Appl. Comput. Math., V.10, N.1, Special Issue, 2010, pp.xx-xx

FUZZY CLUSTERING WITH POLYNOMIAL FUZZIFIER FUNCTION IN CONNECTION WITH M-ESTIMATORS

ROLAND WINKLER[1], FRANK KLAWONN[2], RUDOLF KRUSE[3]

Abstract. The invention of fuzzy set theory by Lotfi A. Zadeh had great impact on linguistic computing and machine controlling. Also data mining algorithms were inspired by this idea, routing back to fuzzy clustering. In this paper, we will explore the connections of fuzzy clustering approaches to hard clustering approaches as well as on statistically motivated algorithms. We will use Klawonn and Höppners idea of connecting hard clustering with fuzzy clustering and specialize their algorithms to define an M-estimator. Furthermore we investigate the transition from M-estimators to fuzzy clustering algorithms. To extend M-estimators to a multi-cluster problem, we use a clever normalization function of the robust weights, similar to fuzzy c-means. The resulting algorithm provides an update strategy that can be used with every robust loss function that can be used for M-estimators. However, the success of the clustering algorithm depends on the properties of the loss function.

Keywords: fuzzy c-means, M-estimators, polynomial fuzzifier function, robust statistics; noise clustering, multiple prototypes, robust statistics

AMS subject classification: ??

1. INTRODUCTION

In this paper, connections between M-estimators and fuzzy clustering are explored, hence a short historic review on the development of both classes of algorithms is given before going into detail.

When Lotfi A. Zadeh first introduced his fuzzy set theory [18] in 1965, it was intended to modulate set memberships that are not well defined. The main goal was to modulate linguistic fuzzy terms like 'warm', 'cold' etc., hence the name fuzzy set theory. 8 years later, in 1973, Dunn [7] combined MacQueens [17] hard c-means (HCM) algorithm with Zadehs idea of fuzzy sets to create the first version of fuzzy c-means. This version was than generalized by Bezdek [2] in 1981 by introducing the concept of the fuzzifier. This last version is now commonly referred as fuzzy c-means (FCM). FCM is usually very stable and it has advantages compared to HCM and therefore and became very popular. However, it lacked the ability to handle outliers. Dave [3] published an extension (NC = noise clustering) to fuzzy c-means in 1991 that can handle noise very well by introducing a noise cluster that has constant distance to all data objects.

The concept of the fuzzifier is changed by Klawonn and Höppner [16] in 2003 by replacing the exponential function with a (parametrized) polynomial of degree 2. This polynomial fuzzifier function can be seen as linear combination of FCM and HCM with the linear parameter replacing the role of the fuzzifier (PFCM = FCM with polynomial fuzzifier function). This type of FCM

[1] German Aerospace Center, Institute of Flight Guidance, Departent of Air Transportation, Lilienthalplatz 7 38108 Braunschweig, Germany e-mail: roland.winkler@dlr.de

[2] Ostfalia University of Applied Sciences, Department of Computer Science, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel, Germany e-mail: f.klawonn@ostfalia.de

and Helmholtz Centre for Infection Research, Bioinformatics and Statistics, Inhoffenstr. 7, D-38124 Braunschweig, Germany, e-mail: frank.klawonn@helmholtz-hzi.de

[3] Otto-von-Guericke-University Magdeburg, Fakultät für Informatik, Building 29, Room 008, Universitätsplatz 2, 39106 Magdeburg, Germany, e-mail: kruse@iws.cs.uni-magdeburg.de

Manuscript received 15 December, 2010.

not only assigns values of the open interval $(0, 1)$, but allows membership values of $[0, 1]$, including 0 and 1.

Parallel to the development of FCM and its descendants, the theory regarding M-estimators (ME) evolved. As clustering in general refers to the problem of multiple class partitioning of data objects, statistical estimators are developed assuming only one true cluster. We concentrate here on estimating the mean of a data set as it is related to finding a proper location for the prototypes in case of FCM.

Maximum-likelihood estimation started way before development of the computer. It probably was invented before 1800 but became popular by publications of Fischer after 1912 [9]. Maximum likelihood estimation had also problems with outliers and therefore the more robust version of 'maximum likelihood type' M-estimators are introduced by Huber [13] in 1964. For M-estimators, the assumption on the data distribution is relaxed by replacing the log-likelihood function with a more robust loss function that is not necessarily based on a probability distribution. However, the estimation result of the M-estimator should be close to fit the parameter to the unpolluted (but unknown) true data distribution.

Since the first introduction of M-Estimator, many other versions of the loss function were developed which leads to M-estimators with different properties. For example: Tukey's biweight [1], Hampel's [11], Cauchy's and Huber's estimator [12, 13].

The problem, targeted by M-estimators can be seen as a special case of clustering, namely a two-class clustering problem with only one 'good' cluster and one noise cluster. Because M-estimators target a subclass of problems of general clustering algorithms, it might be possible to find M-estimators that behave just like clustering algorithms for one-cluster problems. Davé and Krishnapuram showed in [5] that it is indeed possible to define an M-estimator based on NC. In this paper, we show that it is also possible to find an M-estimator, based on PFCM.

The big question is however, is it possible to find clustering algorithms, based on M-estimators? This direction is much harder to accomplish since the problem, targeted by clustering algorithms is more general and therefore, specialized algorithms might not be extendible on the broader class of problems. M-estimators are not designed to handle more than one prototype, which is why they can not be naturally extended to perform well on data sets with several clusters. A straight forward attempt has been made by Frigui and Krishnapuram in [10] by defining the robust c-prototypes (RCP) algorithm. In RCP, the (squared) distance is wrapped by a robust loss function, similarly to M-estimators. The problem with this approach is, that the choice of the loss function is very limited to ensure a closed form for solving the new prototype position (similar to Equation (2) on Page 3 for FCM).

In the next section, the basics regarding fuzzy c-means clustering with and without a noise cluster as well as with and without polynomial fuzzifier function are introduced. Section 3 introduces these clustering algorithms as M-estimators and in Section 4, the introduced fuzzy related M-estimators are compared to other, well known M-estimators. In Section 5, the extension from single prototype M-estimators to multiple prototype fuzzy clustering algorithms is discussed. Finally, the conclusions in Section 6 are followed by the references.

2. FUZZY CLUSTERING, LEAST SQUARES AND M-ESTIMATORS

Although fuzzy c-means (FCM) and noise clustering (NC) are very well known, some mathematical details are needed in the next section. Let $\boldsymbol{X} \subset V$ be a finite set of data objects of a vector space V with $|\boldsymbol{X}| = n$. The clusters are represented by a set of prototypes $\boldsymbol{Y} = \{y_1, \ldots, y_c\} \subset V$ which can be initialized randomly, only the number of prototypes c must be known in advance. Let $1 < \omega \in \mathbb{R}$ be the fuzzifier and $\boldsymbol{U} \in \mathbb{R}^{c \times n}$ be the partition matrix with $u_{ij} \in [0, 1]$ and $\forall j : \sum_{i=1}^{c} u_{ij} = 1$. And finally, let $d : V \times V \longrightarrow \mathbb{R}$ be a distance function with its abbreviation $d_{ij} = d(y_i, x_j)$ and f_{FCM} be the fuzzifier function $f_{\mathrm{FCM}}(u) = u^{\omega}$.

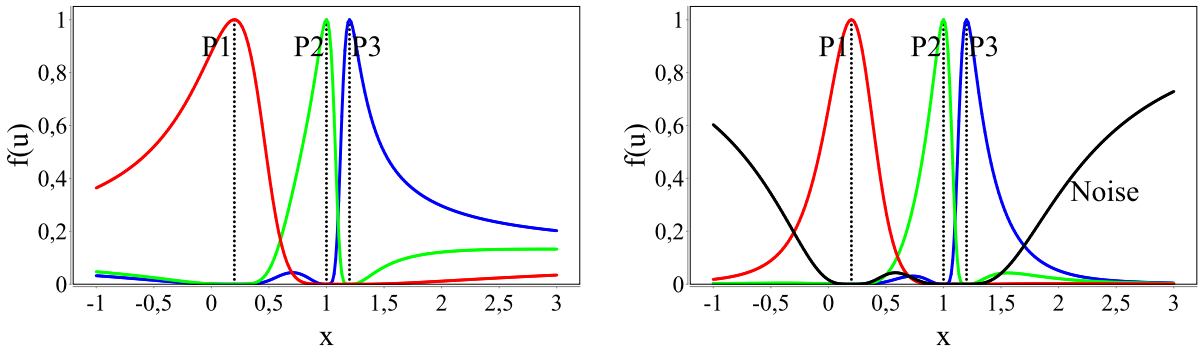

FIGURE 1. Fuzzified membership values of the prototypes of FCM with 3 clusters, without a noise cluster (left) and with a noise cluster (right).

Fuzzy c-means clustering is based on an objective function $\boldsymbol{J}$ that is to be minimized:

$$\boldsymbol{J}_{\mathrm{FCM}}(\boldsymbol{X},\boldsymbol{U},\boldsymbol{Y}) = \sum_{i=1}^{c}\sum_{j=1}^{n} f_{\mathrm{FCM}}(u_{ij})d_{ij}^2 = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{\omega} d_{ij}^2.$$

The minimization of J is done by iteratively updating the members of $\boldsymbol{U}$ and $\boldsymbol{Y}$ and is computed using a Lagrange extension to ensure the constraints $\sum_{i=1}^{c} u_{ij} = 1$. The iteration steps are denoted by a time variable $t \in \mathbb{N}$ denoting $t = 0$ as the initialization step:

$$u_{ij}^{t+1} \overset{\mathrm{FCM}}{=} \frac{(d_{ij}^t)^{\frac{2}{1-\omega}}}{\sum_{k=1}^{c}\left((d_{kj}^t)^{\frac{2}{1-\omega}}\right)} \text{ and} \tag{1}$$

$$y_i^{t+1} \overset{\mathrm{FCM}}{=} \frac{\sum_{j=1}^{n} f_{\mathrm{FCM}}\left(u_{ij}^t\right)\cdot x_j}{\sum_{j=1}^{n} f_{\mathrm{FCM}}\left(u_{ij}^t\right)} = \frac{\sum_{j=1}^{n}\left(u_{ij}^t\right)^{\omega}\cdot x_j}{\sum_{j=1}^{n}\left(u_{ij}^t\right)^{\omega}}. \tag{2}$$

For noise clustering, an additional cluster is specified which is represented by a virtual prototype y_0 which has no location in V. Instead, it has a constant distance $0 < d_{\mathrm{noise}} \in \mathbb{R}$ to all data objects: $\forall j : d_{0j} = d(y_0, x_j) = d_{\mathrm{noise}}$ which is called noise distance. y_0 is not represented as a member of V, which means, it is not updated during the iteration process. However, the membership values are updated as if y_0 was a normal prototype. The noise prototype is introduced to assign higher membership degrees to the noise cluster for all data objects whose distance to regular prototypes exceeds the noise distance. This favours regular prototypes to be better placed in the center of data clusters without being attracted by noise data. Also it suppresses the unwanted effect that membership values converge to $\frac{1}{c}$ for data objects far away from all normal prototypes. Figure 1 shows the influence of the noise cluster to the fuzzified membership values. Three prototypes are placed at 0.2, 1.0 and 1.2 in an one dimensional environment and the y-achsis represents the fuzzified membership values of each prototype.

Klawonn et al. addressed in [15, 16] a different, fuzzy related problem. Namely, that the membership degrees to a prototype are larger than zero for each data object that is not identical to another prototype. In other words, all data objects influence all prototypes. This is not always intuitive or wanted.

Fuzzy clustering became very popular in comparison to hard clustering because it seems to have less problems with local minima [14] and using membership values between 0 and 1 was very helpful to characterize data objects that *can not* be uniquely assigned to any prototype. But in exactly the same way, it prevents a hard classification for data objects that *should* be assigned uniquely to a single prototype. Fuzzy c-means clustering with a polynomial fuzzifier

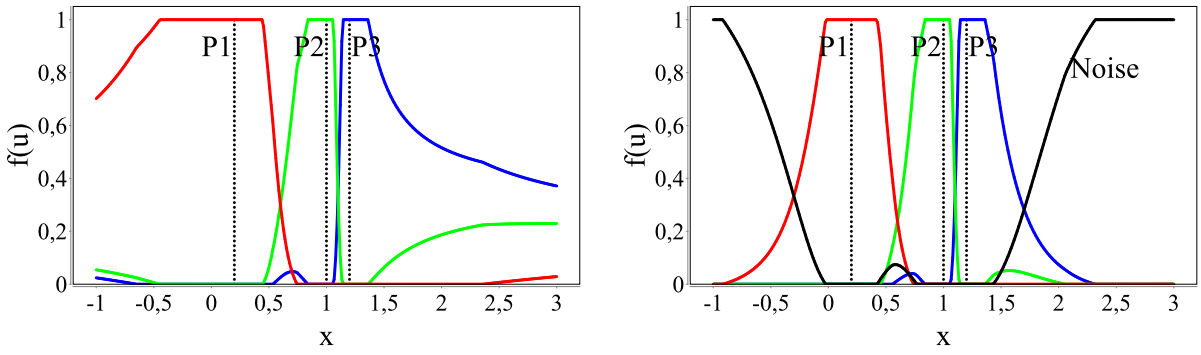

FIGURE 2. Fuzzified membership values of the prototypes of PFCM with 3 clusters, without a noise cluster (left) and with a noise cluster (right).

function (PFCM) (please note the difference to the abbreviation PCM which is usually used for possibilistic c-means clustering) replaces the exponential fuzzifier function $f_{\mathrm{FCM}}(u) = u^{\omega}$ by a polynomial fuzzifier function of the form $f_{\mathrm{PFCM}}(u) = \frac{1-\beta}{1+\beta}u^2 + \frac{2\beta}{1+\beta}u$, $\beta \in [0,1)$. It is in a way a linear combination of hard clustering and FCM with $\omega = 2$. The parameter β describes the ratio of distances at which the clustering result becomes crisp [15, 16].

The new objective function for PFCM differs solely in the fuzzifier function:

$$\boldsymbol{J}_{\mathrm{PFCM}}(\boldsymbol{X},\boldsymbol{U},\boldsymbol{Y}) = \sum_{i=1}^{c}\sum_{j=1}^{n} f_{\mathrm{PFCM}}(u_{ij})d_{ij}^2 = \sum_{i=1}^{c}\sum_{j=1}^{n}\left(\frac{1-\beta}{1+\beta}u_{ij}^2 + \frac{2\beta}{1+\beta}u_{ij}\right)d_{ij}^2.$$

The iterative update process however changes because the constraint of non-negative membership values is not naturally fulfilled as it is in FCM. So if the membership values would be calculated analogously to FCM, negative membership values would occur. To avoid this, the constraint of $u_{ij} \geq 0$ must be ensured by different means. It holds, that $d_{ij} > d_{kj} \Rightarrow u_{ij} \leq u_{kj}$. First, the prototypes are sorted according to their distance to x_j which gives an ordering in membership values as well. Only those prototypes are included into the membership update process, for which the membership degree would be larger than zero. (It is possible that only one prototype would satisfy this condition. In this case, the clustering becomes crisp.) The membership for all other prototypes is set to zero. So the fuzzy membership value is only calculated for a subset of all prototypes and the number of these prototypes is denoted by $\hat{c}_j$. The permutation of the sorted prototypes is denoted by φ.

The update process for the position of the prototypes is similar to FCM, only the fuzzifier function is replaced by f_{PFCM}. The update process for the membership values of PFCM however, has changed to

$$u_{ij}^{t+1} \overset{\mathrm{PFCM}}{=} \begin{cases} \frac{1}{1-\beta}\left(\frac{1+(\hat{c}_j^t-1)\beta}{\sum_{k=1}^{\hat{c}_j^t}\frac{(d_{ij}^t)^2}{(d_{\varphi(k)j}^t)^2}} - \beta\right) & \text{iff} \quad \varphi(i) \leq \hat{c}_j^t \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

Figure 2 shows an example of PFCM with and without the noise clustering. Extending PFCM with a noise cluster works analogously to the transition from FCM to NC.

M-estimators (M-E) are based on the idea of maximum likelihood estimation (MLE). MLE [8] is a method to find the parameters of a statistical model in such a way that it becomes maximal likely to observe the given data set. This proposition already contains the greatest weakness of this method: the data generating process must be known in order to choose the correct model. If the model is chosen (even slightly wrong), no parameter optimization can generate a meaningful result. Often, the correct model is not known or the data set contains outliers that make it

impossible to estimate the correct parameters even if the underlying model might be correct. M-estimators are designed in such a way that they still produce meaningful results, even if the model assumption is not correct or the data contains outliers. MLE is usually performed using the expectation maximization (EM) algorithm [6].

The EM algorithm is based on an objective function as FCM but with a statistical background. As for FCM, let there be c clusters and $\boldsymbol{X}$ be the data set. Consider the random variable $\tilde{X} : \Omega \to V$ that has the data space as domain and let $f_{\tilde{X}} : \boldsymbol{X} \times \boldsymbol{\Theta} \to \mathbb{R}$ be its probability density function that depends on some parameter set $\Theta \in \boldsymbol{\Theta}$. The underlying model assumption is done by choosing the kind of density function $f_{\tilde{X}}$. The likelihood of observing the data set is specified by the likelihood function

$$\boldsymbol{J}_{\mathrm{MLE}}(\boldsymbol{X}, \Theta) = \prod_{j=1}^{n} f_{\tilde{X}}(x_j),$$

which plays the role of the objective function for FCM but in this case, must be maximized. For computational properties, the negative log-likelihood of the data set is optimized instead of the likelihood.

$$-\ln \boldsymbol{J}_{\mathrm{MLE}}(\boldsymbol{X}, \Theta) = -\ln \left(\prod_{j=1}^{n} f_{\tilde{X}}(x_j, \Theta) \right) = \sum_{j=1}^{n} -\ln f_{\tilde{X}}(x_j, \Theta),$$

which in turn must be minimized. The new minimum is located at the same place as the maximum of the likelihood function because the negative logarithm is a strictly decreasing function. Normal statistical functions like normal distributions or even mixtures of normal distributions are very often used. Single outliers can cause significant damage to the parameter estimation, not only for normal distributions. Therefore, in robust statistics the model is replaced by a robust loss function ρ that is not necessarily based on a probability density function $\rho(x_j, \Theta) = -\ln f_{\tilde{X}}(x_j, \Theta)$. Therefore, $\Theta = y$ is interpreted as prototype, $y \in Y$ and the function $g(x, y) = \rho(\|x - y\|)$ is interpreted as a loss function $\rho : \mathbb{R} \to \mathbb{R}$ that takes the distance $\|x - y\|$ as an argument with $\|\cdot\|$ as the Euclidean distance. Finally, the M-estimator (maximum-likelihood-type estimator) is an optimization problem that is based on minimizing the following objective function:

$$\boldsymbol{J}_{\text{M-E}}(\boldsymbol{X}, y) = \sum_{j=1}^{n} \rho(d_j),$$

with $d_j = \|x_j - y\|$. For a local minimum, a necessary condition is, that the derivative is zero. Usually, it is not possible to calculate a global minimum for this objective function directly. Therefore, similar to FCM, an iterative update process is used. Let $\psi(d_j) = \psi(\|x_j - y\|) = \frac{\partial}{\partial y}\rho(\|x_j - y\|)$ be the derivative of ρ and $w_j = w(d_j) = \frac{\psi(d_j)}{d_j}$. The optimization criterion is:

$$0 = \sum_{j=1}^{n} \psi(d_j) \cdot \frac{\partial}{\partial y} d_j = \sum_{j=1}^{n} w_j \cdot d_j \cdot \frac{\partial}{\partial y} d_j = 2 \sum_{j=1}^{n} w_j \cdot (x_j - y). \tag{4}$$

This holds, because d_j has been defined as the Euclidean distance:
$d_j \cdot \frac{\partial}{\partial y} d_j = \|x_j - y\| \cdot \frac{\partial}{\partial y}\|x_j - y\| = \frac{1}{2}\frac{\partial}{\partial y}\|x_j - y\|^2 = (y - x_j)$. From Equation (4) follows by rearranging:

$$y = \frac{\sum_{j=1}^{n} w_j x_j}{\sum_{j=1}^{n} w_j}. \tag{5}$$

Thus, y is the weighted mean of the data set where the weights in turn depend on y in relation to the data objects. This provides an iterative update process for M-estimators. The equation looks very much like the update equation for FCM or PFCM, if the term w_j is replaced by $f_{\mathrm{FCM}}(u_{ij})$ or $f_{\mathrm{PFCM}}(u_{ij})$.

3. M-estimators using fuzzy related error functions

One general goal is, to find an error evaluation function ρ for an M-estimator that shows the same behaviour like PFCM. Therefore, the M-estimators with an error function similar to FCM as Davé and Krishnapuram have discussed in [5] is developed for PFCM. FCM usually is designed for many clusters and one noise cluster in case it is desired. This scenario does not fit to the M-estimator approach because M-estimators are assumed to have one prototype (model). To find an evaluation function ρ that is similar to FCM or PFCM, these clustering approaches have to be restricted to one "normal" and one noise cluster. With only one "normal" prototype, the update equation for the membership values for FCM (1) and PFCM (3), respectively, can be simplified. First, some of the indices can be skipped. $u_j := u_{1j}$ is the membership of the jth data object to the normal cluster. $u_{0j} = 1 - u_j$ is the membership to the noise cluster and of no interest at this point. $y := y_1$ is the location of the one prototype and $d_j := d_{1j}$ is the distance of the jth data object to the prototype. The simplified version of the FCM membership equation update is:

$$u_j^{t+1} \overset{\text{FCM}}{=} \frac{(d_j^t)^{\frac{2}{1-\omega}}}{\left(d_j^t\right)^{\frac{2}{1-\omega}} + d_{\text{noise}}^{\frac{2}{1-\omega}}} = \frac{1}{1 + \left(\frac{d_j^t}{d_{\text{noise}}}\right)^{\frac{2}{\omega-1}}}. \tag{6}$$

Due to the restriction to only 2 clusters, the PFCM update formula becomes very simple in comparison to the original approach. The variable $\hat{c}_j$ can be removed because it can either be 1 which is the crisp case or 2 which is the fuzzy case. In the crisp case, the membership to the normal prototype is either exactly 0 or exactly 1 which does not require an update formula, a simple case distinction will do. Furthermore, the sorting of clusters becomes unnecessary. Equation (3) changes therefore to the following:

$$u_j^{t+1} \overset{\text{PFCM}}{=} \begin{cases} 1 & \text{if} \quad \frac{\left(d_j^t\right)^2}{d_{\text{noise}}^2} \leq \beta \\ \frac{1}{1-\beta}\left(\frac{1+\beta}{1+\frac{\left(d_j^t\right)^2}{d_{\text{noise}}^2}} - \beta\right) & \text{if} \quad \beta < \frac{\left(d_j^t\right)^2}{d_{\text{noise}}^2} < \frac{1}{\beta} \\ 0 & \text{if} \quad \frac{1}{\beta} \leq \frac{\left(d_j^t\right)^2}{d_{\text{noise}}^2} \end{cases}. \tag{7}$$

The update formula for the prototype remains unchanged for both variants of membership functions $f = f_{\text{FCM}}$ and $f = f_{\text{PFCM}}$.

$$y^{t+1} = \frac{\sum_{j=1}^n f(u_j^t) x_j}{\sum_{j=1}^n f(u_j^t)}. \tag{8}$$

Based on the idea in [5, 4], the functions w, ϕ and ρ are defined in such a way that the resulting M-estimator shows the behaviour of FCM. If the behaviour is supposed to be equal, the update procedures for the prototypes must be equivalent. For updating the prototype position, w_j is considered to be a constant in the M-E iterative process very much like u_j in the update process for FCM. From Equation (5) and (8) it can be concluded, that $w_j = w_j(d_j) = f_{\text{FCM}}(u_j)$ must hold while u_j is considered as a function of d_j as it is shown in Equation (6). With $w(d_j) = \frac{\psi(d_j)}{d_j}$

and $\rho(d_j) = \int_0^{d_j} \psi(s)ds$, it is obtained:

$$\begin{aligned} w_{\mathrm{FCM}}(d_j) &= \left(\frac{1}{1+\left(\frac{d_j^2}{d_{\mathrm{noise}}^2}\right)^{\frac{1}{\omega-1}}} \right)^{\omega}, \\ \psi_{\mathrm{FCM}}(d_j) &= d_j \left(\frac{1}{1+\left(\frac{d_j^2}{d_{\mathrm{noise}}^2}\right)^{\frac{1}{\omega-1}}} \right)^{\omega} \quad \text{and} \\ \rho_{\mathrm{FCM}}(d_j) &= \frac{1}{2} d_j^2 \left(\frac{1}{1+\left(\frac{d_j^2}{d_{\mathrm{noise}}^2}\right)^{\frac{1}{\omega-1}}} \right)^{\omega-1}. \end{aligned} \tag{9}$$

The resulting M-estimator behaves just like FCM in the above stated scenario of one "normal" and one noise cluster. Let $\boldsymbol{J}_{\text{M-E FCM}}$ denote the objective function that describes the optimization process for the M-estimator using ρ_{FCM}:

$$\boldsymbol{J}_{\text{M-E FCM}}(\boldsymbol{X}, y) = \sum_{j=1}^{n} \rho_{\mathrm{FCM}}(d_j).$$

The behaviour of PFCM is in some way different from FCM because it explicitly allows crisp results. As for the FCM M-estimator, the PFCM M-estimator equation is defined as follows: $w_{\mathrm{PFCM}}(d_j) = f_{\mathrm{PFCM}}(u_j)$ with u_j is defined by Equation (7). With $f_{\mathrm{PFCM}}(1) = 1$ and $f_{\mathrm{PFCM}}(0) = 0$ and

$$\begin{aligned} f_{\mathrm{PFCM}}(u_j) &= \frac{1-\beta}{1+\beta} u_j^2 + \frac{2\beta}{1+\beta} u_j \\ &= \frac{1-\beta}{1+\beta} \left(\frac{1}{1-\beta}\right)^2 \left(\frac{1+\beta}{1+\frac{d_j^2}{d_{\mathrm{noise}}^2}} - \beta \right)^2 + \frac{2\beta}{1+\beta} \left(\frac{1}{1-\beta} \left(\frac{1+\beta}{1+\frac{d_j^2}{d_{\mathrm{noise}}^2}} - \beta \right) \right) \\ &= \frac{1}{1-\beta^2} \left(\left(\frac{1+\beta}{1+\frac{d_j^2}{d_{\mathrm{noise}}^2}} \right)^2 - \beta^2 \right) \end{aligned} \tag{10}$$

for $u_j \in (0,1)$. This leads to

$$w_{\mathrm{PFCM}}(d_j) = \begin{cases} 1 & \text{if } \frac{d_j^2}{d_{\mathrm{noise}}^2} \leq \beta \\ \frac{1}{1-\beta^2} \left(\left(\frac{1+\beta}{1+\frac{d_j^2}{d_{\mathrm{noise}}^2}} \right)^2 - \beta^2 \right) & \text{if } \beta < \frac{d_j^2}{d_{\mathrm{noise}}^2} < \frac{1}{\beta} \\ 0 & \text{if } \frac{1}{\beta} \leq \frac{d_j^2}{d_{\mathrm{noise}}^2} \end{cases}, \tag{11}$$

$$\psi_{\mathrm{PFCM}}(d_j) = \begin{cases} d_j & \text{if} \quad \frac{d_j^2}{d_{\mathrm{noise}}^2} \leq \beta \\ \frac{d_j}{1-\beta^2}\left(\left(\frac{1+\beta}{1+\frac{d_j^2}{d_{\mathrm{noise}}^2}}\right)^2 - \beta^2\right) & \text{if} \quad \beta < \frac{d_j^2}{d_{\mathrm{noise}}^2} < \frac{1}{\beta} \quad \text{and} \\ 0 & \text{if} \quad \frac{1}{\beta} \leq \frac{d_j^2}{d_{\mathrm{noise}}^2} \end{cases}$$

$$\rho_{\mathrm{PFCM}}(d_j) = \begin{cases} \frac{1}{2}d_j^2 & \text{if} \quad \frac{d_j^2}{d_{\mathrm{noise}}^2} \leq \beta \\ \frac{1}{2}\frac{1}{\beta^2-1}\left[\beta^2\left(d_j^2 + d_{\mathrm{noise}}^2\right) + \frac{(\beta+1)^2 d_{\mathrm{noise}}^4}{d_j^2+d_{\mathrm{noise}}^2}\right] + \rho_1^* - \rho_2^* & \text{if} \quad \beta < \frac{d_j^2}{d_{\mathrm{noise}}^2} < \frac{1}{\beta} \\ \rho_1^* - \rho_2^* + \rho_3^* & \text{if} \quad \frac{1}{\beta} \leq \frac{d_j^2}{d_{\mathrm{noise}}^2} \end{cases}$$

$$= \begin{cases} \frac{1}{2}d_j^2 & \text{if} \quad \frac{d_j^2}{d_{\mathrm{noise}}^2} \leq \beta \\ \frac{1}{2}\frac{1}{\beta^2-1}\left[\beta^2\left(d_j^2 + d_{\mathrm{noise}}^2\right) - (\beta+1)^2 \frac{d_j^2 d_{\mathrm{noise}}^2}{d_j^2+d_{\mathrm{noise}}^2}\right] & \text{if} \quad \beta < \frac{d_j^2}{d_{\mathrm{noise}}^2} < \frac{1}{\beta} \\ \frac{1}{2}d_{\mathrm{noise}}^2 & \text{if} \quad \frac{1}{\beta} \leq \frac{d_j^2}{d_{\mathrm{noise}}^2} \end{cases} .$$

where ρ_1^* to ρ_3^* describe the border values of the resolved integrals. ρ_1^* is the right border in the first case, ρ_2^* is the left boarder for the second case and ρ_3^* is the right border in the second case.

$$\rho_1^* \overset{d_j^2=\beta d_{\mathrm{noise}}^2}{=} \frac{1}{2}\beta d_{\mathrm{noise}}^2, \tag{12}$$

$$\rho_2^* \overset{d_j^2=\beta d_{\mathrm{noise}}^2}{=} \frac{1}{2}\frac{1}{\beta^2-1}\left[\beta^2\left(\beta d_{\mathrm{noise}}^2 + d_{\mathrm{noise}}^2\right) + \frac{(\beta+1)^2 d_{\mathrm{noise}}^4}{\beta d_{\mathrm{noise}}^2 + d_{\mathrm{noise}}^2}\right]$$

$$= \frac{1}{2}\frac{\beta^2+1}{\beta-1}d_{\mathrm{noise}}^2 \text{ and} \tag{13}$$

$$\rho_3^* \overset{d_j^2=\frac{1}{\beta}d_{\mathrm{noise}}^2}{=} \frac{1}{2}\frac{1}{\beta^2-1}\left[\beta^2\left(\frac{1}{\beta}d_{\mathrm{noise}}^2 + d_{\mathrm{noise}}^2\right) + \frac{(\beta+1)^2 d_{\mathrm{noise}}^4}{\frac{1}{\beta}d_{\mathrm{noise}}^2 + d_{\mathrm{noise}}^2}\right]$$

$$= \frac{\beta}{\beta-1}d_{\mathrm{noise}}^2. \tag{14}$$

For example, in the third case the integral can be decomposed as follows:

$$\rho_{\mathrm{PFCM}}(d_j) = \int_0^{d_j} \psi_{\mathrm{PFCM}}(x)dx$$

$$= \underbrace{\int_0^{\sqrt{\beta d_{\mathrm{noise}}^2}} \psi_{\mathrm{PFCM}}(x)dx}_{=\rho_1^*} + \underbrace{\int_{\sqrt{\beta d_{\mathrm{noise}}^2}}^{\sqrt{\frac{1}{\beta}d_{\mathrm{noise}}^2}} \psi_{\mathrm{PFCM}}(x)dx}_{=\rho_3^*-\rho_2^*} + \underbrace{\int_{\sqrt{\frac{1}{\beta}d_{\mathrm{noise}}^2}}^{d_j} \psi_{\mathrm{PFCM}}(x)dx}_{=0} .$$

The case distinction is more or less similar to the M-estimator of Hampel et al. Like $\boldsymbol{J}_{\text{M-E FCM}}$, let $\boldsymbol{J}_{\text{M-E PFCM}}$ denote the objective function that describes the optimization process for the M-estimator using ρ_{PFCM}:

$$\boldsymbol{J}_{\text{M-E PFCM}}(\boldsymbol{X}, y) = \sum_{j=1}^{n} \rho_{\text{PFCM}}(d_j).$$

Initialization can be a quite hard problem, especially for clustering algorithms with multiple prototypes. However, for prototype-based M-estimators like FCM and PFCM related M-estimators, the situation is much better. Due to the fact that there is only one prototype to be initialized, simple statistical methods can be used. For example the multidimensional median, that takes each component of the vector space as single median. Alternatively, a truncated mean with a large truncated area might also be a good initialisation. For simplicity, only the (multidimensional) median is considered. For the PFCM M-estimator, one specific problem can arise. Because it does not "see" the full data set when weights are zero, it is likely, that data objects far away will never influence the calculation process. On the one hand, this is good for the robustness of the algorithm. On the other hand, it can mean that there are no data objects in the calculation area of the prototype. That can happen after initialization. If that happens, either the initialization was not correct or the noise distance parameter d_{noise} was ill chosen.

Depending on the problem, it might be useful to use the FCM M-estimator to find a good initialization for the PFCM M-estimator. The advantage of the FCM M-estimator is that it takes all data objects into account and will end up in close proximity of at least one data object. But also for this, it is easy to construct an example where the prototype ends up in a very small group of data objects that have nothing to do with the majority of the data. These problems are caused by the local nature of PFCM M-estimators. It would be best if some structural knowledge of the data set is available to define a meaningful starting point. Otherwise, several random initializations might lead to good results.

4. Comparing M-estimators

The influence functions (IF) for the fuzzy clustering algorithm induced M-estimators FCM M-E and PFCM M-E can be computed in order to analyse their robust properties. For both, the update function follows Equation 8 which means, y is the weighted mean of the data objects. In this sense, the prototype y has the weight $W_y = \sum_{j=1}^{n} f(u_j) = \sum_{j=1}^{n} w_j$. Let $\tilde{X} : \Omega \to V$ be the probability distribution that describes the data set and $\Delta_{x'}$ be the probability distribution with point mass 1 at x'. Let T be an M-estimator of domain V that follows Equation 5, $t \in [0, 1]$ and the influence function be

$$IF(x', T, \tilde{X}) = \lim_{t \downarrow 0} \frac{T((1-t)\tilde{X} + t\Delta x') - T(\tilde{X})}{t}. \tag{15}$$

For the objective function $J' = \sum_{j=1}^{n}(1-t)f(u_j)\|x_j - y'\|^2 + tf(u_{x'})\|x' - y'\|^2$ with f being a fuzzifier function, the estimator has the form

$$y' = \frac{(1-t)\sum_{j=1}^{n} f(u_j)x_j + tf(u_{x'})x'}{(1-t)\sum_{j=1}^{n} f(u_j) + tf(u_{x'})} = \frac{(1-t)W_y y + tf(u_{x'})x'}{(1-t)W_y + tf(u_{x'})}. \tag{16}$$

This notation is easily extended from the discrete form to probability density functions which can be inserted into the influence function. The prototype weight is defined as $W_y = \int_{x \in V} w(\|x-$

TABLE 1. Several loss functions and their related weight functions

Function and its Parameters	Loss Function $\rho(d)$	Derivative $\psi(d)$	Weight function $w(d)$	Parameter Range
Mean	$\frac{1}{2}d^2$	d	1	$0 \le d$
Median	d	1	$\frac{1}{d}$	$0 \le d$
FCM $d_{\text{noise}} > 0$ $\omega > 1$	$\frac{d^2}{2}\left(\frac{1}{1+\left(\frac{d^2}{d^2_{\text{noise}}}\right)^{\frac{1}{\omega-1}}}\right)^{\omega-1}$	$d\left(\frac{1}{1+\left(\frac{d^2}{d^2_{\text{noise}}}\right)^{\frac{1}{\omega-1}}}\right)^{\omega}$	$\left(\frac{1}{1+\left(\frac{d^2}{d^2_{\text{noise}}}\right)^{\frac{1}{\omega-1}}}\right)^{\omega}$	$0 \le d$
PFCM $d_{\text{noise}} > 0$ $0 \le \beta < 1$	$\frac{1}{2}d^2$ $\frac{1}{2}\frac{1}{\beta^2-1}\left[\beta^2\left(d_j^2+d^2_{\text{noise}}\right) - (\beta+1)^2\frac{d_j^2 d^2_{\text{noise}}}{d_j^2+d^2_{\text{noise}}}\right]$ $\frac{1}{2}d^2_{\text{noise}}$	d $\frac{d_j}{1-\beta^2}\left(\left(\frac{1+\beta}{1+\frac{d_j^2}{d^2_{\text{noise}}}}\right)^2-\beta^2\right)$ 0	1 $\frac{1}{1-\beta^2}\left(\left(\frac{1+\beta}{1+\frac{d_j^2}{d^2_{\text{noise}}}}\right)^2-\beta^2\right)$ 0	$\frac{d^2}{d^2_{\text{noise}}} \le \beta$ $\beta < \frac{d^2}{d^2_{\text{noise}}} < \frac{1}{\beta}$ $\frac{1}{\beta} \le \frac{d^2}{d^2_{\text{noise}}}$
Huber $k > 0$	$\begin{cases}\frac{1}{2}d^2 \\ k\cdot d - \frac{1}{2}k^2\end{cases}$	$\begin{cases}d \\ k\end{cases}$	$\begin{cases}1 \\ \frac{k}{d}\end{cases}$	$0 \le d < k$ $d \ge k$
Hampel $0 < a \le b \le c$ $a_1 = ab - \frac{1}{2}a^2$ $a_2 = \frac{a}{2}(c-b)$	$\begin{cases}\frac{1}{2}d^2 \\ a\cdot d - \frac{1}{2}a^2 \\ a_1 + a_2\left(1-\left(\frac{c-d}{c-b}\right)^2\right) \\ a_1 + a_2\end{cases}$	$\begin{cases}d \\ a \\ a\left(\frac{c-d}{c-b}\right) \\ 0\end{cases}$	$\begin{cases}1 \\ \frac{a}{d} \\ \frac{a}{d}\left(\frac{c-d}{c-b}\right) \\ 0\end{cases}$	$0 \le d \le a$ $a < d \le b$ $b < d \le c$ $c < d$
Cauchy $d_{\text{noise}} > 0$	$\frac{d^2_{\text{noise}}}{2} ln\left(1+\frac{d^2}{d^2_{\text{noise}}}\right)$	$d\left(1+\frac{d^2}{d^2_{\text{noise}}}\right)^{-1}$	$\left(1+\frac{d^2}{d^2_{\text{noise}}}\right)^{-1}$	$0 \le d$
Tukey $d_{\text{noise}} > 0$	$\begin{cases}\frac{1}{6}\left(1-\left(1-\frac{d^2}{d^2_{\text{noise}}}\right)^3\right) \\ \frac{1}{6}\end{cases}$	$\begin{cases}d\left(1-\frac{d^2}{d^2_{\text{noise}}}\right)^2 \\ 0\end{cases}$	$\begin{cases}\left(1-\frac{d^2}{d^2_{\text{noise}}}\right)^2 \\ 0\end{cases}$	$0 \le d < d_{\text{noise}}$ $d_{\text{noise}} \le d$

$y\|)dF$ in the continuous case.

$$\begin{aligned} IF(x', T, \tilde{X}) &= \lim_{t\downarrow 0}\frac{y'-y}{t} \\ &= \lim_{t\downarrow 0}\frac{\frac{(1-t)W_y y + tf(u_{x'})x'}{(1-t)W_y + tf(u_{x'})} - y}{t} \\ &= \lim_{t\downarrow 0}\frac{f(u_{x'})}{W_y + tf(u_{x'}) - tW_y}(x'-y) \\ &= \frac{f(u_{x'})}{W_y}(x'-y) = \frac{w_{x'}}{W_y}(x'-y). \end{aligned} \tag{17}$$

The fuzzified membership value $f(u_{x'})$ or the weight $w_{x'}$ respectively only depend on the distance between x' and y, where W_y is a positive, but fixed value. As Equation (17) shows, in a 1-dimensional environment $(x'-y) = d_{x'}$, hence the influence function is proportional to the ψ function with $IF = \frac{\psi}{W_y}$. The plots for ψ_{FCM} ($\omega = 2$) and ψ_{PFCM} ($\beta = 0.2$) are shown in Figure 3, with $d_{\text{noise}} = 0.2$.

From the loss function ψ, the robust parameters gross error sensitivity γ^*, local shift sensitivity λ^* and the rejection point p^* can be computed. For this analysis, only the 1-dimensional case is considered and the robust parameters are calculated depending on the distance of to the

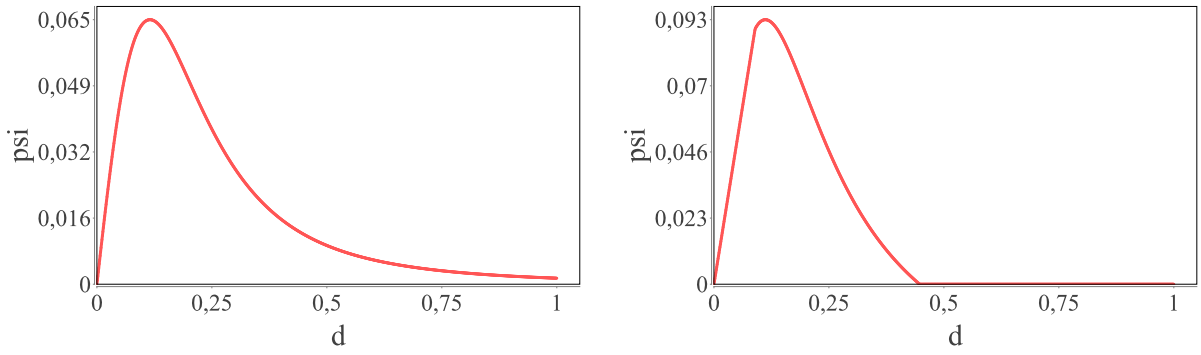

FIGURE 3. The influence function for the FCM M-E (left) and the FCM M-E (right).

prototype: $d = \|x' - y\|$. It is immediately clear that the PFCM M-estimator has a finite rejection point $p^*_{\text{PFCM}} = \sqrt{d^2_{\text{noise}} \frac{1}{\beta}}$ while the rejection point of the FCM M-estimator is infinite.

The gross error sensitivity for FCM γ^*_{FCM} is achieved at a distance of $d_{\gamma^*_{\text{FCM}}} = d_{\text{noise}} \left(\frac{\omega-1}{\omega+1}\right)^{\frac{\omega-1}{2}}$ from which follows that

$$\gamma^*_{\text{FCM}} = \frac{d_{\text{noise}}}{W_y} (2\omega)^{-\omega} (\omega - 1)^{\frac{\omega-1}{2}} (\omega + 1)^{\frac{\omega+1}{2}}. \tag{18}$$

The gross error sensitivity for the PFCM M-estimator can only occur if $\beta \leq \frac{d^2}{d^2_{\text{noise}}} \leq \frac{1}{\beta}$ holds. If $\beta > \frac{d^2}{d^2_{\text{noise}}}$, the influence function is linear which means the supremum is at $\beta = \frac{d^2}{d^2_{\text{noise}}}$ and in case of $\frac{d^2}{d^2_{\text{noise}}} > \frac{1}{\beta}$, the influence function is constantly zero. Calculating the gross error sensitivity for the PFCM M-estimator leads to a very complicated expression: $d = \sqrt{\frac{(1-\beta)^2 n^4}{A} + \frac{A}{\beta^2} - n^2}$ with $A = \left(2\beta^6 n^6 + 4\beta^5 n^6 + 2\beta^4 n^6 + (5\beta^{12} n^{12} + 22\beta^{11} n^{12} + 39\beta^{10} n^{12} + 36\beta^9 n^{12} + 19\beta^8 n^{12} + 6\beta^7 n^{12} + \beta^6 n^{12})^{\frac{1}{2}}\right)^{\frac{1}{3}}$. However, the gross error sensitivity is linear depending on d_{noise} and for $\beta = \frac{1}{2}$, $d_{\gamma^*_{\text{PFCM}}} \approx 0.51 * d_{\text{noise}}$ so that $\gamma^*_{\text{PFCM}} \approx \frac{0.793}{W_y} d_{\text{noise}}$.

The influence functions for FCM and PFCM are continuous and except for PFCM in $d = \sqrt{\beta d^2_{\text{noise}}}$ and $d = \sqrt{\frac{1}{\beta} d^2_{\text{noise}}}$ even smooth. Therefore and with the mean value theorem, the local shift sensitivity λ^* can be largely computed using the derivation of the influence function

$$\lambda^* = \sup_{x \neq x'} \left| \frac{IF(x, T, F) - IF(x', T, F)}{x - x'} \right| = \sup_{x \in V} \left| \frac{\partial}{\partial x} IF(x, T, F) \right|.$$

It should be noted that for $\beta = 1$, the PFCM M-estimator turns into a crisp version of this estimator with a discontinuity at $d = d_{\text{noise}}$. The same effect would occur for $\omega = 1$. Both discontinuity cases suggest that the local shift sensitivity can become very large for a small $\varepsilon > 0$ and $\omega = 1 + \varepsilon$ as well as $\beta = 1 - \varepsilon$ respectively. With a proper use of the parameter, the local shift sensitivity λ^* is bounded and fairly small for both M-estimators.

The breakdown point however depends very much on the initialization of the prototype for the PFCM and FCM M-estimator. The initialization process is not discussed, so the breakdown point can not be finally specified. A good choice is the multidimensional median or an initialization due to data structure knowledge which initializes the prototype in a useful position. Given a useful initialization, the PFCM M-estimator has a breakdown point of 0.5 because of the finite rejection point. In case of FCM, for $\omega < 2$ [5], also the FCM M-estimator has a breakdown point of 0.5.

All other algorithms in Table 1, except the mean, have a breakdown point of 0.5, but their other properties differ. Only Hampels M-E and Tukeys biweight have a finite rejection point.

Also the gross error sensitivity of FCM and PFCM M-estimator is linear dependent on d_{noise}, as it is the case for Cauchys M-estimator and Tukeys biweight. In all cases, the influence function is smooth and therefore, the local shift sensitivity is bounded and for reasonable parameter choices fairly small.

5. Fuzzy clustering approaches induces by M-estimators

The extension of M-estimators of location to the case with more than one location, like in fuzzy clustering is not a trivial task. Frigui and Krishnapuram made a nice proposal in [10] by introducing RPC, a clustering algorithm with robust loss functions. However, the use of loss functions, wrapped around the distance is quite limited because the derivative of the objective function must be solvable for the prototypes positions y_i. Therefore, the objective function

$$\boldsymbol{J}_{\text{RPC}}(\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{Y}) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{\omega} \rho(d_{ij}^2)$$

does not provide the desired effect.

A different attempt how an M-estimator can be extended to multiple prototypes without loosing the characteristics of the original M-estimator is presented here. This attempt should not suffer the same problems as the RPC approach and is therefore not based on an objective function. The idea is basically to use the robust weights of prototypes independently as fuzzy membership values. To avoid a clustering algorithm with independent prototypes, the membership values are normalized to create a clustering algorithm, similar to FCM.

Let f be the fuzzifier function for a fuzzy clustering algorithm as in Equation (8). In a fuzzy algorithm, the sum of membership values is equal to 1. However, the membership values are implicitly determined by the prototypes, the update process just requires the fuzzified membership values (see Equations (2) and (8)). This fact is important since a function like f is usually not specified for M-estimators: $w_j = w(d_j) = f(u_j)$ for a data object x_j in a fuzzy clustering algorithm. The membership value u_j can also be considered to be a function of d_j, $u_j = u(d_j)$ while the function u depends on the fuzzy clustering algorithm: $f(u(d_j)) = w(d_j)$. The problem is, that there is no distinct equation for u_j, if considering an M-estimator. Therefore, the condition of $\sum_j u_j = 1$ is hard to specify.

If going from one prototype to many prototypes considering, $f(u(d_{ij})) = w(d_{ij})$, the result is a clustering algorithm with c independent prototypes. This is not very convenient since it makes the clustering result extremely dependent on the initialization, much like PCM. Therefore, an interdependence among prototypes is necessary to create a reasonable well working clustering algorithm. One way of creating an interdependence is, to normalise the membership values. Let d_{ij} be the distance from prototype y_i to data object x_j and $w_{ij} = w(d_{ij})$ be one of the weight functions from table 1. Let the normalization $g : [0, 1] \to [0, 1]$ be a monotonously increasing function with $g(0) = 0$. In the remaining part of this section, several normalisation methods are discussed for this scenario.

The first normalisation methods are very straight forward.

$$g_1(w_{ij}) = \frac{w_{ij}}{\sum_{k=1}^{c} w_{kj}} \text{ and} \tag{19}$$

$$g_2(w_{ij}) = \frac{w_{ij}}{\sum_{k=1}^{c} w_{kj}} \cdot w_{ij}. \tag{20}$$

.

g_1 maps the weights on the $[0, 1]$ interval in such a way, that the sum of all normalised weights is 1. This is close to FCM, but it does not fit for the robust weight calculation because for some algorithms, the weights can become zero. If all weights but one are zero, the normalisation of a non-zero weight is set to 1, which is not very robust as figure 4-left indicates for Hampel's M-estimator. This problem not only holds for Hampel's M-estimator, but also for Tukeys and the PFCM M-estimator since both can produce 0 weights. The scenario presented in figure 4

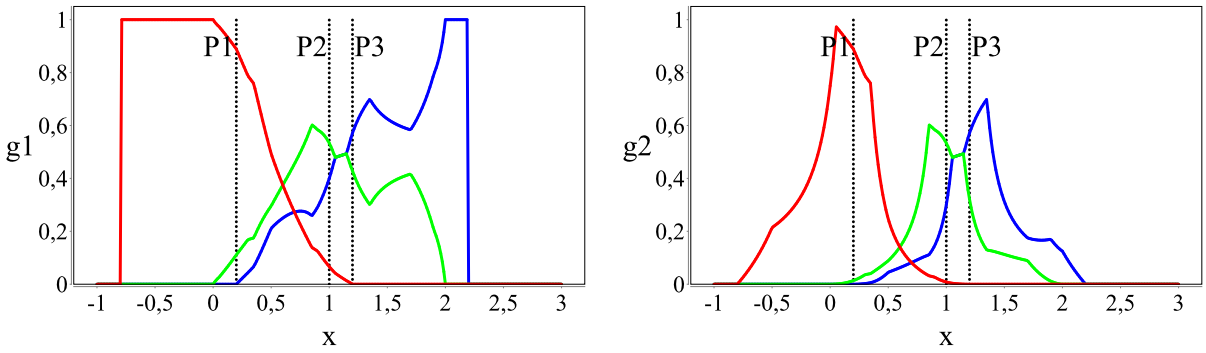

FIGURE 4. The effect of normalisation g_1 (left) and g_2 (right) on Hampel's M-estimator.

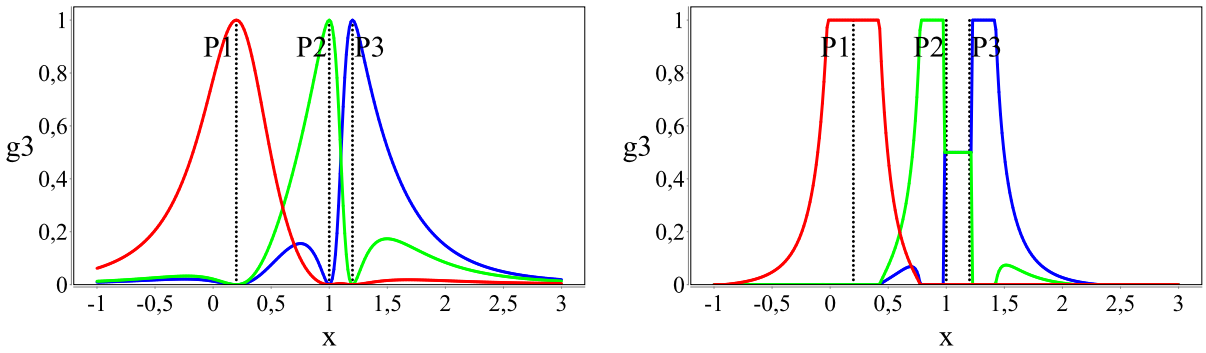

FIGURE 5. The effect of g_3 on the FCM M-estimator (left) and the PFCM M-estimator (right).

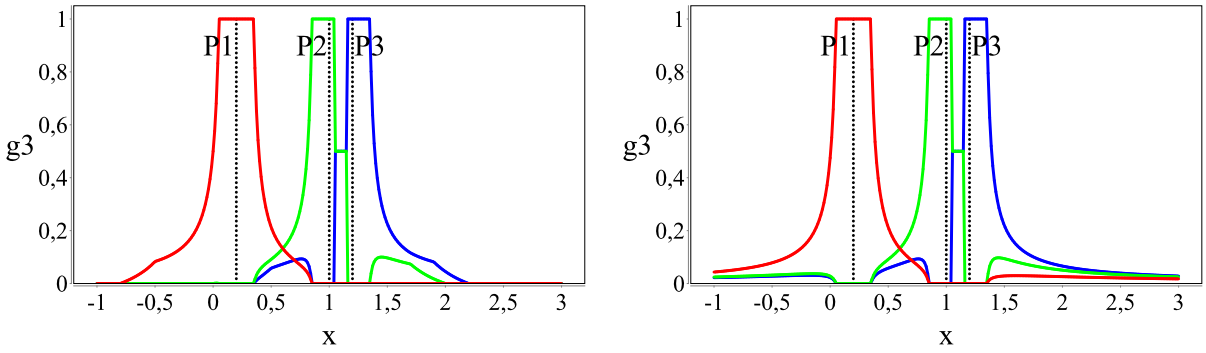

FIGURE 6. The effect of g_3 on Hampel's M-estimator (left) and Hubert's M-estimator (right).

is the same as in figure 1 and 2. Three prototypes are placed at 0.2, 1.0 and 1.2 in an one dimensional environment. The weights from Hampel's M-estimator are normalised according to Equation (19) for the left hand side.

g_2 overcome the problem of g_1 by multiplying the original weight on the normalised value of g_1. This takes the actual robustness of the weight into account. Figure 4-right already indicates to the problem in the area of prototype 2 and 3. The normalized weight values overlap very much which which is not good because data objects in this area have a strong influence on both prototypes. The prototypes tend to go on the same position because data objects in the overlapping area have a strong influence on both. Having two or more prototypes covering

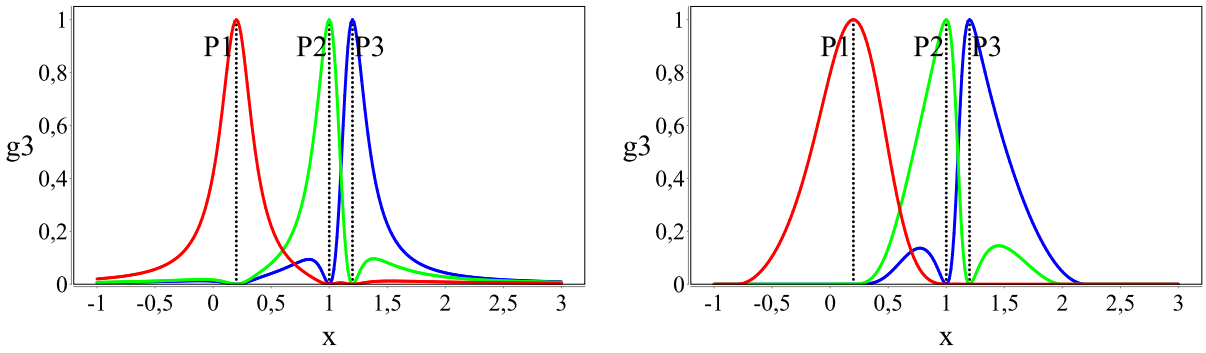

FIGURE 7. The effect of g_3 on Cauchy's M-estimator (left) and Tukeys M-estimator (right).

one cluster usually means that at least one other cluster is uncovered, which is quite bad for a clustering algorithm. The same holds for all six weight functions, presented in table 1.

In FCM as well as in PFCM, close prototypes are pulled apart from the surrounding data objects. This comes from the inverted mean in FCM and PFCM of the form

$$u_{ij} = \frac{\frac{1}{d_{ij}^2}}{\sum_{k=1}^{n} \frac{1}{d_{kj}^2}}.$$

which gives data objects close to an other prototype a membership value of almost 0. As this kind of mean is missing in the two presented normalisation methods, close prototypes are not well separated. Therefore, the third normalisation maps the weights from $[0, 1]$ to $[0, \infty)$ in order to get a more sophisticated mean.

$$g_3(w_{ij}) = \frac{\frac{1}{1-w_{ij}}}{\sum_{k=1}^{c} \frac{1}{1-w_{kj}}} \cdot w_{ij}. \tag{21}$$

g_3 reduces the problem, that two prototypes run into the same location. However, there are situations where this happens. If Huber's, Hampel's or the PFCM M-estimator is used, areas of weight 1 appear. In this case, it is not clear how the data objects are partitioned. This situation can be solved by giving all membership values the same normalised weight. If at least one weight is equal to 1, let $\bar{c} = |\{w_{ij} : w_{ij} = 1, i = 1 \dots c\}|$ be the number of weights of value 1.

$$g_3(w_{ij}) = \begin{cases} 0 & \text{if } w_{ij} < 1 \\ \frac{1}{\bar{c}} & \text{if } w_{ij} = 1 \end{cases}$$

With normalisation g_3, the M-estimator induced clustering algorithms work quite well, see figure 6 for Hamels M-estimator. The FCM and PFCM algorithm give a nice insight about the difference in the presented M-estimators and their original fuzzy clustering algorithms. The extended FCM induced M-estimator, presented in figure 5-left is best to be compared with figure 1-right, when the presentation of the noise cluster is ignored. The fuzzified membership values in figure 5 and the normalised weight values in figure 1 show almost an identical profile. PFCM with noise and the extended PFCM induced M-estimator, presented in figure 2-right and 5-right respectively are also very similar. The obvious difference is in between prototype 2 and 3 where the original PFCM provides a better, fuzzy, transfer of membership values. Also the extended PFCM M-estimator is a little bit steeper than the PFCM algorithm with noise.

The only common problem for weight functions that allow strictly 0 values is, that their prototype is initialised to far away from any data object. This causes the prototype to be 'empty' which makes it impossible to move into a useful direction. Observe figures 8, 9 and 10 for examples of the proposed extended M-estimators. The 'tails' of the prototypes represent

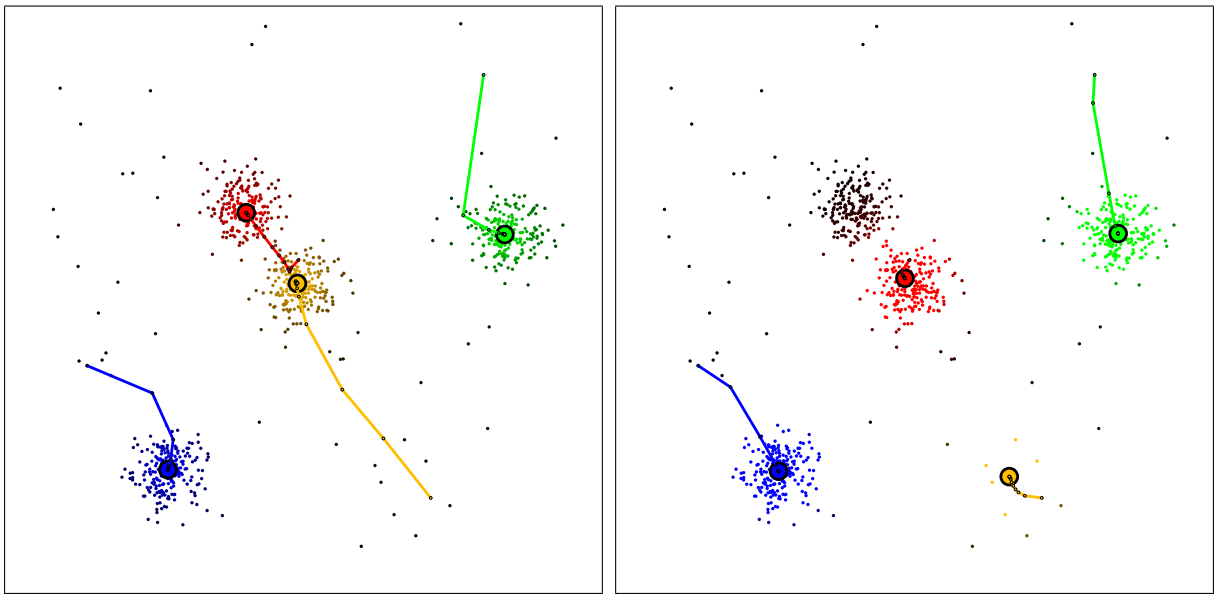

FIGURE 8. Extended FCM M-E (left) and extended PFCM M-E (right) applied on a artificial data set.

the way the prototypes took from their initialisation position to their final position. Noise data objects are presented black while the color intensity of a data object refers to the value of the normalised

The Huber Estimator does not work well because the influence of a data object increases linearly with its distance while the weight of the Huber M-estimator decreases anti-linear ($\frac{1}{\text{distance}}$). Therefore, nearby data objects are not favoured against far away data objects, so the prototypes tend to go into the centre of gravity of the data set. This is usually not a problem in a one-cluster scenario with a small percentage of noise. But with many clusters in a data set, and one prototype should cover just one cluster, most data objects are considered to be noise for this one cluster. In this case, a finite rejection point or fast decreasing weight function would help.

The extended FCM M-E algorithm performs very well as does the extended M-E of Cauchy and Hampel in the example, presented in figure 9. The extended M-E of Cauchy however has in general similar problems like Hubers extended M-E but they do not occur as often. The prototypes of the extended Cauchy M-E tend to run into the centre of gravity and they do not find the centre of Clusters as good as for example the extended FCM M-E, the prototypes are always slightly drawn to the centre of the data set.

The extended Tukey M-E as well as the extended PFCM M-E both have problems finding the final cluster in this example since their remaining prototype is too far away and the membership values of all "good" cluster data objects are 0 or extremely small. This might be a matter of choosing the correct parameter, but the parameter should allow the algorithm to represent the "good" data as well as possible. The parameter should not be chosen in a way that the area with represented weight 1 is larger than the cluster itself in order to make the algorithm working. In a one cluster environment, the (multidimensional) median is usually a good initialisation which is not the case for many clusters.

6. CONCLUSIONS

Fuzzy set theory had a great impact on data mining, clustering in particular. In this paper, we have explored the connection of fuzzy c-means and hard c-means clustering, as well as statistical motivated M-estimators. Although these many methods are not identical, many have similar

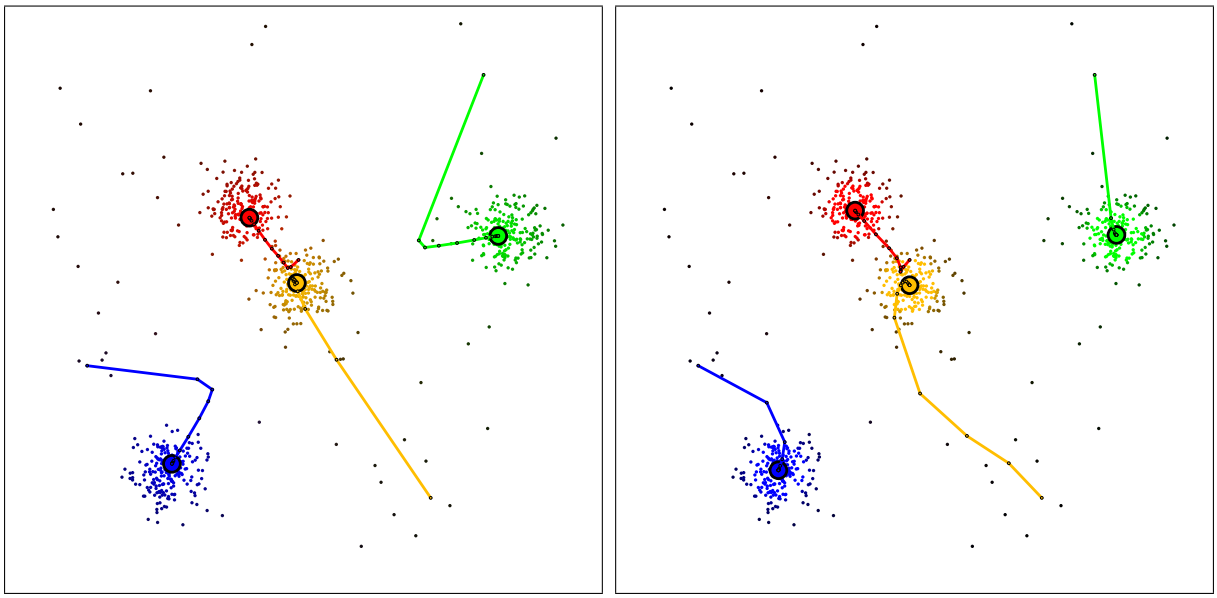

FIGURE 9. Extended Cauchy M-E (left) and extended Hampel M-E (right) applied on a artificial data set.

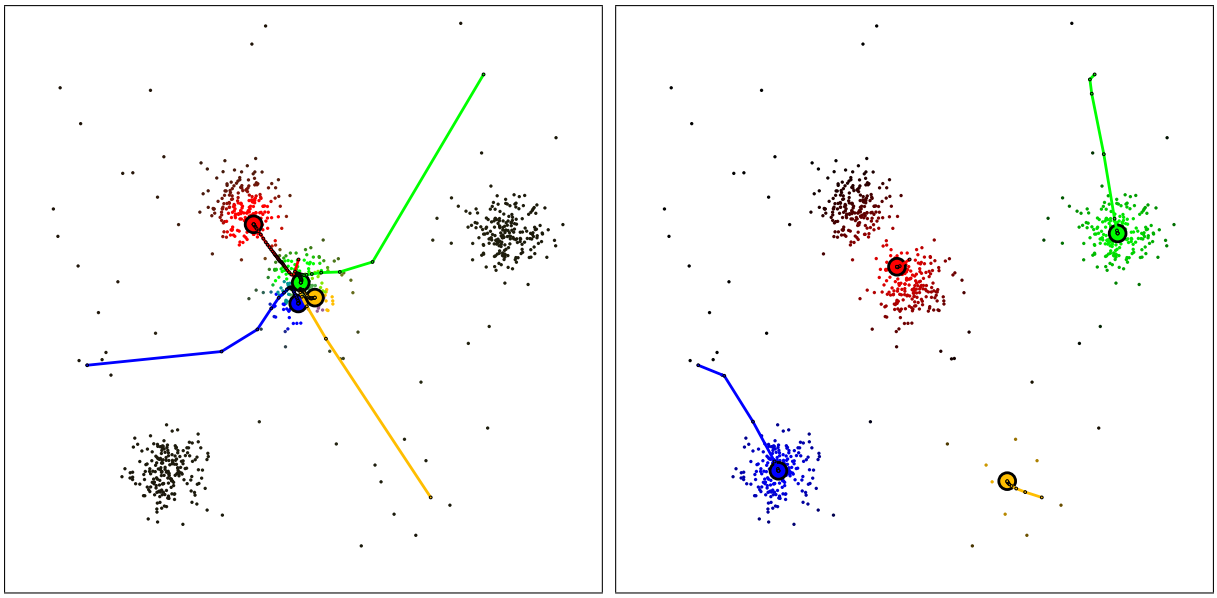

FIGURE 10. Extended Huber M-E (left) and extended Tukey M-E (right) applied on a artificial data set.

properties. The newly introduced PFCM M-estimator has similar properties to well known M-estimators for example Hampel's M-estimator. It is based on its fuzzy version and has all its stability properties and shares the same problems with local optimal solutions. Naturally, the PFCM M-estimator has other robustness properties than the FCM M-estimator because the PFCM M-E has a finite rejection point which is not the case for the FCM M-E.

Because M-estimators target a more simple problem than clustering algorithms, the transition from M-estimators to multi-cluster problems is more complicated. We present an attempt to do so by introducing a clever normalization method. The results are very promising but there is still potential for improvements. Especially because some extended M-estimators do not perform well. One of the main problems is, that some prototypes do not find "good" data because their

radius of rejection is too small. An other problem is, that for close enough prototypes, they can move together on the same location if their weight functions form a plateau of constant values near the prototypes. Also some extended M-estimators like Huber's and Cauchy's have problems with rejecting far away data objects, hence they tend to end up in the centre of gravity of the data set. It might be possible to improve the performance by using a better normalisation or to find a more natural way of extending the weight functions. The fuzzy-clustering induced M-estimator weights however perform quite well.

References

[1] Albert E. Beaton and John W. Tukey. The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16(2):147–185, 1974.

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

[3] Rajesh N. Dave. Characterization and detection of noise in clustering. *Pattern Recogn. Lett.*, 12(11):657–664, 1991.

[4] R.N. Dave and R. Krishnapuram. M-estimators and robust fuzzy clustering. In *Fuzzy Information Processing Society, 1996. NAFIPS. 1996 Biennial Conference of the North American*, pages 400–404, Jun 1996.

[5] R.N. Dave and R. Krishnapuram. Robust clustering methods: a unified view. *Fuzzy Systems, IEEE Transactions on*, 5(2):270–293, May 1997.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Rozal Statistical Society, Ser. B*, 39(1):1–38, 1977.

[7] J.C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Cybernetics and Systems: An International Journal*, 3(3):32–57, 1973.

[8] Fisher. Theory of statistical estimation. In *Proc. Cambridge Philosophical Society*, volume 22, pages 700–725. Cambridge University Press, Cambridge, 1925.

[9] R. A. Fisher. On an absolute criterion for fitting frequency curves. *Messenger of Mathematics*, 41:155160, 1912.

[10] Hichem Frigui and Raghu Krishnapuram. A robust algorithm for automatic extraction of an unknown number of clusters from noisy data. *Pattern Recognition Letters*, 17(12):1223 – 1232, 1996.

[11] Frank R. Hampel, Elvezio M. Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. Wiley-Interscience, New York, revised edition, April 2005.

[12] Peter J. Huber. *Robust statistics*. Wiley, 1981.

[13] P.J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[14] Frank Klawonn. Fuzzy clustering: insights and a new approach. *Mathware and Soft Computing*, 11:125–142, 2004.

[15] Frank Klawonn and Frank Höppner. An alternative approach to the fuzzifier in fuzzy clustering to obtain better clustering. In *EUSFLAT Conf.*, pages 730–734, 2003.

[16] Frank Klawonn and Frank Höppner. What is fuzzy about fuzzy clustering? understanding and improving the concept of the fuzzifier. In *Cryptographic Hardware and Embedded Systems - CHES 2003*, volume 2779 of *Lecture Notes in Computer Science*, pages 254–264. Springer Berlin / Heidelberg, 2003.

[17] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[18] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338 – 353, 1965.

Roland Winkler- Roland Winkler received his Diploma degree in computer science from the University of Magdeburg in 2008. Since then, he is a PhD Student at the German Aerospace Center in Braunschweig, Germany. His scientific interests focus on the dimension problem of data analysis, especially in the domain of clustering and classification. He is also interested in exploiting connections between robust statistics and fuzzy clustering.

Frank Klawonn- Frank Klawonn received his M.Sc. and Ph.D. in mathematics and computer science from the University of Braunschweig in 1988 and 1992, respectively. He is now the head of the Lab for Data Analysis and Pattern Recognition at the University of Applied Sciences in Wolfenbuettel (Germany). His main research interests focus on techniques for intelligent data analysis, especially clustering and classification. He is also the head of the Bioinformatics and Statistics Group at Helmholtz Centre for Infection Research in Braunschweig, Germany and a member of the editorial boards of international journals like "Data Mining, Modelling and Management", "Evolving Systems", "Uncertainty, Fuzziness and Knowledge-Based Systems" "Knowledge Engineering and Soft Data Paradigms", "Fuzzy Sets and Systems", "Hybrid Information Technology" and "Intelligent Systems".

Rudolf Kruse- Rudolf Kruse obtained his diploma (Mathematics) degree in 1979 from University of Braunschweig, Germany, and a PhD in Mathematics in 1980 as well as the venia legendi in Mathematics in 1984 from the same university. Following a short stay at the Fraunhofer Gesellschaft, in 1986 he joined the University of Braunschweig as a professor of computer science. Since 1996 he is a full professor at the Faculty of Computer Science of the Otto-von-Guericke University in Magdeburg where he is leading the computational intelligence research group.
He has carried out research and projects in statistics, artificial intelligence, expert systems, fuzzy control, fuzzy data analysis, computational intelligence, and data mining. His research group is very successful in various industrial applications.
He has coauthored 15 monographs, 15 edited books, as well as more than 330 refereed technical papers in various scientific areas. He is associate editor of several scientific journals. He is a fellow of the International Fuzzy Systems Association (IFSA), fellow of the European Coordinating Committee for Artificial Intelligence (ECCAI) and fellow of the Institute of Electrical and Electronics Engineers (IEEE).