Proximal Optimization for Fuzzy Subspace Clustering

Arthur Guillon, Marie-Jeanne Lesot, Christophe Marsala, Nikhil R. Pal

Proximal Optimization for Fuzzy Subspace Clustering

Arthur Guillon[1], Marie-Jeanne Lesot[1], Christophe Marsala[1], and Nikhil R. Pal[2]

[1] Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 4 place Jussieu 75005 Paris, France
{Arthur.Guillon,Marie-Jeanne.Lesot,Christophe.Marsala}@lip6.fr
[2] Indian Statistical Institute, Calcutta 700 108, W. Bengal, India
nikhil@isical.ac.in

Abstract. This paper proposes a fuzzy partitioning subspace clustering algorithm that minimizes a variant of the FCM cost function with a weighted Euclidean distance and a penalty term. To this aim it considers the framework of proximal optimization. It establishes the expression of the proximal operator for the considered cost function and derives PFSCM, an algorithm combining proximal descent and alternate optimization. Experiments show the relevance of the proposed approach.

Keywords: Fuzzy partitioning clustering, subspace clustering, proximal descent

1 Introduction

Subspace clustering [1] is an unsupervised machine learning task that aims at partitioning data into groups with strong internal similarity and external dissimilarity (just as clustering) while also discovering the best subspaces to represent these clusters. The identified subspaces are required to be minimal, yet sufficient to describe the clusters they contain.

The definition of subspace clustering requires the identification of the clusters and of their subspaces to be simultaneous: indeed, if either clusters or their subspaces are known beforehand, the problem reduces to finding the subspaces or correct description of the clusters, respectively. In addition, as opposed to feature selection, different clusters are most of the time discovered in different subspaces.

As briefly sketched in Section 2, there exist several families of techniques and algorithms to solve the subspace clustering problem, as well as various representations of the subspaces, depending on the intended application of the subspace clustering.

This paper places itself in the partitioning paradigm in a fuzzy setting and produces clusters identified by a center. Moreover, it discovers axis-parallel subspaces, which are thus identified by weights on the original data features. An

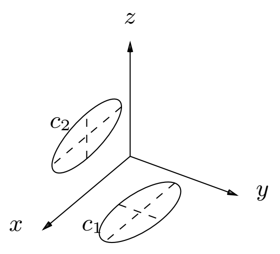

Fig. 1. Two clusters, contained in two different planes: c_1 in (x, y) and c_2 in (x, z).

original cost function formalises these concepts and adds, to a FCM cost function [3] with weighted Euclidean distance, a penalty term expressing constraints to identify the relevant subspaces.

As this penalty term is not differentiable, standard optimization techniques such as alternate optimization are not available. This paper introduces a novel optimization scheme, exploiting tools from the proximal descent theory [8]. The utilisation of such techniques is still relatively new in machine learning and in clustering in particular [9].

This paper proposes an innovative implementation of this theoretical paradigm in the fuzzy subspace clustering framework. It establishes a theorem giving the expression of the proximal operator allowing the optimization of the considered cost function. Finally, it proposes an algorithm, called PFSCM, standing for Proximal Fuzzy Subspace C-Means, using this result to solve the subspace clustering problem through the combination of proximal descent and alternate optimization.

This paper is structured as follows: in Section 2, related works and the scientific context of subspace clustering are summed up. A new cost function is presented and studied in Section 3. In Section 4, the implementation of proximal descent is studied to optimize the proposed function, leading to the update equation from which the PFSCM algorithm is derived. This algorithm is then experimentally validated in Section 5.

2 Related Works

Subspace clustering [1] can be seen as a combination of clustering and feature selection tasks, the latter being performed locally for each cluster. It aims at identifying both a data decomposition into homogeneous and distinct subgroups and the subspaces in which these clusters are defined. Figure 1 gives an example of such clusters, contained in axis-parallel subspaces: although the data are 3-dimensional, cluster c_1 actually lives in the plane $z = 0$ and cluster c_2 in the plane $y = 0$.

A large number of approaches to the subspace clustering problem have been explored in machine learning as well as in data mining or computer vision. A list can be found in [12]. This paper focuses on iterative partitioning techniques. The

k-subspace algorithm [13] generalises the k-means approach, alternating between the assignation of points to the clusters and the estimation of subspaces to fit these clusters. Witten & Tibshirani [14] propose a reformulation of the k-means minimization problem into a maximization problem with a weighted distance. An ℓ_1-based constraint is added in order to produce sparse weight vectors and identify the subspaces. Qiu et al. [10] adapt this framework to fuzzy clustering and compare it to some usual subspace clustering algorithms. Both the crisp and fuzzy variants of these algorithms heavily modify the original k-means function to formulate a maximization problem with a ℓ_1-regularization term, in order to identify minimal subspaces.

Closer to the original k-means paradigm, the fuzzy c-means clustering algorithm [3] has been adapted to the context of subspace clustering. Keller & Klawonn [6] adapt the FCM cost function to use a weighted Euclidean distance. Denoting $(x_i)_{i=1}^n \in \mathbb{R}^d$ the datapoints of dimension d, c the number of clusters, $(u_{ri}) \in [0,1]$ for $i \in \{1, \ldots, n\}$ and $r \in \{1, \ldots, c\}$ the fuzzy membership degree of x_i to cluster C_r, $\mu_r \in \mathbb{R}^d$ the center of cluster C_r and $(w_{rj}) \in [0,1]$ the weight of dimension j for cluster C_r, they study the following cost function:

$$J_{K\&K}(C, U, W) = \sum_{r=1}^{c} \sum_{i=1}^{n} u_{ri}^m \sum_{j=1}^{d} w_{rj}^v (x_{ij} - \mu_{rj})^2 \tag{1}$$

where $m, v \in \mathbb{R}$ are fuzzifiers which can be tuned by the user to specify the level of fuzziness of the corresponding parameters and C, U, W are respectively the matrices containing the centers (μ_r), the memberships (u_{ri}) and the weights (w_{rj}). The function is minimized under the following constraints:

- (C1) $\forall i \in \{1, \ldots, n\}, \sum\limits_{r=1}^{c} u_{ri} = 1$ and (C2) $\forall r \in \{1, \ldots, c\}, \sum\limits_{i=1}^{n} u_{ri} > 0$;
- (C3) $\forall r \in \{1, \ldots, c\}, \sum\limits_{j=1}^{d} w_{rj} = a \neq 0$.

The first two constraints $(C1)$ and $(C2)$ are similar to the FCM ones. Constraint $(C3)$ on the weights (w_{rj}), where a is a user-defined parameter, is specific to the subspace clustering problem and prevents the trivial solution such that $\forall r, \forall j, w_{rj} = 0$. The minimization of Equation (1) under these constraints produces a solution to the fuzzy subspace clustering problem. The computed weights (w_{rj}) indicate how close the points assigned to C_r are in dimension j. Figure 1 illustrates the relation between Equation (1) and subspace clustering: cluster c_1 lies in the (x, y) plane. In the z dimension, its points are very close to its center; therefore, minimizing $J_{K\&K}$ amounts to maximizing w_{1z} rather than w_{1x} and w_{1y}.

Borgelt [4] generalises Keller and Klawonn's work and proposes to slightly change the weights, so that the algorithm completely selects dimensions by attributing a null weight to others. He introduces the following cost function, where the terms u_{ri}^m and w_{rj}^v are replaced with general fuzzification functions g and h [7], which are supposed to be convex and differentiable on the [0,1] interval:

$$J_B(C,U,W) = \sum_{r=1}^{c}\sum_{i=1}^{n} h(u_{ri}) \sum_{j=1}^{d} g(w_{rj})(x_{ij}-\mu_{rj})^2 \tag{2}$$

This function is optimized under the same constraints as $J_{K\&K}$. Experimental results on artificial data show that Borgelt's algorithm better selects subspaces.

Both Keller & Klawonn and Borgelt functions are differentiable in each parameter on the considered domains, which allows to retain the technical framework of fuzzy c-means alternate optimization. They derive their algorithms from the corresponding cost function through the usual Lagrangian technique and obtain three update equations for parameters C, U and W.

3 Proposed Cost Function for Fuzzy Subspace Clustering

In this section, a new cost function is introduced to model the subspace clustering problem and a study of its properties is conducted.

3.1 A Weighted Fuzzy c-Means Function

Using the same notations as in Section 2, we propose the following cost function:

$$J(C,U,W) = \sum_{r=1}^{c}\sum_{i=1}^{n} u_{ri}^{m} \sum_{j=1}^{d} w_{rj}^{2}(x_{ij}-\mu_{rj})^2 + \gamma \sum_{r=1}^{c} \Big| \sum_{j=1}^{d}(w_{rj}) - \alpha \Big| \tag{3}$$

under the classic FCM constraints (C1) and (C2).

The first term is the same as Keller & Klawonn's cost function, except for the weight fuzzifier v which is set to 2, in order to simplify further mathematical analysis of the function: it corresponds to a FCM cost function with a locally weighted Euclidean distance.

The second term adds a cost to the function which prevents the sum of the weights of each cluster C_r from being too far from the user-defined parameter α which plays the same role as a in Keller & Klawonn's (C3) constraint: for $\alpha \neq 0$, it prevents the trivial solution $W = 0$. The user-defined parameter $\gamma \in \mathbb{R}$ is used to balance out the two terms: it only needs to be large enough to penalize trivial solutions. This term can also be interpreted as an "inlined" constraint that incorporates constraint (C3), which does not need to be optimized through the particular Lagrangian method, but rather allows the use of new optimization techniques.

The cost function J in Equation (3) thus conveys the idea of finding a solution to the subspace clustering problem with a relaxed constraint, inspired by ℓ_1-regularization [11].

3.2 Minimization of the Cost Function

Using the cost function J, solving the subspace clustering problem amounts to finding the parameters (C^*, U^*, W^*) that minimize J. This function can be decomposed as follows:

$$J(C,U,W) = F(C,U,W) + \gamma G(W)$$
$$\text{where } F(C,U,W) = \sum_{r=1}^{c}\sum_{i=1}^{n} u_{ri}^m \sum_{j=1}^{d} w_{rj}^2 (x_{ij} - \mu_{rj})^2$$
$$G(W) = \sum_{r=1}^{c} G_r(W_r) = \sum_{r=1}^{c} \Big| \sum_{j=1}^{d} (w_{rj}) - \alpha \Big|$$

The function J verifies several properties of interest, which motivate and validate the technique used in the next section. First, J is a convex function of W, as it is the sum of convex functions.

F is differentiable in all three parameters and it can be shown that its gradient is Lipschitz-continuous for fixed C and U. These properties guarantee good performances of well-known optimization algorithms, such as gradient descent.

For fixed W, minimizing J under constraints is equivalent to minimizing F. As for fuzzy c-means, this can be done through alternate optimization. From the above function and constraints, the two classic update equations for membership degree and cluster centers are derived:

$$u_{ri} = \frac{d_{ri}^{\frac{2}{1-m}}}{\sum_{s=1}^{c} d_{si}^{\frac{2}{1-m}}} \quad \text{where} \quad d_{ri}^2 = \sum_{j=1}^{d} w_{rj}^2 (x_{ij} - \mu_{rj})^2 \tag{4}$$

$$\text{and } \mu_{rj} = \frac{\sum_{i=1}^{n} u_{ri}^m \cdot x_{ij}}{\sum_{i=1}^{n} u_{ri}^m} \tag{5}$$

These two equations are used in the PFSCM algorithm described in Section 4 to update the terms u_{ri} and μ_r in order to find the minimum of J.

Function G is convex but not differentiable in the variable W, which prevents the derivation of an update term for weight optimization and motivates the use of proximal descent, proposed in the next section.

4 Proximal Descent for Weight Optimization

As it is not differentiable everywhere, the function J previously defined cannot be optimized by classic alternate optimization. This section proposes a new algorithm, PFSCM (which stands for Proximal Fuzzy Subspace C-Means), based on an advanced technique of convex optimization: proximal descent [9].

In this section, the parameter of interest is the matrix of weights W, while C and U are fixed. Therefore, $J(C, U, W)$ is noted $J(W)$ for the sake of simplicity.

4.1 Proximal Descent

The cost function has the form $J(W) = F(W) + \gamma G(W)$, where both functions are convex but only F is differentiable and classic optimization techniques thus cannot be applied. As this general form of function has gained interest in the machine learning community (for example, when the second function G is a regularization term), proximal descent has been studied as an alternative to these techniques [2].

When $\gamma = 0$, usual optimization techniques would suggest to seek for the minimum of F (0 in the particular case of Equation (3)) by iterating some update equation. For example, gradient descent considers a general equation of the form $W^{t+1} = W^t - \eta \cdot \nabla F(W^t)$, where t is the iteration index and η is a descent step size. This simple optimization scheme provides an iterative algorithm in order to minimize any convex function F, starting from any W^0 and iterating until convergence.

As the function G is not differentiable, its gradient ∇G does not exist for each W^t. Proximal descent enriches gradient descent in the following way:

$$W^{t+1} = \operatorname{prox}_{\frac{\gamma}{L}G}\left(W^t - \frac{1}{L}\nabla F(W^t)\right) \tag{6}$$

$$\text{where } \operatorname{prox}_{\frac{\gamma}{L}G}(W') = \operatorname*{argmin}_W \left\{\frac{1}{2}\|W - W'\|^2 + \frac{\gamma}{L}G(W)\right\} \tag{7}$$

where $L > 0$ is a descent step size, similar to η. That is, in order to solve a global minimization problem, proximal descent solves a minimization problem as defined by Equation (7) at each step of the iteration.

Proximal descent can be understood as a technique of separating the descent in two phases: first for the function F, then for G. Such a descent scheme is also known as the "forward-backward" algorithm. In order to solve Equation (7), proximal descent approximates F around the current point of the iterative descent, W^t:

$$\begin{aligned}
&\operatorname*{argmin}_W \left\{F(W^t) + \left\langle \nabla F(W^t), W - W^t \right\rangle + \gamma G(W) + \frac{L}{2}\left\|W - W^t\right\|^2\right\} \\
&= \operatorname*{argmin}_W \left\{\frac{1}{2}\left\|W - (W^t - \frac{1}{L}\nabla F(W^t))\right\|^2 + \frac{\gamma}{L}G(W)\right\}
\end{aligned}$$

Here again, if $\gamma = 0$, this problem has a simple solution: gradient descent scheme $W^{t+1} = W^t - \frac{1}{L}\nabla F(W^t)$, hence the scheme given in Equation (6).

The key ingredient to efficiently implement the descent scheme defined by Equation (6) is the notion of proximal operator: it provides a closed-form expression to the optimization problem defined by Equation (7), which is often counter-intuitive, yet simple to implement.

4.2 Efficient Weight Optimization with Proximal Operators

We establish in the following theorem a proximal operator for the penalty term $G(W) = \gamma \sum_{r=1}^{c} \left| \sum_{j=1}^{d} (w_{rj}) - \alpha \right|$. Let K be the vector $(1,\ 1, \dots 1) \in \mathbb{R}^{1 \times d}$, such that $K \cdot K^{\intercal} = d$.

Theorem 1. *Let* $G_r(W_r) = |\sum_{j=1}^{d} (w_{rj}) - \alpha|$ *and* $L \in \mathbb{R}$.

$$\mathrm{prox}_{\frac{\gamma}{L} G_r}(W_r) = W_r + \frac{1}{d} K^{\intercal} \cdot \left(\alpha + \mathrm{prox}_{\frac{\gamma d}{L}|\cdot|}(K \cdot W_r - \alpha) - K \cdot W_r \right) \tag{8}$$

where $\mathrm{prox}_{\lambda|\cdot|}(x) = \mathrm{sign}(x) \max(|x| - \lambda, 0)$.

Moreover, $\mathrm{prox}_{\frac{\gamma}{L} G}(W) = \left(\mathrm{prox}_{\frac{\gamma}{L} G_r}(W_r) \right)_{r=1 \dots c} \in \mathbb{R}^{d \times c}$.

Proof. The proof uses results from [5] and [9]. First, $G_r(W_r) = \phi(K \cdot W_r)$ where $\phi(x) = |x - \alpha|$. Using the translation and semi-orthogonal linear transform properties [5]:

$$\begin{aligned} \mathrm{prox}_{G_r}(W_r) &= W_r + \frac{1}{d} K^{\intercal} \cdot \left(\mathrm{prox}_{\phi}(K \cdot W_r) - K \cdot W_r \right) \\ &= W_r + \frac{1}{d} K^{\intercal} \cdot \left(\alpha + \mathrm{prox}_{d|\cdot|}(K \cdot W_r - \alpha) - K \cdot W_r \right) \end{aligned}$$

Hence the expression of $\mathrm{prox}_{\frac{\gamma}{L} G_r}$ by the postcomposition property [9]. Finally, $\mathrm{prox}_{\frac{\gamma}{L} G}$ is computed using the separable sum property of proximal operators [9]. □

Equation (8) gives the expression of a proximal operator for the G function which can be used to efficiently implement the scheme defined in Equation (6) to update the current estimation of W.

As for gradient descent, the choice of L matters for the actual convergence of the descent, as well as for its speed. We observe that setting $L = \mathrm{trace}(H^{-1})$ yields good results, where H is the Hessian matrix of F (as a function of W). As F is simple enough, H is a diagonal matrix and does not depend on W.

4.3 A Fuzzy Subspace Algorithm: PFSCM

Using the previous mathematical results, we propose the PFSCM algorithm for fuzzy subspace clustering (see Algorithm 1). PFSCM combines alternate optimization of k-means-style algorithms for differentiable parameters with proximal descent for the optimization of the weights.

Initialization is a typical issue of k-means-like algorithms. In this paper, initial centers are randomly chosen and each cluster receives uniform weights for all dimension. As most partitioning algorithms, the number c of clusters to identify must be set by the user, as well as constants $\gamma > 0$ and $\alpha > 0$.

```
Data: $X$: data matrix
Parameters: $c$,$\gamma$,$\alpha$: numbers;
Variables: $\mu$, U, W: arrays;
           W$_{last}$: array
Initialization: $W_r \leftarrow (1,\ 1, \ldots\ 1)$ for each $C_r$;
                $\mu \leftarrow$ random centers
Output: $\mu$, U, W
repeat
    repeat
        Update U according to Equation (4);
        Update $\mu$ according to Equation (5)
    until convergence($\mu$, U);
    repeat
        Update W according to Equation (7)
    until convergence(W);
    W$_{last} \longleftarrow$ W
until convergence(W$_{last}$);
Update U and $\mu$ one last time.
```

Algorithm 1: The proximal fuzzy subspace clustering PFSCM algorithm

The algorithm then iterates the update of all three parameters U, μ and W, much like alternate optimization in k-means algorithm. It consists of two alternate inner loops: the regular parameters μ and U are optimized separately from W, which requires the special optimization procedure described in the previous subsection. Parameters μ and U are optimized one last time at the end of the algorithm, in order to guarantee that the result takes the final computed weights into account.

The convergence criteria are defined as the distance between the current and the previous values of the parameters being optimized. In particular, convergence for (μ, U) is defined as $\|\mu_t - \mu_{t+1}\|_2 < \varepsilon \vee \|U_t - U_{t+1}\|_2 < \varepsilon$.

PFSCM outputs U, C and W. In order to exploit the result of the algorithm, it may be of interest to extract the dimension associated to each cluster. To that aim we propose to post-process the matrix W using an additional parameter *cut* to cut out the irrelevant dimensions in a simple fashion: a dimension j for a cluster C_r is considered relevant if $w_{rj} > cut$.

5 Experimental Study

The proposed PFSCM algorithm has been tested on artificial data in order to study its ability to correctly identify centers of non-circular clusters, as well as the dimensions that are relevant to describe the clusters. The results show the effectiveness of PFSCM in detecting the clusters and their subspaces. Moreover, PFSCM is compared to Keller & Klawonn's algorithm [6] and shows to provide a better estimation of the dimensionality of the subspaces.

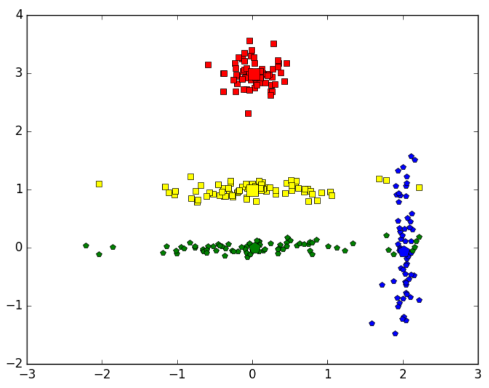

Fig. 2. Clustering example in two dimensions

	Red cluster		Yellow cluster		Green cluster		Blue cluster	
	w_1	w_2	w_1	w_2	w_1	w_2	w_1	w_2
Weights	0.528	0.472	0.063	0.937	0.027	0.973	0.964	0.036

Table 1. Computed weights for the example given in Figure 2. Column w_1 (resp. w_2) denotes the weight associated to the x-axis (resp. y-axis).

5.1 Illustrative Example

This subsection presents an illustrative experiment in $d = 2$ dimensions, similar to the example given in Keller & Klawonn [6] and graphically represented in Figure 2: four clusters are generated, one of them (the top red one in Figure 2) being circular while the others have a very low variance in one dimension. PFSCM is run with $c = 4$, $m = 2$, $\alpha = 1$ and $\gamma = 1000$.

In Figure 2 the points are colored according to the cluster C_r for which u_{ri} is maximum and Table 1 presents the weights computed for each dimension and cluster. It can be observed that PFSCM correctly identifies the desired clusters and their dimensions: the two weights (w_1, w_2) found for the circular cluster are similar, whereas the horizontal (respectively vertical) clusters verify $w_2 \gg w_1$ (respectively $w_1 \gg w_2$).

It is worth noting that, for this specific instance, some points close to the blue cluster are assigned to one of the horizontal clusters, as it minimizes the cost function. This kind of inliers is frequent in subspace clustering problems, and naturally leads to the use of fuzzy membership values (u_{ri}). In a similar fashion, moving the vertical cluster towards the center of the whole figure leads to the "stealing" of some points of the red cluster by the blue one. However, it can be observed that the identified dimensions for the circular cluster stay relatively stable (both $w_{rj} > cut = 0.2$), failing to recognize a non-flat cluster only 4 times out of 100 in the specific situation where all generated centers are vertically aligned.

5.2 Experimental Protocol

Considered Data In order to validate PFSCM, the previous experiment is generalized to higher dimensions, more precisely to artificial data of dimension $d \in \{5, 7, 9, 11, 13, 15\}$. For each experiment, $k = 4$ centers $c_1, \ldots, c_4$ are generated randomly in the hypercube $[-3{,}3]^d$ with a minimum (Euclidean) distance of 0.3 between the centers. Then, d_r dimensions $j_1, \ldots j_{d_r}$ are randomly picked, with d_r randomly chosen between 1 and $d-3$. Dimensions $j_1, \cdots, j_{d_r}$ are thereafter called the "relevant dimensions" for cluster C_r.

For each cluster, 100 points are generated according to a Gaussian distribution, with variance $v < 0.1$ for dimensions $j_1, \ldots, j_{d_r}$ and $v \in [0.5{,}0.9]$ for other dimensions. The generated points in cluster r in dimension j thus follow $X_r \sim \mathcal{N}(c_r, v_j)$.

Algorithm Parameters Keller & Klawonn's algorithm is initialized with FCM centers and uses $m = v = 2$, $a = 1$ and $c = 4$. PFSCM is ran with $m = 2$, $\alpha = 1$, $\gamma = 1000$ and $c = 4$. Both algorithms use the same convergence criterion, with $\varepsilon = 10^{-4}$.

The parameter *cut* is set to $\frac{1}{2d}$, which is a simple rule of thumb to identify the dimensions selected as relevant by the algorithms in each considered dimension d.

Quality Criteria Both algorithms are evaluated on three metrics in order to qualify their results and their ability to discover the desired clusters and subspaces, and their dimensions.

First, let $\delta = \sum_{r=1}^{4} \|\mu_r - c_r\|_2$ be the sum of the Euclidean distances between the generated centers and the computed ones (μ_r): this metric is a standard quality criterion for evaluating the produced clusters. A low value means that the computed centers are close to the original ones.

We also consider θ defined as the percentage of clusters for which all relevant dimensions are correctly identified by the algorithm: the relevant dimensions are correctly identified if $w_{rj} > cut \Leftrightarrow j \in \{j_1, \cdots j_{d_r}\}$.

Finally, for the clusters for which the relevant dimensions have been correctly identified, let the weight ratio $\phi = \frac{\omega_1}{\omega_{j_{d_r}}}$ where ω_1 is the largest computed weight and $\omega_{j_{d_r}}$ the smallest computed weight for the relevant dimensions. This metric computes the distortion of the cluster between the relevant dimensions, as estimated by both algorithms.

5.3 Experimental Results

The results of the experiment are presented in Table 2 in the form of the means and standard deviations of the three criteria, computed over 100 runs of each

	d	δ		ϕ		θ
		Mean	SD	Mean	SD	%
PFSCM	5	0.90	0.67	2.51	1.41	76
	7	0.98	0.81	3.08	1.72	79
	9	0.90	0.50	3.96	2.09	80
	11	0.88	0.33	4.35	2.01	83
	13	0.97	0.40	4.78	1.99	83
	15	0.90	0.10	5.22	1.84	91
K&K	5	1.27	1.03	2.61	1.78	43
	7	1.55	1.38	3.12	2.29	39
	9	1.39	1.18	4.05	3.01	31
	11	1.26	0.90	4.50	3.48	28
	13	1.42	1.29	4.68	3.60	25
	15	1.21	1.06	8.05	3.27	10

Table 2. Comparison between PFSCM and Keller & Klawonn's algorithm

algorithm. Both algorithms sometimes produce bad results, identifying centers too far from the generated ones, which distort the means and standard deviations of the previous metrics. Such outliers (less than 2% of the runs) have been cleaned out of the results.

It can be observed that PFSCM correctly identifies the generated centers (as shown by δ), and produces stable results in each dimension, as shown by the low standard deviation. Moreover the algorithm, along with the proposed cut ratio $cut = \frac{1}{2d}$, performs well in selecting the relevant dimensions of the subspaces (as shown by θ). Finally, the weights ratio ϕ is relatively stable when the number of dimensions increases.

Keller & Klawonn's algorithm correctly identifies the centers (c_r) and the difference with PFSCM is not meaningful. However it appears to miss out some relevant dimensions of the generated subspaces. This is a general feature of the algorithm, which can be seen in [6] as well: although the most relevant dimension is almost always identified, Keller & Klawonn's algorithm gives a much smaller weight to the other relevant dimensions, which is also shown by the larger mean for ϕ. This feature can be modulated by tuning the value of the fuzzifier v, but then this modification affects the weights of each dimension, including the most relevant one.

In summary, PFSCM identifies the same clusters as Keller & Klawonn's algorithm, but produces a better estimation of the dimensions of the subspaces. It is also more regular when the dimension d increases.

6 Conclusion and Future Works

This paper introduces a new approach to solve the fuzzy subspace clustering problem with a cost function involving non-differentiable terms. Advanced optimization techniques are explored, which replace the standard update equations of fuzzy c-means-like algorithms.

Experiments on synthetic data show the relevance of the proposed approach, that appears to correctly identify all the relevant dimensions and not more, whereas Keller & Klawonn's algorithm tends to underestimate the number of relevant dimensions. This provides more information about the importance of each dimension for the subspaces and clusters.

Future works will aim at generalizing this approach around the same key ideas: a differentiable function matching the specification of the problem and one or several penalty functions, expressing constraints on the shape of the solution. The introduction of regularization terms for parameters other than W will also be studied. Finally, more efficient descent schemes will be considered, in order to speed up the descent.

References

1. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications. In: Proc. of the ACM SIGMOD Int. Conf. on Management of Data. pp. 94–105. ACM (1998)
2. Bach, F., Jenatton, R., Mairal, J., Obozinski, G., et al.: Convex optimization with sparsity-inducing norms. Optimization for Machine Learning pp. 19–53 (2012)
3. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Kluwer Academic Publishers, Norwell, MA, USA (1981)
4. Borgelt, C.: Fuzzy subspace clustering. In: Advances in Data Analysis, Data Handling and Business Intelligence, pp. 93–103. Studies in Classification, Data Analysis, and Knowledge Organization, Springer (2010)
5. Combettes, P.L., Pesquet, J.C.: Proximal splitting methods in signal processing. In: Fixed-point algorithms for inverse problems in science and engineering, pp. 185–212. Springer (2011)
6. Keller, A., Klawonn, F.: Fuzzy clustering with weighting of data variables. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 8(06), 735–746 (2000)
7. Klawonn, F., Höppner, F.: What is fuzzy about fuzzy clustering? understanding and improving the concept of the fuzzifier. In: Advances in Intelligent Data Analysis V, pp. 254–264. Springer (2003)
8. Moreau, J.J.: Fonctions convexes duales et points proximaux dans un espace hilbertien. CR Acad. Sci. Paris Sér. A Math 255, 2897–2899 (1962)
9. Parikh, N., Boyd, S.: Proximal algorithms. Foundations and trends in optimization 1(3), 123–231 (2013)
10. Qiu, X., Qiu, Y., Feng, G., Li, P.: A sparse fuzzy c-means algorithm based on sparse clustering framework. Neurocomputing 157, 290–295 (2015)
11. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) pp. 267–288 (1996)
12. Vidal, R.: A tutorial on subspace clustering. IEEE Signal Processing Magazine 28(2), 52–68 (2010)
13. Wang, D., Ding, C., Li, T.: K-subspace clustering. In: Machine learning and knowledge discovery in databases, pp. 506–521. Springer (2009)
14. Witten, D.M., Tibshirani, R.: A framework for feature selection in clustering. Journal of the American Statistical Association 105, 713–726 (2010)