case, and the town characteristics and its associated maintenance cost, in the second one) and these relationships must be compatible with some known data, it is clear that they may be solved by means of DA techniques.

The problem-solving techniques considered will make use of the Evolutionary Algorithms (EAs) [3] in the field of DA. We will analyze different hybrid evolutionary learning processes. Firstly, we will use Genetic Algorithm-Programming (GA-P) [4] algorithms for symbolic regression and later we will use Genetic Algorithms (GAs) [5] and Evolution Strategies (ESs) [6] to design Mamdani and TSK-type Fuzzy Rule-Based Systems (FRBSs) [7–9]. We will consider these two approaches to solve the mentioned problems and we will compare their performance with two other widely known techniques: classical regression (non-linear least squares model fitting to the set of data) and neural methods.

The paper is set up as follows. In Section 2, we introduce the use of the hybrid evolutionary techniques in the field of DA and present the GA-P and Genetic Fuzzy Rule-Based Systems (GFRBSs) [10]. Sections 3, 4, and 5 are devoted to present the two different approaches commented, the use of GA-P algorithms for symbolic regression problems and the use of GAs and ESs to optimize and design FRBSs. In Section 6, the electrical distribution problems are tackled by means of the proposed techniques and their performance is compared with other kind of techniques, classical regression and neural methods. Finally, some concluding remarks are pointed out in Section 7.

2. Preliminaries: Hybrid Evolutionary Techniques for Data Analysis

2.1. Framework

In DA, objects described by some attributes are considered and the specific values of the attributes are the data to be analyzed. Objects can be, for example, things, time series, process states, and so on. The overall goal is to find a structure (information) about these data. This leads to a complexity reduction in the considered application which allows us to obtain improved decisions based on the gained information.

The application of DA has a wide range and occurs in diverse areas where different problem formulations exist.

Different algorithmic methods for DA have been suggested in the literature, as Clustering algorithms [11], regression techniques [12], Neural Networks [13], FRBSs [7], EAs [3], etc.

As regards DA in the light of EAs, a representation of the information structure is considered and evolved until having an abstraction and generalization of the problem, reflected in the fitness function. For example, in [14] different approaches for learning in the framework of GAs are to be found.

Recently a lot of research efforts have been directed towards the combination of different methods for DA. In this way, EAs have been combined with different techniques either to optimize their parameters acting as evolutionary tuning processes or to obtain hybrid DA

methods, for example, evolutionary-neural processes [15], evolutionary regression models [16] and evolutionary fuzzy systems [17].

Next, we briefly introduce two specific evolutionary hybrid approaches, the GA-P to perform symbolic regressions and GFRBSs to optimize and design fuzzy models. Different particular developments in each field will be presented in Sections 3, 4 and 5.

2.2. *GA-P for Symbolic Regression*

Genetic Programming (GP) [16] has emerged as an effective mean of automatically generating computer programs to solve a variety of problems in many different problem domains, including the discovery of empirical formulae from numerical data.

GP methods generate symbolic expressions and can perform symbolic regressions. However, the way in which GP perform symbolic regressions is quite restrictive; the structure of an expression can be changed by crossover and mutation operations, but the value of the constants embedded in it—generated by the implementation program when the GP starts—can only be altered by mutations.

The GA-P [4] performs symbolic regression by combining the traditional GAs with the GP paradigm to evolve complex mathematical expressions capable of handling numeric and symbolic data. The GA-P combines GAs and GP, with each population member consisting of both a string and an expression as it is shown in Fig. 1. The GP part of the GA-P evolves the expression. The GA part concurrently evolves the coefficients used in the expressions. Most of the GA-P's elements are the same as in either of the traditional genetic techniques.

The GA-P and GP make selection and child generation similarly, except that the GA-P's structure requires separate crossover and mutation operators for the expression and coefficient string components. In the GA-P, crossover and mutation take place independently for the coefficient string and the expression component. Mutation and crossover rates for the coefficient string (using traditional GA methods) are independent from the rates for the expression part (using standard GP methods).

By fusing the GA's capability of value optimization and the GP's capability of creating mathematical equations, it improves the ability to describe the data. Therefore, the GA-P is a powerful DA tool.

A complete description of GA-P can be found in [4].

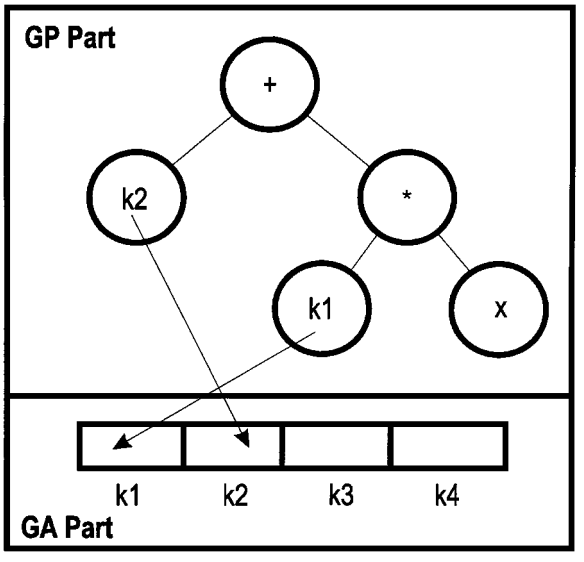


Figure 1. Member of population, GA-P algorithms.

2.3. *Genetic Fuzzy Rule-Based Systems*

Nowadays, one of the most important applications of the Fuzzy Set Theory suggested by Zadeh in 1965 [18] is the FRBSs. 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 [7, 19–21].

An FRBS presents two main components: (1) the Inference System, which puts into effect the fuzzy inference process needed to obtain an output from the FRBS when an input is specified, and (2) the Knowledge Base (KB) representing the known knowledge about the problem being solved, composed of the Rule Base (RB) constituted by the collection of fuzzy rules, and of the Data Base (DB) containing the membership functions defining their semantics.

There exist two different kinds of fuzzy rules in the literature according to the expression of the consequent:

1. Mamdani-type fuzzy rules consider a linguistic variable in the consequent [8]:

$$\text{IF } X_1 \text{ is } A_1 \text{ and } \ldots \text{ and } X_n \text{ is } A_n$$
$$\text{THEN } Y \text{ is } B$$

with $X_1, \ldots, X_n$ and Y being the input and output linguistic variables, respectively, and $A_1, \ldots, A_n$ and B being linguistic labels, each one of them having associated a fuzzy set defining its meaning.

2. TSK fuzzy rules are based on representing the consequent as a polynomial function of the inputs [9]:

$$\text{IF } X_1 \text{ is } A_1 \text{ and } \ldots \text{ and } X_n \text{ is } A_n$$
$$\text{THEN } Y = p_1 \cdot X_1 + \cdots + p_n \cdot X_n + p_0$$

with $X_1, \ldots, X_n$ being the input linguistic variables, Y being the output variable, and $p_0, p_1, \ldots, p_n$ being real-valued weights.

Knowledge-based methods are suitable for fuzzy DA. In this approach, fuzzy If-Then rules are formulated and a process of fuzzification, inference and defuzzification leads to the final decision. Different efforts have been made to obtain an improvement on system performance by incorporating learning mechanisms to modify the rules and/or membership functions in the KB.

With the aim of solving this problem, in the last few years, many different approaches have been presented taking EAs, usually GAs, as a base, to automatically derive the KB, constituting the so called GFRBSs [10] (see Fig. 2). GFRBSs are considered nowadays as an important branch of the Soft Computing area [22]. The promising results obtained by the EAs in the learning or tuning of the KB have extended the use of these algorithms in the last few years, see [23, 24].

It is possible to distinguish among three different groups of GFRBSs depending on the KB components included in the learning process: DB, RB, or both, i.e.,

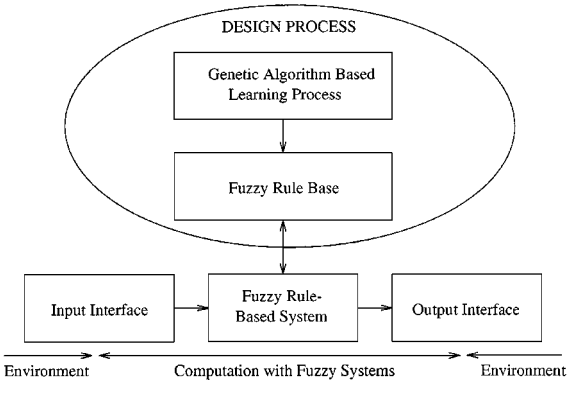


Figure 2. Genetic fuzzy rule-based systems.

KB [10]. The third group may be divided in two different subgroups depending on whether the KB learning is performed in a single process or in different stages. For a wider description of each GFRBS group see [10, 25], and for an extensive bibliography see [23], Section 3.13. Different approaches may be found in [10, 17].

On the other hand, GFRBSs included in the first group are usually called *evolutionary tuning processes*, while the ones belonging to the second and third ones are called *evolutionary learning processes*. In Sections 4 and 5, we will present some specific approaches belonging to the first and third families, respectively.

3. Interval-Valued GA-P for Symbolic Regression

In this Section we will introduce a modified version of the GA-P method, which we will call *Interval GA-P*. This approach—initially developed to solve an specific symbolic regression problem, [26]—is characterized by using interval values, instead of punctual ones, and by combining GA-P with local optimization techniques as well.

Regression analysis is concerned with the approximation of observed data by a function, when some variables (outputs) depend on other (inputs). We will adopt from here some usual conventions in statistical regression, and we will say that the two variables Y and X—where Y is the output we want to model and $X = (X_1, \ldots, X_n)$ is the input—are random variables. We will also understand that the regression analysis involves finding a function g, such that $g(X)$ is an admissible estimation of $E(Y \mid X)$. If the structure of g is unknown, the problem is named *symbolic regression*.

Symbolic regression produces a punctual estimation; anyway, sometimes it is necessary to obtain the margins in which we expect the output Y is, when the input variables X_i are known. Now, we should not look for a function g, but a multivalued mapping $\Gamma_\alpha : \text{Im}(X) \rightarrow I(\mathbb{R})$, where $I(\mathbb{R})$ is the set formed by all closed intervals in $\mathbb{R}$, such that the random set $\Gamma_\alpha \circ X : \Omega \rightarrow I(\mathbb{R})$ verifies

$$P\{\omega \in \Omega \mid Y(\omega) \in \Gamma_\alpha \circ X(\omega)\} \geq 1 - \alpha$$

for a given value of α.

We can assess this interval prediction in some different ways. We think that it is reasonable to admit that,

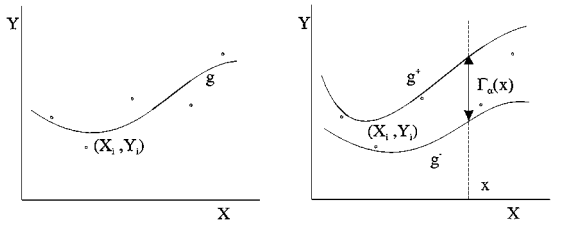


Figure 3. Linear and interval estimation.

given a value for α, the shorter Γ_α is, the better it is. So, if we define

$$\Gamma_\alpha \circ X = [g^- \circ X, g^+ \circ X]$$

for two continuous functions g^+ and g^- (see Fig. 3) the margin of validity will be better when

$$E(g^+ \circ X - g^- \circ X)$$

is as low as possible, constrained by

$$P\{\omega \in \Omega \mid g^- \circ X(\omega) < Y(\omega) < g^+ \circ X(\omega))\} \geq 1 - \alpha.$$

In other words, given a region

$$R_{(g^+,g^-)} = \{(x, y) \in \mathbb{R}^{d+1} \mid g^-(x) < y < g^+(x)\}$$

it must be true that

$$P\{\omega \in \Omega \mid (X, Y)(\omega) \in R_{(g^+,g^-)}\} \geq 1 - \alpha.$$

Let us suppose now that g^+ and g^- also depend on a function $h_\theta : \mathbb{R}^d \to \mathbb{R}$ in the following way:

$$[g^-(x), g^+(x)] = \{t \in \mathbb{R} \mid t = h_\theta(x), \theta \in [\theta_1^-, \theta_1^+] \times \cdots \times [\theta_m^-, \theta_m^+]\}$$

where the expression of h_θ is known except for the value of m parameters θ_i, and h_θ is continuous with respect to θ and x (so g^+ and g^- will also be continuous functions, as we had proposed). Then, for a random sample of size N, obtained from the random vector (X, Y),

$$((X_1, Y_1), \ldots, (X_N, Y_N))$$

we define θ_i^- and θ_i^+ to be the values that minimize

$$\frac{1}{N} \sum_{i=1}^{N} (g^+(X_i) - g^-(X_i))$$

constrained by

$$1 - \epsilon \leq \frac{1}{N} \sum_{i=1}^{N} I_{R_{(g^+,g^-)}}(X_i, Y_i)$$

for a given value of ϵ. Notice that $\alpha \neq \epsilon$; once chosen a value for ϵ, we can only estimate α by means of a second sample

$$((X'_1, Y'_1), \ldots, (X'_M, Y'_M)),$$

independent from the first one, by means of

$$\hat{\alpha}_M = 1 - \frac{1}{M} \sum_{i=1}^{M} I_{R_{(g^+,g^-)}}(X'_i, Y'_i).$$

The random variable $\hat{\alpha}_M$ follows a binomial distribution with parameters M and α and, by the strong law of the large numbers, it converges almost surely to the value α when $M \to \infty$.

In any case, to minimize $E(g^+ \circ X - g^- \circ X)$ with respect to the imposed constraints we should apply non linear constrained optimization techniques (say, for instance, non linear programming). And we cannot forget that the calculus is based in the knowledge of h_θ. Both problems (the search of the analytic expression of h and the values for θ_i^+ and θ_i^-) can be simultaneously solved by applying (with some modifications) the GA-P technique.

The adequacy of function h to a set of points is defined by the separation between g^+ and g^-, and both were defined in terms of h:

$$[g^-(x), g^+(x)] = \{t \in \mathbb{R} \mid t = h_\theta(x), \theta \in [\theta_1^-, \theta_1^+] \times \cdots \times [\theta_m^-, \theta_m^+]\}$$

that is, to find the value of $g^+(x)$ we should find the maximum of h inside the allowed range for its parameters,

$$g^+(x) = \max_{\mathbb{R}} \{h_\theta(x), \theta \in [\theta_1^-, \theta_1^+] \times \cdots \times [\theta_m^-, \theta_m^+]\}.$$

The same result could be applied to g^-. Fortunately, numerical calculus of this minimum and this maximum can be avoided if we choose an adequate representation for the expression part of the GA-P algorithm.

The proposed representation is based in the use of interval arithmetic to perform all operations involved in the expression part (see Fig. 4). That is, we codify

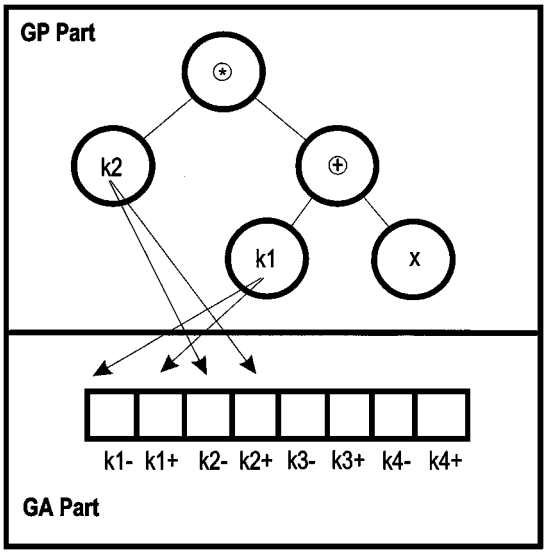


Figure 4. Interval arithmetic GA-P algorithms.

the function in a tree, whose terminal nodes represent intervals $[\theta_i^-, \theta_i^+]$ (that will contain the unknown values of the parameters). The internal nodes represent unary interval operations

$$O_u(A) = \{x \in \mathbb{R} \mid x = o_u(t) \wedge t \in A\}$$

or binary operations

$$O_b(A, B) = \{x \in \mathbb{R} \mid x = o_b(t, u) \wedge t \in A, u \in B\}$$

where $A, B \in I(\mathbb{R})$, $o_a : \mathbb{R} \to \mathbb{R}$ and $o_b : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. Then, the evaluation of the expression part in an input value (point or interval) will be an interval.

A description on the unary and binary operators and the remaining characteristics of the algorithm are to be found in [26].

4. A Genetic Fuzzy Rule-Based System for Defining the Data Base

4.1. Framework

In this section, we deal with the case in which EAs are considered to define the DB. Thus, we refer to GFRBSs belonging to the first group mentioned in Section 2, which are commonly known as *evolutionary tuning processes*. Many of these processes are to be found in the specialized literature (see [23], Section 3.13, and [24], Section 13). They all deal with the problem of refining a preliminary KB obtained from the linguistic information given by human experts, from an automatic learning process based on the numerical information available, or from a method combining both types of information [20].

These kinds of GFRBSs may work over different DB components and adjust its previous definition by adapting it. The components that may be involved in the evolutionary tuning process are the following:

- The definitions of the fuzzy rule membership functions collected in the DB.
- The scaling factors.
- The gain of the different fuzzy partitions considered.

We will briefly introduce two different evolutionary tuning processes in the the following two subsections. The aim of the first method is to adjust the fuzzy membership functions of the different fuzzy partitions considered in a preliminary Mamdani-type KB. On the other hand, the second one works over TSK KBs and refines both the membership functions used in the rule antecedent and the real parameters defining the rule consequent.

Both processes may be used in combination with any generation process capable to obtain a preliminary definition of a KB of the corresponding type, Mamdani or TSK, respectively. In the experiments that will be carried out in Section 6, the Mamdani-type one will be combined with a very known inductive algorithm for deriving Mamdani-type KBs, the Wang and Mendel's (WM) one [27] (see Appendix), and with one of the evolutionary learning processes that will be presented in Section 5. On the other hand, the TSK evolutionary tuning process is used in the TSK GFRBS that will be presented in Section 5 as well.

4.2. A Genetic Tuning Process for Adjusting the Fuzzy Membership Functions in a Mamdani-type Data Base

As all GFRBSs in the same family, the genetic tuning process presented in [25, 28] is based on the existence of a previous definition of the whole KB, i.e., an initial DB and an RB composed of T Mamdani-type fuzzy rules, called **R**.

Each chromosome forming the genetic population will encode a different DB definition that will be

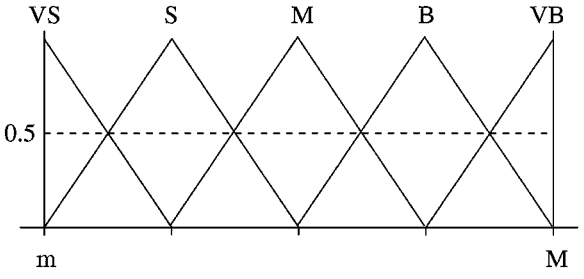


Figure 5. Graphical representation of a possible fuzzy partition.

combined with the existing RB to evaluate the individual adaption.

The GA designed for the tuning process presents real coding issue, uses the stochastic universal sampling [29] as a selection procedure and Michalewicz's non-uniform mutation operator [30]. As regards the crossover operator, the max-min-arithmetical one [31, 32], which makes use of fuzzy tools in order to improve the GA behaviour, is employed.

The primary fuzzy sets considered in the initial linguistic variables fuzzy partitions are triangular-shaped (see Fig. 5). Thus, each one of the membership functions has an associated parametric representation based on a 3-tuple of real values and a primary fuzzy partition can be represented by an array composed of $3 \cdot N$ real values, with N being the number of terms forming the linguistic variable term set. The complete DB for a problem in which m linguistic variables are involved is encoded into a fixed length real coded chromosome C_r built by joining the partial representations of each one of the variable fuzzy partitions as it is shown in the following:

$$C_{ri} = (a_{i1}, b_{i1}, c_{i1}, \ldots, a_{iN_i}, b_{iN_i}, c_{iN_i}),$$
$$C_r = C_{r1}\, C_{r2} \cdots C_{rm}$$

The initial gene pool is created making use of the initial DB definition. This one is encoded directly into a chromosome, denoted as C_1. The remaining individuals are generated by associating an interval of performance $[c_h^l, c_h^r]$ to every gene c_h in C_1, $h = 1 \cdots \sum_{i=1}^{m} N_i \cdot 3$. Each interval of performance will be the interval of adjustment for the corresponding gene, $c_h \in [c_h^l, c_h^r]$.

If $(t \bmod 3) = 1$ then c_t is the left value of the support of a fuzzy number. The fuzzy number is defined by the three parameters (c_t, c_{t+1}, c_{t+2}), and the intervals

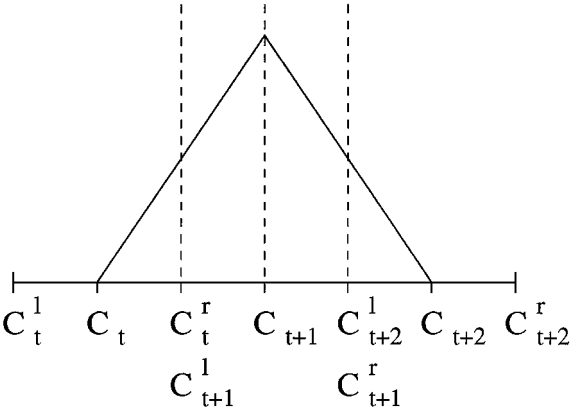


Figure 6. Intervals of performance.

of performance are the following:

$$c_t \in [c_t^l, c_t^r] = \left[c_t - \frac{c_{t+1} - c_t}{2}, c_t + \frac{c_{t+1} - c_t}{2}\right],$$
$$c_{t+1} \in [c_{t+1}^l, c_{t+1}^r] = \left[c_{t+1} - \frac{c_{t+1} - c_t}{2}, c_{t+1} + \frac{c_{t+2} - c_{t+1}}{2}\right],$$
$$c_{t+2} \in [c_{t+2}^l, c_{t+2}^r] = \left[c_{t+2} - \frac{c_{t+2} - c_{t+1}}{2}, c_{t+2} + \frac{c_{t+2} - c_{t+1}}{2}\right]$$

Figure 6 shows these intervals.

Therefore we create a population of chromosomes containing C_1 as its first individual and the remaining ones initiated randomly, with each gene being in its respective interval of performance.

As regards the fitness function, two different definitions for it may be considered. Both of them are based on an application specific measure usually employed in the design of GFRBSs, the mean square error (SE) over a training data set, E_{TDS}, composed of a number of input-output data pairs, $(ex_1^i, \ldots, ex_n^i, ey^i)$. The first definition is constituted directly by this criterion. Therefore, it is represented by the following expression:

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

with $S(ex^l)$ being the output value obtained from the FRBS using the KB $\mathbf{R}(C_j)$, comprising the initial RB definition, $\mathbf{R}$, and the DB encoded in the chromosome C_j, when the input variable values are

$ex^l = (ex_1^l, \ldots, ex_n^l)$, and ey^l is the known desired value.

The second fitness function definition is based on considering the *completeness property*, an important property of KBs [8, 33]. This condition is ensured by forcing every example contained in the training set to be covered by the considered KB to a degree greater than or equal to τ,

$$C_{\mathbf{R}_{(C_j)}}(e_l) = \bigcup_{j=1,\ldots,T} R_j(e_l) \geq \tau, \quad \forall e_l \in E_{\text{TDS}} \quad \text{and} \quad R_j \in \mathbf{R}(C_j)$$

where $\tau \in [0, 1]$ is the minimal training set completeness degree, a value provided by the system designer.

Therefore, we define a *training set completeness degree* of $\mathbf{R}(C_j)$ over the set of examples E_{TDS} as

$$TSCD(\mathbf{R}(C_j), E_{\text{TDS}}) = \bigcap_{e_l \in E_{\text{TDS}}} C_{\mathbf{R}_{(C_j)}}(e_l)$$

and the final fitness function penalizing the lack of the completeness property is:

$$F(C_j) = \begin{cases} E(C_j), & \text{if } TSCD(\mathbf{R}(C_j), E_{\text{TDS}}) \geq \tau \\ \dfrac{1}{2} \displaystyle\sum_{e_l \in E_{\text{TDS}}} (ey^l)^2, & \text{otherwise.} \end{cases}$$

4.3. *An Evolutionary Refinement Process for Tuning the Membership Function and Consequent Definitions in a TSK-type Knowledge Base*

The evolutionary refinement process [34] 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 (a *genetic local search algorithm* [35, 36]), guided by a global error measure over the training data set. We describe the hybrid EA components below.

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

The antecedent fuzzy partitions are encoded in the first part of the chromosomes working in the way shown in the previous section. In this case, each one of the triangular fuzzy sets $D_{ij} = (a_{ij}, b_{ij}, c_{ij})$, $i = 1, \ldots, iv$ (iv = number of input variables), $j = 1, \ldots, N_i$ (N_i = number of fuzzy sets in the ith fuzzy partition), 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:

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

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:

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

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. Figure 7 shows an example of these intervals.

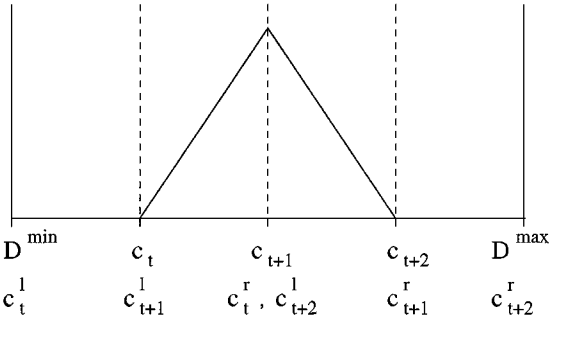


Figure 7. Example of triangular membership function and intervals of performance for the refinement process.

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:

$$C_i^2 = (p_{i0}, p_{i1}, \ldots, p_{iiv}), \quad i = 1, \ldots, m,$$
$$C^2 = C_1^2\, C_2^2 \cdots C_m^2$$

The parameters in C^2 are encoded using the *angular coding* [37], a coding scheme specially designed for TSK rule consequent parameters. It is based on encoding the angle value associated to the TSK rule consequent parameters instead of the tangent one by means of the function

$$C : \mathbf{R} \rightarrow \left(-\frac{\pi}{2}, \frac{\pi}{2}\right), \quad C(y) = \arctan(y)$$

Therefore, the interval of performance of all genes in C^2 is the same, $(-\frac{\pi}{2}, \frac{\pi}{2})$.

The available knowledge in the form of the preliminary definition of the KB being optimized is used to initialize the first population by performing the following three steps:

1. The preliminary definition of the KB taken as process input is encoded directly into a chromosome, denoted as C_1.
2. The following $\frac{M}{2} - 1$ 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 $\frac{M}{2}$ 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.

The fitness function based on the SE measure, E, presented in the previous Section, is considered. On the other hand, the selection mechanism and the genetic operators are the same ones used in the Mamdani-type genetic tuning process: Baker's stochastic universal sampling, Michalewicz's non-uniform mutation and max-min-arithmetical crossover.

Finally, the last genetic operator to be applied consists of an $(1 + 1)$-ES, which plays the role of local search algorithm in the hybrid approach built. In this case, the genetic local search performed is selective: each time a GA generation is performed, the ES is applied over a percentage δ of the best different population individuals existing in the current genetic population.

The coding scheme and the fitness function considered in the $(1 + 1)$-ES 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 when working on the genes in C^1. This is due to the fact that $(1 + 1)$-ESs, usually work over individuals in which the genes are independent and all of them are adapted using the same step size σ. Both assumptions are not right in this case because of two reasons: each three consecutive parameters in C^1 are defined in a different universe thus requiring different order mutations (a step size $\sigma_i = \sigma \cdot s_i$ is going to be used for each gene) and are related among them (they three define a triangular fuzzy set), thus requiring to be adapted incrementally for not obtaining non-meaningful fuzzy sets.

The next algorithm summarizes the application of the adaptation process on a membership function encoded in the parent. With $C_{ij} = (x_0, x_1, x_2)$ being the fuzzy set currently adapted, the steps to follow are:

1. *Compute the step size of the central point*, $s(x_1) \leftarrow \frac{\text{Min}\{x_1 - x_0, x_2 - x_1\}}{2}$.
2. *Generate* $z_1 \sim N(0, \sigma_1^2)$ *and compute* x_1' *in the following way*:

$$x_1' \leftarrow \begin{cases} x_1 + z_1, & \text{if } x_1 + z_1 \in [x_0, x_2] \\ x_0, & \text{if } x_1 + z_1 < x_0 \\ x_2, & \text{if } x_1 + z_1 > x_2 \end{cases}$$

3. *Adapt the remaining two points*:

(a) $s(x_0) \leftarrow \frac{\text{Min}\{x_0 - C_{ij}^{\min}, x_1' - x_0\}}{2}$
Generate $z_0 \sim N(0, \sigma_0^2)$

$$x_0' \leftarrow \begin{cases} x_0 + z_0, & \text{if } x_0 + z_0 \in \left[C_i^l, x_1'\right] \\ C_i^l, & \text{if } x_0 + z_0 < C_i^l \\ x_1', & \text{if } x_0 + z_0 > x_1' \end{cases}$$

(b) $s(x_2) \leftarrow \frac{\text{Min}\{x_2 - x_1', C_{ij}^{\max} - x_2\}}{2}$

Generate $z_2 \sim N(0, \sigma_2^2)$

$$x_2' \leftarrow \begin{cases} x_2 + z_2, & \text{if } x_2 + z_2 \in \left[x_1', C_i^r\right] \\ x_1', & \text{if } x_2 + z_2 < x_1' \\ C_i^r, & \text{if } x_2 + z_2 > C_i^r \end{cases}$$

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. Since all of them are defined over the same interval of performance, $(-\frac{\pi}{2}, \frac{\pi}{2})$, they all will use the same step size $\sigma_i = \sigma \cdot s_i$ with $s_i = 0.00001$.

5. Two Genetic Fuzzy Rule-Based Systems for Learning Mamdani and TSK-type Knowledge Bases

In this section, we will consider two GFRBSs that belong to the third mentioned group, the ones learning the complete KB. The following subsections are devoted to present two evolutionary learning processes capable of generating a whole definition of a Mamdani-type and TSK-type KB from examples, respectively.

5.1. A GFRBS for Learning Mamdani KBs

This genetic learning process is composed of the following three stages [25, 28]:

1. An *inductive generation process* for generating Mamdani-type fuzzy rules from examples, with two components: a *fuzzy rule generating method* based on a non-evolutionary inductive algorithm, and an *iterative covering method* of the example set.
2. A *genetic multisimplification process* for selecting rules, based on a binary coded GA with a genotypic sharing function and a measure of the FRBS performance. It will remove the redundant rules generated by the previous component with the aim of obtaining different simplified KBs presenting the best possible cooperation among the fuzzy rules composing them.
3. The *genetic tuning process* introduced in Section 4.2. It will give the final KB as output by tuning the membership functions in each possible KB derived from the genetic multisimplification process. The most accurate one obtained in this stage will constitute the final output of the whole genetic learning process.

Next subsections will briefly describe the first two learning stages.

5.1.1. The Inductive Generation Process. The generation process is based on a previously defined DB, composed of different fuzzy partitions of the variable spaces, as the one shown in Fig. 5.

The covering method is developed as an iterative process that allows us to obtain a set of fuzzy rules covering the example set. In each iteration, it runs the generating method, obtaining the best fuzzy rule according to the current state of the training set, considers the relative covering value this rule provokes over it, and removes from it the examples with a covering value greater than ϵ, provided by the system designer. It ends up when the training set becomes empty.

Each time the generating method is run, it produces a set of candidate fuzzy rules by generating the fuzzy rule best covering every example from the training set. The accuracy of the candidates is measured by using a multicriteria fitness function, composed of three different criteria measuring the covering that each rule provokes over the training set, which allows us to ensure that the final set of rules generated verify the completeness and consistency properties [8, 33]. Finally, the best fuzzy rule is selected from the set of candidates and given as method output.

The last criteria can be described by means of the following expressions:

(a) *High frequency value*: The frequency of a fuzzy rule, R_i, through the set of examples, E_p, is defined as:

$$\Psi_{E_p}(R_i) = \frac{\sum_{l=1}^{p} R_i(e_l)}{p}$$

(b) *High average covering degree over positive examples*: The set of positive examples to R_i with a compatibility degree greater than or equal to ω is defined as:

$$E_\omega^+(R_i) = \{e_l \in E_p / R_i(e_l) \geq \omega\}$$

with $n_\omega^+(R_i)$ being equal to $|E_\omega^+(R_i)|$. The *average covering degree* on $E_\omega^+(R_i)$ can be defined as:

$$G_\omega(R_i) = \sum_{e_l \in E_\omega^+(R_i)} R_i(e_l)/n_\omega^+(R_i)$$

(c) *Penalization associated to the non satisfaction of the k-consistency property*: The set of the negative examples for R_i is defined as:

$$E^-(R_i) = \{e_l \in E_p / R_i(e_l) = 0 \text{ and } A_i(ex^l) > 0\}$$

An example is considered negative for a rule when it best matches some other rule that has the same antecedent but a different consequent. The negative examples are always considered over the complete training set.

This last criterion penalizes those fuzzy rules with many negative examples with respect to the number of positive examples with a compatibility degree greater than or equal to ω. In this way, it penalizes the non satisfaction of the k-consistency property [33]. The *penalty function on the negative examples set of the rule* R_i will be:

$$g_n(R_i{}^-) = \begin{cases} 1, & \text{if } n^-_{R_i} \leq k \cdot n^+_\omega(R_i) \\ \dfrac{1}{n^-_{R_i} - k n^+_\omega(R_i) + \exp(1)}, & \text{otherwise} \end{cases}$$

with $n^-_{R_i} = |E^-(R_i)|$ being the number of negative examples.

These three criteria are combined into a fitness function using any aggregation function increasing in the three variables. In this paper we work with the product in the following way:

$$F(R_i) = \Psi_{E_p}(R_i) \cdot G_\omega(R_i) \cdot g_n(R_i{}^-)$$

5.1.2. The Genetic Multisimplification Process. Since the generation process works in an iterative way, it may obtain a KB containing redundant rules that do not properly cooperate between them. The aim of this second stage is to simplify the previously generated KB, removing from it the rules not cooperating well.

The main idea of the genetic multisimplification process is that it does not only generate one simplified definition of the previous fuzzy rule set, but several different ones. To do so, it runs the genetic simplification process proposed in [38]. This process is based on a binary-coded GA which encodes the set of rules obtained from the generation process into a fixed-length chromosome. The value 1 means that the rule belongs to the final KB, and the 0 means that it does not. Two-point crossover and uniform mutation operators are used to alter the individuals and the stochastic universal sampling procedure, along with an elitist selection scheme, to perform selection. The fitness function is the F one shown in Section 4.2, which combines an error measure, the SE, and a term penalizing the lack of the encoded KB completeness property.

Each time the genetic simplification process obtains a simplified KB definition, the multisimplification one penalizes the search space zone where it is located, so it will not be generated in future runs. A genotypic sharing scheme [39] is used to penalize individuals according to its space proximity to the previous solutions found. To do so, there is a need to define a *distance metric* which, given two individuals, returns a value of how close they are. We use the Hamming distance due to the fact that this measure is defined to work on a binary space and it is very simple to compute, thus being suitable for our purpose. With $A = (a_1, \ldots, a_m)$ and $B = (b_1, \ldots, b_m)$ being two individuals, the Hamming distance is defined as follows:

$$H(A, B) = \sum_{i=1}^{m} a_i \cdot b_i$$

Making use of this metric, the *modified fitness function* guiding the search on the multisimplification process is based on modifying the value associated to an individual by the basic algorithm fitness function, multiplying it by a *derating function* penalizing the closeness of this individual to the solutions previously obtained. Hence, the modified fitness function used by the multisimplification process is the following:

$$F'(C_j) = F(C_j) \cdot G(C_j, S)$$

where F is the basic genetic simplification process fitness function, $S = \{s_1, \ldots, s_k\}$ is the set containing the k solutions already found, and G is a kind of *derating function*. We consider the following taking into account the fact that the problem we deal with is a minimization one:

$$G(C_j, S) = \begin{cases} \infty, & \text{if } d = 0 \\ 2 - \left(\dfrac{d}{r}\right)^\beta, & \text{if } d < r \text{ and } d \neq 0 \\ 1, & \text{if } d \geq r \end{cases}$$

where d is the minimum value of the Hamming distance between C_j and the solutions s_i included in S, i.e., $d = \text{Min}_i\{H(C_j, s_i)\}$, and the penalization is considered over the most close solution, r is the *niche radius*, and β is the *power factor* determining how concave ($\beta > 1$) or convex ($\beta < 1$) the derating curve is. Therefore, the

penalization given by the derating function takes its maximum value when the individual C_j encodes one of the solutions already found. There is no penalization when C_j is far away from S in a value greater than or equal to the niche radius r.

The algorithm of the genetic multisimplification process is shown below:

1. Initialization: Equate the multisimplification modified fitness function to the basic simplification fitness function: $F'(C_j) \leftarrow F(C_j)$.
2. Run the basic genetic simplification process, using the modified fitness function, keeping a record of the best individual found in the run.
3. Update the modified fitness function to give a depression in the region near the best individual, producing a new modified fitness function.
4. If all simplified KBs desired have not been obtained, return to step 2.

Hence, the number of runs of the sequential algorithm performed is the number of solutions that we desire to obtain, i.e., the number of simplified KBs we will generate. We allow the FRBS designer to decide this number as well as the values of the parameters r and β.

5.2. A GFRBS for Learning TSK KBs

This second GFRBS is based on performing the learning of the KB in different steps as the previous one presented. In this case, it comprises the following two stages [34]:

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 using the angular coding proposed in the previous section and a local measure of error, and takes into account the knowledge contained in this training data subset to improve the search process.
2. The *evolutionary refinement process* introduced in Section 4.3 for adjusting both the consequent and the antecedent parts of the fuzzy rules in the preliminary KB obtained from the first stage.

Next subsections will present the different components of the first learning stage. 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 whole generation process, which makes use of the two previous aspects.

5.2.1. The TSK Rule Consequent Learning Method. In this method, a (μ, λ)-ES 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 being solved. 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 the *angular coding* in Section 4.3.

EA evolution is guided by a fitness function composed of a local measure of error. The expression of the measure used is the following one:

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

where E is the set of input-output data pairs $e_l = (ex_1^l, \ldots, ex_{iv}^l, ey^l)$ located in the fuzzy input subspace defined by the rule antecedent, $h_l = T(A_1(ex_1^l), \ldots, A_{iv}(ex_{iv}^l))$ 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 composition of the remaining vectors, the components of $\vec{\sigma}$ are initiated to 0.001, and the ones in $\vec{\alpha}$, when considered, are set to *arctan* (1).

5.2.2. 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:

$$y_{\text{med}} = \frac{\sum_{e_l \in E} ey^l}{|E|}, \quad y_{\min} = \min_{e_l \in E} \{ey^l\}$$

$$y_{\max} = \max_{e_l \in E} \{ey^l\}, \quad h_{\max} = \max_{e_l \in E} \{h_l\}$$

$$E_\theta = \{e_l \in E / h_l \geq \theta \cdot h_{\max}\}$$

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

1. Generate the $\vec{x}$ part of the first individual, $\vec{x}_1$, initiating parameters x_i, $i = 1, \ldots, iv$, to zero, and parameter x_0 to the angular coding of y_{med}.
2. Generate the $\vec{x}$ part of the following γ individuals, $\vec{x}_2, \ldots, \vec{x}_{\gamma+1}$, with $\gamma \in \{0, \ldots, \mu - 1\}$ defined by the GFRBS designer, initiating parameters x_i, $i = 1, \ldots, iv$, to zero, and x_0 to the angular coding of a value computed at random in the interval $[y_{\min}, y_{\max}]$.
3. Generate the $\vec{x}$ part of the remaining $\mu - (\gamma + 1)$ individuals, $\vec{x}_{\gamma+2}, \ldots, \vec{x}_\mu$, initiating parameters x_i, $i = 1, \ldots, iv$, to the angular coding of values computed at random in the interval $(-\frac{\pi}{2}, \frac{\pi}{2})$, and x_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 GFRBS 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 significant ones from E.

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

$$f : [0, 1] \times \{-1, 1\} \rightarrow \left(-\frac{\pi}{2}, \frac{\pi}{2} \right)$$

$$f(x, z) = z \cdot \frac{\pi}{2} \cdot x^q$$

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

   For $j = \gamma + 2, \ldots, \mu$ do

   (a) For $i = 1, \ldots, iv$ do

      (a1) Generate y at random in $[0, 1]$.
      (a2) Generate z at random in $\{-1, 1\}$.
      (a3) Set x_i to $f(y, z)$.

   (b) Generate the value of x_0:

      (b1) Select e at random from E_θ.
      (b2) Set x_0 to $ey - \sum_{k=1}^{iv} C^{-1}(x_k) \cdot ex_k$, where $C^{-1}(\beta) = \tan(\beta)$ is the inverse of C.

5.2.3. 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 by a normalization process. If the latter is the case, perform a fuzzy partition of the input variable spaces dividing each universe of discourse into a number of equal or unequal partitions, select a kind of membership function and assign one fuzzy set to each subspace. In this paper, we will work with symmetrical fuzzy partitions of triangular membership functions (see Fig. 5).
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 KB.

6. Modeling Electrical Distribution Networks by Means of the Proposed Hybrid Evolutionary Data Analysis Techniques

6.1. Computing the Length of Low Voltage Lines

The first of the problems we will show is that of finding a model that relates the total length of low voltage

Table 1. Notation considered for the low voltage line estimation problem variables.

Symbol	Meaning
A_i	Number of clients in population i
R_i	Radius of i population in the sample
n	Number of populations in the sample
l_i	Line length, population i
$\tilde{l}_i$	Estimation of l_i
s_i	Number of sectors in town i

line installed in a rural town with the number of inhabitants in the town and the mean of the distances from the center of the town to the three furthest clients in it. This model will be used to estimate the total length of line being maintained by one of the companies. We were provided with a sample of 495 towns in which the length of line was actually measured and the company used the model to extrapolate this length over more than 10,000 towns with these properties.

We will limit ourselves to the estimation of the length of line in a town, given the inputs mentioned before. Hence, our objective is to relate the first variable (line length) with the other two ones (population, radius of village), first by classical methods, later by applying the hybrid evolutionary DA techniques presented in this paper. Numerical results will be compared in a next subsection.

Our variables are named as shown in Table 1.

6.1.1. Application of Classical Methods: Classical Regression Adjust. In order to apply classical methods, we needed to make some hypothesis [2]. In the populations that are being studied, electrical networks are star-shaped and arranged in sectors. A main line passes near all clients inside them, and clients are connected to these main lines by small segments (see Fig. 8).

To build a theoretical simplified model we have admitted that:

- Village i comprises s_i sectors. All sectors in the same village cover the same angle $2\theta_i$. Main lines depart from the centre of the village.
- The density of clients is constant inside every sector.
- All sectors in a village have the same radius, R_i and contain a main line of length R_i and as many branches as customers.

If we admit that customers are uniformly distributed, we can approximate the total length by multiplying the mean distance between one of them and the nerve by the number of inhabitants. Let us name this mean distance d_i for population i, and let the sector be $2\theta_i$ wide. Then

$$d_i = \frac{2(1-\cos\theta_i)}{3\theta_i} R_i$$

so cable length will be

$$\tilde{l}_i = s_i\left(R_i + \frac{A_i}{s_i} d_i\right) = s_i R_i + A_i \frac{2(1-\cos\theta_i)}{3\theta_i} R_i$$

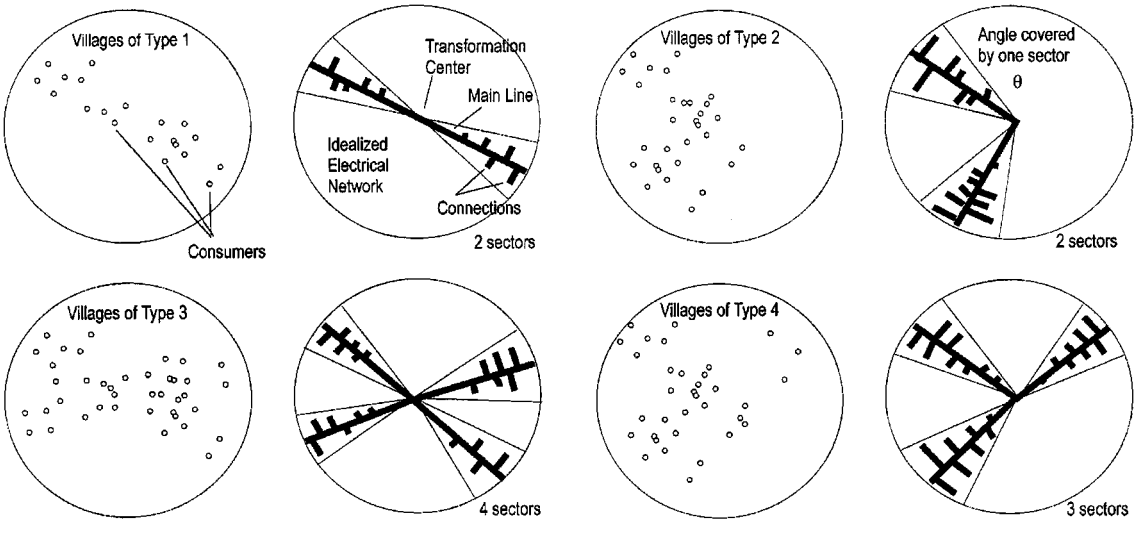


Figure 8. Models of some kind of nuclei.

If the angles θ_i and the numbers s_i were similar enough between them, we could regard them as constants and estimate them by the parameters $\bar{\theta}_i = \theta$ and $\bar{s}_i = s$ of a least squares linear regression

$$\tilde{l}_i / R_i = s + k(\theta) A_i .$$

to a set of pairs $(x, y) = (A_i, l_i / R_i)$.

We can get a better adjust by allowing a certain dependence between the number of sectors, their angles and the number of inhabitants. This can be done by dividing the sample into classes or by mean of a change of variables. Both cases were studied, and the best adjust was obtained with the model

$$\frac{\tilde{l}_i}{R_i} = k_1 A_i^{k_2}$$

6.1.2. GA-P and Interval GA-P Adjust. Let us apply GA-P algorithms to check whether we can obtain a formula that is comparable in complexity with the last one, while getting better adjust to the real data. We will define "simple expression" as a formula that can be codified in a tree with no more than 20 nodes and depending on no more than 10 parameters. Binary operations will be sum, difference, product, ratio and power. The unary operation will be the square root. Other decisions (whose meaning is well known, see for instance [4, 30, 40]) are shown in the Table 2.

We randomly select three individuals every generation. The worst one of them is replaced with the best descendent of the crossover of the remaining ones. Observe that this strategy is elitist and steady state.

6.1.3. GFRBS Fuzzy Modeling. We have considered three different fuzzy models to solve the problem, two Mamdani-type and one TSK-type ones. They have been generated from a two-stage GFRBS composed of the WM RB learning process (see Appendix) and the Mamdani-type genetic tuning process presented in Section 4.2 and from the Mamdani and TSK GFRBSs presented in Sections 5.1 and 5.2. The parameter values considered are shown in Tables 3, 4 and 5, respectively. In all cases, the initial DB considered is constituted by some primary equally partitioned fuzzy partitions formed by *five linguistic terms* with triangular-shaped fuzzy sets giving meaning to them (as shown in Fig. 5), and the adequate scaling factors to translate the generic universe of discourse into the one associated with each problem variable.

Table 2. Parameter values considered for the GA-P process.

Parameter	Decision
Population size	100
Maximum number of generations	1000 (steady state)
Parent selection	(See text)
GA Part encoding	Real
GA Crossover operator	Two points
GP Crossover operation	Standard
GA Cross. probability	0.9
GP Cross. prob. internal nodes	0.9
GP Cross. prob. leaves	0.1
GA Mutation probability	0.01
GP Mutation probability	0.01
Expression part limited to	20 nodes
Complexity individuals initial pop.	20 nodes
Maximum number of parameters	10
Enrichment initial population	1000 individuals
Edition probability	0
Encapsulation probability	0
Permutation probability	0
Decimation	No
ADFs maximum	0
Local GA optimization	Nelder and Mead's simplex

6.1.4. Comparison Between Methods. To compare classical methods, GA-P technique and GFRBS fuzzy modeling we have divided the sample into two sets comprising 396 and 99 samples. SE values over these two sets are labeled *training* and *test*. In this case, we define SE as

$$\frac{1}{2 \cdot N} \sum_{i=1}^{N} (\tilde{l}_i - l_i)^2$$

Table 3. Parameter values considered for the Mamdani-type genetic tuning process.

Parameter	Decision
Population size	61
Maximum number of generations	1000
Non-uniform mutation parameter b	5
Max-min-arithmetical parameter a	0.35
Crossover probability	0.6
Mutation probability	0.1
Fitness function WM tuning	E
Fitness function Mamdani GFRBS tuning	F
Completeness property parameter τ	0.1

Table 4. Parameter values considered for the Mamdani GFRBS.

Parameter	Decision
Generation	
Example covering value ϵ	1.5
Positive example degree ω	0.05
k-consistency property parameter	0.1
Multisimplification	
Population size	61
Maximum number of generations	500
Crossover probability	0.6
Mutation probability	0.1
Completeness property parameter τ	0.1
Number of simplified KBs to generate	3
Niche radius r	10% of initial KB rule number
Power factor β	0.5

and the 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 every generated fuzzy model (see Table 6).

Table 5. Parameter values considered for the TSK GFRBS.

Parameter	Decision
Generation	
Number of parents μ	15
Number of descendents λ	100
Maximum number of generations	500
Parameter γ	$0.2 \cdot \mu = 3$
Parameter θ	0.7
Parameter q	5
Recombination operators considered $\vec{r}$	(3, 2, 0)
Number of parents considered for recombination $\vec{\zeta}$	$(\mu, \mu, 1)$
Refinement	
Population size	61
Maximum number of generations	1000
Non-uniform mutation parameter b	5
Max-min-arithmetical parameter a	0.35
Crossover probability	0.6
Mutation probability	0.1
Selective local search parameter δ	{0, 0.2}
Maximum number of generations for the $(1+1)$-*ES*	25
Standard deviation d considered in the generation of the initial population	0.001

Table 6. Results obtained in the low voltage line estimation problem.

Method	Training	Test	Complexity
Linear	287775	209656	7 nodes, 2 par.
Exponential	232743	197004	7 nodes, 2 par.
Second-order polynomial	235948	203232	25 nodes, 6 par.
Third-order polynomial	235934	202991	49 nodes, 10 par.
Three layer perceptron 2-25-1	169399	167092	102 par.
GA-P	183693	159837	20 nodes, 3 par.
Interval GA-P	192908	158737	16 nodes, 3 par.
WM fuzzy model	175337	180102	13 rules
Mamdani fuzzy model	**150559**	166669	19 rules
TSK fuzzy model (without refinement)	162609	**148514**	20 rules

The parameters of the polynomial models were fitted by nonlinear least squares (Levenberg-Marquardt method); exponential and linear models were fitted by linear least squares and the multilayer perceptron error was minimized with the Conjugate Gradient algorithm. The number of neurons in the hidden layer was chosen to minimize the test error.

We can observe that fuzzy models and GA-P techniques clearly outperform classical non linear regression methods, being equal or superior to Neural Networks. This result has great significance, because it means that Neural Network performance can be achieved with a model with a high descriptive power. WM fuzzy model and the other Mamdani-type one generated from our GFRBS provide the most comprehensive explanation of their functioning, and should be used when a human-readable, rule based, description of the problem is needed. In this case, the genetic-based methods have found very simple structures, comprising only 13 and 19 rules, respectively, and, as may be observed, the fuzzy model obtained from our GFRBS is more accurate to a high degree than the one generated from the WM-based one in the approximation of both data sets.

When a mathematical formula is preferred to the rule bank, GA-P methods provide a suitable expression where the user can select the balance between complexity and precision. We observed that usually Interval GA-P finds a simpler expression than punctual GA-P, besides its convergence is somewhat slower. Observe that Interval GA-P is *not* intended to provide an estimation but a range of values in which the output is, with a probability higher than a preselected value. The number collected in the table is the scoring achieved

by a punctual model formed when every interval of parameters is replaced by its mean point in the final model.

By last, observe that the best precision can only be obtained if we choose the less descriptive of the fuzzy models, TSK. We have to note that the result shown in the table has been obtained after the first learning stage due to the refinement one presents an undesirable overlearning, improving the abstraction capability of the fuzzy model to a high degree but making the generalization one worse. This fuzzy model is only a little more complex than the other two ones (20 rules) and definitely it is the selection that should be made when the precision is more important than the easiness of explanation. Anyway, this fuzzy model has associated a higher level of description than Neural Network models, because of the possibility of interpreting the antecedent part of the fuzzy rules.

6.2. *Computing the Maintenance Costs of Medium Voltage Line*

The second problem has a different nature, since we will not deal with real data but with estimations of minimum maintenance costs which are based on a model of the optimal electrical network for a town. These values are somewhat lower than real ones, but companies are interested in an estimation of the minimum costs. Obviously, real maintenance costs are exactly accounted and hence a model that relates these costs to any characteristics of towns would not be of a great practical significance.

We were provided with data concerning four different characteristics of the towns (see Table 7) and their minimum maintenance costs in a sample of 1059 simulated towns. In this case, our objective was to relate the last variable (maintenance costs) with the other four ones by applying the DA techniques presented in this paper. Numerical results will be compared next.

As regards classical methods, we have considered again linear, polynomial and Neural Network models. The parameters of the polynomial models were fitted using the same method, Levenberg-Marquardt, and the neural model (a three layer perceptron) was trained again with the Conjugate Gradient algorithm. 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. In this case, we used 4 input nodes, 5 hidden nodes, and 1 output node.

Table 7. Notation considered for the medium voltage line maintenance cost estimation problem variables.

Symbol	Meaning
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
y	Maintenance costs of medium voltage line

Table 8. Results obtained in the medium voltage line maintenance cost estimation problem.

Method	Training	Test	Complexity
Linear	164662	36819	17 nodes, 5 par.
Second-order polynomial	103032	45332	77 nodes, 15 par.
Three-layer perceptron 4-5-1	86469	33105	35 par.
GA-P	18168	21884	50 nodes, 5 par.
Interval GA-P	16263	18325	15 nodes, 4 par.
WM fuzzy model	20318	27615	66 rules
Mamdani fuzzy model	19679	22591	63 rules
TSK fuzzy model ($\alpha = 0$)	25579	26450	268 rules
TSK fuzzy model ($\alpha = 0.2$)	**11074**	**11836**	268 rules

GA-P algorithms have been applied with the same parameter values used in the solving of the previous problem. Since the problem tackled now is more complex (four input variables instead of two), in this case we will define "simple expression" as a formula that can be codified in a tree with no more than 50 nodes and depending on no more than 10 parameters. The GFRBSs use the same parameters as well, excepting the example covering value parameter ϵ in the Mamdani GFRBS, that now takes 2.0 as a value.

To compare the mentioned techniques, we have divided the sample into two sets comprising 847 and 212 samples, 80 and 20 percent of the whole data set, respectively. Results obtained in the different experiments developed are shown in Table 8, where column names stand for the same aspects that in the previous section.

In view of them, we can draw similar conclusions to the ones in the previous problem. GA-P techniques and fuzzy models outperform again classical non linear regression methods and Neural Networks. In this case, the WM-based design process has found a structure comprising 66 rules, a little more complex than the one obtained by the Mamdani GFRBS

(63 rules), being more accurate the latter descriptive fuzzy model.

Punctual GA-P found a mathematical expression that can be codified in 50 nodes (note that a second order polynomial needs 77 nodes and that a linear model uses 17) and that explains the data almost identically than the fuzzy model generated from the Mamdani GFRBS. Interval GA-P results reflect the ability of the method to eliminate outliers. The results shown in Interval GA-P row were calculated over the subset of examples that were in the confidence interval (97.9% of total) and therefore it found the simplest expression (15 nodes and it does not use the input x_2, the total area of the town). Interval GA-P can only be used when we are allowed to discard some elements of the sample. If we cannot do that, we should apply punctual GA-P or fuzzy models. For example, if we apply the obtained Interval GA-P model to the whole dataset, without discarding outliers, we obtain roughly the same test error, but a much higher training error (202385) that may not be admissible.

Finally, the TSK fuzzy model has obtained again the best results but its interpretation capability has been very reduced due to the high number of rules in the KB, 268. Anyway, it is the right choice when the main requirement is the model accuracy, performing really better than the neural model. It has to be noted that, in this case, the refinement process has shown very good performance, improving the preliminary TSK fuzzy model obtained from the generation stage to a high degree. It may be observed, as well, the good behaviour of the genetic local search strategy since the best results are obtained when a 20% ($\delta = 0.2$) of the population individuals in each GA generation are refined using the $(1+1)$-*ES*.

7. Concluding Remarks

In this paper, we have solved two real-world models of electrical energy distribution, namely the computation of the low voltage line installed in rural towns and the estimation of the minimum maintenance costs (for medium voltage lines) by means of different DA techniques. We presented two new hybrid evolutionary DA techniques, GA-P algorithms for performing symbolic regressions and GFRBSs for designing and optimizing fuzzy models. We compared these methods with better known DA methods such as classical regression and Neural Networks.

Both hybrid evolutionary techniques have proven to be powerful DA tools capable of making abstraction on the data with good generalization properties in view of the results obtained. The first one allows us to obtain expressions with algebraic operators while the second one is able to generate KBs that are interpretable to a different degree: Mamdani-type fuzzy models give a complete description of the problem-solving, whilst TSK ones give a local linguistic description of it.

Appendix: The Wang and Mendel's Rule Base Generation Process

The inductive KB generation process proposed by Wang and Mendel in [27] has been widely known because of its simplicity and good performance. It is based on working with an input-output data set representing the behaviour of the problem being solved and with a previous definition of the DB composed of the input and output primary fuzzy partitions used. The fuzzy rule structure considered is the usual Mamdani-type rule with n input variables and one output variable presented in Section 2.

The generation of the fuzzy rules of this kind is performed by putting into effect the three following steps:

1. *To generate a preliminary linguistic rule set*: This set will be composed of the fuzzy rule best covering each example (input-output data pair) existing in the input-output data set. The composition of these rules is obtained by taking a specific example, i.e., a $(n+1)$-dimensional real array (n values for the input variables and one for the output one), and setting each one of the rule variables to the linguistic label associated to the fuzzy set best covering every array component.
2. *To give an importance degree to each rule*: Let $R =$ If X_1 is A and X_2 is B then Y is C be the linguistic rule generated from the example (x_1, x_2, y) in a problem in which three variables (two input variables and one output variable) are involved. The importance degree associated to it will be obtained as follows:

$$G(R) = \mu_A(x_1) \cdot \mu_B(x_2) \cdot \mu_C(y)$$

3. *To obtain a final RB by using the preliminary rule set*: If all rules presenting the same antecedent values have associated the same consequent one in the preliminary set, this linguistic rule is automatically

put (only once) into the final RB. On the other hand, if there are conflictive rules, i.e., rules with the same antecedent and different consequent values, the rule considered for the final RB will be the one with higher importance degree.

Acknowledgments

Cordón and Herrera were supported by CYCIT, under Project TIC96-0778 with the University of Granada and Sánchez was supported by Hidroeléctrica del Cantábrico, under Contract CN-96-055-B1 with the University of Oviedo.

References

1. O. Cordón, F. Herrera, and L. Sánchez, "Computing the spanish medium electrical line maintenance costs by means of evolution-based learning processes," in *Proceedings of the Eleventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems* (*IEA-98-AIE*), Castellón, vol. 1, pp. 478–482, 1998.
2. L. Sánchez, "Estudio de la red asturiana de baja tensión rural y urbana," *Technical Report*, Hidroeléctrica del Cantábrico Research and Development Department (in spanish), Asturias, Spain, 1997.
3. T. Bäck, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, 1996.
4. L. Howard and D. D'Angelo, "The GA-P: A genetic algorithm and genetic programming hybrid," *IEEE Expert*, pp. 11–15, 1995.
5. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
6. H.P. Schwefel, *Evolution and Optimum Seeking. Sixth-Generation Computer Technology Series*, John Wiley and Sons, 1995.
7. A. Bardossy and L. Duckstein, *Fuzzy Rule-Based Modeling With Application to Geophysical, Biological and Engineering Systems*, CRC Press, 1995.
8. D. Driankov, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control*, Springer-Verlag, 1993.
9. T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.
10. O. Cordón and F. Herrera, "A general study on genetic fuzzy systems," in *Genetic Algorithms in Engineering and Computer Science*, edited by J. Periaux, G. Winter, M. Galán, and P. Cuesta, John Wiley and Sons, pp. 33–57, 1995.
11. A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.
12. S. Chatterjee and B. Price, *Regression Analysis by Examples*, John Wiley and Sons, 1991.
13. P.D. Wasswerman, *Advanced Methods in Neural Computing*, Van Nostrand Reinhold, 1993.
14. J.J. Grefensette (Ed.), *Genetic Algorithms for Machine Learning*, Kluwer Academic Press, 1994.
15. E. Vonk, L.C. Jain, and R.P. Johnson, *Automatic Generation of Neural Network Architecture Using Evolutionary Computation*, World Scientific, 1997.
16. J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, 1992.
17. F. Herrera and J.L. Verdegay (Eds.), *Genetic Algorithms and Soft Computing*, Physica-Verlag, 1996.
18. L.A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.
19. W. Pedrycz (Ed.), *Fuzzy Modelling: Paradigms and Practice*, Kluwer Academic Press, 1996.
20. L.X. Wang, *Adaptive Fuzzy Systems and Control*, Prentice-Hall, 1994.
21. R.R. Yager and L.A. Zadeh (Eds.), *An Introduction to Fuzzy Logic Applications in Intelligent Systems*, Kluwer Academic Press, 1992.
22. P.P. Bonissone, "Soft computing: The convergence of emerging reasoning technologies," *Soft Computing*, vol. 1, no. 1, pp. 6–18, 1997.
23. 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*, edited by E. Sanchez, T. Shibata, and L. Zadeh, World Scientific, pp. 209–241, 1997.
24. O. Cordón, F. Herrera, and M. Lozano, "On the combination of fuzzy logic and evolutionary computation: A short review and bibliography," in *Fuzzy Evolutionary Computation*, edited by W. Pedrycz, Kluwer Academic Press, pp. 57–77, 1997.
25. O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples," *International Journal of Approximate Reasoning*, vol. 17, no. 4, pp. 369–407, 1997.
26. L. Sánchez, "Interval-valued GA-P Algorithms," Technical Report, Dept. of Computer Science, University of Oviedo, Oviedo, Spain, 1997.
27. L.X. Wang and J.M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.
28. O. Cordón and F. Herrera, "A three-stage method for designing genetic fuzzy systems by learning from examples," in *Proceedings of the Fourth International Conference on Parallel Problem Solving from Nature, PPSN IV, L.N.C.S. 1141*, edited by H.M. Voight, W. Ebeling, E. Rechemberg, and H.P. Schwefel, Springer-Verlag: Berlin, 1996, pp. 720–729.
29. J.E. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proceedings of the Second International Conference on Genetic Algorithms (ICGA'87)*, Hillsdale, 1987, pp. 14–21.
30. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, 1996.
31. F. Herrera, M. Lozano, and J.L. Verdegay, "Tuning fuzzy controllers by genetic algorithms," *International Journal of Approximate Reasoning*, vol. 12, pp. 299–315, 1995.
32. F. Herrera, M. Lozano, and J.L. Verdegay, "Fuzzy connectives based crossover operators to model genetic algorithms population diversity," *Fuzzy Sets and Systems*, vol. 92, no. 1, pp. 21–30, 1997.

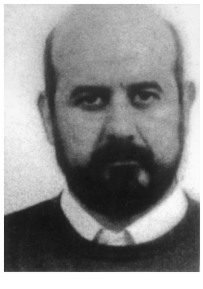
33. A. González and R. Pérez, "Completeness and consistency conditions for learning fuzzy rules," *Fuzzy Sets and Systems*, vol. 96, no. 1, pp. 37–51, 1998.
34. O. Cordón and F. Herrera, "A two-stage evolutionary process for designing TSK fuzzy rule-based systems," Technical Report DECSAI-97115, Dept. of Computer Science and A.I, University of Granada, Spain, 1997.
35. P. Jog, J.Y. Suh, and D.V. Gutch, "The effects of population size, heuristic crossover and local improvement on a genetic algorithm," in *Proceedings of the Third International Conference on Genetic Algorithms* (*ICGA*), pp. 110–115, 1989.
36. J.Y. Suh and D.V. Gutch, "Incorporating heuristic information on genetic search," in *Proceedings of the Second International Conference on Genetic Algorithms (ICGA)*, 1987, pp. 100–107.
37. O. Cordón and F. Herrera, "Evolutionary design of TSK fuzzy rule-based systems using (μ, λ)-evolution strategies," in *Proceedings of the Sixth IEEE International Conference on Fuzzy Systems* (*FUZZ-IEEE'97*), Barcelona, 1997, vol. 1, pp. 509–514.
38. F. Herrera, M. Lozano, and J.L. Verdegay, "A learning process for fuzzy control rules using genetic algorithms," *Fuzzy Sets and Systems*, 1998, to appear.
39. K. Deb and D.E. Goldberg, "An investigation of niche and species formation in genetic function optimization," in *Proceedings of the Second International Conference on Genetic Algorithms* (*ICGA*), Hillsdale, 1989, pp. 42–50.
40. J. Koza and J. Rice, *Genetic Programming*, The MIT Press, 1994.
41. G. Bojadziev, *Fuzzy Sets, Fuzzy Logic, Applications*, World Scientific, 1995.
42. H. Ishibuchi, H. Tanaka, and H. Okada, "An architecture of neural networks with interval weights and its application to fuzzy regression analysis," *Fuzzy Sets and Systems*, vol. 57, pp. 27–39, 1993.

Oscar Cordón was born in Cadiz, Spain, on August 11, 1972. He received his M.S. degree in Computer Science in 1994 and his Ph.D. in Computer Science in 1997, both from the University of Granada, Spain.

He is an Assistant Professor in the Department of Computer Science and Artificial Intelligence at the University of Granada, where 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 modelling, fuzzy logic controllers, fuzzy classification, genetic fuzzy systems, and combination of fuzzy logic and genetic algorithms.

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

Dr. Herrera has co-edited a book, *Genetic Algorithms and Soft Computing* (Physica Verlag, with J.L. Verdegay) and two journal Special Issues, one on "Genetic Fuzzy Systems for Control and Robotics" (*International Journal of Approximate Reasoning*) and "Genetic Fuzzy Systems" (*International Journal of Intelligent Systems*, with L. Magdalena), on the integration of Genetic Algorithms and Fuzzy Logic. 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.



Luciano Sánchez was born in Avilés, Spain, in 1967. He received the Industrial Engineering and the Industrial Engineering Ph.D. degrees from the Universidad de Oviedo, Spain, in 1991 and 1993, respectively.

Since 1991 he has been at the Escuela Tcnica Superior de Ingenieros Industriales de Gijn, where he is now an Assistant Professor. His research interests are in the areas of industrial applications of soft computing, with emphasis on fuzzy rule learning and genetic programming.