

# Exploiting Particle Swarm Optimization to Attune Strong Fuzzy Partitions Based on Cuts

Ciro Castiello and Corrado Mencar

Department of Informatics, University of Bari Aldo Moro, Italy  
{ciro.castiello, corrado.mencar}@uniba.it

## Abstract

Cut-based strong fuzzy partitions (SFP) are characterized by cuts, i.e. points in the universe of discourse where the non-zero membership degrees of the fuzzy sets in the partition is 0.5. Cuts are useful to identify the most representative regions for the fuzzy sets involved in a SFP but pose loose constraints on the slopes of trapezoidal fuzzy sets. We address the problem of optimizing such slopes in order to maximize the performance of fuzzy rule-based systems while keeping cuts constant. This way, model performance is improved and interpretability is preserved. We use Particle Swarm Optimization to perform optimization and we analyze two different approaches for generating solution spaces. We tested the proposed approach on a number of fuzzy rule-based classifiers designed by DC\* (Double Clustering with A\*) on synthetic data. For all the considered models, performance is never degraded but improved in many cases, without violating any interpretability constraint.

**Keywords:** Strong fuzzy partitions, Particle Swarm Optimization, Trapezoidal fuzzy sets, DC\*.

## 1 Introduction

Linguistic variables (LVs) are the founding grounds to establish the knowledge base of a linguistic fuzzy rule base system. Due to their nature, conjugating the symbolic promptness of idiomatic expressions with the mathematical apparatus provided by fuzzy sets, LVs offer the possibility to model the information related to some data domain in a human-comprehensible fashion. Indeed, LVs assume the values of linguistic terms which naturally support the different attempts to un-

derstand the perceived reality in the way it is described and accounted for through natural language. Fuzzy sets represent the semantic interpretations of linguistic terms; an ensemble of fuzzy sets, possibly describing a data feature related to a LV, constitutes a fuzzy partition. As implied by its name, a fuzzy partition provides a way to split the domain associated to a LV, according to some prearranged granularity level. Adjusting a fuzzy partition is a crucial task since it must preserve co-intension with the user's understanding of the concepts at hand [6]. In other words, a fuzzy partition should capture a relationship among fuzzy sets which is consistent with the cognitive system characterizing the common sense of humans. Moreover, interpretability issues must be taken in special account, because the mere employment of fuzzy logic does not guarantee a knowledge representation which is ultimately interpretable [1].

For all those reasons, the design of fuzzy partitions is a topic largely discussed in literature and the adoption of Strong Fuzzy Partitions (SFPs) is acknowledged as a best practice among scholars [3]. SFP is a special case of fuzzy partitions whose fuzzy sets express membership degrees summing up to one when applied to each single element inside the Universe. Such a characterization enables the fulfillment of some notable interpretability constraints and paves the way to understand fuzzy membership in terms of similarity with respect to a prototype. In fact, the lower a membership value turns out to be for a fuzzy set, the higher it grows for another, thus implying a shift from a fully qualifying concept to a different one.

Different strategies may be followed to put into practice the realization of a SFP [2]. On the one hand, the prototype-centered connotation directly suggests that a SFP can be conveniently obtained after a clustering process performed on data. The identified clusters are often projected over the data dimensions (so that they become a reference for designing the fuzzy sets constituting a single partition) and sometimes addi-

tional information is employed to better delineate the clusters morphology. This can be achieved by considering cuts over each dimension, i.e. points delimiting the boundaries amongst clusters, useful to identify the most representative regions for the fuzzy sets involved in a SFP<sup>1</sup>. On the other hand, SFPs are characterized by the shape of their fuzzy sets. Different options may be considered, ranging from the simplicity of triangular fuzzy sets (that fit the prototype-centered organization of knowledge in a straightforward way), to the versatility of Gaussian sets (which assign a non-zero membership value to every element inside the Universe).

In a previous work of ours we analyzed the problem of designing SFPs based on cuts and we demonstrated that they *cannot* be always defined in terms of triangular fuzzy sets (as long as well-formed partitions are sought after). We also showed the feasibility to resort to trapezoidal fuzzy sets in this matter, and we proposed some methods to design trapezoidal SFPs starting from cuts. Here we go one step further by exploring the possibility to derive SFPs from cuts through the employment of Particle Swarm Optimization (PSO). In this attempt, we are interested in exploiting in a better way the information coming from the analysis of data, so that a proper configuration of fuzzy sets can be identified ensuring both the interpretability of the resulting SFP and its capability to model the data when included in a fuzzy rule-based classifier.

PSO is a computational method inspired by social interaction mechanisms occurring in nature. An optimization process is carried on by managing an ensemble (swarm) of simple entities (particles) which iteratively explore a search space aiming at the minimization of an objective function. The movement of the swarm is properly directed as a whole, combining local information with neighbourhood interactions. The adoption in our research of such a computational method is motivated by its simplicity and fast rate of convergence. Actually, our numerical experiments show that the employment of PSO proves to be effective in enhancing the quality of the fuzzy sets configurations in terms of accuracy, while preserving the overall interpretability of SFPs which are automatically derived in the desired well-formed fashion.

The rest of the paper is organized as follows. Section 2 introduces the issues related to the design of trapezoidal SFPs from cuts and the optimization performed through swarm of particles. In Section 3 we provide some details related to the specific setting of

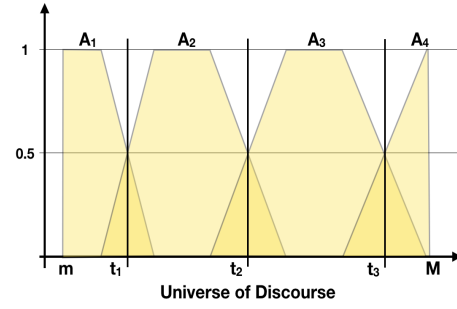


Figure 1: A strong fuzzy partition derived from 3 cuts.

our proposal. Section 4 describes and discusses the experimental results. Final notes and remarks are reported in Section 5.

## 2 Preliminaries

Let  $U = [m, M] \subseteq \mathbb{R}$  be a Universe of Discourse and let  $A_1, A_2, \dots, A_{n+1}$  be a sequence of normal and convex fuzzy sets defined on  $U$ . Such a sequence of fuzzy sets constitutes a Strong Fuzzy Partition provided that:

$$\forall x \in U : \sum_{i=1}^{n+1} \mu_{A_i}(x) = 1$$

being  $\mu_{A_i}$  the membership function related to the fuzzy set  $A_i$ .

It is possible to consider a sorted sequence of points  $t_1, t_2, \dots, t_n$  laying on  $U$ . We are going to refer to these points as *cuts*: they can be properly exploited to organize the structure of a SFP. In fact, with reference to the aforementioned sequence of fuzzy sets, each couple of cuts  $t_i, t_{i+1}$  may represent the lower and upper bounds of the interval representing the 0.5-cut of the fuzzy set  $A_{i+1}$  (for  $i = 1, \dots, n-1$ ). This kind of configuration is exemplified in Figure 1 for  $n = 3$ , where the partition has been completed on the extreme sides by considering  $m$  and  $M$  to produce the fuzzy sets  $A_1$  and  $A_4$ .

Moving from those assumptions, we are going to contextualize in the following the key-elements of our proposal, namely the trapezoidal SFPs (together with some previously introduced mechanisms to derive them from cuts) and the PSO search method (which is employed here as an original proposal to automatically derive SFPs from cuts).

### 2.1 Trapezoidal SFPs from cuts

The membership function of a trapezoidal fuzzy set is a piecewise linear function constrained by four parameters  $a, b, c, d \in \mathbb{R}$ . A trapezoidal fuzzy set  $T[a, b, c, d]$

<sup>1</sup>Such representative regions can be conveniently associated to the 0.5-cut of the fuzzy set, i.e. the crisp set of points inside the Universe whose membership degree is at least equal to 0.5. Notice that  $\alpha$ -cuts should not be confused with cuts as delimiting points in the Universe.

can be defined for each  $x \in U$  as follows:

$$T[a, b, c, d](x) = \begin{cases} \frac{x-a}{b-a}, & x \in ]a, b[ \\ 1 & x \in [b, c] \\ \frac{x-d}{c-d}, & x \in ]c, d[ \\ 0 & x \leq a \vee x \geq d \end{cases}$$

It should be observed that: i) a trapezoidal fuzzy set is well-formed provided that  $a \leq b \leq c \leq d$ ; ii) it could be the case that a trapezoidal fuzzy set collapses to a triangular fuzzy set whenever  $b = c$ .

In [9] we analyzed the problem to design SFPs from cuts and we were able to determine that confining the choice of fuzzy sets to triangular shapes is a limiting strategy. In fact, there exist configurations of cuts which cannot be handle by algorithms producing triangular SFPs only. On the contrary, resorting to the more general case of trapezoidal shapes appears to be a more flexible and less biased approach. To produce SFPs composed by trapezoidal fuzzy sets  $T[a_i, b_i, c_i, d_i]$  ( $i = 1, \dots, n+1$ ) the following conditions must hold:

$$\begin{cases} a_1 = b_1 = m, \\ a_{i+1} = c_i, & (i = 1, \dots, n) \\ b_{i+1} = d_i, & (i = 1, \dots, n) \\ c_{n+1} = d_{n+1} = M \end{cases} \quad (1)$$

Additionally, the following condition must hold to organize the SFP given a set of cuts  $t_1, t_2, \dots, t_n$ :

$$b_{i+1} = 2t_i - a_{i+1} \quad (i = 1, \dots, n) \quad (2)$$

which is of course equivalent to  $d_i = 2t_i - c_i$ . This additional condition is consistent with the idea that the cuts stand as symmetry axes for each couple of descending and ascending sides of two consecutive trapezoidal fuzzy sets.

Three mechanisms have been proposed to design trapezoidal SFPs from cuts [9]. The Constant Slope (CS) approach imposes to all the involved fuzzy sets a common slope (properly derived from the cuts configuration) for their slanting sides. The Variable Fuzziness (VF) approach attunes the slopes in accordance with the distance of each couple of subsequent cuts. The Core Points (CP) approach follows VF but needs some additional information (possibly a-priori known) to refine the slopes. All of them represent heuristic methods allowing to deterministically identify three well-formed trapezoidal SFPs from cuts out of a number of possibilities which are virtually infinite. For such a reason, we judged it could be worthwhile to further explore this space of possibilities by resorting to the particular search method introduced in the following section.

## 2.2 Particle Swarm Optimization

Particle Swarm Optimization (PSO) has been introduced around two decades ago as a technique for optimization of continuous nonlinear functions [7]. Since then, PSO gained increasing attention among researchers who applied this model in a plethora of fields and developed variations and novel versions of the original technique, so that nowadays we can refer to a particle swarm “paradigm” [10].

The idea underlying PSO is to develop a computational mechanism for optimization by borrowing from nature some kind of behavioural algorithms relying more on social interaction than individual capabilities. Bird flocks and fish schools are able to coordinate their movements (which involve several participants at the same time) to reach a specific goal: collecting food, avoiding obstacles, escaping predators, etc. Analogously, in PSO a set of particles is in motion to find a solution inside a search space (related to a given problem) driven by an objective function that must be minimized. Each particle adjusts its movement according to its own actual and best positions as well as considering those pertaining to the swarm (or part of it). By means of an iterative process, PSO goes through the analysis of several positions (i.e. problem solutions) in the search space converging towards some local/global optima. PSO shares similarities with other classes of population-based algorithms, like Genetic Algorithms, where the metaphor of movement and mutual influencing of trajectories is replaced by evolution and mutual exchanging of (genetic) information [5]. However, PSO does not require the implementation of specific operators or dedicated individual representations.

Formally, let  $f : \mathbb{R}^R \rightarrow \mathbb{R}$  be an objective function to be minimized, defined on a  $R$ -dimensional search space

$$E = \bigotimes_{r=1}^R ([l_r, u_r]) \subseteq \mathbb{R}^R. \quad (3)$$

Let the swarm be composed by  $S$  particles. Each  $i$ th particle is characterized at time  $\tau$  by a number of  $R$ -dimensional vectors: a position vector  $x_i(\tau)$  (coordinates of the current particle position in  $E$ ), a velocity vector  $v_i(\tau)$  (involved in the particle movement), a memory vector  $p_i(\tau)$  (storing the best particle position registered so far), and a vector  $p_g(\tau)$  related to the surrounding particles (storing the best position registered so far in the particle neighbourhood). At each time step, the vectors  $x_i(\tau)$ ,  $v_i(\tau)$  are updated for every particle according to the following formulas:

$$\begin{cases} v_i(\tau + 1) = \omega v_i(\tau) + C_1 \phi_1(p_i(\tau) - x_i(\tau)) \\ \quad \quad \quad + C_2 \phi_2(p_g(\tau) - x_i(\tau)) \\ x_i(\tau + 1) = x_i(\tau) + v_i(\tau + 1) \end{cases}$$

where  $\omega$  is the so-called “inertia weight” (a positive value generally lesser than one involved as a friction coefficient),  $C_1, C_2$  are non-negative coefficients, and  $\phi_1, \phi_2$  are randomly generated values in  $[0, 1]$ . (In this work, we set  $\omega = C_1 = C_2 = 0.5$  as it is common in literature.) The process is iterated evaluating at each step the objective function on every particle and registering the best of its value in view of converging to a solution for the underlying problem at hand.

### 3 Applying PSO to design trapezoidal SFPs

Since we are willing to exploit PSO to determine the structure of a trapezoidal SFP based on cuts, the involved particles must be properly conceived at first. From (1) we learn that a sequence

$$\langle c_i, d_i, a_{i+1}, b_{i+1} \rangle, \quad i = 1, \dots, n,$$

should be provided, which reduces to

$$\langle c_i, d_i \rangle, \quad i = 1, \dots, n,$$

due to the redundancy in the couples  $(c_i, a_{i+1})$ ,  $(d_i, b_{i+1})$ . Indeed, from (2) we observe that a single parameter is needed, therefore we assume that

$$c_i, \quad i = 1, \dots, n,$$

is the only free variable whose optimal value we expect to be sought by PSO.

As concerning the objective function, we should keep in mind that the partitioning process is generally related to the clustering of a (possibly) multi-dimensional set of data. In this sense, the partitioned Universe  $U$  may be the domain of one of the involved data feature. As an example, if a classification problem has to be carried out by properly partitioning a multi-dimensional space, then the objective function must be related to the classification accuracy demonstrated by the fuzzy classifier resulting from the clustering process.

The domain of the objective function is the search space where the particles are supposed to move. As reported in (3), such a domain is composed by a number of bounded intervals: in our context we can identify them as the ranges  $[l_i, u_i]$  inside which the particles (corresponding to the parameters  $c_i$ ) are supposed to move. Given the sequence of cuts  $t_1, t_2, \dots, t_n$  on  $U$ , completed by the extreme points  $m$  and  $M$ , it is straightforward to observe that

$$\begin{cases} c_1 \in [l_1, u_1] \subseteq [m, t_1[, \\ c_i \in [l_i, u_i] \subseteq [t_{i-1}, t_i[, \quad (i = 2, \dots, n) \end{cases} \quad (4)$$

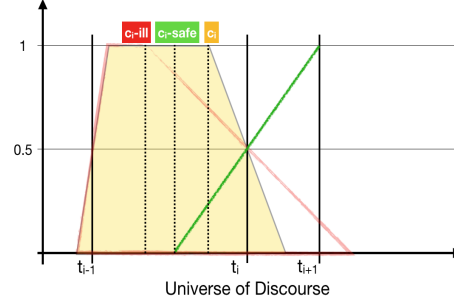


Figure 2: Different possibilities to place a point  $c_i$  on the Universe of Discourse: points on the left of  $c_{i\text{-safe}}$  generate an erroneous crossing between the  $i$ -th fuzzy set and the subsequent cut  $t_{i+1}$ .

However, some notes of caution must be addressed concerning the actual span of those ranges.

The placement of a point  $c_i$  inside a range reverberates through the adjacent intervals and may be disruptive on the well-formedness of the trapezoidal fuzzy sets. Firstly, the inequality  $b_i \leq c_i$  must be obviously verified for each  $i = 1, \dots, n$ . Additionally, there is the need to consider the mutual cut positions, to avoid some erroneous crossing with fuzzy sets. As an example, consider the scenario depicted in Figure 2. While choosing a placement to properly design the  $i$ -th fuzzy set,  $c_i$  appears clearly to be a suitable point. However, the choice of  $c_{i\text{-ill}}$  would produce a trapezoidal set (represented by a blurred red line in figure) crossing the subsequent cut  $t_{i+1}$ . That would prevent to obtain a well-formed  $(i + 1)$ -th trapezoidal set, from resulting  $d_i = b_{i+1} > t_{i+1}$  which would lead to  $b_{i+1} \geq c_{i+1}$  (since it can be easily shown that  $\forall i : t_i \geq c_i$ ). In such a case, therefore, the actual range where  $c_i$  takes place must be reduced by identifying a lower bound. It corresponds to the point labelled in figure as  $c_{i\text{-safe}}$ : it can be obtained by symmetry considering the reciprocal positions of cuts  $t_i, t_{i+1}$  (green line in figure).

To take into account the correct definition of the ranges involved in (4), a first approach consists in determining the lower and upper bounds moving from the leftmost fuzzy set. In practice, the first range is fixed at start as follows:

$$\begin{cases} [l_1, u_1[ = [m, t_1[ & \text{if } t_2 - t_1 \geq t_1 - m, \\ [l_1, u_1[ = [2t_1 - t_2, t_1[ & \text{otherwise} \end{cases}$$

and then the other ranges are fixed accordingly, checking the possibility to set  $l_i = b_i$ :

$$\begin{cases} [l_i, u_i[ = [b_i, t_i[ & \text{if } t_{i+1} - t_i \geq t_i - t_{i-1}, \\ [l_i, u_i[ = [2t_i - t_{i+1}, t_i[ & \text{otherwise} \end{cases}$$

Such a strategy, which will be referred as Leftmost Slope Constraint (LSC) in the following, ensures the

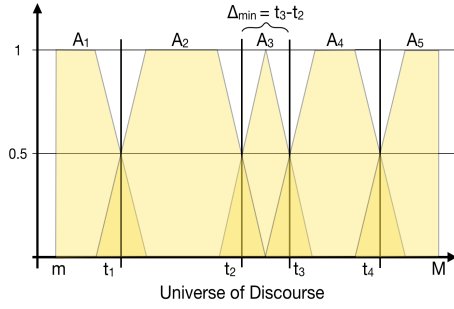


Figure 3: An example of trapezoidal SFP designed by the CS method: the slopes for the fuzzy sets are drawn from the triangle centred on the minimum cuts distance.

realization of well-formed trapezoidal SFPs, but gives rise to a side effect. Whenever  $b_i$  cannot be set as the lower bound of some specific range, the subsequent interval collapses to  $t_{i+1}$ .<sup>2</sup> This implies a diminution of the exploratory power of the PSO, which is obliged to reduce the  $i + 1$ -th fuzzy set to a singleton.

An alternative approach for the definition of the ranges involved in (4) is based on one of the three heuristic methods mentioned in Section 2.1, namely the Constant Slope method which is briefly detailed in the following. The CS heuristics aims at identifying the most specific fuzzy set to be included into the SFP: it corresponds to the triangular fuzzy set centred on the minimum interval  $\Delta_{\min}$  defined by the cuts  $t_1, \dots, t_n$  on  $U$ . Assuming to include  $m$  and  $M$  into the cuts sequence as  $t_0$  and  $t_{n+1}$ , respectively:

$$\Delta_{\min} = \min\{(t_{i+1} - t_i) | i = 0, 1, \dots, n\}$$

The slopes of the triangular fuzzy set, i.e. the values  $\frac{1}{\Delta_{\min}}, -\frac{1}{\Delta_{\min}}$ , are adopted as the slopes of all the remaining trapezoidal fuzzy sets to be drawn to complete the SFP. An example of SFP designed through the CS method is represented in Figure 3.

This kind of construction may be the foundation for defining the bounds for the PSO search: such an approach will be referred as Constant Slope Constraint (CSC) in the following. In practice, the ranges to search the points  $c_i$  may be limited by the fixed slope identified through the CS method, thus yielding:

$$c_i \in [l_i, u_i] = \left[ t_i - \frac{\Delta_{\min}}{2}, t_i \right], \quad i = 1, \dots, n$$

Such a position ensures that the resulting SFP is composed by well-formed trapezoidal sets. In fact, if we

<sup>2</sup>Strictly speaking, the interval approaches  $[t_{i+1} - \epsilon, t_{i+1}]$  for  $\epsilon > 0$  arbitrarily small. In a computer simulation, this limiting case is approximated to  $t_{i+1}$ .

re-write  $b_i$  and  $c_{i-\text{safe}}$  respectively as  $(2t_{i-1} - c_{i-1})$  and  $(2t_i - t_{i+1})$ , then it can be easily shown that

$$t_i - \frac{\Delta_{\min}}{2} \geq \max[(2t_i - t_{i+1}), (2t_{i-1} - c_{i-1})]$$

By contradiction, from

$$t_i - \frac{\Delta_{\min}}{2} < 2t_i - t_{i+1}$$

would result

$$\frac{\Delta_{\min}}{2} > -2t_i + t_{i+1} + t_i = t_{i+1} - t_i = \Delta_i$$

which is absurd by definition of  $\Delta_{\min}$ . Also by contradiction, from

$$t_i - \frac{\Delta_{\min}}{2} < 2t_{i-1} - c_{i-1}$$

would result

$$t_i - \frac{\Delta_{\min}}{2} < 2t_{i-1} - (t_{i-1} - \frac{\Delta_{\min}}{2})$$

that is

$$\Delta_{\min} > t_i - t_{i-1} = \Delta_i$$

which is again absurd by definition of  $\Delta_{\min}$ .

The LSC and CSC methods are therefore two alternative approaches enabling the application of PSO to design trapezoidal SFPs from cuts. Their feasibility can be assessed by experimenting on data, as we show in the following section.

## 4 Numerical results

The cuts which control the definition of a SFP over a Universe  $U$  typically derive from some clustering process applied to a set of data. In this sense,  $U$  stands as a single dimension of the multi-dimensional space where the data are included. To test the effectiveness of the previously introduced LSC and CSC methods, we considered a number of datasets and we analyzed them through the employment of DC\* [8]. DC\* is a specific algorithm designed to perform a double clustering process devoted to extract interpretable fuzzy granules of information from data and to express them in form of fuzzy rules. DC\* is an acronym for Double Clustering with A\*: a first clustering of data is performed using a prototype-guided algorithm; then the derived prototypes are projected on each dimension and those projections are further clustered by exploiting the capabilities of the A\* search algorithm. More precisely, cuts are identified on every single dimension as the mid-points between each couple of projections related to prototypes of different classes. The goal of the second clustering process, which operates on all



Dataset	# Input	# Classes	# Samples
SD1	2	3	200
SD2	2	3	400
SD3	2	3	400
SD4	2	2	5300
SD5	2	3	300
SD6	2	3	600
SD7	2	2	500
SD8	2	2	300
SD9	2	3	600

Table 1: Description of the datasets involved in the experimental session.

dimensions simultaneously, is to provide a configuration of cuts (i.e. a subset of the whole cuts ensemble) splitting the space into the minimum number of multi-dimensional intervals including same-class prototypes only. Once identified such a special configuration, the cuts on each dimension provide the patterns to design the SFPs underlying the final fuzzy rule base which embeds the knowledge extracted from data. Further details on DC\* can be found in [4] where the VF method has been applied to derive trapezoidal partitions from cuts.

In the following, we are going to use DC\* simply as a tool to obtain a configuration of cuts from the analysis of data: we are not interested in directly evaluating its capabilities, but rather in assessing the predictive behaviour of the resulting system while different approaches to generate SFPs are applied. In the same way, we included in our experiments a number of bi-dimensional synthetic datasets, for the sake of plain representation and better comprehension of the numerical results. Such datasets refer to classification problems and are depicted in Figures 4-5 where the cut configurations produced by DC\* are reported together with the data points. Table 1 sums up the main characteristics of the datasets. As can be observed, the datasets differ in the number of classes and datapoints. Also, the application of DC\* produced cuts that in some cases are in agreement with the data distribution, while in some other cases they appear to be less appropriate for discriminating among classes.

To evaluate the performance of the fuzzy rule bases corresponding to the derived SFPs we employed a couple of methods. For each datapoint fed to the rule base, the MAX method assigns the class emitted by the single rule characterized by the highest activation strength. As an alternative, the SUM method assigns the class emitted by the rules adding up the highest cumulative activation strength. In Table 2 and Table 3 we show the performance results related to the MAX and SUM methods, respectively. For each of them, the

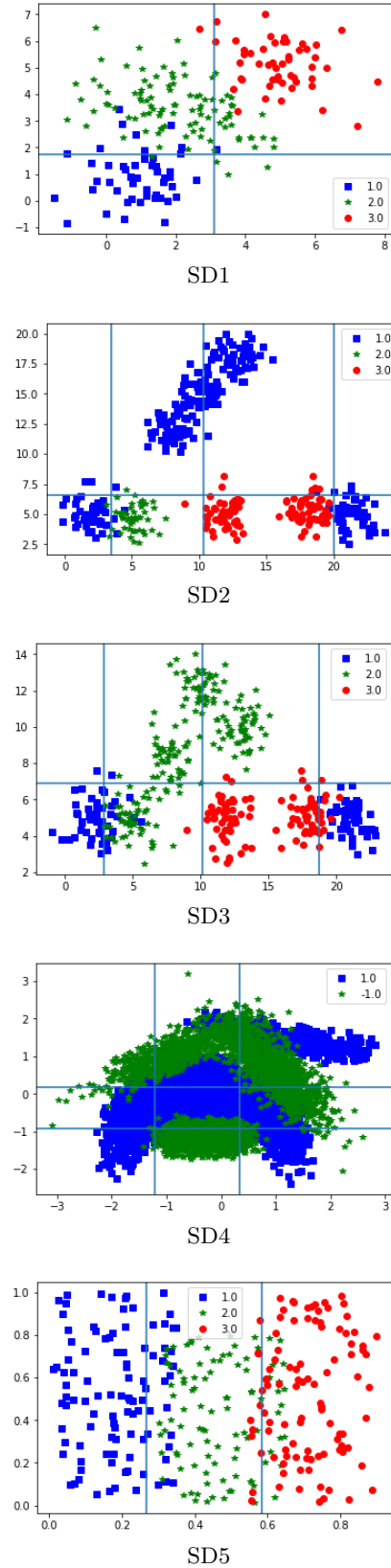


Figure 4: The datasets SD1-SD5 adopted for the numerical simulation.

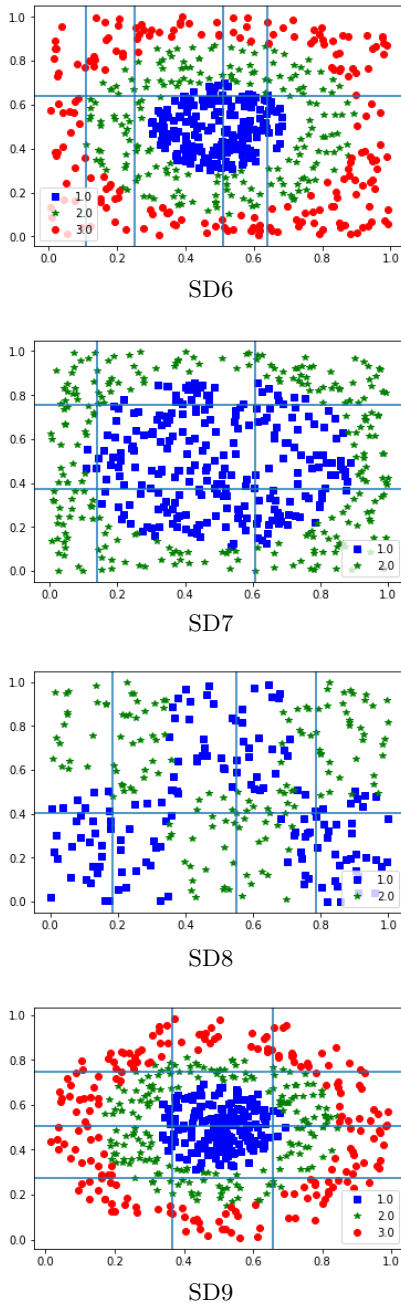


Figure 5: The datasets SD6-SD9 adopted for the numerical simulation.

numerical outcomes are reported by telling apart the accuracy values obtained through the heuristic methods (CS, VF, and CP methods) and those obtained through the application of PSO (LSC and CSC methods). In some cases (SD1, SD5) the reported results are identical in the tables: that is due to the presence of a single rule for each class in the corresponding fuzzy classifiers. This implies the equivalence between the MAX and SUM methods in such a specific case. Values reported in bold highlight the situations where

either LSC or CSC (or even both of them) are able to outperform the heuristic methods.

It can be observed that PSO is able to match the heuristic methods results in every experimental session and in several cases it can even improve their performance while preserving the prefixed interpretability constraints. Moreover, we note that the application of the SUM method generally determines a higher degree of variability in the classification results when compared to the MAX inference method. However, in both cases the search for a suitable SFP attuning performed by PSO proves to be profitable.

In Figure 6 we provide a comparison of the SFPs obtained by applying the different methods put in action during the experimental session. The SD7 dataset has been chosen as an example and only one dimension has been considered.

In summary, we can state that PSO provides a suitable way to derive interpretable fuzzy partitions, all the more so when we consider also computational efforts: in every experiment (excepting the one related to SD4, which includes a huge number of datapoints) the PSO search has been conducted on the basis of 100 iterations running out within a range of  $\sim 1-5$  minutes.

## 5 Conclusions

Constructing a fuzzy partition over a data domain is not a trivial task: several approaches can be followed yielding different outcomes both in terms of accuracy and interpretability of the fuzzy rule system which can be later established on the top of the partitioning results. It may be the case that additional information is injected to optimize the process, such as cuts which are useful to attune the structure of the partition. In this paper we studied the employment of a PSO search to define SFPs based on cuts. The SFPs are meant

Dataset	Heuristic methods			PSO methods	
	CS	VF	CP	LSC	CSC
SD1	82.00	82.00	82.00	82.00	82.00
SD2	96.25	96.25	96.25	96.25	96.25
SD3	92.50	92.50	92.50	92.50	<b>92.75</b>
SD4	67.30	67.30	66.32	<b>67.32</b>	67.30
SD5	83.67	83.67	83.67	83.67	83.67
SD6	58.67	58.67	58.67	58.67	58.67
SD7	68.00	68.60	62.40	<b>68.60</b>	68.40
SD8	66.67	66.67	66.67	66.67	66.67
SD9	67.83	67.83	65.83	<b>68.83</b>	<b>68.17</b>

Table 2: Accuracy (%) of fuzzy classifiers embedding the SFPs obtained through the application of different strategies, using the MAX inference method.

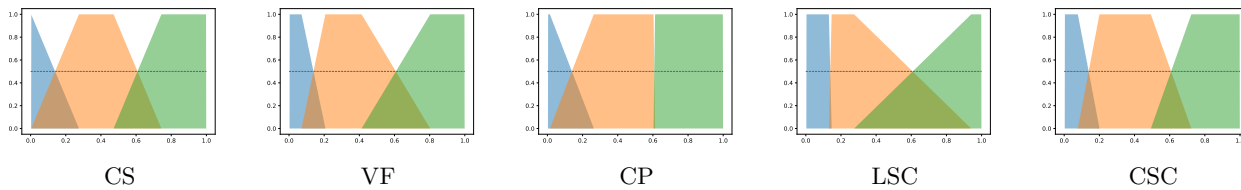


Figure 6: Comparison of computed SFPs for dataset SD7 (first dimension).

Dataset	Heuristic methods			PSO methods	
	CS	VF	CP	LSC	CSC
SD1	82.00	82.00	82.00	82.00	82.00
SD2	86.75	94.50	95.25	<b>96.25</b>	<b>96.25</b>
SD3	81.75	89.75	92.50	<b>95.00</b>	<b>95.00</b>
SD4	65.34	65.34	66.32	<b>67.34</b>	<b>67.98</b>
SD5	83.67	83.67	83.67	83.67	83.67
SD6	58.17	59.33	58.83	<b>61.00</b>	<b>60.83</b>
SD7	69.00	70.40	61.80	<b>73.80</b>	70.40
SD8	61.33	65.33	66.00	<b>67.00</b>	<b>67.00</b>
SD9	71.50	71.33	66.00	69.33	<b>71.83</b>

Table 3: Accuracy (%) of fuzzy classifiers embedding the SFPs obtained through the application of different strategies, using the SUM inference method.

to be used for building up fuzzy classifiers in form of fuzzy rule bases, which should ensure high accuracy performance as well as interpretability of the embedded knowledge. Experimental sessions conducted through the aid of the DC\* algorithm provide encouraging results regarding the suitability of the proposed approach. Some prospects for future work can be outlined too. Particularly, there is room to devise different strategies to further explore the search space investigated by the PSO (as alternative options for LSC and CSC methods). Also, a deeper understanding of the intrinsic differences between the MAX and SUM inference methods—especially in terms of interpretability of inference—would be beneficial for better assessing the experimental results.

### Acknowledgement

The research is partially supported by Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) under grant PON ARS01\_00141 "CLOSE". The authors are members of the INdAM Research group GNCS.

### References

- [1] J. M. Alonso, L. Magdalena, HILK++: an interpretability-guided fuzzy modeling methodology for learning readable and comprehensible fuzzy rule-based classifiers, *Soft Computing* 15 (2011) 1959–1980.
- [2] J. Casillas, O. Cordón, F. Herrera, L. Magdalena, *Accuracy improvements in linguistic fuzzy modeling*, Vol. 129, Springer, 2013.
- [3] J. Casillas, O. Cordón, F. Herrera, L. Magdalena, *Interpretability issues in fuzzy modeling*, Vol. 128, Springer, 2013.
- [4] C. Castiello, A. M. Fanelli, M. Lucarelli, C. Mencar, *Interpretable fuzzy partitioning of classified data with variable granularity*, *Applied Soft Computing* 74 (2019) 567–582.
- [5] R. C. Eberhart, Y. Shi, *Comparison between genetic algorithms and particle swarm optimization*, in: V. Porto, N. Saravanan, D. Waagen, A. Eiben (Eds.), *Evolutionary Programming VII*, EP 1998, Vol. 1447, Springer, Berlin, Heidelberg, 1998, pp. 611–616.
- [6] S. Guillaume, B. Charnomordic, *Generating an interpretable family of fuzzy partitions from data*, *IEEE Transactions on Fuzzy Systems* 12 (2004) 324–335.
- [7] J. Kennedy, R. Eberhart, *Particle swarm optimization*, in: *Proc. International Conference on Neural Networks (ICNN'95)*, Vol. 4, Perth, WA, Australia, 1995, pp. 1942–1948.
- [8] M. Lucarelli, C. Castiello, A. M. Fanelli, C. Mencar, *Interpretable knowledge discovery from data with DC\**, in: *Proc. 7th Conf. of the Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (IFSA-EUSFLAT 2015)*, Gijón, Asturias (Spain), 2015, pp. 815–822.
- [9] C. Mencar, M. Lucarelli, C. Castiello, A. M. Fanelli, *Design of strong fuzzy partitions from cuts*, in: *Proc. 8th Conf. of the European Society for Fuzzy Logic and Technology (EUSFLAT 2013)*, Milan, Italy, 2013, pp. 424–431.
- [10] R. Poli, J. Kennedy, T. Blackwell, *Particle swarm optimization. an overview*, *Swarm Intelligence* 1(1) (2007) 33–57.