International Journal of Approximate Reasoning 64 (2015) 1–19

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Granular fuzzy models: Analysis, design, and evaluation

Orion F. Reyes-Galaviz [a,*], Witold Pedrycz [a,b,c]

[a] Department of Electrical & Computer Engineering, University of Alberta, Edmonton T6R 2V4 AB, Canada
[b] Department of Electrical and Computer Engineering, Faculty of Engineering, King Abdulaziz University, Jeddah, 21589, Saudi Arabia
[c] Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland

ARTICLE INFO

Article history:
Received 11 December 2014
Received in revised form 1 April 2015
Accepted 9 June 2015
Available online 22 June 2015

Keywords:
Conditional Fuzzy C-Means
Induced information granules
Granularity optimization
Granular fuzzy modeling
Differential evolution

ABSTRACT

The study is concerned with a design of granular fuzzy models. We exploit a concept of information granularity by developing a model coming as a network of intuitively structured collection of interval information granules described in the output space and a family of induced information granules (in the form of fuzzy sets) formed in the input space. In contrast to most fuzzy models encountered in the literature, the results produced by granular models are information granules rather than plain numeric entities. The design of the model concentrates on a construction of information granules that form a backbone of the overall construct. Interval information granules positioned in the output space are built by considering intervals of equal length, equal probability, and developing an optimized version of the intervals. The induced fuzzy information granules localized in the input space are realized by running a conditional Fuzzy C-Means (FCM). The performance of the model is assessed by considering criteria of coverage and information specificity (information granularity). Further optimization of the model is proposed along the line of an optimal re-distribution of input information granules induced by the individual interval information granules located in the output space. Experimental results involve some synthetic low-dimensional data and publicly available benchmark data sets.

1. Introductory comments

In spite of the truly remarkable diversity of architectures of fuzzy models and ensuing design approaches [8,13,17–22,25,35,51], the models share one common feature. Fuzzy models, regardless of the use of the technology of fuzzy sets, produce *numeric* results. In contrast, linguistic models [5,10–12,15,40,42,43,52,53,59] form an interesting conceptual and design alternative in the plethora of fuzzy models in the sense their structure is intuitively appealing and the results are inherently coming as information granules (fuzzy sets). In the construction of fuzzy models, various clustering techniques are often used, in particular Fuzzy C-Means (FCM) [9]. In [57], a modified FCM algorithm, combined with a back-propagation algorithm, was implemented to construct detailed fuzzy models by using a weighted FCM algorithm. In [58], FCM was used to generate discrete interval-valued type-2 fuzzy models. A fuzzy C-Regression model is proposed in [27] to automatically determine a suitable number of rules from a given fuzzy model. In [26], a fuzzy clustering technique was developed to generate hyperplane-like clusters, which helped improve the estimation of Takagi–Sugeno (T–S) fuzzy models. Another clustering technique, based on a combination of FCM and switching regression algorithms [10] was used to construct enhanced fuzzy models with an improved fuzzy clustering and better structure identification. A different clustering approach helps

* Corresponding author.
E-mail addresses: reyesgal@ualberta.ca (O.F. Reyes-Galaviz), wpedrycz@ualberta.ca (W. Pedrycz).

http://dx.doi.org/10.1016/j.ijar.2015.06.005
0888-613X/

identify T–S fuzzy models from clusters obtained with a modified Gath–Geva algorithm [6]. In this approach, the best number of clusters is determined by using a validity measure. A face recognition approach [29] used partition matrices obtained with the FCM algorithm as input patterns to train artificial neural networks (ANNs). Other architectures that are similar to ANNs, are the neural fuzzy controller (NEFCON) [30–32] and neuro-fuzzy classification (NEFCLASS) model [33]. These structures use fuzzy sets as weights, and can be interpreted as linguistic "if–then" rules. The obtained results are defuzzified (decoded) producing numeric values. In contrast to our model, these architectures do not induce fuzzy classification rules by granulating the data; instead the fuzzy partition predefined in each input dimension is adjusted by a training mechanism.

In linguistic models clustering is used more intensively, leading to the formation of information granules in the output space and subsequently producing induced information granules positioned in the input space. One of the earliest models falling under this category was proposed in [50], where the FCM algorithm was used to identify the structure of a fuzzy model in many-input single-output systems. In this approach, the ordinary fuzzy partition of the input space is avoided. Instead, the output data is clustered and the clusters (prototypes) are projected onto the individual input coordinates. This produces fuzzy rules that can be defuzzified into numeric entities. In our approach, since the output space is granulated with the optimized interval-based information granules, the results are produced in the form of intervals. Similar interval-based clusters were proposed in [28] to granulate time-series and improve the forecasting accuracy of a model. These intervals are optimized by gradually adjusting their widths until they become more reasonable and informative. In our study, an optimization mechanism is used to adjust the widths of the intervals to improve performance of the granular model. In other studies [1,3,54,56], the information granules are induced by projecting hyperbox clusters from the multidimensional input space onto each input coordinate spaces. While this approach exhibits some similarity to the approach pursued here, it differs in the way the input clusters are induced, which is done from information granules located in the output space. In more comparable approaches, a set of intervals formed in the output space is used as classes to turn an approximation problem into a classification task, and to induce the formation of hyperboxes [2] and hyper-ellipsoids [55] in the input space, nonetheless the values at the end are defuzzified into numeric values. In [45], information granulation is used to induce consistent granular structures, which help build a knowledge distance that is used to form a lattice model that reveals the essence of information granularity. Lattice computing is used in several studies to enhance fuzzy models, where intervals are used to represent fuzzy sets, either described as α-cuts [39], or as fuzzy membership functions. Since these intervals are partially-ordered, they are considered as a lattice, hence mathematical lattice theory can be applied to build tunable fuzzy models [19–21,34]. In [46] and [47], the objective function of the clustering algorithm was modified to introduce a directionality component to the formation of the prototypes, the resulting information granules are used to form "if–then" rules. Redundant rules were merged and conflicting rules were eliminated to increase the coverage of important regions in the data. Also in [38] particle swarm optimization (PSO) was used to construct interval-based cluster centers (prototypes), where the widths of the intervals are adjusted through a granulation–degranulation scheme. In this way, a granular version of the original data was formed. By using a coverage criterion as an objective function, it was possible to allocate the granularity of the data.

To put the subject matter in a more general setting, granular fuzzy models are essentially about building models at the level of information granules, not numeric evidence. Since experimental numeric data lead to information granules, these are structured in the form of information granules. Subsequently, these information granules are linked together by developing a backbone (blueprint) of the model. In the design of such linkages it has to be acknowledged that in most situations models are direction-oriented constructs. Clustering itself produces direction-free building blocks. To bring a directionality component onto the granular fuzzy models, the formation of information granules is realized in two phases. First, information granules are constructed in the output space. Second, for a given information granule in the output space, a collection of induced information granules is built by engaging some clustering mechanism. In this sense, information granules emerging in the input space are directly implied by the structure already established in the output space. Furthermore the associations among information granules in the input and output spaces are transparent and in this way we can easily form the linkages of the model.

The objective of this study is to design and develop a granular fuzzy model that takes the input and target data to form an information granule. Having these granules available, the main goal is to build a model at the level of information granules. The design process consists of several main phases: 1) defining and refining intervals in the output space; 2) completing conditional fuzzy clustering on the input data induced by the intervals constructed; and 3) optimizing the intervals and alternatively, the number of clusters in each interval. We are also interested in studying the impact the number of intervals, size of each interval, number of clusters per interval, and the fuzzification factor of the clustering method have in the construction of these information granules. Throughout the study, we demonstrate that in granular modeling, information granules play an important and multifaceted role:

- The model is built as a network of associations among information granules. This supports interpretability of the model, which becomes easily translated into a collection of rules with condition and conclusion parts being formed by the constructed information granules.
- Information granules form conceptual sound building blocks (supported by data) and in light of their functionality, can be used in the formation of a variety of relationships among input and output variables.

- Information granules serve as sound descriptors of data. Each information granule comes with its own well-defined semantics and as such the granules can be associated with a certain linguistically sound meaning. In this way one can produce a general sound view at the data.

In comparison with the earlier studies on linguistic models reported in [16,24,25,40–42], the research discussed here exhibits several novel facets and raises and addresses new interesting issues. The question of optimization of input information granules is of paramount relevance and this is addressed here. The optimization criteria, both coverage and specificity are also studied here. The ensuing Differential Evolution (DE) [44,48,49] is invoked to serve as a key optimization vehicle. A number of vital design issues concerning parameters of the information granules are also raised and discussed.

The study is structured as follows. In Section 2, the architecture of the model is briefly described. Since the model heavily relies on information granules, it is essential to focus on its design. The details of the construction of the information granules and their optimization are presented in Section 3, where we focus on a specialized fuzzy clustering algorithm, namely a context-based FCM. A series of numeric experiments are presented in Section 4 and they show the performance of the model. In the experiments, we use synthetic data and selected data coming from publicly available machine-learning repositories. The model is further refined by applying Differential Evolution (DE) as being explained in Section 5. We also provide more numerical experiments. The concluding comments are covered in Section 6.

2. A general architecture of the model and its underlying processing

The model proposed in this study is inherently structured around information granules and from an architectural perspective it arises as a network of connected information granules. The general topology of the model is shown in Fig. 1; see also [42]. It becomes beneficial to analyze to this figure in more detail to gain a better view at the topology and the ensuing design of the model.

As visualized there, we encounter a collection of connected information granules. There is a collection of information granules in the form of intervals $B_1, B_2, \ldots, B_p$ positioned in the output space. Let $B_h = [b_h^-, b_h^+]$, $h = 1, 2, \ldots, p$ denote the lower and upper bounds of the hth interval, while p is the number of intervals. These intervals form a partition of the output space (meaning that those intervals are pairwise disjoint and cover the entire output space). Each B_h is projected onto the input space and reflected there through a collection of information granules (fuzzy sets) in $\mathbb{R}^n$ located at the first layer of the structure of the model, where $\mathbb{R}$ is the set of real numbers and n is the dimensionality of the input data (input space). In general, for each B_h, there could be a certain number of clusters formed, say $c_1, c_2, \ldots,$ and c_p, respectively.

There are two essential key features of the model. First, we build as a collection of information granules (both in the input and output space) and those in the input space are directly induced by the output information granules (see Fig. 2). Second, the results produced by the model are *granular* rather than *numeric* ones. This becomes apparent by looking at the computing realized by the model.

Any numeric input datum $\boldsymbol{x}_k$, $k = 1, 2, \ldots, N$ "activates" the input information granules and produces the corresponding degrees of membership, where N is the number of data. Those being produced by the fuzzy sets implied by the same output interval are then aggregated (summed) as $z_h(\boldsymbol{x}_k) = \sum_{i=1}^{c} u_{hik}$, where u_{hik} is the ith membership degree in the hth

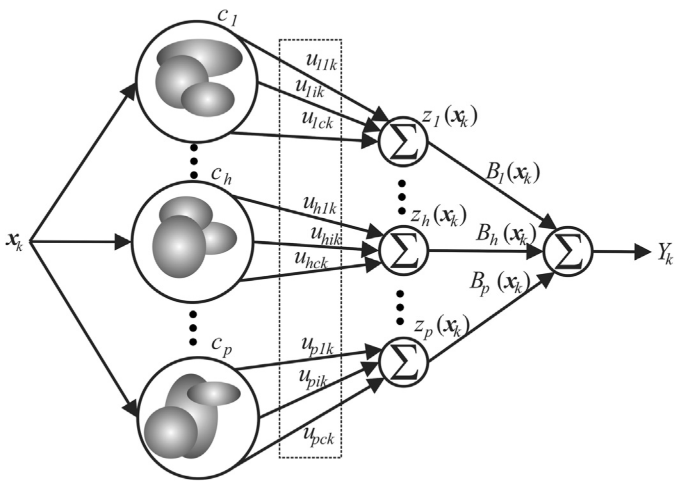

Fig. 1. General architecture of the granular fuzzy model, where a collection of induced information granules c_h is formed by engaging a context-based clustering mechanism, giving rise to several membership degrees u_{hik}, which are summed at each context h to trigger the corresponding activation level $z_h(\boldsymbol{x}_k)$.

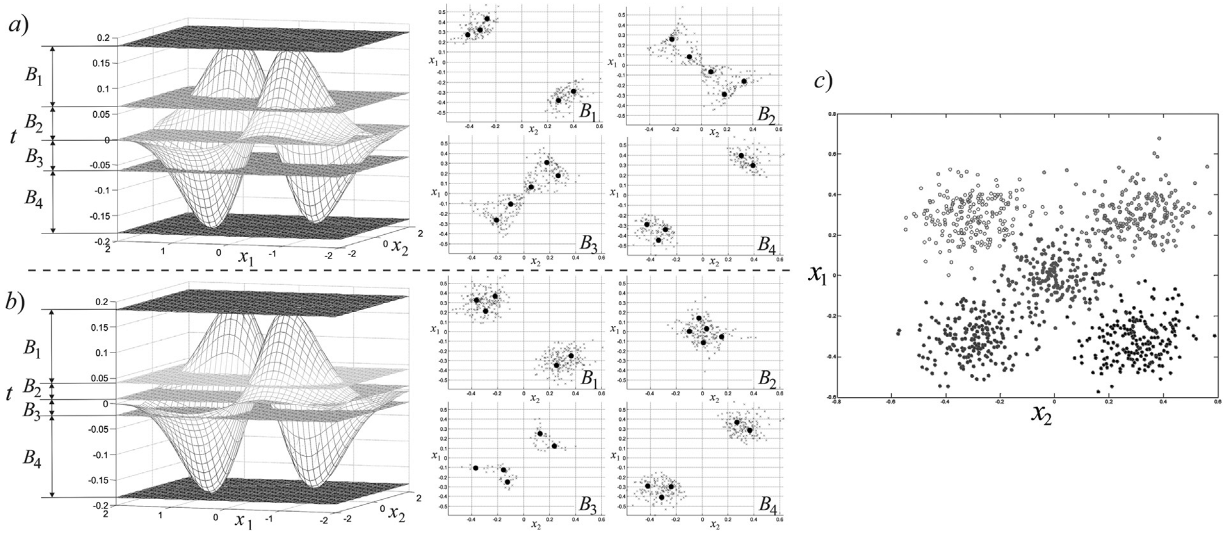

Fig. 2. Output information granules and their influence on the clusters induced in the input space. The intervals placed in the output space (a) exhibit different widths than the intervals in (b). Five clusters are considered for each interval h and the prototypes are represented by black dots.

context for the kth input datum, thus giving rise to the overall activation level $z_h(\boldsymbol{x}_k)$ for the hth interval. In the sequel, the summation of the partial results is performed as follows:

$$Y_k = \sum_{h=1}^{p} \big(z_h(\boldsymbol{x}_k) \otimes B_h(\boldsymbol{x}_k)\big) \tag{1}$$

where $z_h(\boldsymbol{x}_k)$ and $B_h(\boldsymbol{x}_k)$ indicate that these variables are dependent of the input datum $\boldsymbol{x}_k$. We denote the algebraic operation by $\otimes$ to emphasize that the underlying computing operates on a collection of intervals, e.g. $a \otimes R = [a * r^-, a * r^+]$, note that $R = [r^-, r^+]$ and the symbol $*$ is used as the "conventional" multiplication. Following calculations known in interval mathematics [23], the final result is an interval with bounds determined on a basis of the bounds of the individual intervals and their associated activation levels, namely:

$$\begin{aligned} Y_k &= \big[y_k^-, y_k^+\big] = \sum_{h=1}^{p} \big(z_h(\boldsymbol{x}_k) * \big[b_h^-(\boldsymbol{x}_k), b_h^+(\boldsymbol{x}_k)\big]\big) \\ y_k^- &= \big(z_1(\boldsymbol{x}_k) * b_1^-(\boldsymbol{x}_k)\big) + \big(z_2(\boldsymbol{x}_k) * b_2^-(\boldsymbol{x}_k)\big) + \cdots + \big(z_p(\boldsymbol{x}_k) * b_p^-(\boldsymbol{x}_k)\big) \\ y_k^+ &= \big(z_1(\boldsymbol{x}_k) * b_1^+(\boldsymbol{x}_k)\big) + \big(z_2(\boldsymbol{x}_k) * b_2^+(\boldsymbol{x}_k)\big) + \cdots + \big(z_p(\boldsymbol{x}_k) * b_p^+(\boldsymbol{x}_k)\big) \end{aligned} \tag{2}$$

As usual in system modeling, a granular fuzzy model is constructed on a basis of certain data which come in the form of a collection of input–output data pairs $(\boldsymbol{x}_k, t_k)$, where $\boldsymbol{x}_k \in \mathbb{R}^n$ and $t_k \in \mathbb{R}$. The formation of information granules in the output space is guided by some optimization criterion. Once they have been constructed, they induce a family of information granules in the input space.

Considering the architecture of the granular model portrayed in Fig. 1, it can be translated into a series of rules in a straightforward fashion as the condition and conclusion components are shown explicitly in the structure. The clusters (described by their prototypes) form the condition part of the rules while the intervals in the output space are located in the conclusion part of the rule. For the hth interval B_h, we have c clusters positioned in the input space and consequently a union of the input fuzzy sets, thus giving rise to the rule of the following format:

$$\text{if } A_{h1} \text{ or } A_{h2} \text{ or } \ldots \text{ or } A_{hc} \text{ then } B_h \tag{3}$$

where $A_{h1}, A_{h2}, \ldots, A_{hc}$ are the fuzzy sets in $\mathbb{R}^n$ with the membership functions computed on a basis of a set of prototypes $\{\boldsymbol{v}_{h1}, \boldsymbol{v}_{h2}, \ldots, \boldsymbol{v}_{hc}\}$ respectively, where $\boldsymbol{v}_{hi}$, $i = 1, 2, \ldots, c$, describes the coordinates of the ith cluster center of the hth context computed in the clustering stage. In the sequel, if more detailed interpretability is required, the prototypes can be projected onto the individual input variables and then the condition part can be read as a Cartesian product of fuzzy sets defined in the individual input spaces, say $A_{hi} = A_{hi}(1) \times A_{hi}(2) \times A_{hi}(n)$, that reads as

$$A_{hi}(1) \text{ and } A_{hi}(2) \text{ and } \ldots \text{ and } A_{hi}(n) \tag{4}$$

As portrayed in Fig. 2c, a two-dimensional synthetic dataset consisting of 1000 points around five points is randomly generated by using a normal distribution. This leads to five clouds of overlapping data. A symmetric function is then used to calculate a target output value described as $t = x_1 * x_2 * \exp(-x_1^2 - x_2^2)$. Furthermore, the output space is divided into four intervals B_1, B_2, B_3, and, B_4, with a different size and position of the cutoff points (Fig. 2*a* and *b*), in order to construct five clusters in each induced input space. As depicted there, these clusters have a direct influence on the data and on the distribution of the prototypes (black dots) induced in the input space. Henceforth, the formation of the output information granules is crucial and several strategies of their development are to be investigated. The input information granules are constructed for the individual output information granules and those are built by using an augmented version of Fuzzy C-Means (FCM) [9], known as a so-called conditional or context-based FCM [36,37]. The detailed construction of all information granules is covered in the ensuing sections.

3. The development of the model

In this section, we discuss the main design phases of the model. Information granules are viewed as key conceptual components and those will be looked at in great detail. Here we engage a certain optimization process.

3.1. The design of information granules

3.1.1. Construction of output information granules

As previously discussed, the interval information granules formed in the output space are critical to the functioning of the overall model as subsequently they shape up a collection of information granules emerging in the input space. Several intuitively appealing ways of building a collection of intervals are sought:

Equal size intervals. We make the intervals of the same length by splitting the range of the output space into intervals (bins) of equal size. The method is simple however the intervals formed in this manner do not reflect the nature of the data: there could be a few data points for which a separate interval is formed while a large number of data (exhibiting potentially high diversity) are accommodated within the same interval (which could have a detrimental impact on the performance of the model).

Equal probability (EP) intervals. The intervals are formed in such a way so that each interval includes the same number of data (a fraction p/N, to be specific). In other words, the intervals are relatively narrow if there is a large number of data in this region, and become broad if the density of data is low.

Irrespectively of the way the intervals are formed, there is a common limitation associated with them – they do not reflect the nature of input data (by being exclusively focused on the output space and not taking into consideration the input–output relationships).

Optimization of intervals. The intervals are optimized by forming them in a way it models the input–output dependencies (which are the crux of any model). With this regard, we formulate a suitable performance index and establish an optimization scheme.

3.1.2. Construction of input information granules

Information granules in the input space are determined on a basis of the already formed intervals in the output space. The rationale here is to map the output interval to the corresponding regions of the input. The essence is to run clustering, but in contrast to the "standard" FCM, here we invoke a so-called *conditional* FCM [36,37]. We cluster the data that are falling within a context of B_h – in other words, the clustering is conditioned by the output interval.

For each of data set falling within B_h, the task is to determine its structure and obtain a collection of cluster centers (prototypes) by clustering the data with FCM. For simplicity let us consider that the number of clusters c produced for each interval is equal. The FCM algorithm is realized iteratively in each context h by updating the values of the partition matrix and the prototypes. To update the entries of the partition matrix we use the following expression [9,39],

$$u_{hik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\boldsymbol{x}_{hk} - \boldsymbol{v}_{hi}\|}{\|\boldsymbol{x}_{hk} - \boldsymbol{v}_{hj}\|} \right)^{\frac{2}{m-1}}} \tag{5}$$

where $i = 1, 2, \ldots, c$, $k = 1, 2, \ldots, N$, $h = 1, 2, \ldots, p$. Furthermore, u_{hik} represents the element of the partition matrix induced by the ith cluster and the kth data in the hth context, $\boldsymbol{x}_{kh}$ is the kth data in the hth context, $\| \cdot \|$ stands for some distance function, and m (>1.0) is the fuzzification coefficient. Assuming that the Euclidean distance is used, the cluster centers (prototypes) are calculated as follows,

$$\boldsymbol{v}_{hi} = \frac{\sum_{k=1}^{N} u_{hik}^{m} \boldsymbol{x}_{hk}}{\sum_{k=1}^{N} u_{hik}^{m}} \tag{6}$$

The value of the fuzzification coefficient m affects the shape of the membership functions (clusters) being generated. With the lower values of m, the membership functions tend to resemble characteristic functions of sets, whilst obtaining fewer elements with intermediate membership values. Higher values of m produce spiky membership functions with a profound rippling effect, the elements tend to show more local minima, and the values start to position closer to the averages

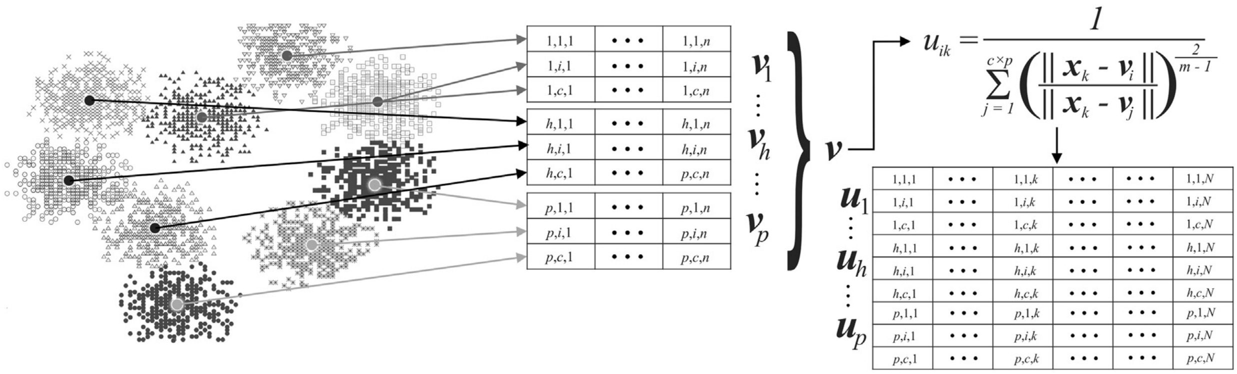

Fig. 3. Calculation of the new partition matrix by using the prototypes obtained in the clustering stage by applying (5) and (6).

of the modes. A common value found in the literature is $m = 2$, which constitutes a reasonable compromise between set-like membership functions and membership function exhibiting excessive oscillations in the membership grades produced by high values of m.

As portrayed in Fig. 3, the prototypes obtained for each context by using (5) and (6), are combined and utilized to determine a new partition matrix for the input data, now having $c \times p$ clusters. The elements of the partition matrix that correspond to each interval are summed as $z_h(x_k) = \sum_{i=1}^{c} u_{hik}$. Since in the traditional FCM the partition matrix has to satisfy the condition $\sum_{i=1}^{c} u_{ik} = 1$, then in this case the next condition has to be satisfied,

$$\sum_{h=1}^{p} z_h(x_k) = 1 \tag{7}$$

Each interval impacts the performance of the clustering mechanism: if $z_h > z_{h+1}$ then the data placed under B_h produces a partition matrix whose result will have a higher membership degree in its corresponding node, henceforth it produces an interval that is formed around its corresponding numeric output value.

3.2. Performance evaluation: coverage and specificity criteria

To evaluate the granular fuzzy model, two criteria are taken into account: coverage and specificity. Here the dominant index is concerned with the coverage of the target data, and this coverage can be affected by a number of different design parameters. These include the number of intervals, the number of clusters in the input space, and the fuzzification coefficient. The selection of these initial values is problem dependent, and finding them is a matter of further optimization.

The key objective studied here is the coverage criterion expressed as follows:

$$f_1 = \frac{1}{N}\sum_{k=1}^{N}(t_k \in Y_k) \tag{8}$$

where the belongingness predicate returns 1 if t_k belongs to Y_k. For the specificity criterion, the length of the output intervals is taken into consideration. There is a compelling reason considering it: if the length is broad then the specificity of the information granule is low, and if the length of the interval decreases, the specificity starts to increase. The measure of specificity is computed as the following sum

$$f_2 = \frac{1}{N}\sum_{k=1}^{N}\left(y_k^+ - y_k^-\right) \tag{9}$$

These coverage and specificity measures are in conflict as the number of elements covered by the information granule increases, the length of the interval might increase and cause a decrease in the specificity, thus becoming less detailed. In order to resolve this conflict and to calibrate the impact of the specificity criterion we consider the following form of f_2:

$$f_2 = e^{-\alpha(l/L)} \tag{10}$$

where α is a non-negative parameter, l is the total average length of the output intervals, and L is the total range of the output variable (i.e. $L = t_{\max} - t_{\min}$). With the modification of the second objective, we maximize the performance index as $Q = f_1 \times f_2$, but in this case we have to take into consideration the value of α, given that a value close to zero can have little to no impact in the specificity criterion, whereas large values of α could have a significant impact in the final coverage. Finding a suitable value of α may call for another optimization problem.

4. Experiments

To illustrate clearly the design process of the proposed granular model, we discuss the following numeric example.

4.1. Synthetic single-input single-output data

A one-dimensional synthetic dataset is generated by using the function $t = \sin(5x)/5x$. The input variable ranges from 1.15 to 7.00; in total 586 data points are generated lying within an interval ranging from -0.091 to 0.128. The data are randomly separated into 70% and 30% to train and test the granular model. The output intervals are formed by using the EP criterion. In the first experiment $p = 5$ intervals were used. For each experiment, the induced data are clustered by using different combinations of the fuzzification coefficients $m \in \{1.3, 1.5, 2, 3\}$ and the number of clusters $c \in \{6, 9, 12, 15\}$. We only report the best and the worst results for all the combinations being tested.

In the reported experiments, the best result is obtained when $m = 1.5$ and $c = 15$, with 95.39% of the data covered and an average length of $l = 19\%$, refer to Fig. 4. If the number of intervals is increased to $p = 9$ (Fig. 5), the best coverage is obtained with the same parameters, and 93.59% of data gets covered, but in this case the average length is $l = 11\%$, making the network of information granules more specific. There is a significant impact on the coverage with different values. Nevertheless, these results are bound to change when a different dataset is used. The final intervals or *envelopes* displayed in Figs. 2 and 3 are represented as red dots. This because the synthetic data are previously separated, henceforth the results show the intervals of the training and testing data in two different plots, with the results of the interval covering only the training or test sets. If both images are to be superimposed, the total interval can be observed.

4.2. Real-world data

For these experiments we used three publicly available datasets [4,7]: Boston housing, delta elevators, and delta ailerons datasets. The Boston housing dataset contains prices of houses from suburbs in Boston. This dataset has 13 continuous values, including the target attribute which is the median value of owned-occupied homes in $1000's with a range of 5 to 50, one binary-valued attribute, and a total of 506 instances [7]. The delta elevators and delta ailerons datasets are obtained from the task of controlling the elevators and the ailerons from an F16 aircraft but both datasets have different domains. The delta elevators dataset contains 9517 examples, with six continuous values, including the target attribute with a range of -0.014 to 0.013, and one integer value. The delta ailerons dataset contains 7129 instances, with six continuous values, which include the target value with a range of -0.0021 to 0.0022 [4].

For all datasets, the same procedure applied to the synthetic data is used now; the instances are randomly separated into the training and testing sets. Once the intervals have been formed with the EP criterion, the coverage of the granular model is evaluated. A number of experiments were completed to explore different outcomes of the model by experimenting with different values of the parameters, these included $p \in \{2, 3, 4, 5\}$, $m \in \{1.1, 1.3, 1.5, 2, 3\}$, and 2 to 12 clusters for each data set. We concentrate on the best and worst results, and also include results where both coverage and specificity show an acceptable performance.

By using the EP strategy and the Boston housing dataset (Table 1), the best result was found with a fuzzification factor of $m = 2.0$ and $c = 10$ clusters, and when the output space comes with two intervals, 90.51% of the data is covered. With three intervals, 82.81% of the data is covered, with the same fuzzification coefficient, but in this case $c = 12$. These results show how increasing the number of intervals decreased the coverage of the data, nevertheless the information granules are more specific. It is noticeable that the fuzzification coefficient plays an important role in the final output of the model. With more intervals, higher fuzzification coefficient, and more clusters the coverage does not increase or starts to decrease. This is a reason why we have only selected the results obtained with up to five intervals.

For the delta elevators dataset, the best results are found when the output space is split into two intervals; with $m = 3.0$, $c = 5$ with 98.23% of the data set covered. On the other hand, with the three intervals and $c = 8$, we observe good coverage and narrower intervals, where 94.91% of the data is covered, see Table 2.

When testing the model with the delta elevators dataset and by partitioning the output space into two intervals, 99.78% of the data is covered when $c = 11$, and by using four output intervals and $c = 10$ then 95.55% data is covered and the specificity increases; for both results $m = 2.0$ and these results are shown in Table 3. Notice that the average length of the intervals is reflective of the number of partitions tested in all datasets, which is owing to the data being separated into equal quantities when using the EP criterion, but only for the delta elevators dataset, the specificity increases with four intervals and becomes reflected in the produced narrower intervals.

Until now, the results show how the specificity decreases as the coverage increases. This behavior is illustrated in Fig. 6. For this plot, the coverage and specificity are mapped by using f_1 and $1 - f_2$ from (8) and (9) respectively, so *specificity* values closer to 1 mean narrower information granules. Overall, by using the EP criterion, the coverage achieves a higher coverage but the specificity significantly increases.

5. Further refinements of the model

So far the model has been built by practically *hardwiring* the existing information granules with the relationships among them being established in a very intuitive fashion, and pivot entirely upon some explicit point of view articulated by the

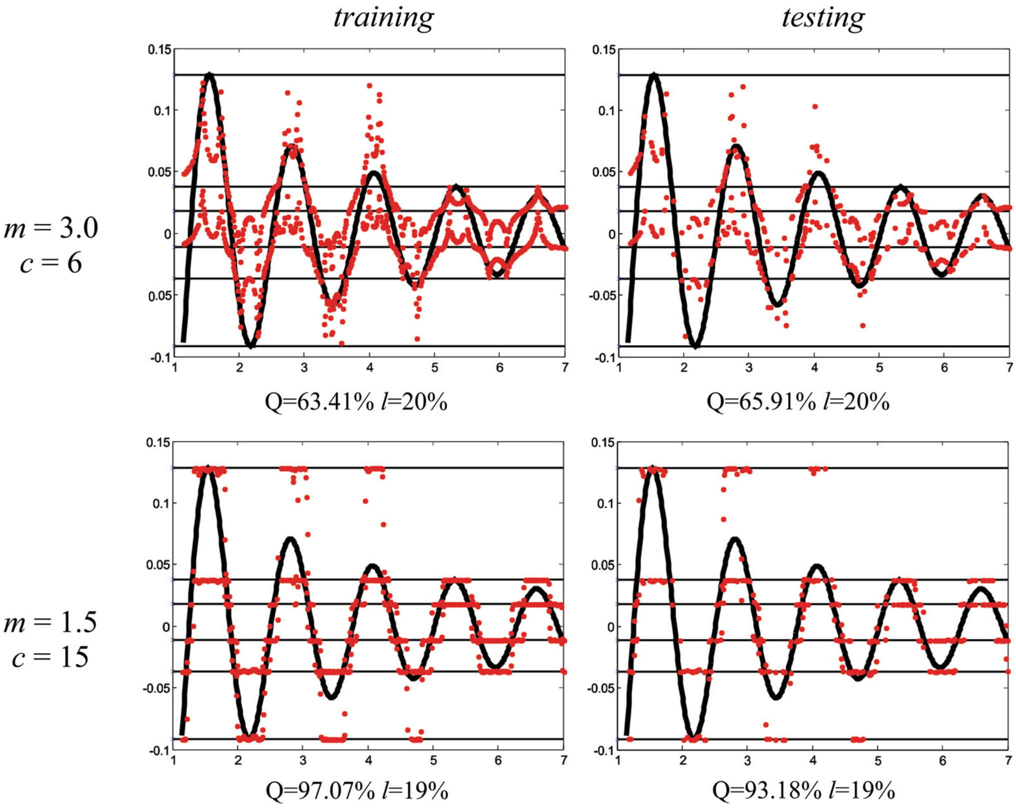

Fig. 4. Worst and best results for the total coverage obtained for synthetic data, using $p = 5$ intervals, different fuzzification coefficients, and different number of clusters, the intervals are obtained by using EP. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

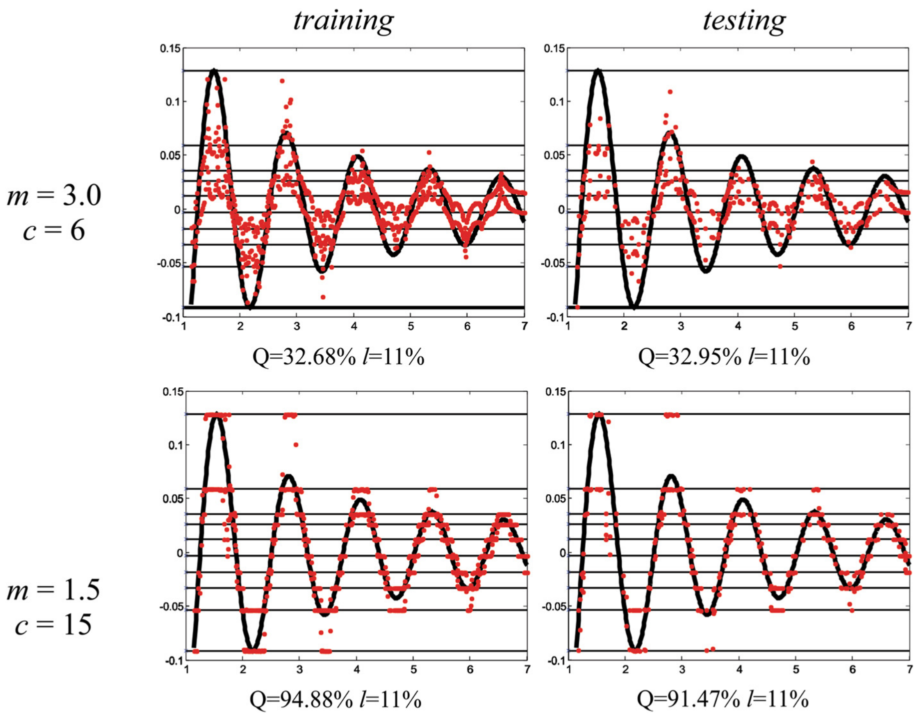

Fig. 5. Worst and best results for the total coverage obtained for synthetic data and 9 intervals, the intervals are obtained by using EP. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

Table 1
Worst and best overall results obtained for the Boston housing dataset and by using EP with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c = 2$		$c = 2$	
		Q	l	Q	l
1.1	Training	44.07	0.28	35.31	0.21
	Testing	40.13	0.26	34.87	0.20
		2 intervals		*3 intervals*	
		$c = 10$		$c = 12$	
2.0	Training	**90.96**	0.50	83.05	0.34
	Testing	**89.47**	0.50	82.24	0.34

Table 2
Worst and best overall results obtained for the delta elevators dataset and by using EP with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c = 10$		$c = 9$	
		Q	l	Q	l
1.1	Training	41.93	0.25	35.90	0.22
	Testing	41.73	0.25	35.22	0.22
		2 intervals		*3 intervals*	
		$c = 5$		$c = 8$	
3.0	Training	**98.23**	0.50	94.92	0.33
	Testing	**98.23**	0.50	94.90	0.33

Table 3
Worst and best overall results obtained for the delta ailerons dataset and by using EP with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c = 4$		$c = 3$	
		Q	l	Q	l
1.1	Training	60.62	0.26	53.33	0.25
	Testing	60.69	0.26	53.86	0.25
		2 intervals		*4 intervals*	
		$c = 11$		$c = 10$	
2.0	Training	**99.78**	0.50	95.87	0.20
	Testing	**99.77**	0.50	94.81	0.20

designer. No refinements were considered for the previous design, thus there is still some room for improvement. In this section, we discuss two approaches that bring some algorithmic enhancements to the topology of the granular model. In the first approach we optimize the sizes of the output information granules, given a number of cutoff points, to maximize the coverage of the data. The second approach is concerned with a distribution of the overall number of information granules. Both of these approaches are optimized by using Differential Evolution (DE) whereas (8) is treated as the fitness function.

5.1. Optimization of output information granules

While the EP criterion used in the original construction is sound to some extent, an improved strategy could be developed. Considering the same performance index as before (8), we optimize the coverage by adjusting the position of cutoff points. In other words, the optimization task is expressed as follows:

$$Q = \sum_{k=1}^{N} (t_k \in Y_k) \rightarrow \max \tag{11}$$

In case t_k falls within the Y_k interval, a value of 1 is returned (*True*). We use the count of the elements that fall within the information granule. The aim is to maximize the number of elements covered by the interval Y_k. For this task, Differential Evolution (DE) is applied to optimize the cutoff points of the output intervals. To initialize the algorithm, an initial population is prepared by randomly generating a set of individuals as vectors of random values in the range $[0, 1]$,

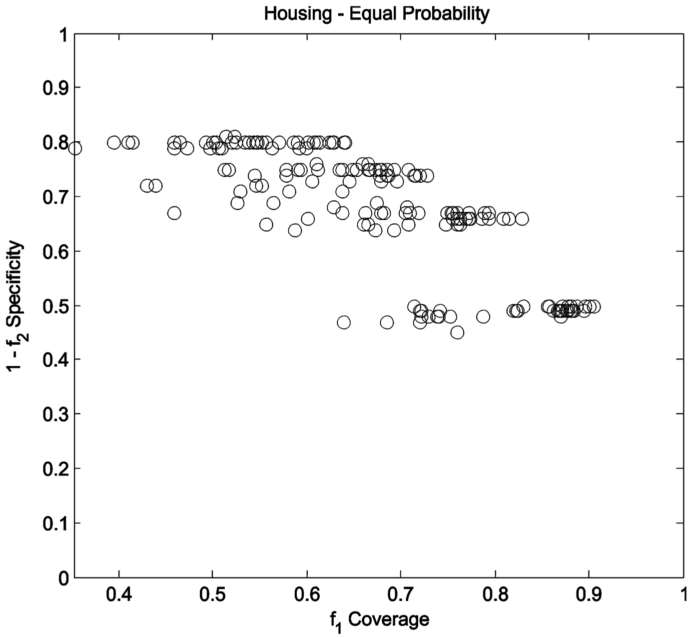

Fig. 6. Coverage vs. specificity for the Boston housing dataset when using EP intervals.

all with a size equal to the number of intervals (p) to represent the width of each interval. These individuals are later normalized into phenotypes that represent the length of the intervals in the output range, and consequently, the cutoff points that granulate (divide) the output space:

$$\boldsymbol{B} = [B_1, B_2, \ldots, B_p] = \left[b_1^-, b_1^+; b_2^-, b_2^+; \ldots; b_p^-, b_p^+\right] \tag{12}$$

Obviously $b_1^- = t_{\min}$ and $b_p^+ = t_{\max}$. For the sake of completeness, the DE algorithm is described in Appendix A. In this study, a population of 30 individuals and 50 generations are used. We observed that the performance index did not improve after more generations and/or when increasing the population sizes over 30 individuals. To test the optimization algorithm, the synthetic data from the previous experiments is used along with the same combinations of number of clusters and fuzzification coefficients. Furthermore, for comparison purposes the data previously separated to train and test the granular model are the same as in the previous experiments.

The initial coverage obtained by using the EP criterion is used as a reference point for the DE algorithm. By partitioning the output space into five partitions, the best coverage is observed when $m = 1.5$ and $c = 12$, where a total of 99.66% of the data set gets covered (Fig. 7). It can be observed how the worst and best results improve when compared to the results displayed in Fig. 4, therefore the network of information granules is optimized, and the specificity slightly decreases as the coverage increases.

When the output space consists of nine intervals, refer to Fig. 8, the best result is obtained with the same fuzzification coefficient and number of cluster observed in the previous results, here 98.46% of the data is covered, and a small increase is observed in the specificity of the information granules, hence we have better coverage with narrower intervals.

When working with the real-world datasets, the overall results are increased, but the toll we have to pay is a decrease in the specificity of the information granules. For the Boston housing dataset (Table 4), if the output space is partitioned into two intervals, the best result is obtained with $m = 2.0$ and $c = 5$, where 95.65% of the data are covered, but the specificity substantially decreases. If the output space comes with four intervals, a sound result is obtained when $m = 2.0$ and $c = 8$, where 82.81% of the data is covered, which is the same coverage as with the previous experiment (Table 1) but with the improved specificity.

With the delta elevators dataset (Table 5), and by using two partitions, the best coverage is obtained when $m = 3.0$ and $c = 8$, here a total of 99.51% of data gets covered. When the output space is divided into three intervals, an acceptable result is obtained with the same fuzzification coefficient and $c = 3$, where 96.29% of the data is covered, showing better coverage with the specificity not being affected, compared to the results displayed in Table 2.

By using the delta ailerons dataset and two partitions in the output space, the best result is obtained when $m = 2.0$ and $c = 11$, where 99.82% of the data is covered, refer to Table 6. If the output space is split to five intervals, 95.64% of the data gets covered when $c = 5$, and specificity of the information granules improves, compared with the results previously obtained with EP (Table 3). Notice how the worst results improve on coverage and specificity for this dataset.

With these results in place, it is observed how DE improves the coverage of the output data, but in some cases the specificity is sacrificed in the process. This behavior is depicted in Fig. 9, where the coverage achieves its highest result with

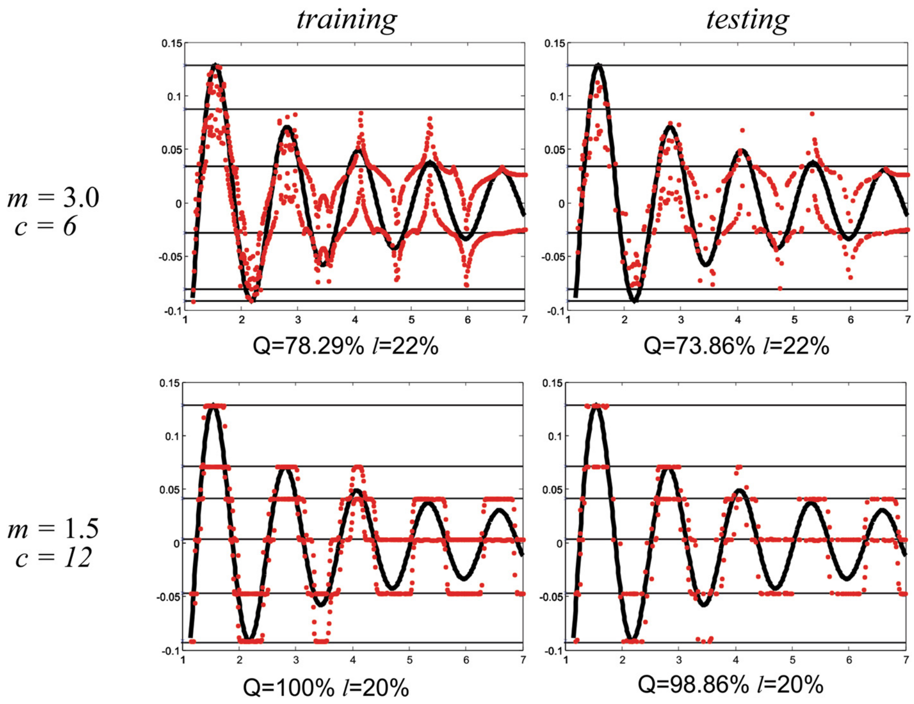

Fig. 7. Worst and best results for the total coverage obtained for synthetic data, 5 intervals, different number of fuzzification coefficients, and different number of clusters using DE. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

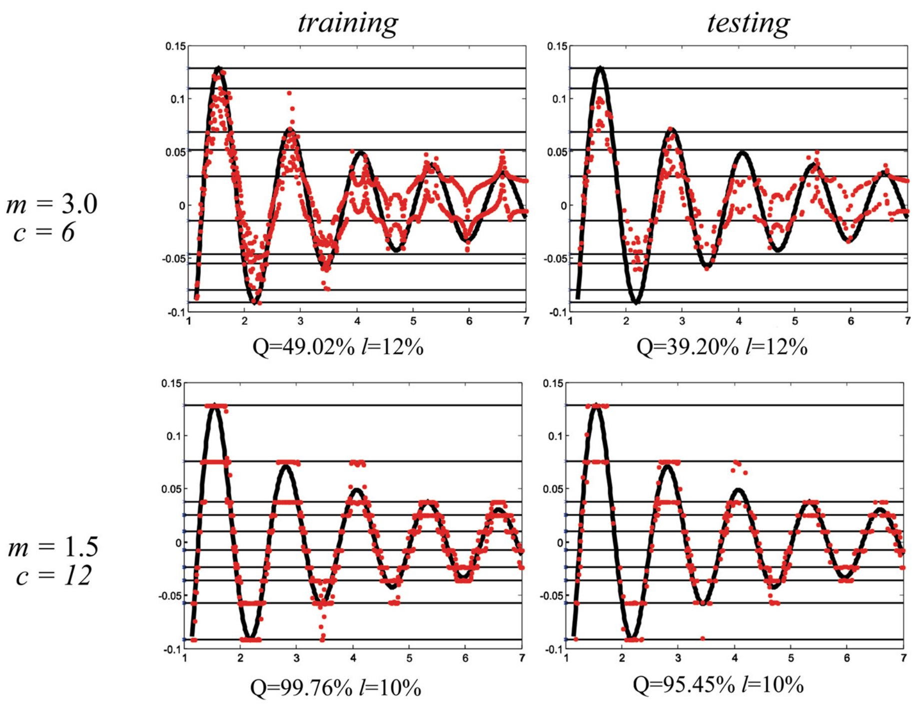

Fig. 8. Worst and best results for the total coverage obtained for synthetic data, 9 intervals, different number of fuzzification coefficients, and different number of clusters using DE. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

less specific intervals. Nonetheless, acceptable results can be found, where good data coverage is obtained with narrower information granules. A second approach is studied where the number of clusters is distributed across the input data, this might help reduce the impact that the coverage has on the specificity of the network of information granules.

Table 4
Worst and best results obtained for the Boston housing dataset when using DE with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c=2$		$c=2$	
		Q	l	Q	l
1.1	Training	57.34	0.39	51.41	0.32
	Testing	53.29	0.39	46.71	0.31
		2 intervals		*4 intervals*	
		$c=5$		$c=8$	
2.0	Training	**96.05**	0.73	84.18	0.31
	Testing	**94.74**	0.71	79.61	0.31

Table 5
Worst and best results obtained for the delta elevators dataset with the use of DE with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c=2$		$c=2$	
		Q	l	Q	l
1.1	Training	56.54	0.35	46.10	0.28
	Testing	54.54	0.35	46.16	0.29
		2 intervals		*3 intervals*	
		$c=8$		$c=3$	
3.0	Training	**99.61**	0.50	96.17	0.33
	Testing	**99.05**	0.50	96.57	0.33

Table 6
Worst and best results obtained for the delta ailerons dataset when using DE with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c=5$		$c=5$	
		Q	l	Q	l
1.1	Training	77.33	0.24	69.54	0.18
	Testing	77.33	0.24	71.01	0.18
		2 intervals		*5 intervals*	
		$c=11$		$c=5$	
2.0	Training	**99.86**	0.50	95.95	0.19
	Testing	**99.72**	0.50	94.90	0.19

5.2. Distribution of the number of clusters (information granules)

Different number of clusters behind each output interval could be beneficial to have a smaller impact on the specificity of the information granules. Here the allocation of information granularity is being carried by the input information granules, by distributing the number of clusters across the input data. We require a total number of information granules to be retained and the optimization process is again carried by DE.

The intervals are initially constructed by using the EP criterion, and the information granules are further developed in the input space to increase the coverage of the output data. To achieve this, we take into account the number of clusters c at each node and the number of partitions p. It is obvious up to this point that the size of the final collection of prototypes has $c \times p$ clusters, so by taking this amount of prototypes as a reference, we use DE as an optimization mechanism to distribute the number of clusters across input nodes and take into account the following conditions; $\sum_{h=1}^{p} c_h = c \times p$ and $c_h \geq 2$.

With the help of the DE algorithm, many different allocations of clusters are tested to enhance the performance of the information granules, and the final output of the model. For this version of the DE algorithm, the size of the individuals is again equal to the number of contexts (p), and in the optimization process the genotypes, which initially assumes values in the range [0, 1], are normalized into a phenotype in the form $c = [c_1, c_2, \ldots, c_h]$, which correspond to the number of clusters in each context. The objective of the DE algorithm is to find an individual that satisfies both conditions and increases the coverage of the target data by using (11). Hence these two circumstances are used as the fitness function in the optimization process. In these experiments, a population of 20 individuals and 50 generations are used; again a larger population size and/or number of generations do not improve the performance of the model.

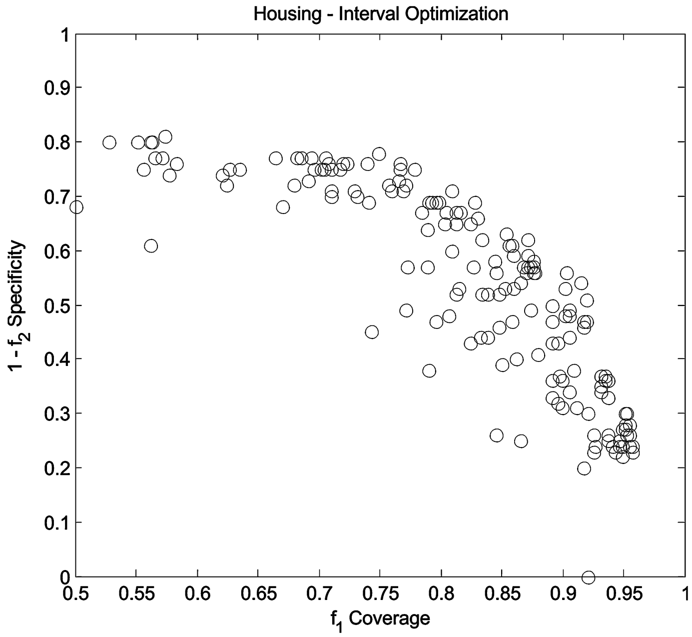

Fig. 9. Coverage vs. specificity for the Boston housing dataset and when optimizing the intervals with DE.

Table 7
Worst and best results obtained for the Boston housing dataset using DE to distribute the number of clusters, with selected values of the parameters.

Fuzzification coefficient (m)		4 intervals			5 intervals	
		$c = 2$			$c = 2$	
		Q	l		Q	l
1.1	Training	48.31	0.29		38.42	0.22
	Testing	40.13	0.27		36.84	0.20
		2 intervals			*5 intervals*	
		$c = 10$			$c = 12$	
2.0	Training	**92.09**	0.50	3.0	85.59	0.30
	Testing	**91.45**	0.50		79.61	0.29

By using the synthetic data and five partitions, the best coverage is obtained when $m = 1.5$ and $c = 15$, where the total number of training and testing sets covered is 99.75% and 97.15% respectively, or 98.98% of the total data; the final distribution of cluster centers that are associated with the interval formation in the output space is 11, 12, 19, 14, and 19. The total average length of the output intervals is $l = 19\%$; by comparing the results for the EP criterion in Section 4, we observe that the coverage increases while the specificity does not change. If the number of partitions is increased to nine, the best coverage is obtained with the same parameters as in the previous experiments. Here 99.76% of the training data and 96.59% of the testing data are covered or 98.17% of the total data. The distribution of the cluster centers in this case is 12, 14, 17, 17, 13, 13, 14, 18, and 17; the coverage is also increased in these experiments and the specificity does not change ($l = 10\%$), compared to the specificity from the experiments in Section 5.1, where the output intervals are optimized with DE.

With the Boston housing dataset (Table 7), if the output space is partitioned into two intervals, the best coverage of the training and testing sets is obtained when $m = 2.0$ and $c = 10$, with 91.90% of the total data covered, the distribution of the cluster centers in this case is 11 and 9. It is noticeable that by using this method there is no direct impact on the specificity, since there is no substantial decrease of specificity as observed when the output intervals were optimized. If the output space is divided into five partitions, $m = 3.0$, and $c = 12$, then 83.79% of the data gets covered, and the distribution of cluster centers is 35, 5, 2, 2 and 19, here we observe a good coverage and better specificity than similar results shown in Tables 1 and 4.

When the delta elevators dataset is considered (Table 8), with two partitions, the best result is obtained when $m = 3.0$ and $c = 12$, where 99.39% of the total data is covered and the specificity remains the same, the cluster centers in each partition are 11 and 13. If the output space is split into five intervals, a good coverage is obtained with the same fuzzification

Table 8
Worst and best results obtained for the delta elevators dataset with the use of DE to distribute the number of clusters, with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c = 2$		$c = 2$	
		Q	l	Q	l
1.1	Training	45.92	0.30	40.45	0.24
	Testing	46.23	0.30	40.91	0.24
		2 intervals		*5 intervals*	
		$c = 12$		$c = 6$	
3.0	Training	**99.49**	0.50	95.51	0.33
	Testing	**99.16**	0.50	95.27	0.33

Table 9
Worst and best results obtained with the delta ailerons dataset with the use of DE to distribute the number of clusters, with selected values of the parameters.

Fuzzification coefficient (m)		*4 intervals*		*5 intervals*	
		$c = 9$		$c = 4$	
		Q	l	Q	l
1.1	Training	68.08	0.30	61.04	0.23
	Testing	66.81	0.29	60.54	0.23
		2 intervals		*4 intervals*	
		$c = 10$		$c = 5$	
3.0	Training	**99.80**	0.50	96.63	0.20
	Testing	**99.77**	0.50	95.65	0.20

factor and $c = 6$, where 95.44% of the data is covered, the final distribution of cluster centers is 11, 2, 3, 2, and 12. In this case the specificity also remains the same but the coverage improves when compared to the results shown in Table 2.

The last dataset tested is the delta ailerons dataset (Table 9), where the best coverage obtained when the output space has two partitions, $m = 3.0$ and $c = 10$ is 99.79%, the distribution of the clusters is 8 and 12. And if four intervals are used, the total coverage is 96.34% when $c = 5$, the distribution of the clusters in each interval is 10, 2, 2, and 6; notice how the specificity is not affected while the coverage increases, when comparing with the similar results shown in Table 3.

Overall the coverage increases, when compared to all the results obtained with the EP criterion. In these experiments an improvement is observed on the specificity in some cases, while in others there is no change, this is illustrated in Fig. 10, where the mapping of the results show a similar behavior as to those observed in Fig. 6, but upon closer inspection, it can be seen how the coverage increases and the specificity is not affected. This is different to the results depicted in Fig. 9, where the specificity significantly decreases, which signifies that the re-arrangement of the clusters also helps in the optimization of the granular model, and does not have a negative impact on the size of the information granules.

Once the model has been optimized, it can be translated into a set of rules. As an example, let us take a two-dimensional dataset, with three intervals, $c = 3$, and $m = 1.5$. After optimization, the intervals resulted in $B_1 = [-0.18, -0.04]$, $B_2 = [-0.04, -0.04]$, and $B_3 = [0.04, 0.18]$. The extraction of the rules is straightforward, as shown in Fig. 11*a*. By using the clusters described by their prototypes to form the condition part, and the optimized intervals to form the conclusion, the rules will read as:

– if $A_{1,1}$ or $A_{1,2}$ or $A_{1,3}$ then $[-0.18, -0.04]$,

The projection of the prototypes can be seen in Fig. 11*b*, where each prototype is projected onto the individual input variables. In this case the condition part reads as

– $A_{1,1} = A_{1,1}(1)$ and $A_{1,1}(2)$

To show the impact of different values of α, and by considering the coverage and specificity criterions, we used the performance index ($Q = f_1 \times f_2$) from (8) and (10) as the fitness function for the DE algorithm. In these experiments the Boston housing dataset is used, the output space is partitioned into three intervals and $c = 5$, $m = 2.0$, since these are the parameters where the best results were found: these are shown in Table 10. As it is observed, after $\alpha = 0.2$ the specificity increases but the coverage decreases. Here we can say that an optimal value of α is around 0.2, since the coverage is still acceptable, and there is an increment in the specificity of the information granules.

In summary, we studied several different techniques applied to the construction of the output information granules, we envisioned equal length intervals, but as shown in Fig. 12a, their size and position heavily depend on the distribution of the

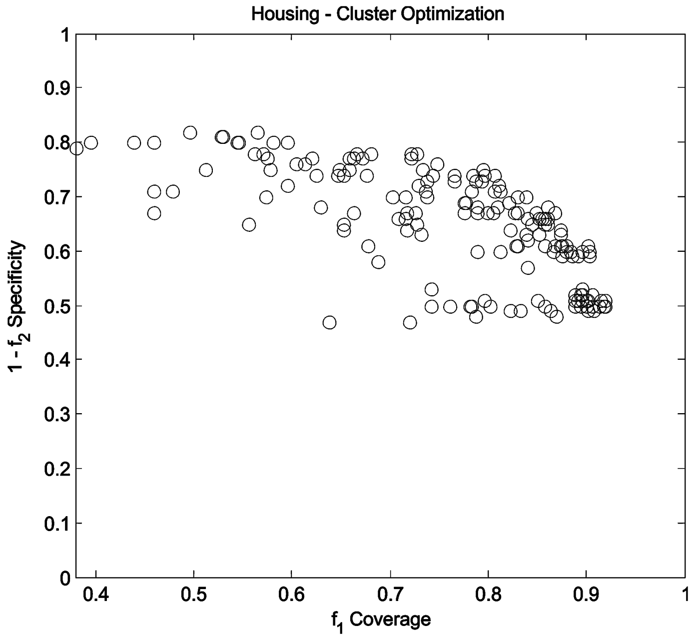

Fig. 10. Coverage vs. specificity for the Boston housing dataset when re-arranging the number of clusters.

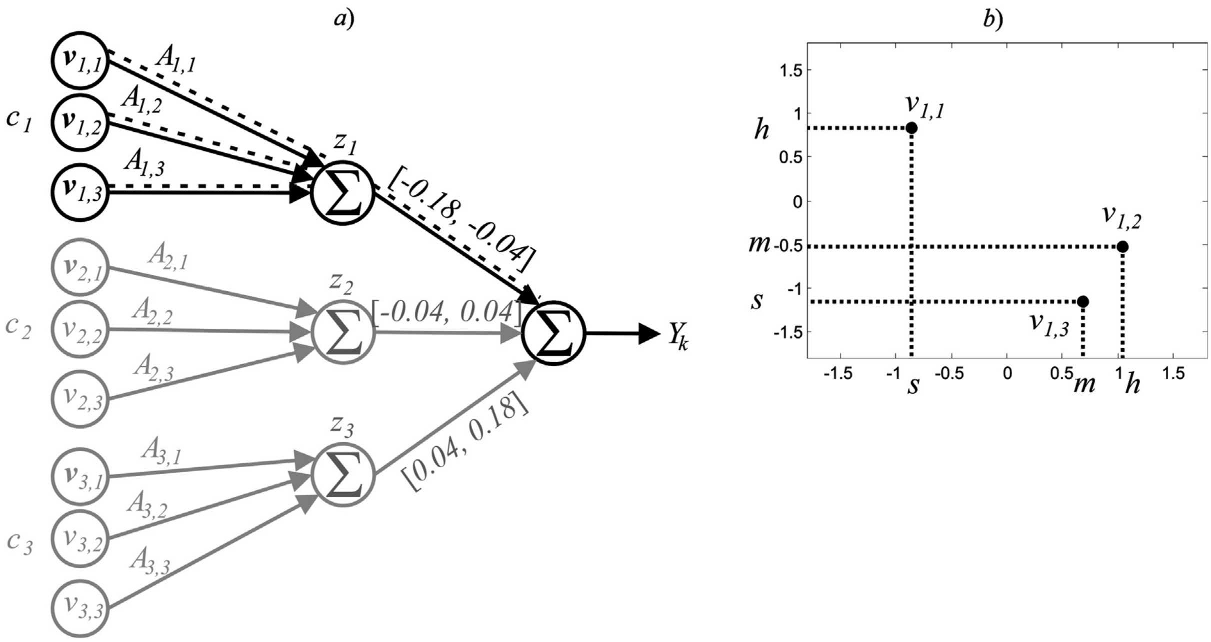

Fig. 11. Optimized granular model with *a*) its interpretation in the form of "if–then" statements and *b*) the projection of the prototypes to their individual input space, where *s*, *m*, and *h* are *small*, *medium*, and *high* labels of the prototypes.

Table 10
Impact of different incremental values of α on the specificity and coverage of the output data, where *lgt*, *tr*, and *te* are, length, training set and testing set, respectively.

α	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00	1.10	1.20	1.30
lgt	0.72	0.52	0.44	0.37	0.40	0.36	0.35	0.35	0.36	0.36	0.35	0.35	0.33	0.33
tr	96.05	95.81	90.11	88.98	89.55	88.14	87.57	88.42	88.42	88.70	87.85	87.57	60.73	58.76
te	94.74	92.84	88.26	76.97	78.29	76.97	75.00	76.97	75.66	76.32	75.00	73.68	52.63	51.32

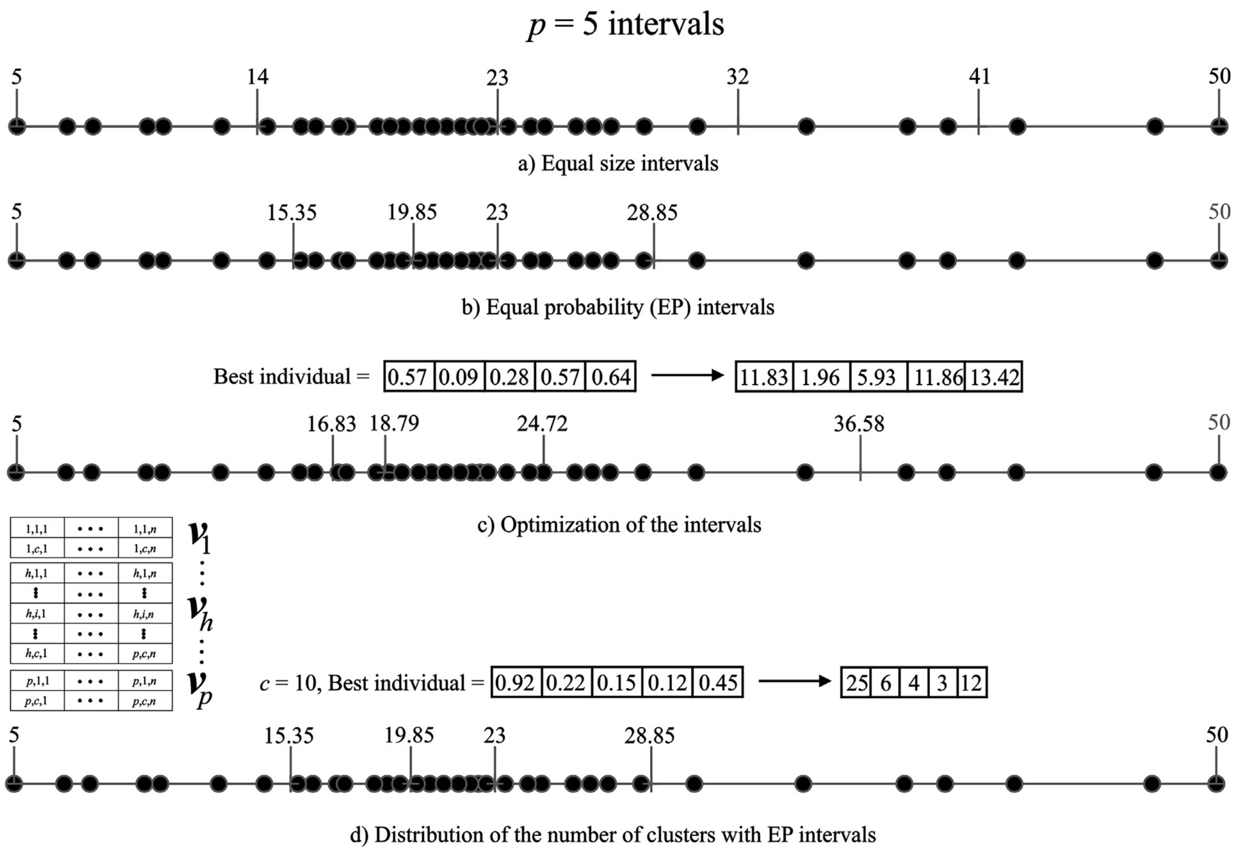

Fig. 12. Techniques applied to the construction of the output information granules.

data. So it was decided to split the output data into equal amounts, or equal probability (Fig. 12b), but this method required an almost manual construction of the information granules. Instead, we let DE fine-tune the intervals, as depicted in Fig. 12c. This method proved to achieve a better coverage, but in some cases a decrease in the specificity was observed. Alternatively, it was decided to optimize the number of clusters positioned in the induced data, and use the intervals constructed with the EP criterion, as illustrated in Fig. 12d. This method yielded good coverage, with little to no impact on the specificity. At the end, it was also shown that different factors affect the final output of the model, namely the amount of intervals and their size, the amount of induced clusters formed in the input space along with their distribution, and the value of the fuzzification factor. Henceforth, it is ideal to have an optimization algorithm which intuitively constructs the network of information granules, and improves the overall performance of the granular fuzzy model, this means finding a model that produces specific (narrow) information granules, and at the same time, covers most of the data.

6. Conclusions

Granular fuzzy models offer a new and attractive venue of system modeling by building models at the level of information granules, which are regarded as essential building blocks. The leitmotiv of the underlying design is that the fuzzy model is formed at the level of information granules and associations among them. Once a collection of intuitively appealing associations among information granules has been formed, their detailed numerically-inclined optimization can be carried out. We demonstrated a number of attractive possibilities including the optimization of output information granules (which is crucial as all input information granules are induced by them), re-distribution of input information granules and a range of parametric optimization dealing with the parameters of the clustering method. While the coverage criterion was intensively exploited in this study, further investigations of the bi-criteria scenario involving both coverage and specificity could be thoroughly investigated.

There are two interesting avenues of further studied directly linked with the undertaken research reported here. The topology of the proposed granular model is general – further realizations including other formalisms of information granules (rough sets, probabilistic sets) are worth exploring. Second, the assessment of the quality of the model can be pursued by admitting further generalizations of the output intervals and making them granular intervals (viz. intervals whose bounds are not numbers but information granules themselves).

Acknowledgements

The first author fully acknowledges support from the National Council of Science and Technology (CONACYT), Mexico under grant No. 213501/308562. He also acknowledges the assistance provided by The Autonomous University of Tlaxcala (UAT), Mexico. The second author fully acknowledges support from the Canada Research Chairs (CRC), Natural Sciences and Engineering Research Council of Canada (NSERC), and the National Centre for Research (NCN) under grant No. UMO-2012/05/B/ST6/03068.

Appendix A

Differential Evolution (DE) is a stochastic, population-based search strategy developed by Price and Storn [49]. This algorithm exhibits a clear resemblance with other evolutionary algorithms (EA), but it differs considerably in the sense that distance and direction information from the current population is used to guide the search process. Another difference is that mutation is applied first to generate a trial vector, and secondly a crossover operator is utilized to produce an offspring.

In this study, we utilized two instances of DE algorithms, each one having different objective: i) optimize the intervals used to granulate the target vector, which will induce the information granules in the input space, and ii) distribute the number of clusters for each induced input node. Both algorithms have the same objective; optimize the network of information granules by considering criteria of coverage and information specificity, and in the case of the second version of DE, consider the previous criteria and fulfill two conditions ($\sum_{h=1}^{p} c_h = c \times p$, $c_h \geq 2$). To achieve this, a population of s individuals is prepared, each representing a vector $\boldsymbol{d}$ initialized by using a normal random distribution in $[0, 1]$, with a size equal to the number of intervals or $q = p$. For each parent $\boldsymbol{d}_g(t)$, where $g = 1, 2, \ldots, s$, we generate a trial vector $\boldsymbol{o}_g(t)$ as follows [14]: select a target vector $\boldsymbol{d}_{g_1}(t)$ from the population, then randomly select two individuals $\boldsymbol{d}_{g_2}(t)$ and $\boldsymbol{d}_{g_3}(t)$ such that $g \neq g_1 \neq g_2 \neq g_3$. With these individuals, the trial vector is computed by mutating the target vector as follows:

$$\boldsymbol{o}_g(t) = \boldsymbol{d}_{g_1}(t) + \beta\big(\boldsymbol{d}_{g_2}(t) - \boldsymbol{d}_{g_3}(t)\big) \tag{A.1}$$

where $\beta \in (0, \infty)$ is a scaling factor, used to control the amplification of the differential variation. Next we apply a discrete recombination of the trial vector $\boldsymbol{o}_g(t)$ and the parent vector $\boldsymbol{d}_g(t)$ to produce offspring $\boldsymbol{d}'_g(t)$, as follows:

$$d'_{gq}(t) = \begin{cases} o_{gq}(t) & \text{if } q \in \Theta \\ d_{gq} & \text{otherwise} \end{cases} \tag{A.2}$$

where $d_{gq}(t)$ is the qth element of vector $\boldsymbol{d}_g(t)$, and Θ is a set of element indices that will suffer mutation, or a set of crossover points. The crossover points are randomly selected from a set of possible crossover points by using *binomial crossover* [14,44,48], where a *probability of crossover* value is used; the larger the value of the probability, the more crossover points are selected. In other words, more elements of the trial vector will be used to produce offspring. At the end the fitness of the offspring is compared to the fitness of the parent, if the coverage of the offspring is higher, it replaces its parent, if the parent's coverage is better it remains in the population.

With DE, the larger the population, more differential vectors become available, and more directions can be explored. But it should be kept in mind that large population sizes increase the computational complexity per generation, and can cause the search to degrade to a parallel random search. Small population sizes would decrease the exploration abilities of the algorithms; however, the size of the population is always problem dependent.

References

[1] S. Abe, M.-S. Lan, R. Thawonmas, Tuning of a fuzzy classifier derived from data, in: Proceedings of the Third IEEE Conference on IEEE World Congress on Computational Intelligence, Jun. 1994, in: Fuzzy Systems, vol. 2, 1994, pp. 786–791.

[2] S. Abe, M.-S. Lan, A function approximator using fuzzy rules extracted directly from numerical data, in: Proceedings of 1993 International Joint Conference on Neural Networks, IJCNN '93-Nagoya, Oct., vol. 2, 1993, pp. 1887–1892.

[3] S. Abe, M.-S. Lan, A method for fuzzy rules extraction directly from numerical data and its application to pattern classification, IEEE Trans. Fuzzy Syst. 3 (1) (Feb. 1995) 18–28, http://dx.doi.org/10.1109/91.366565.

[4] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, J. Mult.-Valued Log. Soft Comput. 17 (2–3) (2010) 255–287.

[5] R. Alcala, M.J. Gacto, F. Herrera, A fast and scalable multiobjective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems, IEEE Trans. Fuzzy Syst. 19 (4) (Aug. 2011) 666–681, http://dx.doi.org/10.1109/TFUZZ.2011.2131657.

[6] J. Abonyi, R. Babuska, F. Szeifert, Modified Gath–Geva fuzzy clustering for identification of Takagi–Sugeno fuzzy models, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 32 (5) (Oct. 2002) 612–621, http://dx.doi.org/10.1109/TSMCB.2002.1033180.

[7] K. Bache, M. Lichman, UCI machine learning repository [Online], Available: http://archive.ics.uci.edu/ml, Dec. 11, 2013.

[8] P. Baranyi, The Generalized TP model transformation for TS fuzzy model manipulation and generalized stability verification, IEEE Trans. Fuzzy Syst. 22 (4) (Aug. 2014) 934–948, http://dx.doi.org/10.1109/TFUZZ.2013.2278982.

[9] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Springer-Verlag, US, 1981.

[10] J. Casillas, O. Cordon, M.J. Del Jesus, F. Herrera, Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction, IEEE Trans. Fuzzy Syst. 13 (1) (Feb. 2005) 13–29, http://dx.doi.org/10.1109/TFUZZ.2004.839670.

[11] O. Cordon, F. Herrera, A proposal for improving the accuracy of linguistic modeling, IEEE Trans. Fuzzy Syst. 8 (3) (Jun. 2000) 335–344, http://dx.doi.org/10.1109/91.855921.

[12] O. Cordon, F. Herrera, I. Zwir, Linguistic modeling by hierarchical systems of linguistic rules, IEEE Trans. Fuzzy Syst. 10 (1) (Feb. 2002) 2–20, http://dx.doi.org/10.1109/91.983275.
[13] Z. Deng, Y. Jiang, K.-S. Choi, F.-L. Chung, S. Wang, Knowledge-leverage-based TSK fuzzy system modeling, IEEE Trans. Neural Netw. Learn. Syst. 24 (8) (Aug. 2013) 1200–1212, http://dx.doi.org/10.1109/TNNLS.2013.2253617.
[14] A.P. Engelbrecht, Computational Intelligence: An Introduction, 2nd ed., Wiley & Sons, London, UK, 2007.
[15] A.E. Gaweda, J.M. Zurada, Data-driven linguistic modeling using relational fuzzy rules, IEEE Trans. Fuzzy Syst. 11 (1) (Feb. 2003) 121–134, http://dx.doi.org/10.1109/TFUZZ.2002.803491.
[16] Y.H. Han, K.C. Kwak, An optimization of granular network by evolutionary methods, in: 9th WSEAS International Conference on Artificial Intelligence, 2010, pp. 65–70.
[17] J.-S.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cybern. 23 (3) (May/Jun. 1993) 665–685, http://dx.doi.org/10.1109/21.256541.
[18] C.-F. Juang, C.-Y. Chen, An interval type-2 neural fuzzy chip with on-chip incremental learning ability for time-varying data sequence prediction and system control, IEEE Trans. Neural Netw. Learn. Syst. 25 (1) (Jan. 2014) 216–228, http://dx.doi.org/10.1109/TNNLS.2013.2253799.
[19] V.G. Kaburlasos, S.E. Papadakis, A granular extension of the fuzzy-ARTMAP (FAM) neural classifier based on fuzzy lattice reasoning (FLR), Neurocomputing 72 (10–12) (2009) 2067–2078, http://dx.doi.org/10.1016/j.neucom.2008.06.024.
[20] V.G. Kaburlasos, Granular Fuzzy Inference System (FIS) design by lattice computing, in: Hybrid Artificial Intelligence Systems, in: Lect. Notes Comput. Sci., vol. 6077, 2010, pp. 410–417.
[21] V.G. Kaburlasos, G.A. Papakostas, T. Pachidis, A. Athinellis, Intervals' Numbers (INs) interpolation/extrapolation, in: IEEE International Conference on Fuzzy Systems (FUZZ), 2013, pp. 1–8.
[22] N.K. Kasabov, Qun Song, DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction, IEEE Trans. Fuzzy Syst. 10 (2) (Apr. 2002) 144–154, http://dx.doi.org/10.1109/91.995117.
[23] R.B. Kearfott, Interval computations: introduction, uses, and resources, Euromath Bull. 2 (1) (1996) 95–112.
[24] K.C. Kwak, An optimization of granular networks based on PSO and two-sided Gaussian contexts, Int. J. Adv. Res. Artif. Intell. 1 (9) (2012), http://dx.doi.org/10.14569/IJARAI.2012.010901.
[25] K.C. Kwak, W. Pedrycz, A design of genetically oriented linguistic model with the aid of fuzzy granulation, in: IEEE International Conference on Fuzzy Systems (FUZZ), 18–23 Jul. 2010, 2010, pp. 1–6.
[26] A. Lemos, W. Caminhas, F. Gomide, Multivariable Gaussian evolving fuzzy modeling system, IEEE Trans. Fuzzy Syst. 19 (1) (Feb. 2011) 91–104, http://dx.doi.org/10.1109/TFUZZ.2010.2087381.
[27] C. Li, J. Zhou, B. Fu, P. Kou, J. Xiao, T–S fuzzy model identification with a gravitational search-based hyperplane clustering algorithm, IEEE Trans. Fuzzy Syst. 20 (2) (Apr. 2012) 305–317, http://dx.doi.org/10.1109/TFUZZ.2011.2173693.
[28] W. Lu, X. Chen, W. Pedrycz, X. Liu, J. Yang, Using interval information granules to improve forecasting in fuzzy time series, Int. J. Approx. Reason. 57 (February 2015) 1–18, http://dx.doi.org/10.1016/j.ijar.2014.11.002.
[29] J. Lu, X. Yuan, T. Yahagi, A method of face recognition based on fuzzy c-means clustering and associated sub-NNs, IEEE Trans. Neural Netw. 18 (1) (Jan. 2007) 150–160, http://dx.doi.org/10.1109/TNN.2006.884678.
[30] D. Nauck, F. Klawonn, R. Kruse, Combining neural networks and fuzzy controllers, in: Fuzzy Logic in Artificial Intelligence, in: Lect. Notes Comput. Sci., vol. 695, 1993, pp. 35–46.
[31] D. Nauck, R. Kruse, A fuzzy neural network learning fuzzy control rules and membership functions by fuzzy error backpropagation, IEEE International Conference on Neural Networks 2 (1993) 1022–1027, http://dx.doi.org/10.1109/ICNN.1993.298698.
[32] D. Nauck, R. Kruse, R. Stellmach, New learning algorithms for the neuro-fuzzy environment NEFCON-I, in: Proceedings of Neuro-Fuzzy-Systems, 1995, pp. 357–364.
[33] D. Nauck, U. Nauck, R. Kruse, Generating classification rules with the neuro-fuzzy system NEFCLASS, in: Biennial Conference of the North American, Fuzzy Information Processing Society, NAFIPS, Jun. 1996, pp. 466–470.
[34] S.E. Papadakis, V.G. Kaburlasos, Piecewise-linear approximation of non-linear models based on probabilistically/possibilistically interpreted intervals' numbers (INs), Inf. Sci. 180 (24) (2010) 5060–5076, http://dx.doi.org/10.1016/j.ins.2010.03.023.
[35] M. Pratama, S.G. Anavatti, P.P. Angelov, E. Lughofer, PANFIS: a novel incremental learning machine, IEEE Trans. Neural Netw. Learn. Syst. 25 (1) (Jan. 2014) 55–68, http://dx.doi.org/10.1109/TNNLS.2013.2271933.
[36] W. Pedrycz, Conditional fuzzy C-means, Pattern Recognit. Lett. 17 (6) (May 1996) 625–631, http://dx.doi.org/10.1016/0167-8655(96)00027-X.
[37] W. Pedrycz, Conditional fuzzy clustering in the design of radial basis function neural networks, IEEE Trans. Neural Netw. 9 (4) (Jul. 1998) 601–612, http://dx.doi.org/10.1109/72.701174.
[38] W. Pedrycz, A. Bargiela, An optimization of allocation of information granularity in the interpretation of data structures: toward granular fuzzy clustering, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 42 (3) (2012) 582–590, http://dx.doi.org/10.1109/TSMCB.2011.2170067.
[39] W. Pedrycz, F. Gomide, Fuzzy Systems Engineering: Toward Human-Centric Computing, 1st ed., Wiley–IEEE Press, Hoboken, NJ, 2007.
[40] W. Pedrycz, K.C. Kwak, Linguistic model as a framework of user-centric system modeling, IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum. 36 (4) (Jul. 2006) 727–745, http://dx.doi.org/10.1109/TSMCA.2005.855755.
[41] W. Pedrycz, K.C. Kwak, Boosting of granular models, Fuzzy Sets Syst. 157 (22) (Nov. 2006) 2934–2953, http://dx.doi.org/10.1016/j.fss.2006.07.005.
[42] W. Pedrycz, A.V. Vasilakos, Linguistic models and linguistic modeling, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 29 (6) (Dec 1999) 745–757, http://dx.doi.org/10.1109/3477.809029.
[43] C. Porcel, A.G. López-Herrera, E. Herrera-Viedma, A recommender system for research resources based on fuzzy linguistic modeling, Expert Syst. Appl. 36 (3, part 1) (Apr. 2009) 5173–5183, http://dx.doi.org/10.1016/j.eswa.2008.06.038.
[44] K.V. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer-Verlag, 2005.
[45] Y. Qian, H. Zhang, F. Li, Q. Hu, J. Liang, Set-based granular computing: a lattice model, Int. J. Approx. Reason. 55 (3) (March 2014) 834–852, http://dx.doi.org/10.1016/j.ijar.2013.11.001.
[46] M. Setnes, Supervised fuzzy clustering for rule extraction, in: IEEE International Fuzzy Systems Conference Proceedings, FUZZ-IEEE '99, Aug., vol. 3, 1999, pp. 1270–1274.
[47] M. Setnes, Supervised fuzzy clustering for rule extraction, IEEE Trans. Fuzzy Syst. 8 (4) (Aug. 2000) 416–424, http://dx.doi.org/10.1109/91.868948.
[48] R. Storn, On the usage of differential evolution for function optimization, in: Biennial Conference of the North American Fuzzy Information Processing Society, NAFIPS, Jun. 1996, 1996, pp. 519–523.
[49] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (4) (Dec. 1997) 341–359, http://dx.doi.org/10.1023/A:1008202821328.
[50] M. Sugeno, T. Yasukawa, A fuzzy-logic-based approach to qualitative modeling, IEEE Trans. Fuzzy Syst. 1 (1) (Feb. 1993) 7, http://dx.doi.org/10.1109/TFUZZ.1993.390281.
[51] Y. Sun, G. Feng, J. Cao, A new approach to dynamic fuzzy modeling of genetic regulatory networks, IEEE Trans. Nanobiosci. 9 (4) (Dec. 2010) 263–272, http://dx.doi.org/10.1109/TNB.2010.2082559.
[52] Y. Tang, J. Zheng, Linguistic modelling based on semantic similarity relation among linguistic labels, Fuzzy Sets Syst. 157 (12) (Jun. 2006) 1662–1673, http://dx.doi.org/10.1016/j.fss.2006.02.014.

[53] K. Trawinski, O. Cordon, L. Sanchez, A. Quirin, A genetic fuzzy linguistic combination method for fuzzy rule-based multiclassifiers, IEEE Trans. Fuzzy Syst. 21 (5) (Oct. 2013) 950–965, http://dx.doi.org/10.1109/TFUZZ.2012.2236844.
[54] R. Thawonmas, S. Abe, A fuzzy classifier based on partitioned hyperboxes, in: IEEE International Conference on Neural Networks, 3–6 Jun. 1996, vol. 2, 1996, pp. 1097–1102.
[55] R. Thawonmas, S. Abe, Function approximation based on fuzzy rules extracted from partitioned numerical data, IEEE Trans. Syst. Man Cybern., Part B, Cybern. 29 (4) (Aug. 1999) 525–534, http://dx.doi.org/10.1109/3477.775268.
[56] N. Tschichold-Gürman, Generation and improvement of fuzzy classifiers with incremental learning using fuzzy RuleNet, in: Proceedings of the 1995 ACM Symposium on Applied Computing, SAC '95, ACM, New York, NY, USA, 1995, pp. 466–470.
[57] G.E. Tsekouras, On the use of the weighted fuzzy c-means in fuzzy modeling, Adv. Eng. Softw. 36 (5) (May 2005) 287–300, http://dx.doi.org/10.1016/j.advengsoft.2004.12.001.
[58] O. Uncu, I.B. Turksen, Discrete interval type 2 fuzzy system models using uncertainty in learning parameters, IEEE Trans. Fuzzy Syst. 15 (1) (Feb. 2007) 90–106, http://dx.doi.org/10.1109/TFUZZ.2006.889765.
[59] Y. Xie, V.V. Raghavan, P. Dhatric, X. Zhao, A new fuzzy clustering algorithm for optimally finding granular prototypes, Int. J. Approx. Reason. 40 (1–2) (July 2005) 109–124, http://dx.doi.org/10.1016/j.ijar.2004.11.002.