Problems of Fuzzy c-Means Clustering and Similar Algorithms with High Dimensional Data Sets

Roland Winkler and Frank Klawonn and Rudolf Kruse

Abstract Fuzzy c-means clustering and its derivatives are very successful on many clustering problems. However, fuzzy c-means clustering and similar algorithms have problems with high dimensional data sets and a large number of prototypes. In particular, we discuss hard c-means, noise clustering, fuzzy c-means with polynomial fuzzifier function and its noise variant. A special test data set that is optimal for clustering is used to show weaknesses of said clustering algorithms in high dimensions. We also show that a high number of prototypes influences the clustering procedure in a similar way as a high number of dimensions. Finally, we show that the negative effects of high dimensional data sets can be reduced by adjusting the parameter of the algorithms, i.e. the fuzzifier, depending on the number of dimensions.

1 Introduction

Clustering high dimensional data has many interesting applications. For example clustering similar music files, semantic web applications, image recognition or biochemical problems. Many tools today are not designed to handle hundreds of dimensions, or in this case, it might be better to call it degrees of freedom. Many clustering approaches work quite well in low dimensions, but especially the fuzzy c-means algorithm (FCM), [4, 2, 8, 10] seems to fail in high dimensions. This paper is dedicated to give some insight into this problem and the behaviour of FCM as well as its derivatives in high dimensions.

Roland Winkler
German Aerospace Center Braunschweig e-mail: roland.winkler@dlr.de

Frank Klawonn
Ostfalia, University of Applied Sciences e-mail: f.klawonn@ostfalia.de

Rudolf Kruse
Otto-von-Guericke University Magdeburg e-mail: kruse@iws.cs.uni-magdeburg.de

The algorithms that are analysed and compared in this paper are hard c-means (HCM), fuzzy c-means (FCM), noise FCM (NFCM), FCM with polynomial fuzzifier function (PFCM) and PFCM with a noise cluster (PNFCM) that is an extension of PFCM in the same way like NFCM is an extension of FCM. All these algorithms are prototype based and gradient descent algorithms. Previous to this paper, an analysis of FCM in high dimensions is presented in [12] which provides a more extensive view on the high dimension problematic but solely analysis the behaviour of FCM. Not included in this paper is the extension by Gustafson and Kessel [7] because this algorithm is already unstable in low dimensions. Also not included is the competitive agglomeration FCM (CAFCM) [6] the algorithm is not a gradient descent algorithm in the strict sense.

A very good analysis of the influence of high dimensions is to the nearest neighbour search is done in [1]. The nearest neighbour approach can not be applied directly on clustering problems. But the basic problem is similar and thus can be used as a starting point for the analysis of the effects of high dimensional data on FCM as it is presented in this paper.

We approach the curse of dimensionality for the above mentioned clustering algorithms because they seem very similar but perform very differently. The main motivation lies more in observing the effects of high dimensionality rather than producing a solution to the problem. First, we give a short introduction to the algorithms and present a way how to test the algorithms in a high dimensional environment in the next section. In Section 3, the effects of a high dimensional data set are presented. A way to use the parameters of the algorithms to work on high dimensions is discussed in section 4. We close this paper with some last remarks in section 5, followed by a list of references.

2 The algorithms and the test environment

A cluster is defined as a subset of data objects of the data set X that belong together. The result of a (fuzzy) clustering algorithm is a (fuzzy) partitioning of X. All discussed algorithms in this paper proceed by a gradient descent strategy. The method of gradient descent is applied to an objective function with the following form: Let $X = \{x_1, \dots, x_m\} \subset \mathbb{R}^n$ be a finite set of data objects of the vector space $\mathbb{R}^n$ with $|X| = m$. The clusters are represented by a set of prototypes $Y = \{y_1, \dots, y_c\} \subset \mathbb{R}^n$ with $c = |Y|$ be the number of clusters. Let $f : [0,1] \to [0,1]$ be a strictly increasing function called the fuzzifier function and $U \in \mathbb{R}^{c \times m}$ be the partition matrix with $u_{ij} \in [0,1]$ and $\forall j : \sum_{i=1}^{c} u_{ij} = 1$. And finally, let $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be the Euclidean distance function and $d_{ij} = d(y_i, x_j)$.

The objective function J is defined as

$$J(X,U,Y) = \sum_{i=1}^{c} \sum_{j=1}^{m} f(u_{ij}) d_{ij}^2. \tag{1}$$

The minimisation of J is achieved by iteratively updating the members of U and 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 $t \in \mathbb{N}$ with $t = 0$ as the initialisation step for the prototypes. The algorithms HCM, FCM and PFCM have each a different fuzzifier function. The variances NFCM and PNFCM use a virtual noise cluster with a user specified, constant noise distance to all data objects: $\forall j = 1..m$, $d_{0j} = d_{\text{noise}}$, $0 < d_{\text{noise}} \in \mathbb{R}$. The noise cluster is represented in the objective function as additional cluster with index 0 so that the sum of clusters is extended to $i = 0$ to c.

To have a first impression on the behaviour of the algorithms in question, we apply them on a test data set T_d. T_d is sampled from a set of normal distributions representing the clusters and 10% of normal distributed noise. In T_d, the number of clusters is set to $c = 2n$, in our examples we use values of $n = 2$ and $n = 50$. The performance of a clustering algorithm applied on T_d is measured by the number of correctly found clusters and correctly represented number of noise data objects. A cluster counts as found if at least one prototype is located in the convex hull of the data objects of that cluster.

HCM [11] is not able to detect noise and it is not a fuzzy algorithm. The fuzzifier function is the identity: $f_{\text{HCM}}(u) = u$ and the membership values are restricted to $u_{ij} \in \{0, 1\}$. If applied on T_{50}, HCM finds around 40 out of 100 clusters.

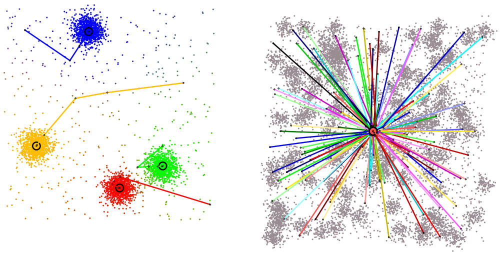

Fig. 1 FCM, applied on T_2 (left) and T_{50} (right)

The fuzzifier function for FCM [4, 2] is an exponential function with $f_{\text{FCM}}(u) = u^{\omega}$ and $1 < \omega \in \mathbb{R}$. In figure 1, the prototypes are represented as filled circles, their 'tails' represent the way the prototypes took from their initial- to their final location. The devastating effect of a high dimensional data set to FCM is obvious: the prototypes run straight into the centre of gravity of the data set, independently of their initial location and therefore, finding no clusters at all. NFCM [3] is one of the two algorithms considered in this paper that is able to detect noise. The fuzzifier function for NFCM is identical to FCM: $f_{\text{NFCM}} = f_{\text{FCM}}$. Apart from the fact that all data objects have the highest membership value for the noise cluster, the behaviour of the algorithm does not change compared to FCM. PFCM [9] is a mixture of HCM and FCM, as the definition of the fuzzifier function shows: $f_{\text{PFCM}}(u) = \frac{1-\beta}{1+\beta}u^2 + \frac{2\beta}{1+\beta}u$. This fuzzifier function creates an area of crisp membership values around a prototype while outside of these areas of crisp membership values, fuzzy values are assigned. The parameter β controls the size of the crisp areas: the low value of β means a small crisp area of membership values.

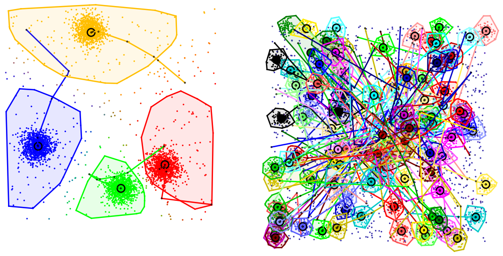

Fig. 2 PFCM, applied on T_2 (left) and T_{50} (right)

In the 2-dimensional example in Figure 2, the surrounded and slightly shaded areas represents the convex hull of all data objects with membership value 1. On the right hand side of the figure, it can be seen, that PFCM does not show the same ill behaviour as FCM and NFCM, PFCM finds approximately 90 out of 100 clusters.

PNFCM (presented along with PFCM in [9]) is the second algorithm considered in this paper that is able to detect noise. The fuzzifier function of PNFCM is again, identical to PFCM but the objective function is again modified by the noise cluster extension. If PNFCM is applied on T_{50}, the result is quite different to PFCM. Due to the specified noise distance, it is very likely that prototypes are initialized so far away from the nearest cluster, that all of its data objects are assigned crisp to the noise cluster. That is true for all prototypes which implies, that no cluster is found and all data objects are assigned as noise. The presented algorithms are in particular interesting because FCM produces useless results, HCM works with a lucky initializian, but their combination PFCM can be applied quite successfully. Since using a noise cluster has at least for PNFCM a negative effect, it is interesting to analyse its influence further.

We want to identify structural problems with FCM (and alike) on high dimensional data sets. Considering real data sets exemplary is not enough to draw general conclusions, therefore, we consider only one but for clustering optimal data set. Let $D = \{x_1, \ldots, x_c\} \subset \mathbb{R}^n$ be a data set that contains of $c > n$ clusters, with one data object per cluster. The clusters (data objects) in D are located on an n-dimensional hypersphere surface and are arranged so that the minimal pairwise distance is maximised. The general accepted definition of clustering is: the data set partitioning should be done in such a way, that data objects of the same cluster should be as similar as possible while data objects of different clusters should be as different as possible. D is a perfect data set for clustering because its clusters can be considered infinitely dense and maximally separated. There is one small limitation to that statement: c should not be extremely larger than n ($c < n!$) because the hypersphere surface might be too small for so many prototypes, but that limitation is usually not significant. Algorithms with problems on D will have even more problems on other data sets because there is no ’easier’ data set than D. Especially if more, high-dimensional problems occur like overlapping clusters or very unbalanced cluster sizes.

As the example in Figure 1-right has shown, the prototypes end up in the centre of gravity for FCM and NFCM. To gain knowledge why this behaviour occurs (and why not in the case of PFCM), the clustering algorithms are tested in a rather artificial way. The prototypes are all initialised in the centre of gravity (COG) and then moved iteratively towards the data objects by ignoring the update procedure indicated by the clustering algorithms. Let α control the location of the prototypes: $\alpha \in [0,1]$, $x_i \in \mathbb{R}^n$ the i’th data object with $cog(D) \in \mathbb{R}^n$ the centre of gravity of data set D implies $y_i : [0,1] \rightarrow \mathbb{R}^n$ with $y_i(\alpha) = \alpha \cdot x_i + (1-\alpha) \cdot cog(D)$ and finally: $d_{ij}(\alpha) = d(y_i(\alpha), x_j)$. Since the membership values are functions of the distance values and the objective function is a function of membership values and distance values, it can be plotted as a function of α.

3 The effect of high dimensions

In this section, the effects of the high dimensional data sets like D are presented. There are two effects on high dimensional data sets that have strong influence on the membership function: the number of prototypes and the number of dimensions. HCM is not farther analysed because its objective function values do not change due to the number of dimensions or the number of prototypes.

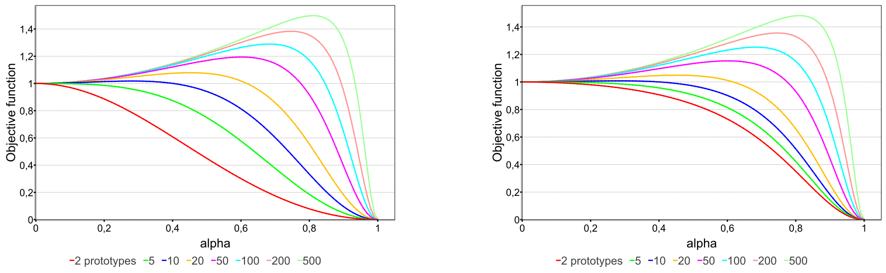

Fig. 3 Objective function plots for FCM (left) and NFCM (right)

However, for FCM and NFCM, these factors have a strong influence on the objective function. In Figure 3, the objective functions for these two algorithms are plotted for a variety of dimensions, depending on α. For convenience, the objective function values are normalized to 1 at $\alpha = 0$. The plots show a strong local maximum between $\alpha = 0.5$ and $\alpha = 0.9$. Winkler et.al showed in [12] that the number of dimensions effects the objective function by the height of this local maximum. The number of prototypes however influences the location of the maximum: the higher the number of prototypes, the further right the local maximum can be observed.

Since these are gradient descent algorithms, the prototypes will run into the centre of gravity if they are initialized left of the local maximum which is exactly what is presented in figure 1-right. Since the volume of an n-dimensional hypersphere increases exponentially with its radius, it is almost hopeless to initialize a prototype near enough to a cluster so that the prototype converges to that cluster. For this example, in 50 dimensions and with 100 prototypes, the converging hypershere radius is 0.3 times the feature space radius which means, the hypervolume is $7.2 \cdot 10^{-27}$ times the volume of the feature space.

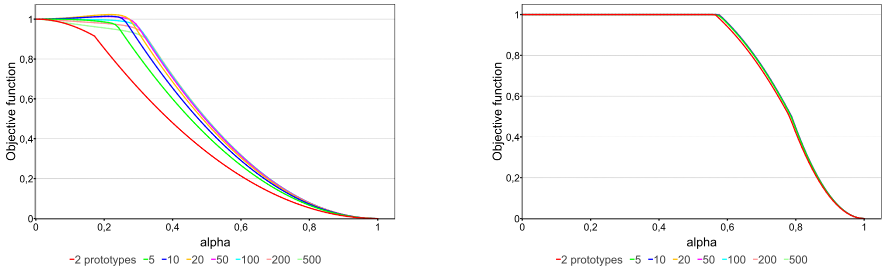

Fig. 4 Objective function plots for PFCM (left) and PNFCM (right)

As presented in Figure 4-left, PFCM does not create such a strong local maximum as FCM, also the local maximum that can be observed is very far left. That is the reason why PFCM can be successfully applied on a high dimensional data set. The situation is quite different for PNFCM, see 4-right. The fixed noise distance is chosen appropriate for the size of the clusters but the distance of the prototypes to the clusters is much larger. Therefore, all data objects have membership value 0 for the prototypes which explains the constant objection function value.

4 How to exploit the algorithm parameters to increase their effectiveness

As the section title indicates, it is possible to exploit the parameters ω and d_{noise} of FCM, NFCM and PNFCM to tune the algorithms so that they work on high dimensions. The term 'exploit' is used because the fuzzifier ω and the noise distance d_{noise} are chosen dimension dependent and not in order to represent the properties of the data set. Let $\omega = 1 + \frac{2}{n}$ and $d_{\text{noise}} = 0.5\log_2(D)$, the parameter β remained at 0.5. Setting the fuzzifier near 1, creates an almost crisp clustering and by setting the noise distance larger its effect is reduced. That way, FCM and NFCM become similar to HCM and PNFCM becomes similar to PFCM. The results of the adapted parameter are shown in Figure 5 for FCM, NFCM and PNFCM for D with 100 dimensions. The objective function plots are all very similar to PFCM which would imply that they work just as well.

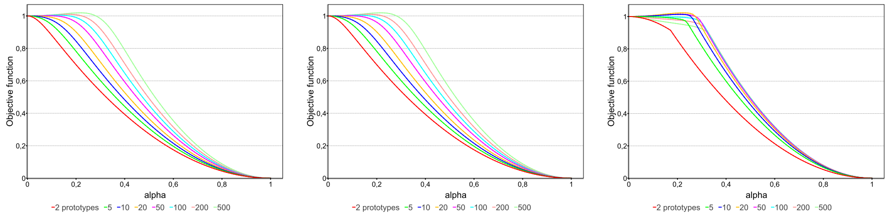

Fig. 5 Objective function plots for FCM (left), NFCM (middle) and PNFCM (right) with dimension dependent parameters

To test that, we apply each algorithm 100 times on T_{50}, the results are presented in Table 1 as mean and sample standard deviation in braces. The found clusters column is the most important, the other two are just for measuring the performance of recognizing noise data objects. The test clearly shows the improvement by adjusting the parameters according to the number of dimensions.

Algorithm	Found clusters		Correctly clustered noise		Incorrect clustered as noise	
HCM	42.35	(4.65)	0	(0)	0	(0)
FCM	0	(0)	0	(0)	0	(0)
NFCM	0	(0)	1000	(0)	10000	(0)
PFCM	90.38	(2.38)	0	(0)	0	(0)
PNFCM	0	(0)	1000	(0)	10000	(0)
Adjusted Parameter						
FCM AP	88.09	(3.58)	0	(0)	0	(0)
NFCM AP	88.5	(3.37)	999.77	(0.58)	1136.0	(344.82)
PNFCM AP	92.7	(2.67)	995.14	(3.12)	96.0	(115.69)

Table 1 Performance overview of T_{50} with 100 data objects for each cluster, 1000 noise data objects and each algorithm is applied performed 100 times. The mean value and (sample standard deviation) are displayed.

5 Conclusions

The two algorithms HCM and FCM do not work on high dimensions properly. It is very odd therefore that a combination of them in form of PFCM works quite well. We have shown that the reason for this effect is a very small local minimum of PFCM compared to FCM in the COG. We presented, that FCM, NFCM and PNFCM can be tuned in such a way that their objective function shows a similar behaviour in our test as PFCM, in which case the clustering result is similar on the test data set T_{50}. The question remains why this local minimum occurs. A possible explanation is presented in [1, 5] as they identify the effect of distance concentration as being the most problematic in having a meaningful nearest neighbour searches. Further work is needed here for a deeper understanding of the effect of distance concentration in relation to clustering. It sounds logical that a clustering algorithm, that is based on the spacial structure of the data, can not work well if all data objects from all points of view and for all (variance limited) data distributions seem to be equally distant. But that poses the question if clustering algorithms in general can produce meaningful results at on arbitrary high dimensional data sets without having some special features that reduces the complexity of the problem.

The knowledge, gained from the presented experiments is, that prototype based clustering algorithms seem to need a crisp component. But from the failure of HCM, it might as well be learned that only crisp assignments of data objects are not good enough. 'Almost crisp' clustering algorithms like the however, perform quite well, at least on T_{50}. That is no guaranty that they will work on high dimensional real data sets, but they are not as hopeless as FCM. However, this also means that the fuzzifier in case of FCM and NFCM as well as the noise distance in case of NFCM and PNFCM has to be used to counter the high dimensional effects. This is very unsatisfying as it prevents a suitable modelling of the data set and a better way would be to adapt the algorithm rather than exploiting its parameters.

References

1. Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is nearest neighbor meaningful? In *Database Theory - ICDT'99*, volume 1540 of *Lecture Notes in Computer Science*, pages 217–235. Springer Berlin / Heidelberg, 1999.
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. 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.
5. Robert J. Durrant and Ata Kabán. When is 'nearest neighbour' meaningful: A converse theorem and implications. *Journal of Complexity*, 25(4):385 – 397, 2008.
6. Hichem Frigui and Raghu Krishnapuram. A robust clustering algorithm based on competitive agglomeration and soft rejection of outliers. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:550, 1996.
7. Donald E. Gustafson and William C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *IEEE*, volume 17, pages 761–766, Jan. 1978.
8. F. Höppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis*. John Wiley & Sons, Chichester, England, 1999.
9. 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.
10. Rudolf Kruse, Christian Dring, and Marie-Jeanne Lesot. *Advances in Fuzzy Clustering and its Applications*, chapter Fundamentals of Fuzzy Clustering, pages 3–30. John Wiley & Sons, 2007. ISBN: 978-0-470-02760-8.
11. Hugo Steinhaus. Sur la division des corps materiels en parties. *Bull. Acad. Pol. Sci., Cl. III*, 4:801–804, 1957.
12. Roland Winkler, Frank Klawonn, and Rudolf Kruse. Fuzzy c-means in high dimensional spaces. International Journal of Fuzzy System Applications (to appear), 2011.