Studying Possibility in a Clustering Algorithm for RBFNN Design for Function Approximation

A. Guillén H. Pomares I. Rojas J. González
L.J. Herrera F. Rojas O. Valenzuela

April 18, 2007

Abstract

The function approximation problem has been tackled many times in the literature by using Radial Basis Function Neural Networks (RBFNNs). In the design of these neural networks there are several stages where, the most critical stage is the initialization of the centers of each RBF since the rest of the steps to design the RBFNN strongly depend on it. The Improved Clustering for Function Approximation (ICFA) algorithm was recently introduced and proved successful for the function approximation problem. In the ICFA algorithm a fuzzy partition of the input data is performed but, a fuzzy partition can behave inadequately in noise conditions. Possibilistic and mixed approaches, combining fuzzy and possibilistic partitions, were developed in order to improve the performance of a fuzzy partition. In this paper, a study of the influence of replacing the fuzzy partition used in the ICFA algorithm with the possibilistic and the fuzzy-possibilistic partitions will be done. A comparative analysis of each kind of partition will be performed in order to see if the possibilistic approach can improve the performance of the ICFA algorithm both in normal and in noise conditions. The results will show how the employment of a mixed approach combining fuzzy and possibilistic approach can lead to improve the results when designing RBFNNs.

1 Introduction

The function approximation problem can be formulated as, given a set of observations $\{(\vec{x}_k; y_k), k = 1, ..., n\}$ with $y_k = F(\vec{x}_k) \in \mathbb{R}$ and $\vec{x}_k \in \mathbb{R}^d$, it is desired to obtain a function $\mathcal{F}$ so $y_k = \mathcal{F}(\vec{x}_k) \in \mathbb{R}$ with $\vec{x}_k \in \mathbb{R}^d$. To solve this problem, Radial Basis Function Neural Networks (RBFNN) have been used because of their capability as universal approximators [15, 14].

One of the methodologies to design an RBFNN is performed by following a sequence of stages: 1) initialization of the centers, 2) initialization of the radii, and 3) computation of the weights. The use of a clustering algorithm is a common solution for a first initialization of the centers [9, 21]. These clustering algorithms were designed for classification problems [8] instead of for

the function approximation problem so the results they provide can be improved significantly. The main elements that cause that clustering algorithms do not perform well for the functional approximation problem are:

- Clustering algorithms try to classify the set of input data assigning a set of predefined labels, however, in the function approximation problem, the output of the function belongs to a continuous interval.
- Clustering algorithms do not use the information provided by the function output ignoring the variability of the function.

In the function approximation problem, the information provided by the output of the function to be approximated is needed to obtain a correct placement of the centers. Centers must be placed in the areas where the function is more variable because more RBFs will be needed to be able to model the variations of the function, meanwhile, in the areas where the function is not that variable, less RBFs will be needed to approximate the function.

It is necessary to design a clustering algorithm oriented to the function approximation problem. The Clustering for Function Approximation [6] (CFA) and its Improved version (ICFA) [7] algorithms were designed specifically for this task. These two algorithms defined a new approach to consider the output of the function as a relevant information to place the centers. The ICFA algorithm was developed to improve several flaws existing in the CFA algorithm. Within the modifications of the original algorithm, a fuzzy partition of the data was introduced. There are several kinds of possibilistic partitions that can improve the performance of the fuzzy partition as it has been shown in [20, 10, 12]. In this paper, the fuzzy partition of the ICFA algorithm will be replaced by the several possibilistic approaches found in the literature and a comparative analysis will be made for each of them.

2 RBFNN Description

An RBFNN $\mathcal{F}$ with fixed structure to approximate an unknown function F with n inputs and one output starting from a set of values $\{(\vec{x}_k; y_k); k = 1, ..., n\}$ with $y_k = F(\vec{x}_k) \in \mathbb{R}$ and $\vec{x}_k \in \mathbb{R}^d$, has a set of parameters that have to be optimized: the centers, the radii and the weights of the RBFs. An RBFNN is defined as:

$$\mathcal{F}\left(\vec{x}_k; C, R, \Omega\right) = \sum_{i=1}^{m} \phi(\vec{x}_k; \vec{c}_i, r_i) \cdot \Omega_i \tag{1}$$

where $C = \{\vec{c}_1, ..., \vec{c}_m\}$ is the set of RBF centers, $R = \{r_1, ..., r_m\}$ is the set of values for each RBF radius, $\Omega = \{\Omega_1, ..., \Omega_m\}$ is the set of weights and $\phi(\vec{x}_k; \vec{c}_i, r_i)$ represents an RBF. The activation function most commonly used for classification and regression problems is the Gaussian function because it is continuous, differentiable, it provides a softer output and improves the interpolation capabilities [17]. The procedure to design an RBFNN for functional approximation problem is shown below:

1. Initialize RBF centers C
2. Initialize the radius R for each RBF
3. Calculate the optimum value for the weights Ω
4. Apply local search algorithm to adjust centers and radii

The initialization of the centers is very important because, if an incorrect initialization of the centers is performed, the approximation error could be increased. The reason for this is because, during the execution of a local search algorithm to make a fine tunning of the centers and the radii, there is a possibility of falling into a bad local minimum. The CFA algorithm was designed for this initialization task, providing very good results in comparison with other clustering techniques and later, the ICFA algorithm was presented as an improved version, overcoming the results of its predecessor.

3 Possibilistic approaches to clustering

In this section, all the main clustering algorithms that use a possibilistic approach will be presented following a chronological order, starting from the oldest algorithm. All these algorithms define a distortion function to be minimized and the minimum is reached iteratively by an alternating optimization mechanism. Let $U^p = [u^p_{ik}]$ be the matrix containing all the possibilistic memberships, $U^f = [u^f_{ik}]$ the matrix containing the fuzzy memberships, and $C = [\vec{c_i}]$ the matrix containing the center positions for $i = 1...m$ and $k = 1...n$. The elements that are modified on each iteration are: the possibilistic membership, the centers, and, in case the algorithm has a mixed approach, the fuzzy membership will also be updated. Each algorithm proposes their respective equations, but they all follow the same scheme:

Do
  Calculate the new U^p_i
  If it is a mixed approach, calculate U^f_i
  Calculate the new C_i
  i=i+1
While($||C_{i-1}$-$C_i|| < threshold$)

In all of the algorithms presented below there is an optional part that reiterates the main part of the algorithm in order to improve the shape of the membership functions. The initialization of the centers does not require the membership functions to keep their interpretability so this optional part is not executed in any of the algorithms.

3.1 Possiblistic C-means (PCM)

This algorithm was proposed in [10], and tries to solve the problem of the presence of outliers that can affect the fuzzy partition deforming the shape of the clusters.

In the Fuzzy C-means [2] (FCM) algorithm, the distortion function to be minimized is defined as:

$$J_{h_f}(U^f, C; X) = \sum_{k=1}^{n} \sum_{i=1}^{m} (u_{ik}^f)^{h_f} D_{ik}^2 \tag{2}$$

where D_{ik} is the inner product induced norm, h_f is a weighing exponent (usually equal to 2), and u_{ik}^f is restricted to the following constraints:

- $\sum u_{ik}^f = 1 \ \forall k = 1...n$
- $0 < \sum u_{ik}^f < n \ \forall i = 1...m.$

The constraint $\sum_{i=1}^{m} u_{ik}^f = 1$ can present a problem when there is an outlier that does not belong to any cluster. For example, if there is a noisy point in the middle of two centers that determine two clusters, the membership value could be 0.5 and 0.5 respectively, when it is obvious that the point should not belong to any of the clusters [10]. The idea of possibilistic clustering comes from relaxing the previous constraint that forces all centers to share the input vectors. In the previous example, each cluster could assign a possibility value of membership of 0.25 or even less to the noisy point, meaning that the outlier input does not belong to any cluster. If this constraint is relaxed, the problem of finding the trivial solution of assigning zero to all the membership values arises. To avoid this situation, it is necessary to add a new addend to the distortion function defined by the FCM obtaining the following one:

$$J_{h_p}(U^p, C; X) = \sum_{k=1}^{n} \sum_{i=1}^{m} (u_{ik}^p)^{h_p} D_{ik}^2 + \sum_{i=1}^{m} \eta_i \sum_{k=1}^{n} (1 - (u_{ik}^p)^{h_p}) \tag{3}$$

where η_i are suitable positive numbers. In the distortion function, the first term demands that the distance from the feature vectors to the centers be as small as possible. The second term forces the membership value u_{ik}^p be as big as possible. The value of η_i determines the distance at which the membership value of a point in a cluster becomes 0.5 so it has to be chosen depending on how big the clusters are desired. The authors propose the following equation to calculate its value:

$$\eta_i = K \frac{\sum_{k=1}^{n} (u_{ik}^p)^{h_p} D_{ik}^2}{\sum_{k=1}^{n} (u_{ik}^p)^{h_p}} \tag{4}$$

where K is a positive number that usually is chosen to be 1. This approach is robust because noisy points will not affect significantly the possibilistic partition as they would do in a fuzzy partition. This approach presents a problem because the distortion function can be minimized placing all the centers in the same position [1]. Another big issue is the election of the values for η_i and K that can deteriorate the performance of the algorithm.

3.2 Fuzzy Possibilistic C-means (FPCM)

In this approach developed in [12], a combination of a fuzzy partition and a possibilistic partition is presented. The authors assert that the membership value of the fuzzy partition is important to be able to assign a hard label to classify an input vector, but at the same time, it is very useful to use the typicality (possibility) value to move the centers properly in presence of outliers. The distortion function to be minimized is:

$$J_{h_f,h_p}(U^f, C, U^p; X) = \sum_{k=1}^{n}\sum_{i=1}^{m}((u_{ik}^f)^{h_f} + (u_{ik}^p)^{h_p})D_{ik}^2 \tag{5}$$

with the following constraints:

$$\sum_{i=1}^{m} u_{ik}^f = 1 \ \forall k = 1...n \tag{6}$$

$$\sum_{k=1}^{n} u_{ik}^p = 1 \ \forall i = 1...m \tag{7}$$

Let $U^p = [u_{ik}^p]$, then, the constraint shown above requires each row of U^p to sum up to 1 but its columns are free up to the requirement that each column contains at least one non-zero entry. Thus, there is a possibility of input vectors not belonging to any cluster.

3.3 Modified Possiblistic C-means (MPCM)

The authors in [19] implement a modification of the PCM that solves the problem that PCM presents when it minimizes the distortion function placing the centers in the same position. To avoid the problem, a mutual repulsion of the clusters is proposed to be able to separate the cluster centers. The clustering repulsion is implemented by adding a cluster repulsion term to the PCM distortion function:

$$J_{h_p}(U^p, C; X) = \sum_{k=1}^{n}\sum_{i=1}^{m}(u_{ik}^p)^{h_p}D_{ik}^2 + \sum_{i=1}^{m}\eta_i\sum_{k=1}^{n}(1-u_{ik})^h + \sum_{i=1}^{m}\gamma_i\sum_{j=1,j\neq i}^{m}\frac{1}{D_{ij}^2} \tag{8}$$

where γ_i is a weighing factor defined as $\gamma_i = \gamma \sum_{k=1}^{n} u_{ik}$. The parameter γ is a positive number that increases or decreases how much the centers will be repulsed.

The MPCM algorithm presents an interesting approach to avoid the problem in the PCM algorithm by making a repulsion between the centers. Although this approach works well for clustering problems, it might not fit for the function approximation problem. In this problem, the repulsion factor given by γ_i can deteriorate the quality of the results because it is based on the execution of the FCM algorithm which can be very similar for equally distributed input vectors, making the repulsion factor almost equal for all the centers. Hence, some centers will not be able to come closer and the target function might require may centers being very close to model the areas where the output is more variable. On top of this, the election of a proper value for K when calculating η_i is still a crucial problem for the performance of the algorithm. For these reasons, this possibilistic approach will not be considered to modify the ICFA algorithm, not appearing in the following sections.

3.4 Improved Possibilistic C-means (IPCM)

Another improved version of the PCM algorithm that combines fuzzy and possibilistic memberships is proposed in [20]. To be able to keep the fuzzy approach and use a possibilistic partition, the authors replace the similarity criteria in the FCM distortion function by the distortion function defined in the PCM algorithm, obtaining the following distortion function to be minimized:

$$J_{h_f,h_p}(U^p, U^f, C; X) = \sum_{k=1}^{n}\sum_{i=1}^{m} (u_{ik}^f)^{h_f}(u_{ik}^p)^{h_p} D_{ik}^2 + \sum_{i=1}^{m} \eta_i \sum_{k=1}^{n} (u_{ik}^f)^{h_f}(1-u_{ik}^p)^{h_p} \tag{9}$$

where u_{ik}^p is the possibilistic membership of $\vec{x}_k$ in the cluster i, u_{ik}^f is the fuzzy membership of $\vec{x}_k$ to the cluster i, h_p and h_f are the weighting exponents for the possibilistic and the fuzzy membership functions, and η_i is a scale parameter that is calculated using:

$$\eta_i = \frac{\sum_{k=1}^{n} (u_{ik}^f)^{h_f}(u_{ik}^p)^{h_p} D_{ik}^2}{(u_{ik}^f)^{h_f}(u_{ik}^p)^{h_p}}. \tag{10}$$

As in the previous methods, an iterative process drives the algorithm to reach a minimum of the distortion function defined in (9).

3.5 Possibilistic Fuzzy C-means

This algorithm [13] is proposed in order to solve some drawbacks presented in the FPCM algorithm regarding the possibilistic membership values. The objective function that has to be minimized is obtained in the same way that

the PCM was presented but, instead of using the FCM as start algorithm, the FPCM is used:

$$J_{h_f,h_p}(U^p, U^f, C; X) = \sum_{k=1}^{n}\sum_{i=1}^{m}((a \cdot u_{ik}^f)^{h_f} + b \cdot (u_{ik}^p)^{h_p})D_{ik}^2 + \sum_{i=1}^{m}\eta_i \sum_{k=1}^{n}(u_{ik}^f)^{h_f}(1 - u_{ik}^p)^{h_p} \tag{11}$$

where $a > 0$, $b > 0$, $h_f > 1$, $h_p > 1$ and η_i are user defined constants. This objective function, for certain values of the user defined parameters, includes the ones of the FCM, PCM and FPCM algorithms. Therefore, the sections below do not reference this algorithm explicitly although implicitly is included in the comparisons. A further study can be done analyzing which values can be more appropriate for the function approximation problem and how the possibilistic membership is influenced by the expected output, which is the element that adds the supervising feature to the clustering algorithms.

4 Clustering for function approximation problems

In this section, the ICFA algorithm will be described. This algorithm was presented as a modification of the CFA algorithm, improving its performance and increasing the algorithm speed.

4.1 Improved Clustering for Function Approximation (ICFA)

Even though the CFA algorithm improves the performance in comparison with other clustering algorithms, it has some flaws that can be improved. Within those flaws, it can be mentioned the way the partition of the input data is done, the complex migration process, the algorithm speed, the existence of some parameters that have to be set in order to obtain good solutions, and the fact that the convergence to a minimum of the distortion function is not guaranteed. An improved version of the CFA algorithm was presented in [7], this algorithm adds several modifications that make the ICFA algorithm improve significantly the results provided by the CFA algorithm.

4.1.1 Objective Function and Iterative Process

To be able to place the centers closer to the areas where the target function is more variable, a change in the similarity criteria used in the clustering process is needed. Each center is assigned an estimated output value that is used to calculate the difference between the output of a center and the output of the input vectors it owns:

$$w_{ki} = |F(\vec{x}_k) - o_i| \tag{12}$$

where $F(\vec{x}_k)$ is the function output for the input $\vec{x}_k$ and o_i is the estimated output of $\vec{c}_i$.

The parameter w is introduced to modify the values of the distance between a center and an input vector. The smaller w is, the more the distance D_W between the center and the vector will be reduced. The distance is now calculated by:

$$D_{ikW} = \|\vec{x}_k - \vec{c}_i\| \cdot w_{ki}. \tag{13}$$

Proceeding this way, D_{ikW} will be small if the center is near the input vector and they have similar output values. Thus a center can own input vectors that are far from it if they have similar output values, and will not own input vectors that, even though are near the center, have a big difference in the output values. This will allow the algorithm to place more centers where the output of the target function to be approximated is more variable. The distortion function to be minimized is redefined as:

$$J_{h_f}(U, C, W) = \sum_{k=1}^{n} \sum_{i=1}^{m} (u_{ik}^f)^{h_f} D_{ikW}^2 \tag{14}$$

where $h_f > 1$ is a parameter to control the degree of sharing of an input vector in the fuzzy partition and usually is equal to 2. This function is minimized applying the LS method, obtaining the following equations that will converge to the solution:

$$u_{ik}^f = \left(\sum_{j=1}^{m} \left(\frac{D_{ikW}}{D_{jkW}} \right)^{\frac{2}{h_f - 1}} \right)^{-1} \tag{15}$$

$$\vec{c}_i = \frac{\sum_{k=1}^{n} (u_{ik}^f)^{h_f} \vec{x}_k w_{ki}^2}{\sum_{k=1}^{n} (u_{ik}^f)^{h_f} w_{ki}^2} \tag{16}$$

$$o_i = \frac{\sum_{k=1}^{n} (u_{ik}^f)^{h_f} y_k d_{ik}^2}{\sum_{k=1}^{n} (u_{ik}^f)^{h_f} d_{ik}^2} \tag{17}$$

where d_{ik} is the euclidean distance between $\vec{c}_i$ and $\vec{x}_k$.

4.1.2 Migration Step

The ICFA algorithm performs a migration step in order to avoid some local minima. The migration only considers the centers that have a distortion value above the average. The distortion of a center is the contribution to the error of the function to be minimized. The center to be migrated will be the one that has been assigned the smallest value of distortion. The destination of the migration

will be the center that has been assigned the biggest value of distortion. If the error is smaller than the one before the migration step, the migration is accepted, otherwise is rejected.

5 Using possibility in clustering for Function Approximation

This section will describe the algorithms derived from using a possiblistic approach in the ICFA algorithm. The FCM algorithm, since it assigns membership degrees, is less sensitive to noisy points than the hard approach because noisy data will have smaller membership values than the non noisy ones. However, a small value can still be significantly high to damage the performance. The possibilistic approach was designed to solve this problem by allowing all the centers to have a small membership values for the same input vector. When the ICFA algorithm was presented, the substitution of a hard partition by a fuzzy one was shown to be successful, so it is reasonable to study the behavior of the algorithm using a possibilistic partition instead of a fuzzy one.

The algorithms that will be presented in the next subsections are the result of replacing the fuzzy partition used in the ICFA algorithm by a possibilistic one.

5.1 Possibilistic clustering algorithms for function approximation general scheme

All the previous algorithms described above follow almost the same scheme as the ICFA algorithm. There are some differences such as the starting point of the algorithm and the initialization of some specific parameters for each algorithm. In general, the main scheme that all the algorithms follow is shown in Figure 1.

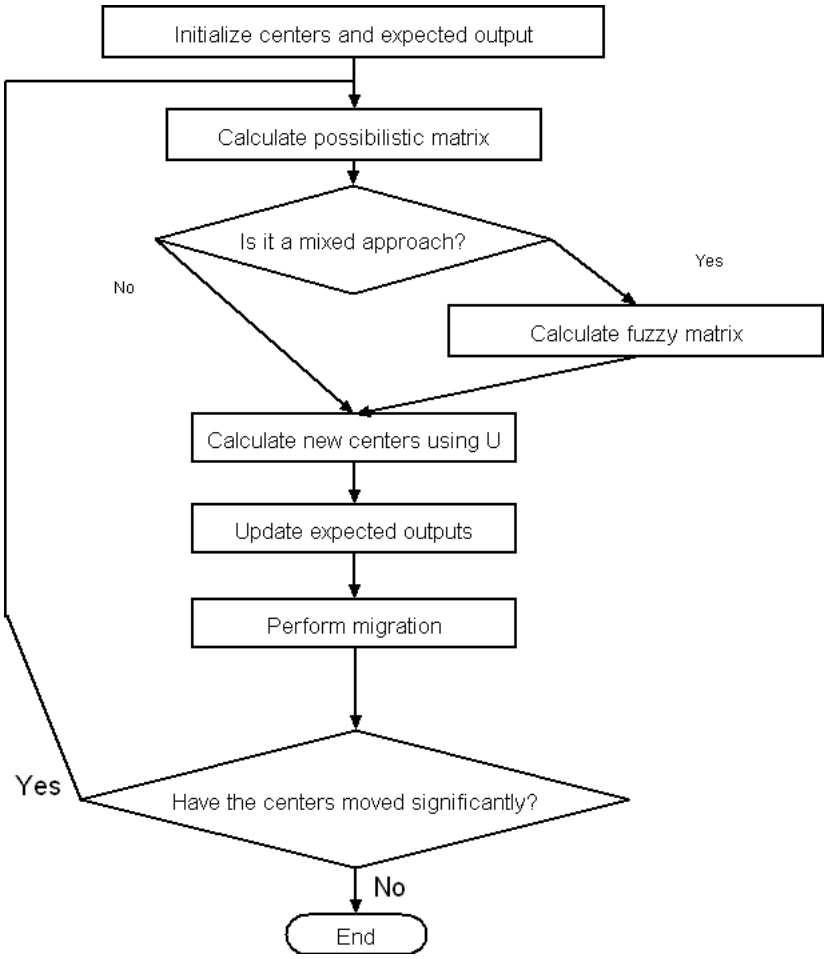

Figure 1: General scheme for the possibilistic approaches

Before starting the calculation of the position of the centers, a first value has to be assigned. In the algorithms PCFA and IPCFA, the execution of the FCM algorithm is mandatory since the parameter η_i has to be calculated using the membership functions generated. However, in the ICFA algorithm, in order to obtain robustness, a fixed starting point is assigned to the centers. Since the FPCFA does not require the execution of the FCM algorithm, the centers will be assigned the same positions used in the ICFA, this is, all the centers are distributed uniformly through the input data space. For the initialization of the expected outputs a value of 1 is assigned in all the algorithms, thus, all the centers will be influenced by the output in the same conditions.

From the execution time point of view, the algorithms that only have one kind of membership (FCM and PCM) are faster than the ones that have two kinds (FPCM and IPCM). This is because they only have to compute one ma-

trix with membership values instead of two membership matrixes (fuzzy and possibilistic). However, when the block where the membership matrixes are computed is compared using the $Big - O$ notation, the four of them belong to $O(m \times n)$ since $O(m \times n) + O(m \times n) = O(m \times n)$ where m is the number of centers and n the number of input vectors.

The migration introduced by the ICFA algorithm has been adapted to the possibilistic approaches. The distortion values depend on the kind of partitioning the algorithm uses because each algorithm has its own distortion function. The distortion of a center will be the distortion that a center adds to the distortion function to be minimized. The selection procedure is performed as it is done in the ICFA algorithm and, once the two centers are selected, the repartition of the input vectors between them is done following the algorithm shown below:

- Calculate vectors belonging to the center $\vec{c}_i$ with maximum distortion:
  $Vectors = \{\vec{x}_{ik} \mid u_{ik}^f > 0.5, \quad \forall k = 1...n \ , \ \forall i = 1...m \ \}$
- Place destination center $\vec{c}_i$ and source centers $\vec{c}_j$:
  Let min be a function returning the minimum values of all the coordinates of the vectors contained in a set
  Let max be a function returning the maximum values of all the coordinates of the vectors contained in a set
  $\vec{left} = min(Vectors)$
  $\vec{right} = max(Vectors)$
  $\vec{mid} = \frac{\vec{right} - \vec{left}}{2}$
  $\vec{c}_i = \vec{mid} + \frac{\vec{mid}}{2}$
  $\vec{c}_i = \vec{mid} - \frac{\vec{mid}}{2}$
- Apply Fuzzy C-means to the set $Vectors$ using $\vec{c}_i$ and $\vec{c}_j$ as initial positions
- Update membership (possibilistic or fuzzy) matrix with the new configuration and calculate global distortion

If the distortion is smaller than the one before the migration step, the migration is kept, else, the migration is not considered. The global algorithm to perform the migration step is shown in Figure 2.

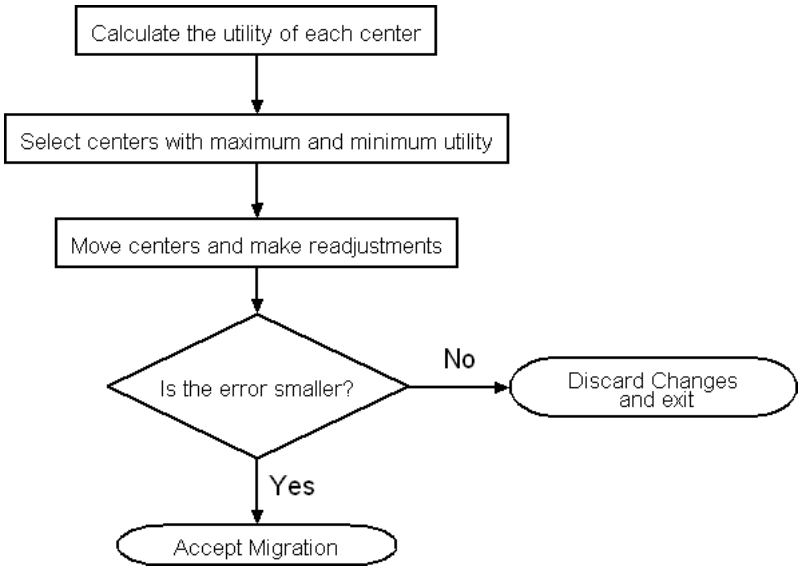

Figure 2: Migration algorithm

In the followings subsections, the adaptation of the ICFA algorithm to a possibilistic approach will be studied. These algorithms will keep the structure just presented in this section.

5.2 Possibilistic CFA (PCFA)

This algorithm introduces the pure possibilistic partition in the ICFA algorithm. To do that, the distortion function has to be redefined as shown below:

$$J_{h_p}(U^p, C, W; X) = \sum_{k=1}^{n}\sum_{i=1}^{m}(u_{ik}^p)^{h_p} D_{ikW}^2 + \sum_{i=1}^{m}\eta_i \sum_{k=1}^{n}(1-u_{ik}^p)^{h_p} \tag{18}$$

The main difference with respect to the distortion function defined by the PCM algorithm is the addition of the weighing parameter that allows the algorithm to use the information provided by the target function.

Since the distortion function has changed, adding a new variable o_i, the iterative method that drives to the minimum has to be recalculated. The procedure to obtain the equations is the same than the one used to obtain the equations for the previous algorithms. The new equations are:

$$u_{ik}^p = \left(\left(1 + \frac{D_{ikW}^2}{\eta_i}\right)^{\frac{1}{h_p - 1}}\right)^{-1} \tag{19}$$

$$\vec{c}_i = \frac{\sum_{k=1}^{n} (u_{ik}^p)^{h_p} \vec{x}_k w_{ki}^2}{\sum_{k=1}^{n} (u_{ik}^p)^{h_p} w_{ki}^2} \tag{20}$$

$$o_i = \frac{\sum_{k=1}^{n} (u_{ik}^p)^{h_p} y_k d_{ik}^2}{\sum_{k=1}^{n} (u_{ik}^p)^{h_p} d_{ik}^2} \tag{21}$$

5.3 Fuzzy-Possibilistic CFA (FPCFA)

In the FPCM, a mixed approach was presented, adding a typicality value to the membership function. The adaptation of the ICFA to this kind of partition is very straightforward, obtaining the following distortion function:

$$J_{h_f,h_p}(U^f, C, U^p, W; X) = \sum_{k=1}^{n} \sum_{i=1}^{m} ((u_{ik}^f)^{h_f} + (u_{ik}^p)^{h_p}) D_{ikW}^2 \tag{22}$$

restricted to the same constraints than the FPCM one.

As in the previous case, the equations for the iterative method have to be recalculated, obtaining the ones below:

$$u_{ik}^f = \left(\sum_{j=1}^{m} \left(\frac{D_{ikW}}{D_{jkW}} \right)^{\frac{2}{h_f - 1}} \right)^{-1} \tag{23}$$

$$u_{ik}^p = \left(\sum_{j=1}^{n} \left(\frac{D_{ikW}}{D_{ijW}} \right)^{\frac{2}{h_p - 1}} \right)^{-1} \tag{24}$$

$$\vec{c}_i = \frac{\sum_{k=1}^{n} ((u_{ik}^f)^{h_f} + (u_{ik}^p)^{h_p}) \vec{x}_k w_{ki}^2}{\sum_{k=1}^{n} ((u_{ik}^f)^{h_f} + (u_{ik}^p)^{h_p}) w_{ki}^2} \tag{25}$$

$$o_i = \frac{\sum_{k=1}^{n} ((u_{ik}^f)^{h_f} + (u_{ik}^p)^{h_p}) y_k d_{ik}^2}{\sum_{k=1}^{n} ((u_{ik}^f)^{h_f} + (u_{ik}^p)^{h_p}) d_{ik}^2} \tag{26}$$

5.4 Improved Possibilistic CFA (IPCFA)

This new algorithm uses a possibilistic partition and a fuzzy partition, combining both approaches as it was done in IPCM [20]. The objective function to be minimized is defined as:

$$J_{h_f,h_p}(U^p, U^f, C, W; X) = \sum_{k=1}^{n}\sum_{i=1}^{m}(u_{ik}^f)^{h_f}(u_{ik}^p)^{h_p}D_{ikW}^2 + \sum_{i=1}^{m}\eta_i\sum_{k=1}^{n}(u_{ik}^f)^{h_f}(1-u_{ik}^p)^{h_p} \tag{27}$$

The computation of η_i at the beginning of the algorithm requires the calculation of the distance between the centers and the input vectors. This distance must not be weighted using w because the estimated outputs, in the initialization of the algorithm, are not appropriate to measure the influence of the output on the position of the centers. Therefore, η_i is calculated only once at the beginning of the algorithm using the euclidean distance as follows:

$$\eta_i = \frac{\sum_{k=1}^{n}(u_{ik}^f)^{h_f}d_{ik}^2}{(u_{ik}^f)^{h_f}} \tag{28}$$

As in all the previous algorithms based on a fuzzy or a possibilistic partition, the solution is reached by an alternating optimization approach where all the elements defined in the function to be minimized (Eq. 27) are updated iteratively. For this new algorithm, the equations are:

$$u_{ik}^p = \frac{1}{1+\left(\frac{D_{ikW}}{\eta_i}\right)^{\frac{1}{h_p-1}}} \tag{29}$$

$$u_{ik}^f = \frac{1}{\sum_{j=1}^{m}\left(\frac{(u_{ik}^p)^{(h_p-1)/2}D_{ikW}}{(u_{jk}^p)^{(h_p-1)/2}D_{jkW}}\right)^{\frac{2}{h_f-1}}} \tag{30}$$

$$c_i = \frac{\sum_{k=1}^{n}(u_{ik}^p)^{h_p}(u_{ik}^f)^{h_f}x_k w_{ki}^2}{\sum_{k=1}^{n}(u_{ik}^p)^{h_p}(u_{ik}^f)^{h_f}w_{ki}^2} \tag{31}$$

$$o_i = \frac{\sum_{k=1}^{n}(u_{ik}^p)^{h_p}(u_{ik}^f)^{h_f}y_k d_{ik}^2}{\sum_{k=1}^{n}(u_{ik}^p)^{h_p}(u_{ik}^f)^{h_f}d_{ik}^2} \tag{32}$$

6 Experimental analysis of the algorithms

This section will analyze the behavior of the previous clustering algorithms. For the sake of clarity, we will use a one dimensional function f_1 (Fig. 3) defined as:

$$f_1 = \frac{\sin(25X)}{e^{(7X)}}, \tag{33}$$

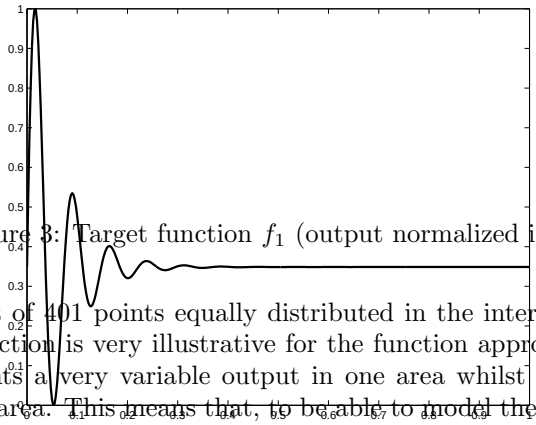

Figure 3: Target function f_1 (output normalized in the interval [0,1])

A set of 401 points equally distributed in the interval $[0, 1]$ was generated. This function is very illustrative for the function approximation problem since it presents a very variable output in one area whilst it is almost constant in another area. This means that, to be able to model the function, it is necessary to place more centers where the function is more variable. If no mechanism is applied to use the information provided by the output of the function, the final position of the centers will be incorrect and the approximation error after the local search process will be big.

The RBFNNs that will approximate the target function f_1, will be generated following the procedure shown below:

- Initialize radii using the K-Nearest Neighbors [11] algorithm with K=1
- Obtain the weights optimally by solving a linear equation system [6]
- Apply a local search algorithm to optimize the centers and the radii, in our case the $Levenberg - Marquardt$ algorithm [6].

Since the number of the centers that are necessary to model the target function is not known, the algorithms will be run using several values for the number of centers. Once the every clustering algorithm has been executed executed and the corresponding RBFNNs generated, the Normalized Root-Mean-Squared-Error (Eq. 34) will be computed both for training and test data in order to determine the quality of the approximation.

$$\mathbf{NRMSE} = \sqrt{\frac{\sum_{k=1}^{n} \left(y_k - \mathcal{F}(\vec{x}_k; C, R, \Omega)\right)^2}{\sum_{k=1}^{n} \left(y_k - \bar{Y}\right)^2}} \tag{34}$$

where $\bar{Y}$ is the average of the outputs of the target function

6.1 PCFA

The performance of the PCM algorithm depends on the election of the parameter η for each center. This parameter determines the distance at which the membership value of an input vector to a center becomes 0.5 and it needs the setting of another parameter K which is a positive real number. The value assigned to η has to agree with the desired width of the cluster so, if a high value is assigned, the cluster width will be big and all the centers will be moved to the same position [1]. For the example, using the function f_1, if K is small, the centers will be placed uniformly trough the input vectors space, since the vectors are equidistributed and the width of the clusters is small. When this value becomes bigger, the centers tend to move to the middle of the input vectors until they all are placed in the center of the input vector space.

The PCFA algorithm suffers from the same problem as the PCM algorithm. If the values for η and K are calculated using the equations that the authors propose (Eq. 4), it tends to find identical clusters, placing all the centers in the same position but, due to the addition of the parameter w, this position is closer to the area where the function is more variable.

Since the value proposed by the authors to initialize K and η is not good enough, the problem of selecting a correct value arises. For the function approximation problem it is not possible to decide the desired width of each cluster previously to the initialization of the centers. Because of this, the computation of η using the membership functions obtained by the FCM algorithm will not be too appropriate. The parameter K can increase or decrease the value of η, therefore, the main issue is to select the right value for this parameter.

The smaller K becomes, the more distributed and separated are the centers will be. The reason for this is because K makes the algorithm converge faster so it keeps the original distribution obtained by the first initialization step with the FCM algorithm. If K is big, all the centers will be placed in the same position in the area where the function is more variable. The balance is reached when the value of K is big enough to allow the centers to move to the area where the function is more variable but small enough to converge before they concentrate in the same position.

In Figure 4, several executions with five centers for several values of K are shown. In the first executions, it is not possible to see all the centers since they are placed in the same position. When the value of K is decreased, the centers start to separate and become more disperse.

Table 1 shows the results of the approximation error once the centers and the radii were initialized and fine tuned using the local search algorithm. The PCFA algorithm does not perform well and it is very difficult to find a correct value for K.

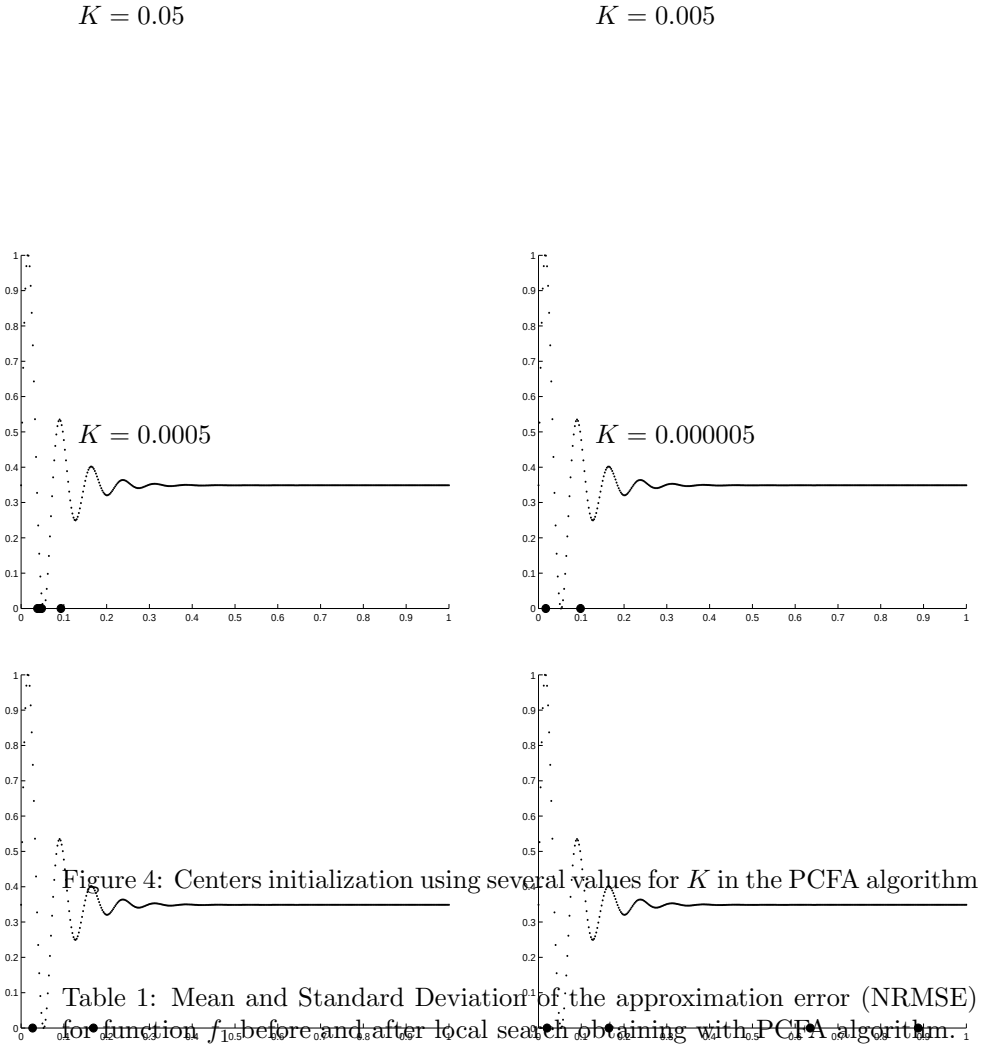

Figure 4: Centers initialization using several values for K in the PCFA algorithm

Table 1: Mean and Standard Deviation of the approximation error (NRMSE) for function f_{10} before and after local search obtaining with PCFA algorithm.

		PCFA
Centers	K	Approximation Error
6	0.000005	0.392(0.007)
	0.00005	0.381(0.112)
7	0.00005	0.473(0.226)
	0.0005	0.459(0.260)
8	0.00005	0.089(0.014)
	0.0005	0.276(0.204)
9	0.00005	0.430(0.004)
	0.0005	0.399(0.296)

6.2 FPCFA

The FPCM algorithm solves successfully the problem presented in the PCM algorithm avoiding the undesired effect of placing the centers in the same position. For f_1, the algorithm places the centers uniformly throughout the input domain.

The adaptation of the fuzzy-possibilistic partition presented in FPCM to the function approximation problem performs very good, placing the centers where the function is more variable and keeping a compromise with the distances between the centers. The are two parameters that can be tuned in order to make a better placement of the centers. These two parameters are h_f and h_p, the exponents for the membership and the typicality matrix.

The parameter h_f controls how fuzzy the partition will be. If this parameter is close to one, the sharing degree of an input vector will be small and the partition will be almost hard. As the value of h_f is increased, the sharing degree of the input vectors will also be increased. The consequence of this is that, for small values of h_f, the centers will be more distributed because each one of them will have its partition well defined, so if a center owns one area where the function is more variable, no other centers will be placed there and they will be distributed in other areas. If h_f is big, the sharing degree will be bigger so all the centers will own to a great extent other input vectors. This will make that the areas where the function is less variable will not belong exclusively to any center in particular and no center will be placed there. The parameter h_p has the same behavior than the parameter h_f although its influence is smaller. The reason for this is because the constraint applied to the possibilistic membership (Eq. 7) affects only to the centers and the constraint applied to the fuzzy membership (Eq. 6) affects the input vectors. Since there are more input vectors than centers, it is logical that h_f has more influence than h_p in the behavior of the algorithm.

To illustrate the previous paragraph, several executions for different values of h_f and h_p were done, the results of these executions are shown in Figure 5. As in the previous subsection, in the figure, sometimes it cannot be seen all the centers because there is an overlap between them. If both h_f and h_p are near one, the centers will keep a compromise between the concentration of centers in the areas where the function is more variable and where it is not that variable. If h_f is near one and h_p becomes bigger, the centers will be concentrated is a major way in the areas where the function is more variable, not placing centers where the function is not variable. If h_f is big and h_p is small, the centers will be more distributed and if h_f is big and h_p is big, the centers will be extremely concentrated where the function is more variable. From these executions we can deduce that is advisable to select an small value for h_f and a similar value for h_p.

Table 2 shows the approximation errors for the function f_1 using several values for h_f and h_p. The FPCFA algorithm provides very good results not only in the approximation error, that is small, but also in the robustness of the different solutions, because it always finds the same configuration. It is important

to remark that independently of the values of h_f and h_p, the algorithm presents a good performance for this example.

$h_f = 1.15 \;\; h_p = 1.15$ $\qquad$ $h_f = 1.15 \;\; h_p = 4$

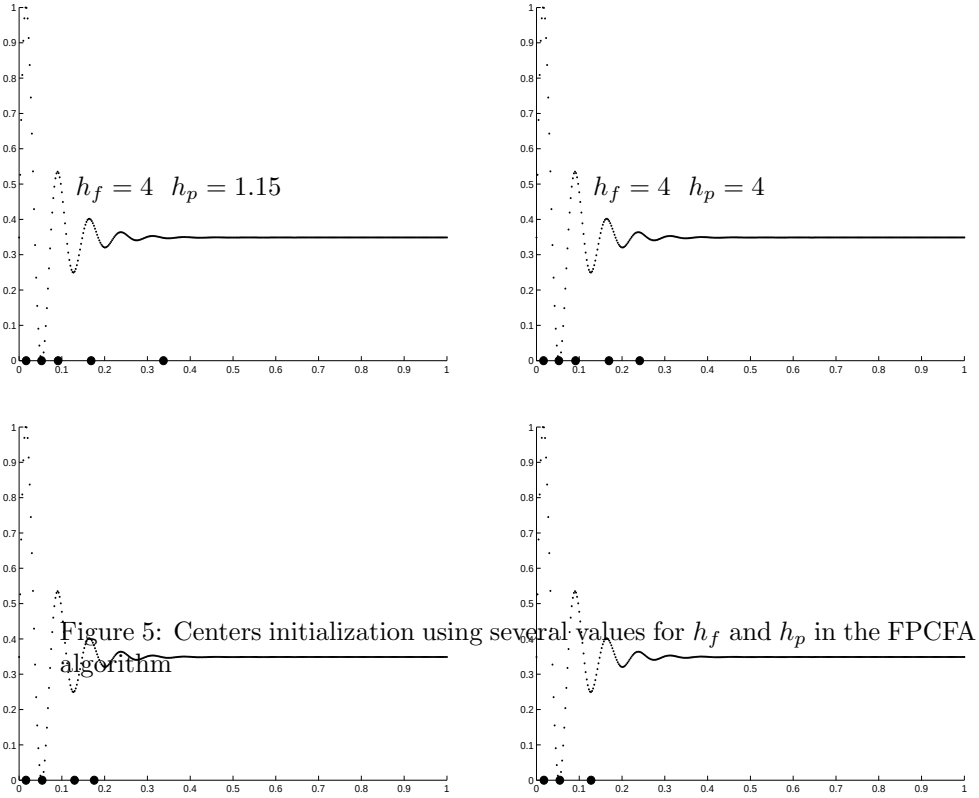

Figure 5: Centers initialization using several values for h_f and h_p in the FPCFA algorithm

Table 2: Mean and Standard Deviation of the approximation error (NRMSE) for function f_1 using the FPCFA algorithm.

FPCFA			
Centers	h_f	h_p	Approximation Error
6	1.5	1.5	0.244(0)
	1.5	2	0.043(0)
	2	1.5	0.014(0)
	2	2	0.025(0)
7	1.5	1.5	0.019(0)
	1.5	2	0.013(0)
	2	1.5	0.092(0)
	2	2	0.020(0)
8	1.5	1.5	0.004(0)
	1.5	2	0.067(0)
	2	1.5	0.005(0)
	2	2	0.066(0)
9	1.5	1.5	0.024(0.000)
	1.5	2	0.015(0.000)
	2	1.5	0.004(0.000)
	2	2	0.021(0.000)

6.3 IPCFA

The IPCM presented a mixed approach to clustering as it was done in the FPCM. As the FPCM algorithm, it distributes uniformly the centers through the input space demonstrating its ability to fix the problems existing in the PCM.

In this algorithm, there are some parameters that have to be set before it execution. As in the FPCM case, two exponents, one for each kind of membership function have to be set and the other parameter is η, which is the same than the one defined for the PCM algorithm.

In this algorithm, the distortion function is more influenced by the fuzzy membership value, because in the distortion function (Eq. 27) the fuzzy membership value is multiplying the rest of the elements in the equation. Then when h_f is small, the variations of h_p are not too significant, because the fuzzy membership values are big and the variations in the possibilistic membership will not influence the distortion function. The behavior of the h_f parameter is the same than the parameter h in the previous algorithm. The smaller the h_f, the more distributed through the input vectors the centers will be. Conversely, the bigger the h_f, the more concentrated in the area where the function is more variable the centers will be. As the value of h_f increases, h_p becomes more important because it will have more influence on the distortion function.In this case, if h_p is small, the centers will be more dispersed because the possibility

values will be bigger so the centers will not share the input vectors to a great extent. If h_p is big, the centers will be more concentrated in the variable area because, as it occurs with the fuzzy membership, the areas where the function is not variable will be shared by all the centers and no center will be placed on those areas. As in the previous cases, several executions are shown in Figure 6 and the approximation errors after the execution of the algorithm are shown in Table 3.

The selection of the parameter η still influences the behavior of the algorithm but in this case, the initialization that was proposed for the PCM algorithm works good enough to avoid the situation of identical center positions.

$h_f = 1.15$ $h_p = 1.15$ $h_f = 1.15$ $h_p = 4$

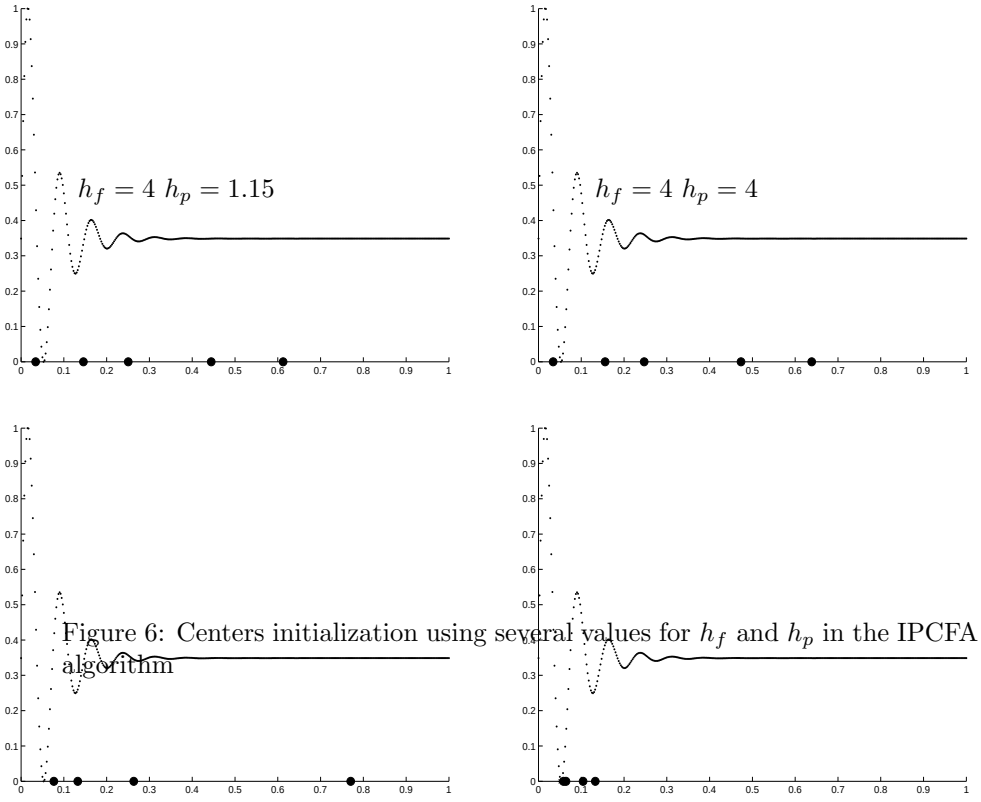

Figure 6: Centers initialization using several values for h_f and h_p in the IPCFA algorithm

Table 3: Mean and Standard Deviation of the approximation error (NRMSE) for function f_1 obtained using the IPCFA algorithm.

IPCFA			
Centers	h_f	h_p	Approximation Error
6	1.5	2	0.038(0.030)
	2	2	0.265(0.289)
	2	1.5	0.082(0.059)
7	1.5	2	0.102(0.064)
	2	2	0.070(0.090)
	2	1.5	0.075(0.043)
8	1.5	2	0.018(0.012)
	2	2	0.045(0.021)
	2	1.5	0.030(0.042)
9	1.5	2	0.095(0.058)
	2	2	0.013(0.015)
	2	1.5	0.073(0.089)

7 Experimental results

In this section the best algorithms analyzed in the previous subsection will be compared with the ICFA algorithm. The bests algorithms were the ones that combined a mixed approach (fuzzy and possiblistic) because their results are not so dependent on the values of the parameters that have to be set. The PCFA algorithm is not considered because its performance is very poor, it lacks of robustness and needs a human expert to set the value of the parameters K and η_i.

The experiments will consist in the approximation of a one dimensional function in absence and in presence of noise; then, a two dimensional function where the center positions are known will be used to see how near to the original centers the algorithms locate the centers. After those experiments, two synthetic functions with the presence of noise and two real world problems will be studied.

7.1 One dimension function

The comparison of the fuzzy approach and the new approaches will be performed with the function f_1 introduced in the previous section, in normal conditions and in the presence of noise.

7.1.1 Absence of noise

The algorithms were executed with the best combination of the parameters described above, the results are shown in Table 4.

The results show how the FPCFA outperforms the other two algorithms in a very significant way. The approximation error obtained is smaller for any number of centers chosen, and as ICFA, the robustness is very high. The IPCFA has a similar performance than the ICFA even improving the results of this one, although there is a big drawback, the IPCFA has a big standard deviation since it is quite dependent of the starting point of the algorithm.

Table 4: Mean and Standard Deviation of the approximation error (NRMSE) for function f_1.

Centers	**ICFA**	**FPCFA**	**IPCFA**
6	0.280(0)	0.014(0)	0.265(0.289)
7	0.073(0)	0.092(0)	0.070(0.090)
8	0.033(0)	0.005(0)	0.045(0.021)
9	0.045(0)	0.004(0)	0.013(0.015)

7.1.2 Presence of noise

The function was distorted with the addition of Gaussian noise with variance 0.1. The generated RBFNN was tested with the original data points in absence of noise, the results are shown in Table 5. The results show how the FPCFA algorithm outperforms again the other two algorithms, indicating that the RBFNN generated is more able to interpolate the original function.

Table 5: Mean and Standard Deviation of the approximation error (NRMSE) for function f_1 using the test data.

Centers	**ICFA**	**FPCFA**	**IPCFA**
6	0.141(0)	0.141(0)	0.125(0.018)
7	0.120(0)	0.079(0)	0.103(0.043)
8	0.089(0)	0.077(0)	0.100(0.054)
9	0.075(0)	0.049(0)	0.078(0.035)

7.2 Two dimensional function

In this experiment a two dimensional function is generated using a gaussian RBFNN (Eq. 35) over a grid of 25x25 points using the parameters in Table 6, that where randomly extracted using a random number generator.

The RBF function used to compute the output is:

$$e^{-\frac{||\vec{x}_k-\vec{c}_i||^2}{r_i^2}} \tag{35}$$

$\vec{c}_i$	r_i	$weights$
7.0026 3.7539	0.8687	0.0377
8.8390 1.9328	4.4933	0.7343
9.4336 20.5478	3.7750	0.3106
15.3469 12.7493	2.6281	0.8203
24.8801 10.8553	4.8836	0.8785

Table 6: Parameters for the function f_2

The approximation errors are shown in Table 7 where the FPCFA algorithm has an optimum performance. Figure 7 a) shows the contour of the target function and the position of the centers, then, in Figure 7 b),c), and d) the positions where the three algorithms have placed the centers are shown. The original centers are represented by hollow circles and the centers initialized by the algorithms are represented by solid diamonds, when the initialized center is in the same position of the original center, the combination of the diamond and the hollow circle becomes a solid circle. The FPCFA algorithm placed the centers in the same positions of the original ones, the ICFA and IPCFA algorithms also placed four out of five centers correctly. However, the IPCFA algorithm obtains a slightly better performance than the ICFA since it is able to identify the area where the function has its maximum variability.

Table 7: Mean and Standard Deviation (in brackets) of the approximation error (NRMSE) for function f_2.

	NRMSE
ICFA	0.0077 (0)
FPCFA	1.7301e-008 (0)
IPCFA	0.0062 (0.0035)

f_2 $ICFA$

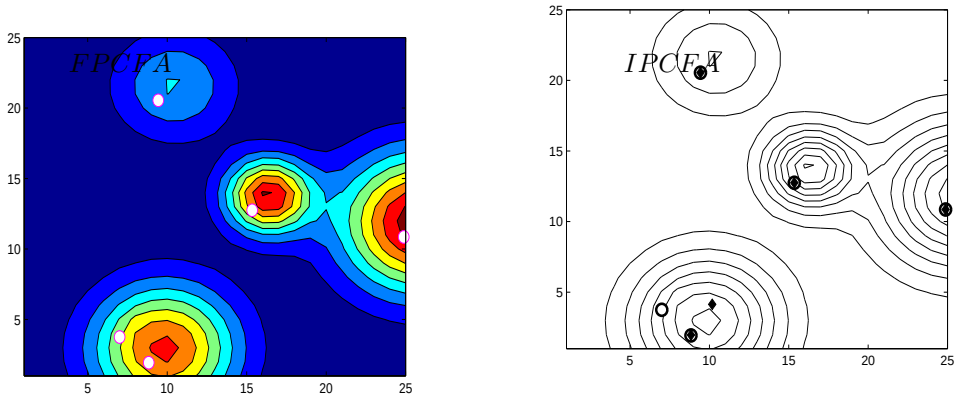

Figure 7: Original centers for function f_2 (circles) and the different initializations performed by the ICFA,FPCFA and IPCFA algorithms (diamonds)

7.3 Further Experimental Results using both artificial and real data

In this subsection, the three algorithms were executed using two two-dimensional functions with the addition of noise and with two real world examples.

7.3.1 Synthetic Function f_3 and f_4

The target functions f_3 and f_4 were presented in [3] and they have been used as a benchmark in [16][5][4]. The function f_3 is defined as:

$$f_3(x_1, x_2) = \frac{1 + sin(2x_1 + 3x_2)}{3.5 + sin(x_1 - x_2)} \quad x_1, x_2 \in [-2, 2] \tag{36}$$

and function f_4 is defined as:

$$f_4(x_1, x_2) = 1.9[1.35 + e^{x_1} sin(13(x_1 - 0.6)^2) e^{-x_2} sin(7x_2)] \; x_1, x_2 \in [0, 1] \quad (37)$$

The training data set was obtained by selecting randomly points in the original function and adding them positive and negative Gaussian noise with variance 0.05.

The algorithms ICFA, FPCFA and IPCFA were executed using several values for the parameters that have to be set before running the algorithms. The results showed how all the algorithms have a very similar performance when the mean error is compared. However, the ICFA algorithm showed a slightly better performance, being more independent of the value of the parameter h to obtain the solutions. Nevertheless, the FPCFA algorithm, for concrete values of their parameters outcome the results provided by the ICFA and the IPCFA algorithms.

7.3.2 Servo-motor Rise Time

The third experiment is taken from the UCI Machine Learning Repository [18] and consists, as it is described in the UCI database, of a system involving a servo amplifier, a motor, a lead screw/nut, and a sliding carriage of some sort. The output value is a rise time, or the time required for the system to respond to a step change in a position set point. This is collection of data covers an extremely non-linear phenomenon: predicting the rise time of a servomechanism in terms of two (continuous) gain settings and two (discrete) choices of mechanical linkages. From the original 167 instances, the first 126 were used for training and the last 41 to test the RBFNN generated. The results (Table 8)showed again how the three algorithms have a similar performance, although the IPCFA algorithm outperforms slightly the results obtained by the other two algorithms. This is due to the overfitting that suffer the ICFA and FPCFA algorithms which are able to adjust more accurately the training data, loosing the generalization capabilities. In real world data sets, it is very difficult to obtain significant or representative data, making very difficult to obtain very accurate models. The results show how the FPCFA is the less dependent of the values of its parameters since it has the smallest standard deviations and this algorithm, for concrete values of its parameters obtains the best performance with both the train and test data.

Table 8: Mean test error (NRMSE) for the servo-motor problem using several values for h_f and h_p

Clusters	**ICFA**	**FPCFA**	**IPCFA**
5	0.378(0.091)	0.419(0.046)	0.373(0.101)
6	0.403(0.127)	0.427(0.068)	0.404(0.101)
7	0.377(0.015)	0.353(0.060)	0.369(0.082)
8	0.377(0.056)	0.296(0.063)	0.284(0.021)
9	0.972(0.876)	0.420(0.102)	0.620(0.390)

8 Conclusions

The initialization of the centers when designing an RBFNN to approximate an input/output mapping, is a very important step that has a big repercussion on the posterior stages of the design. The ICFA algorithm was designed specifically for this task showing how the approximation error can be diminished if an appropriate initialization of the centers is carried out. The ICFA algorithm was inspired in clustering techniques where a fuzzy partition of the data is performed. Within clustering techniques, the possibilistic algorithms were presented as an alternative to the commonly used fuzzy algorithms. The introduction of the possibilistic membership has the objective of reducing the effect of the noise in the input data because a fuzzy partition can behave inadequately in noise conditions.

In this paper, a study of the influence of replacing the fuzzy partition used in the ICFA algorithm with the possiblistic and the fuzzy-possiblistic partitions has been done. The conclusions of this study are:

- The pure possibilistic approach, where only the typicality of an input vector in a cluster is considered, is not too appropriate for the function approximation problem.
- The mixed approaches were a fuzzy partition is combined with a possibilistic partition fits better to the function approximation problem and can improve, in some cases, the results provided by the algorithm that uses a classical fuzzy partition.
- Comparing the two mixed approaches, the FPCFA algorithm is more robust and requires less parameters to bet set, plus it does not need the prior execution of the FCM algorithm to initialize any parameter, not like in the IPCFA algorithm.
- The ICFA still has an advantage over the FPCFA: it only requires one parameter to be set, and it is not so dependant on it when is compared with the other approaches.

- From the computational cost point of view, the ICFA is faster than any other possibilistic approach because it does not need to execute the FCM algorithm at the beginning of the algorithm and it only requires the computation of one membership matrix.
- The experiments showed how a possibilistic partition can outperform the fuzzy one in real world and synthetic problems. These results showed that the FPCFA algorithm can improve the results of the ICFA algorithm.

References

[1] M. Barni, V. Capellini, and A. Mecocci. Comments on 'A possibilistic approach to clustering'. *IEEE Transactions on Fuzzy Systems*, 4:393–396, June 1996.

[2] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum, New York, 1981.

[3] V. Cherkassky and H.Lari-Najafi. Constrained topological mapping for nonparametric regression analysis. *Neural Networks*, 4(1):27–40, 1991.

[4] J. H. Friedman. Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19:1–141, 1991.

[5] J.H. Friedman. Projection pursuit regression. *Journal of the American Statistical Association*, 76:817–823, 1981.

[6] J. González, I. Rojas, H. Pomares, J. Ortega, and A. Prieto. A new Clustering Technique for Function Aproximation. *IEEE Transactions on Neural Networks*, 13(1):132–142, January 2002.

[7] A. Guillén, I. Rojas, J. González, H. Pomares, L.J. Herrera, O. Valenzuela, and A. Prieto. Improving Clustering Technique for Functional Approximation Problem Using Fuzzy Logic: ICFA algorithm. *Lecture Notes in Computer Science*, 3512:272–280, June 2005.

[8] J. A. Hartigan. *Clustering Algorithms*. New York: Wiley, 1975.

[9] N. B. Karayiannis and G. W. Mi. Growing radial basis neural networks: Merging supervised and unsupervised learning with network growth techniques. *IEEE Transactions on Neural Networks*, 8:1492–1506, November 1997.

[10] R. Krishnapuram and J. M. Keller. A Possibilistic Approach to Clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98–110, May 1993.

[11] J. Moody and C.J. Darken. Fast learning in networks of locally-tunned processing units. *Neural Computation*, 1(2):281–294, 1989.

[12] N. R. Pal, K. Pal, and J. C. Bezdek. A Mixed C–Means Clustering Model. In *Proceedings of the 6th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'97)*, volume 1, pages 11–21, Barcelona, July 1997.

[13] N.R. Pal, K. Pal, J.M. Keller, and J.C. Bezdek. A Possibilistic Fuzzy c-Means Clustering Algorithm. *IEEE Transactions on Fuzzy Systems*, 13(4):517–530, August 2005.

[14] J. Park and J. W. Sandberg. Universal approximation using radial basis functions network. *Neural Computation*, 3:246–257, 1991.

[15] T. Poggio and F. Girosi. Networks for approximation and learning. In *Proceedings of the IEEE*, volume 78, pages 1481–1497, 1990.

[16] H. Pomares. Nueva metodología para el diseño automático de sistemas difusos. *PhD. dissertation, University of Granada, Spain*, January 2000.

[17] I. Rojas, M. Anguita, A. Prieto, and O. Valenzuela. Analysis of the operators involved in the definition of the implication functions and in the fuzzy inference proccess. *International Journal of Approximate Reasoning*, 19:367–389, 1998.

[18] C.L. Blake S. Hettich and C.J. Merz. UCI repository of machine learning databases. 1998.

[19] H. Timm and R. Kruse. A Modification to Improve Possibilistic Fuzzy Cluster Analysis. In *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems*, volume 2, pages 1460 –1465, May 2002.

[20] J. Zhang and Y. Leung. Improved possibilistic C–means clustering algorithms. *IEEE Transactions on Fuzzy Systems*, 12:209–217, 2004.

[21] Q. Zhu, Y. Cai, and L. Liu. A global learning algorithm for a RBF network. *Neural Networks*, 12:527–540, 1999.