Multi-Prototypes Convex Merging Based K-Means Clustering Algorithm

Dong Li, Shuisheng Zhou, Tieyong Zeng, and Raymond H. Chan.

Abstract—K-Means algorithm is a popular clustering method. However, it has two limitations: 1) it gets stuck easily in spurious local minima, and 2) the number of clusters k has to be given a priori. To solve these two issues, a multi-prototypes convex merging based K-Means clustering algorithm (MCKM) is presented. First, based on the structure of the spurious local minima of the K-Means problem, a multi-prototypes sampling (MPS) is designed to select the appropriate number of multi-prototypes for data with arbitrary shapes. Then, a merging technique, called convex merging (CM), merges the multi-prototypes to get a better local minima without k being given a priori. Specifically, CM can obtain the optimal merging and estimate the correct k. By integrating these two techniques with K-Means algorithm, the proposed MCKM is an efficient and explainable clustering algorithm for escaping the undesirable local minima of K-Means problem without given k first. Two theoretical proofs are given to guarantee that the cost of MCKM (MPS+CM) can achieve a constant factor approximation to the optimal cost of the K-Means problem. Experimental results performed on synthetic and real-world data sets have verified the effectiveness of the proposed algorithm.

Index Terms—K-Means, multi-prototypes, multi-prototypes sampling, convex merging.

1 Introduction

Clustering analysis is one of the important branches in machine learning [1], [2], which has extensive applications in different fields, for example, artificial intelligence [3], pattern recognition [4], image processing [5], etc. The goal of the clustering algorithm is to separate a data set into multiple clusters so that the objects in the same cluster are highly similar. Many types of clustering algorithms have been studied in the literature, see [6] and the references therein.

As a popular clustering paradigm, partition-based methods believe that data set can be represented by cluster prototypes. They require one to specify the number of clusters k a priori and update the clusters by optimizing some objective functions. The most representative partition-based clustering algorithms is the K-Means algorithm [7], [8], which aims to divide the data set into k clusters so that the sum of squared distances between each sample to its corresponding cluster center is the smallest. However, because the K-Means algorithm is NP-hard, it easily gets stuck in spurious local minima [9], [10]. Besides, k has to be given first.

To avoid bad local minima in the K-Means algorithm, numerous remedies have been proposed. Most of them can be classified into three strategies. The first strategy focuses on initialization selection. Pena *et al.* [11] concluded that the quality of the solution and running time of the K-Means algorithm highly depends on the initialization techniques. A good initialization can find better local minima or even global minima. K-Means++ [12] was proposed to initialize K-Means by choosing the centers with specific probabilities, and the result is $\mathcal{O}(\log k)$-competitive with the optimal result. K-Means|| [13] was presented to obtain a nearly optimal result by an over-sampling technique after a logarithmic number of iterations. An improved K-Means++ with local search [14] was developed to achieve a constant approximation guarantee to the global minima with $\mathcal{O}(k \log \log k)$ local search steps.

The second strategy focuses on theoretical innovations in the model frameworks. A relaxation method for K-Means [15] was designed to construct the objective of K-Means into the so-called 0-1 semidefinite programming (SDP), and solve it by the linear programming and SDP relaxations. Then, a feasible solution is obtained by principal component analysis. Experimental results show that the 0-1 SDP for K-Means always find a global minima for $k = 2$ ([16] also summarized similar results). Coordinate descent method for K-Means [10] was provided to get better local minima by reformulating the objective of K-Means as a trace maximization problem and solving it with coordinate descent scheme.

The third strategy focuses on adjustment to local minima based on various heuristics and empirical observations. Usually, the adjustment scheme is the splitting and merging of prototypes [17], [18], [19], [20], [21].

All the methods above have achieved better local minima or global minima on the data sets with subclusters that are uniform size and linearly separable from each other. This is not surprising as K-Means-type algorithms often produce clusters of relatively uniform size, even if the data sets have varied cluster sizes. This is called the ”uniform effect” [22].

- *D. Li, S. Zhou are with School of Mathematics and Statistics, Xidian University, Xi'an 710071, China (E-mail: lidong_xidian@foxmail.com; sszhou@mail.xidian.edu.cn).*
- *T. Zeng is with the Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong. E-mail: zeng@math.cuhk.edu.hk*
- *R.H. Chan is with the Department of Mathematics, City University of Hong Kong, 83 Tat Chee Ave, Hong Kong, and with the Hong Kong Centre for Cerebro-Cardiovascular Health Engineering, 19 W Ave, Science Park, Hong Kong. E-mail: raymond.chan@cityu.edu.hk*

Manuscript received xxxx, 2022; revised xxxx, 2022. This work was supported by the National Natural Science Foundation of China under Grants No. 61772020; HKRGC Grants Nos. CUHK14301718, CityU11301120, and C1013-21GF; and CityU Grant 9380101 (Corresponding author: Shuisheng Zhou.)

The Euclidean distance squared error criterion of K-Means-type algorithms therefore tends to work well on the data sets with subclusters that are uniform size and linearly separable from each other. This limits the performance of the algorithms on the data sets with special patterns, such as the data sets with subclusters that are non-uniform, with subclusters that are non-convex, and with subclusters that are skewed-distributed, etc.

To address the aforementioned problem, the over-parametrization learning framework, as a promising and empirical approach, has been applied to the clustering algorithms [23], [24], [25], [26], [27]. Graph-based Multi-prototype Competitive Learning (GMPCL) [28] was proposed to first obtain the coarse clustering by a graph-based method, and then refine these clusters by multiprototype competitive learning for the non-linearly separable data. Self-adaptive Multiprototype-based Competitive Learning (SMCL) [29] was developed for imbalanced data clustering. SMCL first selects multi-prototypes in an adaptive way, and then merges the prototypes based on a new separation metrics. Finally, the best number of clusters and clustering result were determined using a new internal clustering metrics. Concept Factorization with Local Centroids [30] was proposed with the aim of capturing the manifold structure of data by introducing multi-prototypes, and formulating the clustering problem as a bipartite graph partitioning task. Overall, these multi-prototypes clustering algorithms share a common procedure, which involves generating multi-prototypes that are more suited for modeling clusters with arbitrary shape and size compared to a single prototype, and merging the prototypes into a given cluster number based on some similarity measures.

However, most existing multi-prototypes methods simply use a predefined number of multi-prototypes and the selection skills lack theoretical guarantees. Therefore, a convex clustering model was introduced in [31] to overcome these two issues. The model is formulated as a convex optimization problem based on the over-parametrization and sum-of-norms (SON) regularization techniques. There are other variants of convex clustering models, see [32], [33] and the references therein. In general, convex clustering is solved by the alternating minimization algorithm (AMA) and the alternating direction method of multipliers (ADMM) [34].

In convex clustering models, the number of over-parametrization is set to the number of samples, and then the samples are classified into different clusters by tuning the regularization parameter. Inevitably, its computational complexity is very high, where each iteration of the ADMM solver is of complexity $\mathcal{O}(n^2p)$. Here, n is the number of samples and p is the dimensionality of the samples. Recently, a novel optimization method, called the semismooth Newton-CG augmented Lagrangian method [35], was proposed to solve the large-scale problem for convex clustering. We emphasize that since these clustering models are convex, there are theoretical guarantee to recover their global minima.

The above approaches rarely analyzed the structures of the local minima, so there is a lack of explanation and understanding of the approaches. Recently, Qian *et al.* [36] investigated the structures under a probabilistic generative model and proved that there are only two types of spurious local minima of K-Means problem under a separation condition. More precisely, all spurious local minima can only be of two structures: (i) the multiple prototypes lie in a true cluster, and (ii) one prototype is put in the centroid of multiple true clusters. In this paper, these two structures are called over-refinement and under-refinement of the true clusters, respectively. Naturally, to get better local minima or global minima, we should refine the prototypes such that one prototype lies in one true cluster. This inspires us to explore an efficient and explainable approach for finding better local minima.

Another line of research focuses on the cluster number k. Most methods require k to be given a priori. In general, k is unknown. Clustering by passing messages between data points [37], called affinity propagation (AP), is designed to generate high-quality clusters without given k, by iteratively exchanging valuable messages between samples. The similarity matrix between pairs of samples is used as input. By adding an entropy penalty term to K-Means to adjust the bias, *unsupervised* K-Means clustering algorithm (U-K-Means) [38] can automatically find the optimal k without giving any initialization and parameter selection. An over-parametrization learning procedure is established to estimate the correct k in K-Means for the arbitrary shape data sets [29], [39]. We remark that the convex clustering models mentioned above, e.g., [31], can also find the correct k by tuning the regularization parameter and the number of neighboring samples.

In this paper, we propose an efficient and explainable multi-prototypes K-Means clustering algorithm for recovering better local minima without the cluster number k being given a priori. It is called **MCKM** (**m**ulti-prototypes **c**onvex merging based **K-M**eans clustering algorithm). It has two steps. The first step is guided by the aim that the final structure of the minima should have (i) at least one or more prototypes located in a true cluster, and (ii) no prototypes are at the centroid of multiple true clusters. Along this line, an over-parametrization selection technique, called multi-prototypes sampling (MPS), is put forward to select the appropriate number of multi-prototypes. Then in the second step, a merging technique, called convex merging (CM), is developed to get better local minima without k being given.

The main contributions of this paper are as follows:

1) An appropriate number of multi-prototypes can be selected by MPS to adapt to data with arbitrary shapes.
2) CM can get better local minima without k being given. It obtains the optimal merging and estimates the correct k, because it treats the merging task as a convex optimization problem.
3) The combined method **MCKM** is an efficient and explainable K-Means algorithm that can escape the undesirable local minima without given k. Two theoretical proofs are given to guarantee that the cost of MCKM (MPS+CM) can achieve a constant factor approximation to the optimal cost of the K-Means problem.
4) Experiments on synthetic and real-world data sets illustrate that MCKM outperforms the state-of-the-art algorithms in approximating the global minima of K-Means, and accurately evaluates the correct k. In addition, MCKM excels in computational time.

The paper is organized as follows. Section 2 reviews the related works. The research motivation is described in Section 3 and the new algorithm is presented in Section 4. The experimental results with discussion are reported in Section 5 and Section 6 concludes the paper.

Notations: Let data set be $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ with sample $\mathbf{x}_j \in \mathbb{R}^p$, and the cluster centers be $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k]$, where $\mathbf{v}_i \in \mathbb{R}^p$ is the prototype of the cluster $\mathcal{C}_i$ for $i = 1, 2, ..., k$. Denote $\|\cdot\|$ the vector 2-norm or the Frobenius norm of a matrix. The distances between $\mathbf{x}_j$ and the prototypes $\mathbf{V}$ are $d_{ij} = \|\mathbf{x}_j - \mathbf{v}_i\| (i = 1, \cdots, k)$ and the closest distance is denoted as $D(\mathbf{x}_j)$. The membership grade matrix is denoted by $\mathbf{U} = [u_{ij}] \in \mathbb{R}^{k\times n}$, where u_{ij} represents the grade of the jth sample belonging to the ith cluster. The optimal cost of K-Means on data set $\mathbf{X}$ is denoted by ${J_{\mathbf{X}}}^{\text{opt}}$ and the corresponding optimal clusters are denoted by $\mathcal{C}^{\text{opt}}$.

2 Related Work

In this section, some improved K-Means-type clustering algorithms and the clustering algorithms based on over-parametrization learning are briefly recalled.

First of all, the K-Means problem aims to find the k partitions of $\mathbf{X}$ by minimizing the sum of squared distances between each sample to its nearest center. The underlying objective function is expressed as follows:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V}} J_{\mathbf{X}}(\mathbf{U},\mathbf{V}) &= \sum_{i=1}^{k}\sum_{j=1}^{n} u_{ij}\|\mathbf{x}_j - \mathbf{v}_i\|^2, \\ s.t. \quad &\sum_{i=1}^{k} u_{ij} = 1, u_{ij} \in \{0,1\}. \end{aligned} \tag{1}$$

To solve problem (1), iterative optimization algorithms are usually employed to approximate the global minima of the K-Means problem [40]. Among these algorithms, the most commonly used is the K-Means algorithm in [7].

2.1 K-Means and K-Means++ Algorithms

The K-Means algorithm [7], as the most popular clustering algorithm, is a heuristic method. First, k initial cluster centers are set as initializations, and then an iterative algorithm, called Lloyd's algorithm, is implemented. For an input of n samples and k initial cluster centers, Lloyd's algorithm consists of two steps: the assignment step assigns each sample to its closest cluster:

$$u_{ij}^{(t+1)} = \begin{cases} 1, & d_{ij}^{(t)} = \min_{1\le c\le k} d_{cj}^{(t)} \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where t is the iteration number. Then, the update step replaces the k cluster centers with the centroid of the samples assigned to the corresponding clusters:

$$\mathbf{v}_i^{(t+1)} = \frac{\sum\limits_{j=1}^{n} \left(u_{ij}^{(t+1)}\right) \mathbf{x}_j}{\sum\limits_{j=1}^{n} \left(u_{ij}^{(t+1)}\right)}. \tag{3}$$

The algorithm alternately repeats the two steps until convergence is achieved. The K-Means algorithm easily gets stuck in spurious local minima because of the non-convexity and non-differentiability of (1).

As studied in [11], a good initialization makes Lloyd's algorithm perform well. Therefore, K-Means++ algorithm [12] was proposed as a specific way of choosing the prototypes $\mathbf{V}^{(0)}$ for K-Means. In the first step, K-Means++ selects an initial prototype $\mathbf{v}_1$ uniformly at random from the data set. In the second step, each subsequent initial centroid $\mathbf{v}_i, i = 2, 3, ..., k$, is chosen by maximizing the following probability with respect to the previously selected set of prototypes:

$$\frac{D(\mathbf{x}_j)^2}{\sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2}, \quad j = 1, 2, ..., n. \tag{4}$$

Then, the algorithm repeats the second step until it has chosen a total of k prototypes. The sampling skill used in the second step is called ”D^2 sampling”. We note that it achieves approximation guarantees, as stated in the following **Theorem** 1.

Theorem 1. *[12] For any data set $\mathbf{X}$, if the prototypes are constructed with K-Means++, then the corresponding objective function $J_{\mathbf{X}}$ satisfies $E[J_{\mathbf{X}}] \le 8(\ln k + 2){J_{\mathbf{X}}}^{opt}$.*

Thus, K-Means++ algorithm is fast, simple, and $\mathcal{O}(\log k)$-competitive with the optimal result.

2.2 Split-merge K-Means Algorithm

Split-merge K-Means Algorithm (SMKM) [21] was introduced to reduce the cost of the K-Means problem (1) by a new splitting-merging step, which is able to generate better approximations of the optimal cost of the K-Means problem. It consists of the following two steps:

1) Splitting step: 2-Means is applied to each cluster $\mathcal{C}_i$ to get $\{\mathbf{v}_{i_1}, \mathbf{v}_{i_2}\}$, and then $\mathcal{C}_{i_{\text{split}}}$ is selected as the split cluster based on the following:

$$i_{\text{split}} = \underset{i\in\{1,2,...,k\}}{\arg\max} \left[J_{\mathcal{C}_i}(\mathbf{U}, \mathbf{v}_i) - J_{\mathcal{C}_i}(\mathbf{U}, \{\mathbf{v}_{i_1}, \mathbf{v}_{i_2}\})\right]. \tag{5}$$

2) Merging step: the pair of clusters with the smallest merging error increment can be merged together. Specifically, $\mathcal{C}_i$ and $\mathcal{C}_c$ are merged if

$$i, c = \underset{i,c\in\{1,2,...,k+1\}, i\neq c}{\arg\min} f_{i,c} \tag{6}$$

where $f_{i,c} = J_{\mathcal{C}_{i,c}}(\mathbf{U}, \mathbf{v}_{i,c}) - [J_{\mathcal{C}_i}(\mathbf{U}, \mathbf{v}_i) + J_{\mathcal{C}_c}(\mathbf{U}, \mathbf{v}_c)]$, $\mathcal{C}_{i,c} = \mathcal{C}_i \cup \mathcal{C}_c$, and $\mathbf{v}_{i,c} = \frac{|\mathcal{C}_i|\cdot\mathbf{v}_i + |\mathcal{C}_c|\cdot\mathbf{v}_c}{|\mathcal{C}_i| + |\mathcal{C}_c|}$.

SMKM repeats alternately the splitting and merging steps until convergence is achieved. In conclusion, the splitting step reduces the K-Means approximation error, while the merging step increases it. Hence, the quality of the local minima can be improved when the splitting-merging step reduces the cost of the K-Means.

2.3 Self-adaptive Multiprototype-based Competitive Learning

Self-adaptive Multiprototype-based Competitive Learning (SMCL) [29] is over-parametrization method based on the framework of K-Means competitive learning for imbalanced data clustering. It consists of the following two steps:

1) Prototype number selection (PNS): the number of multi-prototypes is determined in self-adaptive way. First, given the predefined number of multi-prototypes K, the update of prototypes is as follows:

$$\mathbf{v}_i^{(t+1)} = \begin{cases} \mathbf{v}_i^{(t)} + K\alpha_c(\mathbf{x}_j - \mathbf{v}_i^{(t)}), & i = I_j^*, \\ \mathbf{v}_i^{(t)} - K\eta_c\beta_j\alpha_c(\mathbf{x}_j - \mathbf{v}_i^{(t)}), & \forall i \neq I_j^*. \end{cases} \tag{7}$$

where α_c is learning rate, η_c is tuning parameter, $\beta_j = \exp\left(-\frac{\|\mathbf{v}_i^{(t)} - \mathbf{x}_j\|^2 - \|\mathbf{v}_{I_j^*}^{(t)} - \mathbf{x}_j\|^2}{\|\mathbf{v}_{I_j^*}^{(t)} - \mathbf{v}_i^{(t)}\|^2}\right)$, $i \neq I_j^*$, and $I_j^* = \arg\min_{1 \leq i \leq K}\{d_{ij}\}$.
After each update of the prototypes, K needs to be adjusted if $\max_i(\|\mathbf{v}_i^{(t+1)} - \mathbf{v}_i^{(t)}\|^2) < \eta$ is met. The guidelines for each prototype are as follows:

$$\begin{cases} \mathbf{v}_i^{(t)} & \text{is deleted,} \quad \text{if} \quad n_i < \theta, \\ \mathbf{v}_I^{(t)} & \text{is added,} \quad \text{otherwise.} \end{cases} \tag{8}$$

where $n_i = |\{j \mid j \in \mathcal{C}_i\}|$, θ is frequency parameter, $I = \arg\max_{1 \leq i \leq K}\{n_i\delta_i\}$. $\varrho_j^i = \sum_{z \in \mathcal{C}_i} [\![d(\mathbf{x}_j, \mathbf{x}_z) < \epsilon]\!]$, ϵ is density parameter, $[\![\cdot]\!]$ is the indicator function which returns 1 if the statement is true and 0 otherwise. $\delta_i = \max_{j \in \mathcal{C}_i} \min_{z \in \mathcal{C}_i : \varrho_z > \varrho_j} \frac{d(\mathbf{x}_j, \mathbf{x}_z)}{\bar{d}_i}$, $\bar{d}_i$ is the averaged pairwise distance of all samples in $\mathcal{C}_i$. Finally, PNS terminates if $\max_i(\|\mathbf{v}_i^{(t+1)} - \mathbf{v}_i^{(t)}\|^2) \geq \eta$ is met, and the result of PNS is $\hat{K}$ subclusters, $\mathcal{C}^{\text{PNS}}$.

2) Subcluster Grouping with Model Selection (SGMS): After obtaining $\mathcal{C}^{\text{PNS}} = \{\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_{\hat{K}}\}$, 1-d binary Gaussian mixture probability density function is used to compute the separation measure S_{iz} between $\mathcal{C}_i$ and $\mathcal{C}_z$ for $1 \leq i < z \leq \hat{K}$. The samples in $\mathcal{C}_i$ and $\mathcal{C}_z$ are first projected into the line between $\mathbf{v}_i$ and $\mathbf{v}_z$:

$$\mathbf{x}_p{}^{(i)} = \frac{(\mathbf{x}^{(i)} - \mathbf{v}^{\text{m}})^{\text{T}}(\mathbf{v}_i - \mathbf{v}_z)}{\|\mathbf{v}_i - \mathbf{v}_z\|^2}, \mathbf{x}^{(i)} \in \mathcal{C}_i \tag{9}$$

where $\mathbf{v}^{\text{m}} = (\mathbf{v}_i + \mathbf{v}_z)/2$. The projection calculation of the samples in $\mathcal{C}_z$ is the same as Eq. (9). Therefore, the mean and variance of the projected samples, μ_i, μ_z, σ_i and σ_z can be computed. Then, S_{iz} is computed by:

$$\begin{aligned} S_{iz} =& \frac{1}{\min f(\mathcal{A})}, \mathcal{A} = \{-0.5, -0.49, ..., 0.49, 0.5\}, \\ f(\mathcal{A}) =& \frac{|\mathcal{C}_i|}{|\mathcal{C}_i| + |\mathcal{C}_z|} p(\mathcal{A}|\mu_i, \sigma_i^2) + \frac{|\mathcal{C}_z|}{|\mathcal{C}_i| + |\mathcal{C}_z|} p(\mathcal{A}|\mu_z, \sigma_z^2). \end{aligned} \tag{10}$$

SGMS merges only the two closest subclusters at a time, until all subclusters are merged into one. The initialization is $K = \hat{K}$, and based on $com_{K-1} = \min_{1 \leq i < z \leq K} S_{iz}$, two clusters are merged to obtain $\mathcal{C}(K-1)$ until $K = 1$. Meanwhile, global separability $sep_{K-1} = \max_{1 \leq i \leq K-1} \sum_{j \in \mathcal{C}_i} (\frac{g_j}{q})$ is computed, where g_j is the number of samples whose q-nearest neighbors are not in $\mathcal{C}_i$. Finally, the best number of clusters is $K^* = \arg\min_{1 \leq K, K' \leq \hat{K}-1}(\frac{sep_K}{\max_{K''}\{sep_{K'}\}} + \frac{com_K}{\max_{K'}\{com_{K'}\}})$. And the best clusters are $\mathcal{C}(K^*)$.

SMCL is a two-step clustering algorithm for imbalanced data without given k, which selects multi-prototypes in an adaptive way, and then merges the prototypes using SGMS. Finally, a new evaluation metrics is used to determine the best number of clusters and clustering result. However, its computational complexity is very high, and too many parameters need to be predefined.

2.4 Convex Clustering Algorithm

Convex clustering model [31] formulates the clustering task as a convex optimization problem by adding a sum-of-norms (SON) regularization to control the trade-off between the model error and the number of clusters. To reduce the computational burden of evaluating the regularization terms, the weight, $W = [w_{ij}]$, is introduced. The objective function of convex clustering model is expressed as follows:

$$\min_{\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_n \in \mathbb{R}^p} \frac{1}{2}\sum_{j=1}^{n}\|\boldsymbol{\mu}_j - \mathbf{x}_j\|^2 + \gamma\sum_{i<j} w_{ij}\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|_p, \tag{11}$$

where $\gamma > 0$ is a tuning parameter and the p-norm with $p \geq 1$ ensures the convexity of the model. Here, w_{ij} is a nonnegative weight given by:

$$w_{ij} = \begin{cases} \exp(-\kappa\|\mathbf{x}_i - \mathbf{x}_j\|^2), & \text{if} \quad (i,j) \in E; \\ 0, & \text{otherwise,} \end{cases} \tag{12}$$

where $E = \cup_{j=1}^{n}\{l = (i,j) | i \in \text{KNN}(j)\}$, KNN$(j)$ is the index set of the q-nearest neighbors of $\mathbf{x}_j$ for $j = 1, 2, ..., n$, and κ is a given positive constant.

After the optimal solutions $\boldsymbol{\mu}_1^*, ..., \boldsymbol{\mu}_n^*$ of (11) are obtained, the samples are assigned to be in one cluster if and only if their optimal solutions $\boldsymbol{\mu}^*$ are the same. Convex clustering, based on over-parametrization learning, can avoid bad local minima, cluster arbitrary shape data sets, and get the cluster number [32], [33]. However, its computational complexity is very high, which still remains challenging for large-scale problems. Meanwhile, the number of neighboring samples q is generally selected empirically. If it is too small, convex clustering will not achieve the perfect recovery. Conversely, the computational burden cannot be reduced.

3 Motivation

The above-mentioned algorithms can avoid the bad local minima of K-Means problem. However, their structure is rarely involved in the studies, so that the recovery approaches are not well understood. For example, in [21], the splitting and merging steps are performed for selecting better local minima. But only a prototype is split by 2-Means, and only a pair of prototypes are merged in each iteration. This lack of explanation of the structure of better local minima inspires us to come up with a new algorithm.

3.1 Recover the Better Local Minima Based on Multi-Prototypes Technique

In this subsection, an important theorem in [36], which describes the structure of spurious local minima of K-Means problem under convex data, is recalled for convenience.

Theorem 2. *For well-separated mixture models, all spurious local minima solutions $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_1, ..., \mathbf{v}_k]$ of $\mathbf{X}$ involves the following configurations: (i) multiple prototypes $\{\mathbf{v}_i\}$ lie in a true* *cluster and (ii) one prototype $\mathbf{v}_i$ is put in the centroid of multiple true clusters.*

Note that the above configurations (i) and (ii) are referred to as the over-refinement and under-refinement of the true clusters, respectively. Importantly, **Theorem** 2 gives the general splitting-merging approaches an explanation and understanding for the better local minima. In detail, the splitting and merging steps remove under-refinement and over-refinement of the true clusters, respectively.

Borrowing the precise characterization of local minima in **Theorem** 2, the splitting technique is theoretically a good way to remove under-refinement. However, it does not determine exactly how many prototypes to split into, so as to eliminate under-refinement in the clustering result. Therefore, in this paper, a multi-prototypes technique, as an over-parametrization approach, is exploited instead of implementing an unsatisfactory splitting step. The multi-prototypes technique can avoid under-refinement. In detail, by setting a number larger than k as the number of clusters, the multi-prototypes technique aims to achieve that at least one prototype or multiple prototypes lie in a true cluster. Then when coupled with the K-Means algorithm, one expects that only exact refinement or over-refinement of the true clusters exist after the K-Means algorithm. Hence, a better local minimum can be recovered by simply merging the prototypes in this structures. In conclusion, the multi-prototypes technique is a suitable alternative to the splitting step for removing under-refinement of the true clusters in the clustering result.

3.2 Select the Appropriate Number of Multi-Prototypes

To eliminate under-refinement of the true clusters in the clustering result, we need to set a number of multi-prototypes larger than k. However, a predefined number of multi-prototypes may not work for different data sets. In order to illustrate these issues, a set of experiments is carried out on three synthetic data sets, D1, D2, and D3, as shown in Fig. 1. See Section 5.2 for the details of the data sets.

In Fig. 1, the clustering results of K-Means with different predefined numbers of multi-prototypes on the three synthetic data sets are displayed, where the true number of clusters for the three data sets is $k = 2$. In (1a)–(1c), the number of multi-prototypes is set to be small; in (1d)–(1f), the number of multi-prototypes is set to be large.

From the illustration, it can be summarized that if the given number of prototypes is too small, the multi-prototypes for the different data distributions are inaccurate and some prototypes may lie in the centroid of multiple true clusters. Conversely, if the number is too large, some prototypes lie in the overlapping area between the true clusters, noises samples and outliers. Clearly, the larger the given number of prototypes, the better the representation of the multi-prototypes for the different data distributions. However, if the number of multi-prototypes exceeds a certain number, the representation approximates density clustering and the computational complexity is too high. An appropriate number of multi-prototypes should be closely related to the data distribution. Therefore, we design a multi-prototypes selection technique to sample an appropriate number of multi-prototypes based on the data distribution, where the samples are gradually selected as prototypes by D^2 sampling until the latest selected sample has little improvement in the data representation. The details are presented in Section 4.

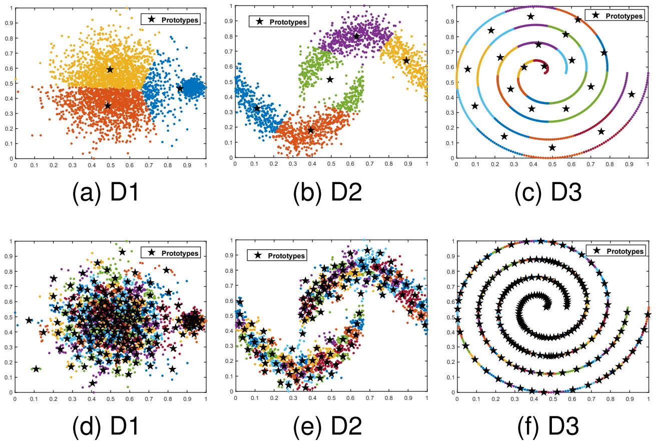


Fig. 1. Clustering results of K-Means with the multi-prototypes on D1, D2, and D3. In (a)–(c), the given number of multi-prototypes is too small; in (d)–(f), the given number of multi-prototypes is too large. The plots clearly show some final prototypes always lie in the overlapping area between the different true clusters, noises samples and outliers when the number of multi-prototypes is not selected properly.

4 Multi-Prototypes Convex Merging Based K-Means Clustering Algorithm

In this section, the multi-prototypes convex merging based K-Means clustering algorithm (MCKM) is proposed to recover better local minima without given the cluster number k a priori. In the first step of MCKM, a multi-prototypes sampling (MPS) first selects a suitable number of multi-prototypes for better data representation. It provides an explainable approach to refine or over-refine clusters based on the structure of the local minima. Furthermore, a theoretical proof is given, which guarantees that the multi-prototypes selected by MPS can achieve a constant factor approximation to the global minima of K-Means problem. Then in the second step, a merging technique, convex merging (CM), recovers the better local minima. CM can obtain the optimal merging and estimate the correct cluster number because it treats the merging task as a convex optimization problem. The overall process of MCKM is as follows:

$$\mathbf{X} \xrightarrow{\text{MPS (Alg. 1)}} \{\mathbf{V}_{\text{MPS}}, \mathcal{C}^{\text{MPS}}\} \xrightarrow{\text{CM (Alg. 2)}} \mathcal{C}^{\text{MCKM}}.$$

MPS and CM are explained in Subsections 4.1 and 4.2, respectively.

4.1 Multi-Prototypes Sampling (MPS)

In the K-Means algorithm, the multi-prototypes are constructed to represent the data structure. To quantify the representation ability of the multi-prototypes, a reconstruction criterion is introduced, see [41], [42]:

$$R(s) = \sum_{j=1}^{n} \|\mathbf{x}_j - \hat{\mathbf{x}}_j(s)\|^2, \tag{13}$$

where $\hat{\mathbf{x}}_j(s) = \sum_{i=1}^{s} u_{ij}\mathbf{v}_i / \sum_{i=1}^{s} u_{ij}$. It gives the reconstructed value with the current prototypes $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_s\}$ and assignment coefficients, u_{ij}, obtained by (2). Note that the lower value of the reconstruction criterion, the better the representation ability of the multi-prototypes. Meanwhile, it can be inferred that the value of the reconstruction criterion decreases with the increasing number of multi-prototypes. However, it is not expected that the number of multi-prototypes is very large, as analyzed in Subsection 3.2. Therefore, a new ratio, called relative reconstruction rate with respect to the number of multi-prototype, is defined as follows:

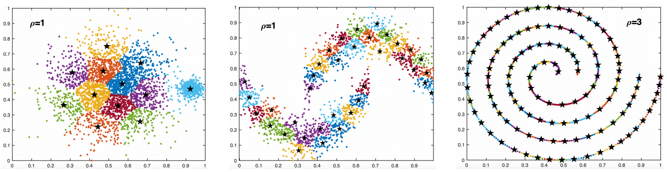
$$\frac{R(s-1) - R(s)}{R(s-1)}. \tag{14}$$

The relative reconstruction rate can be utilized as a measure of the improvement of the representation ability of the new multi-prototypes set after adding a prototype to the multi-prototypes set. If the new multi-prototypes set has little improvement in the relative reconstruction rate after adding a prototype, the new prototype should not be added. Hence, the number of multi-prototypes can be selected based on (14), where the quantization of little improvement is equivalent to $\frac{R(s-1)-R(s)}{R(s-1)} \le \varepsilon$ by setting a small threshold ε.

As the analysis in Subsection 3.2 shows, a predefined number of multi-prototypes is difficult to be set properly, and MPS can select a suitable number using the relative reconstruction rate. MPS randomly picks an initial sample into cluster $\mathcal{C}$, and then proceeds D^2 sampling, where the sample is selected with the probability (4) and added to $\mathcal{C}$ in each iteration. MPS converges until the new multi-prototypes set has little improvement in representing the data set after adding the latest selected sample. Finally, the K-Means algorithm is performed with the selected prototypes on the data set, and the final result is obtained. The proposed MPS is presented in Algorithm 1.

Algorithm 1 Multi-Prototypes Sampling (MPS)

```
Input: Date set $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$, the threshold $\varepsilon$;
Output: The multi-prototypes $\mathbf{V}_{\text{MPS}}$, the number of the
    multi-prototypes $s^*$.
Pick a sample $\mathbf{x}^{(1)}$ randomly and $\mathbf{V} = \{\mathbf{x}^{(1)}\}$;
Compute $R(1)$ based on $\mathbf{V}$, Eq. (2) and Eq. (13);
Set $s := 2$ and $R = R(1)$;
while $s \le n$ do
    Sample $\mathbf{x}^{(s)}$ with probability $\frac{D(\mathbf{x}^{(s)})^2}{\sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2}$ based on the
    current $\mathbf{V}$ and add it to $\mathbf{V}$;
    Compute $R(s)$ based on $\mathbf{V}$, Eq. (2) and Eq. (13);
    if $\frac{R-R(s)}{R} \le \varepsilon$ then
        Break;
    else
        $R = R(s)$;
        $s = s + 1$;
    end if
end while
Run K-Means with the selected prototypes $\mathbf{V}$ on $\mathbf{X}$;
Obtain the multi-prototypes $\mathbf{V}_{\text{MPS}}$, the corresponding
clusters $\mathcal{C}^{\text{MPS}}$ and the number of the multi-prototypes
is $s^* = |\mathbf{V}_{\text{MPS}}|$.
```



(a) D1, $s^* = 12$ (b) D2, $s^* = 25$ (c) D3, $s^* = 156$

Fig. 2. The results of MPS on D1, D2, and D3 with the appropriate ρ, where the black stars are the final multi-prototypes and s^* is the number of the multi-prototypes.

We have the following comments for MPS:

- As shown above, MPS is an unsupervised technique that does not require the number of clusters in advance. By introducing the relative reconstruction rate in (14), MPS has the ability to select the appropriate number of multi-prototypes.
- In MPS, ε is a key parameter to tune the number of multi-prototypes selected by the algorithm. Evidently, the smaller ε is, the more number of multi-prototypes are sampled, and vice versa. Here, ε is empirically set as follows:

$$\varepsilon = \frac{1}{\rho\sqrt{n * p}} \tag{15}$$

  where n is the number of samples, p is the dimensionality of samples, and ρ is a positive constant. We present the results of MPS with the appropriate ρ on D1, D2, and D3, as shown in Fig. 2. Furthermore, we show by experiments in Subsection 5.2.1 that by changing ρ appropriately, MPS allows the clustering results of K-Means to better adapt to the arbitrary shape data sets.
- The computational complexity of MPS is $\mathcal{O}(\frac{1}{2}np(1+s^*)s^* + nps^*t_{\text{K-Means}})$, where s^* is the number of multi-prototypes by MPS, and $t_{\text{K-Means}}$ is the number of iterations of K-Means algorithm.
- We can prove that the multi-prototypes obtained by MPS achieve a constant factor approximation to the global minima of K-Means problem, see below.

Theorem 3. *For any data set $\mathbf{X}$, if the prototypes are constructed with MPS, then the corresponding objective function $J_{\mathbf{X}}^{MPS}$ satisfies:*

$$E[J_{\mathbf{X}}^{MPS}] \le 2(1-\varepsilon)(3J_{\mathbf{X}}{}^{opt} + 2n_a \triangle),$$

where ε is the termination threshold for MPS, $n_a = |\{\mathbf{x} | \|\mathbf{v}(\mathbf{x}) - \mathbf{v}^*(\mathbf{x})\| \ge \|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\|\}|$ and $\triangle = \varepsilon J_{\mathbf{X}}^{MPS}$.

The proof is given in **Appendix** A. Thus given a small ε, the iterations of MPS can continuously optimize $\triangle$ to achieve the desired approximate upper bound on the global minima. After Algorithm 1, the multi-prototypes need to be merged to recover better local minimum. In the next subsection, the merging technique CM is presented.

4.2 Multi-Prototypes Convex Merging (CM)

In this part, a merging technique, called convex merging (CM), is proposed to recover better local minima in the case of unknown number of clusters. CM, derived from the convex clustering paradigm [31], formulates the merging of the multi-prototypes task as a convex optimization problem by adding a sum-of-norms (SON) regularization to control the trade-off between the model error and the number of clusters. The model is as follows:

$$\min_{\boldsymbol{\mu}_1,\dots,\boldsymbol{\mu}_{s^*}\in\mathbb{R}^p} \frac{1}{2}\sum_{i=1}^{s^*}\|\boldsymbol{\mu}_i-\mathbf{v}_i\|^2+\gamma\sum_{i<j} w_{ij}\|\boldsymbol{\mu}_i-\boldsymbol{\mu}_j\|, \qquad (16)$$

where $\gamma > 0$ is a tuning parameter, s^* is number of the multi-prototypes by MPS, and the norms chosen ensure the convexity of the model. Here, w_{ij} is chosen based on the number of neighboring samples q, and (12).

After solving (16), the optimal solutions, $\boldsymbol{\mu}_1^*, \dots, \boldsymbol{\mu}_{s^*}^*$, are obtained. Then the multi-prototypes in $\mathbf{V}_{\text{MPS}}$ are assigned based on the following criteria: for any i, $i^{'} \in \{1, 2, \dots, s^*\}$, $\mathbf{v}_i$ and $\mathbf{v}_{i'}$ can be assigned to the same cluster if and only if their optimal solutions $\|\boldsymbol{\mu}_i^* - \boldsymbol{\mu}_{i'}^*\| \leq \eta$ for a given tolerance η. Otherwise, i and $i^{'}$ are assigned to the different clusters. Accordingly, the optimal clusters of the multi-prototypes, $\mathcal{C}^{\text{CM}} = \{\mathcal{C}_1^{\text{CM}}, \mathcal{C}_2^{\text{CM}}, \dots, \mathcal{C}_{k^*}^{\text{CM}}\}$, are formed, where k^* is the estimated number of clusters.

Finally, the samples are merged into the clusters in $\mathcal{C}^{\text{CM}}$ and we get the final clusters of the data set $\mathcal{C}^{\text{MCKM}}$. In detail, $\mathbf{x}_j \in \mathcal{C}_l^{\text{MCKM}}$, if $\mathbf{x}_j \in \mathcal{C}_i^{\text{MPS}}$ and $\mathbf{v}_i \in \mathcal{C}_l^{\text{CM}}$ for $i = 1, 2, \dots, s^*$, $j = 1, 2, \dots, n$, $l = 1, 2, \dots, k^*$.

In (16), γ regulates both the assignment of the multi-prototypes and the number of clusters. When $\gamma = 0$, each prototype occupies a unique cluster of its own. For a sufficiently large γ, all multi-prototypes are assigned to the same cluster.

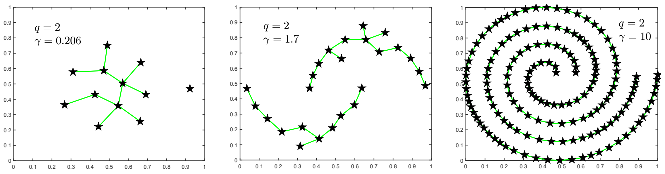
The algorithm of CM is summarized in Algorithm 2. The details of the optimization process can be referred to **Appendix** B and the related studies [34], [35].

Algorithm 2 Convex Merging (CM)

```
Input: The multi-prototypes $\mathbf{V}_{\text{MPS}}$ with $s^*$, the corresponding clusters $\mathcal{C}^{\text{MPS}}$, the number of neighboring samples $q$, the tuning parameter $\gamma$ and the termination $\eta$;
Output: The clusters of the data set $\mathcal{C}^{\text{MCKM}}$ and the estimated number of clusters $k^*$.
1: Optimize (16) on the results of MPS, $\mathbf{V}_{\text{MPS}}$, and get the optimal solution $\boldsymbol{\mu}^* = \{\boldsymbol{\mu}_1^*, \dots, \boldsymbol{\mu}_{s^*}^*\}$;
2: Form the optimal clusters of the multi-prototypes, $\mathcal{C}^{\text{CM}} = \{\mathcal{C}_1^{\text{CM}}, \mathcal{C}_2^{\text{CM}}, \dots, \mathcal{C}_{k^*}^{\text{CM}}\}$, based on $\boldsymbol{\mu}^*$;
3: Obtain the clusters of the data set $\mathcal{C}^{\text{MCKM}}$ based on the clusters $\mathcal{C}^{\text{MPS}}$ and $\mathcal{C}^{\text{CM}}$, and the estimated number of clusters $k^* = |\mathcal{C}^{\text{MCKM}}|$.
```

We have the following comments for CM:

- Because the objective of CM is convex, the global minima of the merging of the multi-prototypes for a given suitable γ is unique and is easier to obtain than the traditional merging techniques [31], [32].
- We present the results of CM on the multi-prototypes of MPS for D1, D2, and D3 with the appropriate q and γ, as shown in Fig. 3. By choosing appropriate q and γ in (16), the prototypes fusion path can be generated, which enhances the explainable and comprehension of recovering the better local minima.



(a) D1, $k^* = 2$ (b) D2, $k^* = 2$ (c) D3, $k^* = 2$

Fig. 3. The results of CM on the multi-prototypes of MPS for D1, D2, and D3 with the appropriate q and γ, where the black stars are the multi-prototypes of MPS and the green lines are the minimum spanning tree of each CM subcluster. k^* is the estimated number of clusters.

- Originating from CC model [31], CM has the property that the value of γ is inversely proportional to the estimation of the number of clusters k^*. Based on monotonicity, a suitable γ is sure to allow MCKM to evaluate the correct cluster number.
- The computational complexity of CM is $\mathcal{O}((s^*)^2 p t_{\text{ADMM}})$, where $s^* \ll n$, and t_{ADMM} the number of iterations of ADMM solver.
- We can prove that the final clustering result of MCKM achieves a constant factor approximation to the global minima of K-Means problem, see below.

Theorem 4. *For any data set* $\mathbf{X}$*, if the final prototypes are constructed with MCKM (MPS+CM), then the corresponding objective function* $J_{\mathbf{X}}^{MCKM}$ *satisfies:*

$$J_{\mathbf{X}}^{opt} \leq J_{\mathbf{X}}^{MCKM} \leq 2J_{\mathbf{X}}^{MPS} + 2n_b J_{\mathbf{V}_{MPS}}^{CM}.$$

where $n_b = \max_{1 \leq i \leq s^*} |\mathcal{C}_i^{MPS}|$*, and* $|\mathcal{C}_i^{MPS}|$ *is the cardinality of* $\mathcal{C}_i^{MPS}$*.*

The proof is given in **Appendix** C. According to **Theorem** 3 and 4, the objective function of MCKM (MPS+CM), still achieves a constant factor approximation to the global minima of K-Means problem. In summary, MCKM can achieve better local minima in theory.

In the next section, several experiments are performed to illustrate the performance of the proposed algorithm.

5 Experimental Results

To verify the effectiveness and efficiency of the proposed algorithm, experiments are carried out on synthetic and real-world data sets. We compare our method with four other clustering algorithms: 1) K-Means algorithm [7]; 2) Split-Merge K-Means algorithm (SMKM) [21]; 3) Affinity Propagation algorithm (AP) [37]; 4) Self-adaptive multiprototype-based competitive learning (SMCL) [29]; and 5) Convex clustering (CC) [31]. These algorithms are chosen because they use different techniques to achieve the good approximation of the global minima. Specifically, K-Means, SMKM and AP usually perform well on relatively uniform size and linearly separable convex data sets. SMCL and CC can handle the clustering of non-convex and skewed data sets without given the cluster number. Moreover, AP, SMCL and CC can estimate the correct cluster number by selecting appropriate hyper-parameters.

All experiments were run on a computer with an Intel Core i7-6700 processor and a maximum memory of 8GB. The computer runs Windows 7 with MATLAB R2017a. The experimental setup and the evaluation metrics used for clustering performance are described below. The termination parameter $\eta = 10^{-6}$ for all algorithms except SMCL which is empirically set at 0.001. AP contains two parameters: damping factor λ and value of the diagonal of the similarity matrix Λ. In AP, $\lambda = 0.98$ for all data sets. SMCL contains six parameters: learning rata α_c, tuning parameter η_c, frequency parameter θ, density parameter ϵ, the number of neighboring samples q, and the predefined number of multi-prototypes K. In SMCL, $\theta = 0.01n$, $\epsilon = \{\epsilon \mid \frac{1}{n}\sum_{z\neq j}[\![d(\mathbf{x}_j, \mathbf{x}_z) < \epsilon]\!] = 0.02n\}$, $q = 5$ and $K = 2$ for all data sets. The positive constant $\kappa = 0.9$ in (12) for CC and MCKM. The remaining parameters need to be fine-tuned in the experiments.

5.1 Evaluation Metrics

In order to evaluate the performances of the clustering algorithms, three metrics are used. They are: the F-measure ($\mathbf{F}^*$), Normalized Mutual Information (**NMI**), and Adjusted Rand Index (**ARI**) [43], [44], [45]. They measure the agreement with the ground truth and the clustering results. Let n be the total number of samples, $\{\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_k\}$ be the partition of the ground truth, and $\{\hat{\mathcal{C}}_1, \hat{\mathcal{C}}_2, \cdots, \hat{\mathcal{C}}_{\hat{k}}\}$ be the partition by an algorithm. Denote $|\cdot|$ the cardinality of a set. Let $\hat{n}_i = |\hat{\mathcal{C}}_i|$, $n_l = |\mathcal{C}_l|$, and $n_i^l = |\mathcal{C}_l \cap \hat{\mathcal{C}}_i|$, where $i = 1, 2, \cdots, \hat{k}$ and $l = 1, 2, \cdots, k$. Then the measure $F(l,i) = \frac{2n_i^l}{n_l + \hat{n}_i}$ is the harmonic mean of the precision and recall of $\mathcal{C}_l$ and its potential prediction $\hat{\mathcal{C}}_i$. The overall F-measure $\mathbf{F}^*$, **NMI** and **ARI** are defined as follows:

$$\mathbf{F}^* = \sum_{l=1}^{k} \frac{n_l}{n} \max\{F(l,i)|i = 1, \cdots, \hat{k}.\}, \tag{17}$$

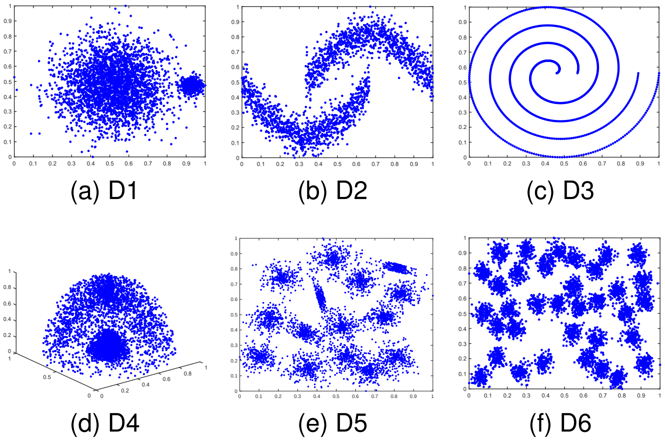
$$\mathbf{NMI} = \frac{\sum_{i=1}^{\hat{k}} \sum_{l=1}^{k} n_i^l \log(\frac{n \cdot n_i^l}{\hat{n}_i \cdot n_l})}{\sqrt{\left(\sum_{i=1}^{\hat{k}} \hat{n}_i \log(\frac{\hat{n}_i}{n})\right)\left(\sum_{l=1}^{k} n_l \log(\frac{n_l}{n})\right)}}, \tag{18}$$

$$\mathbf{ARI} = \frac{\sum_{i=1}^{\hat{k}} \sum_{l=1}^{k} \binom{n_i^l}{2} - \sum_{i=1}^{\hat{k}} \binom{t_i}{2} \sum_{l=1}^{k} \binom{s_l}{2} / \binom{n}{2}}{\frac{1}{2}\left(\sum_{i=1}^{\hat{k}} \binom{t_i}{2} + \sum_{l=1}^{k} \binom{s_l}{2}\right) - \sum_{i=1}^{\hat{k}} \binom{t_i}{2} \sum_{l=1}^{k} \binom{s_l}{2} / \binom{n}{2}}, \tag{19}$$

where $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, $s_l = \sum_{i=1}^{\hat{k}} n_i^l$, and $t_i = \sum_{l=1}^{k} n_i^l$.

5.2 Experiments on Synthetic Data Sets

Six normalized synthetic data sets are selected for clustering in the first set of experiments, see Fig. 4. They include unbalanced data set, non-convex data sets, and convex data sets with large number of clusters. The detailed information on the data sets is given in Table 1, where n is the number of training size, p is the dimensionality of samples, and k is the true number of clusters. In order to have a better understanding of MCKM, the performance of the multi-prototypes sampling (MPS) and the convex merging (CM) are shown respectively in Sections 5.2.1 and 5.2.2.


(a) D1 (b) D2 (c) D3

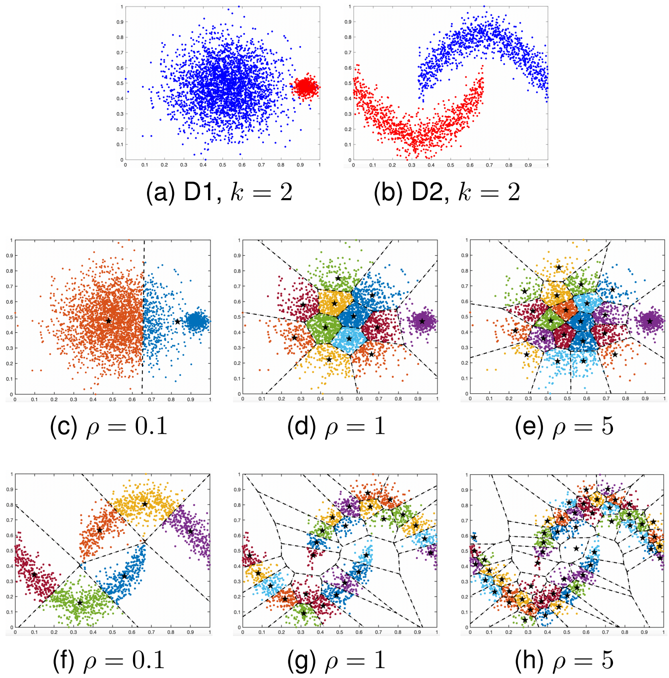
(d) D4 (e) D5 (f) D6

Fig. 4. Six synthetic data sets. They include unbalanced data set, non-convex data set, and convex data set with a large number of clusters.

5.2.1 *Parameter Sensitivity*

The selection of parameters (ρ, q and γ) is crucial for MCKM (MPS+CM). Therefore, we analyze the sensitivity of parameters through visualization on the synthetic data sets.

1) **ρ for MPS:** Here we show that MPS can adapt to the arbitrary shape data sets by choosing appropriate ρ in (15). To illustrate these, MPS is performed on D1 and D2 with $\rho = 0.1, 1, 5$ respectively. The results and corresponding Voronoi partition are shown in Fig. 5.

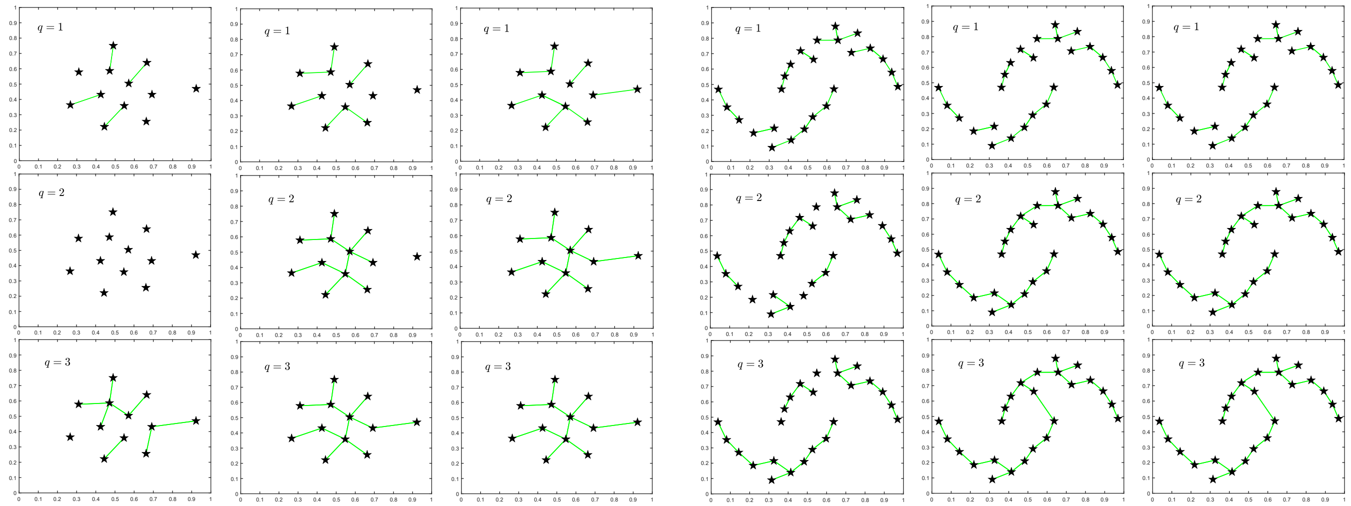

(a) D1, $k = 2$ (b) D2, $k = 2$

(c) $\rho = 0.1$ (d) $\rho = 1$ (e) $\rho = 5$

(f) $\rho = 0.1$ (g) $\rho = 1$ (h) $\rho = 5$

Fig. 5. The true clusters of D1 and D2 in 5a and 5b. The results of MPS and the corresponding Voronoi partition on D1 and D2 with $\rho = 0.1, 1, 5$ in 5c-5e and 5f-5h, respectively, where the black stars are the final multi-prototypes.

The true clusters of D1 and D2 are shown in Fig. 5a and 5b. When $\rho = 0.1$, we observe from the corresponding Voronoi partition in Fig. 5c and 5f that some prototypes obtained by MPS are put in the centroid of the two true



(a) $\gamma = 0.15$ (b) $\gamma = 0.206$ (c) $\gamma = 0.3$ (d) $\gamma = 0.5$ (e) $\gamma = 1.7$ (f) $\gamma = 5$

Fig. 6. The results of CM on the multi-prototypes of MPS for D1 and D2 with different q and γ in 6a-6c and 6d-6f, respectively, where the black stars are the multi-prototypes of MPS with $\rho = 1$ on D1 and D2. The green lines are the minimum spanning tree of each CM subcluster.

clusters. When $\rho = 5$, some prototypes obtained by MPS lie in the outliers on D2, as shown in Fig. 5h. For D1, when $\rho = 1$ and $\rho = 5$, there is no under-refinement structure of the true clusters, as shown in Fig. 5d and 5e. Naturally, we use a small number of multi-prototypes for the next CM. Therefore, in MPS, $\rho = 1$ is appropriate for D1 and D2.

From Fig. 4, it can be found that the value of ρ is directly proportional to the number of multi-prototypes s^*. Therefore, based on this monotonicity, MPS allows the clustering results of K-Means to better adapt to the arbitrary shape data sets. In detail, for the arbitrary shape data set, MPS with an appropriate ρ can achieve that each true cluster have one or more prototypes, and none of the prototypes are put in the centroid of multiple true clusters. Based on the results of MPS, the better local minima of K-Means can be obtained by the subsequent convex merging.

2) **q and γ for CM**: After MPS, CM is applied to merge the multi-prototypes to get the local minima of K-Means problem. Here we show that the prototypes fusion path is generated by choosing appropriate q and γ in (16), which enhances the explainable and comprehension of recovering the better local minima. To illustrate these, CM is performed on the multi-prototypes of MPS for D1 and D2 with $q = 1, 2, 3$ respectively. And $\gamma = 0.15, 0.206, 0.3$ for D1; $\gamma = 0.5, 1.7, 5$ for D2. The results are shown in Fig. 6.

In Fig. 6, it can be observed that when q is fixed, an increasing γ leads to the merging of more multi-prototypes. This finding is consistent with the conclusion drawn in Subsection 4.2. When γ is fixed and $q = 1$, under-merging is often observed in CM, indicating that some prototypes that should have been merged are not merged, but there are no undesired mergers. However, when $q = 3$, CM often exhibits over-merging, meaning that some prototypes that should not have been merged are merged, even when γ is small. Therefore, we suggest setting q to 1 or 2 in most cases, as choosing a smaller q ensures that the prototypes that are merged are highly similar.

Additionally, the appropriate γ varies for different data sets and is difficult to determine empirically. However, in [35], a comprehensive theoretical proof has been provided to demonstrate that an appropriate γ must exist for a given data set. Moreover, CM has the property that the value of γ is inversely proportional to the estimation of the number of clusters. Therefore, based on this monotonicity, an appropriate γ can always be selected.

5.2.2 Performance of MCKM

The clustering results of MCKM and of the other four algorithms are plotted in Fig. 7. The metric results of $\mathbf{F}^*$, **NMI**, and **ARI** are shown in Table 1, where k^* is the number of clusters obtained by the algorithms. Suitable hyperparameters are selected for AP, SMCL, CC and MCKM and are listed in the second column of Table 1. Furthermore, the running times of the algorithms are displayed in Table 2, where the values are averaged over 20 trials.

From the results, we have the following findings.

- From Fig. 7 and the corresponding Table 1, MCKM is competent for the clustering of the arbitrary shape data sets, including unbalanced data set, non-convex data sets, and convex data sets with a large cluster number. Its clustering results are comparable to those of the state-of-the-art clustering algorithms, or even better, see, for example, the results of D3, D4, and D6.
- In terms of getting the number of clusters, AP, SMCL, CC and MCKM have the same performance when the true number of clusters is small. However, when the true number of clusters is relatively large, only AP, CC and MCKM still work well by choosing a suitable parameter. In detail, AP benefits from the iterative update of valuable information passed between samples; CM and MCKM inherits the advantage of convex clustering. CM and CC solve the similar convex optimization model as in (11) and (16). In summary, AP, CC and MCKM are outstanding in getting the number of clusters.

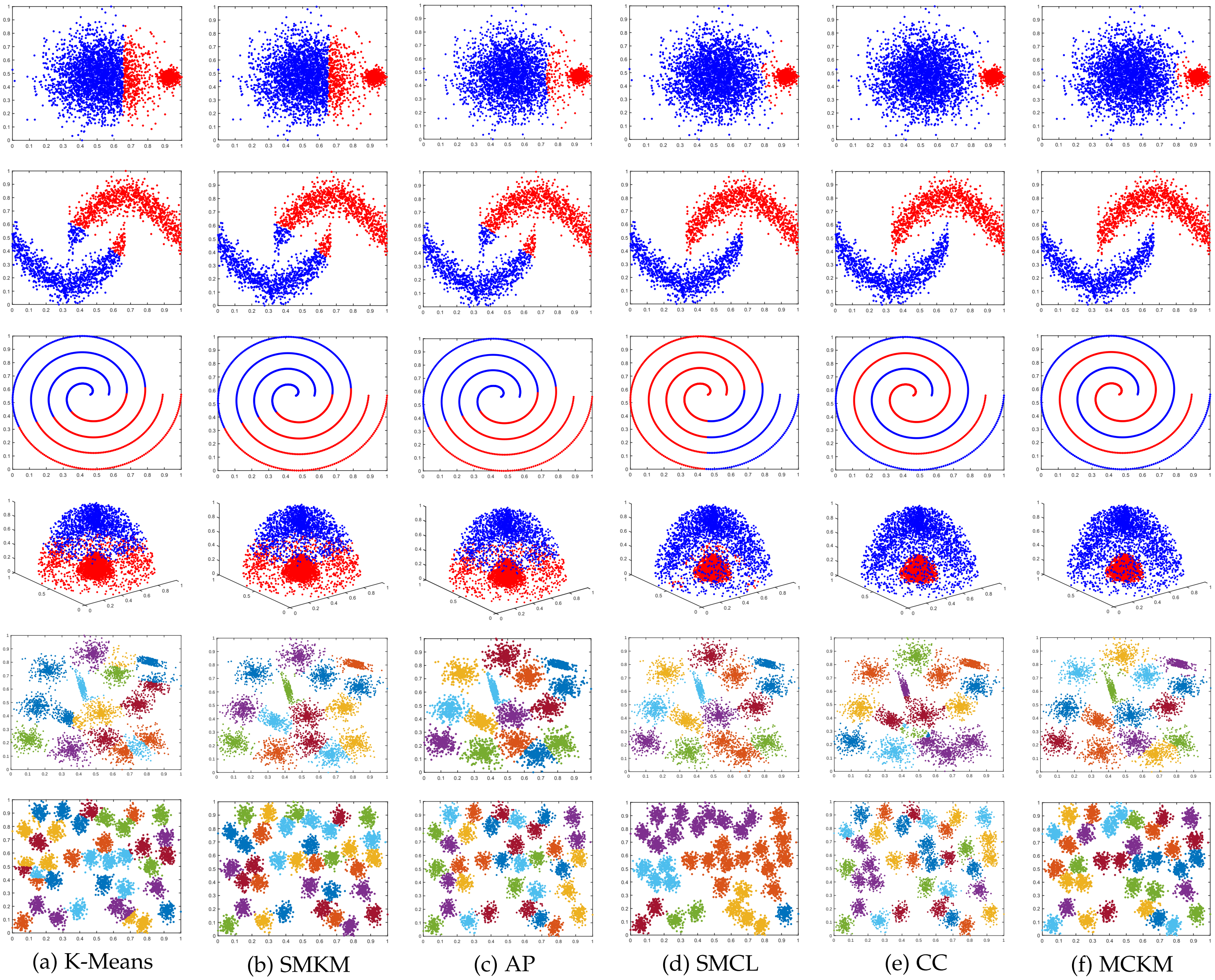


Fig. 7. The clustering results of K-Means [7], SMKM [21], AP [37], SMCK [29], CC [31] and the proposed MCKM on the six synthetic data sets.

- From Table 2, we see that the running time of K-Means and SMKM is much less than that of AP, SMCL and CC. Obviously, the estimation of the number of clusters is very time-consuming for the clustering algorithms. Comparing the four algorithms without the cluster number given a priori, i.e., AP, SMCL, CC, and MCKM, MCKM has significantly higher efficiency. In particular, MCKM is even more efficient than SMKM on convex data sets with a large number of clusters, see D5 and D6.

5.3 Experiments on Real-world Data Sets

The second set of experiments are performed on real-world data sets selected from UCI Machine Learning Repository[1]. The detailed information on the data sets, the clustering performances and the hyper-parameters used are given in Table 3. In MCKM, the constant $\rho = 1, 0.8, 1.6, 1, 2$ are chosen empirically for MPS on HTRU2, Iris, Wine, X8D5K, and Statlog respectively. The running time of the algorithms are displayed in Table 4, where the values are averaged over 20 trials.

From the results, we obtain the following findings.

- In terms of the clustering results, MCKM outperforms the other algorithms on almost all real-world data sets.
- In terms of evaluating the cluster number, CC and MCKM still perform better than the other algorithms.
- In terms of the running time of the algorithms, although MCKM is not as efficient as the other algorithms where the cluster number are given a priori, i.e., K-Means and SMKM, it is the best among the algorithms that do not require the cluster number. In particular, the running time of MCKM is about 38% of that of CC on average.

In conclusion, based on the structure of spurious local minima of the K-Means problem, MCMK provides an explainable two-stage approach for recovering a better local minima, in which oversampling is performed by MPS, and then the multi-prototypes are merged by CM. Moreover, the experiments on synthetic and real-world data sets show that

[1] https://archive.ics.uci.edu/ml/index.php

TABLE 1
The evaluation of the clustering results of the different algorithms on the six synthetic data sets. The best results are shown in boldface.

Algorithms	Parameter	F*	NMI	ARI	k^*
	D1 ($n = 3500, p = 2, k = 2$)				
K-Means	$k = 2$	0.8574	0.4481	0.4999	–
SMKM	$k = 2$	0.8574	0.4481	0.4999	–
AP	$\Lambda = -200$	0.9574	0.7064	0.8042	**2**
SMCL	$\alpha_c = \eta_c = 0.005$	0.9878	0.8748	0.9373	**2**
CC	$q = 5; \gamma = 5$	**0.9939**	**0.9267**	**0.9681**	**2**
MCKM	$q = 2; \gamma = 0.206$	0.9899	0.8909	0.9475	**2**
	D2 ($n = 2000, p = 2, k = 2$)				
K-Means	$k = 2$	0.9160	0.5839	0.6921	–
SMKM	$k = 2$	0.9160	0.5839	0.6921	–
AP	$\Lambda = -200$	0.9255	0.6176	0.7241	**2**
SMCL	$\alpha_c = \eta_c = 0.005$	0.9965	0.9669	0.9860	**2**
CC	$q = 5; \gamma = 20$	**1.0000**	**1.0000**	**1.0000**	**2**
MCKM	$q = 2; \gamma = 1.7$	0.9995	0.9943	0.9980	**2**
	D3 ($n = 1000, p = 2, k = 2$)				
K-Means	$k = 2$	0.5180	0.0006	-0.0002	–
SMKM	$k = 2$	0.5178	0.0006	-0.0002	–
AP	$\Lambda = -60$	0.5138	0.0003	-0.0004	**2**
SMCL	$\alpha_c = \eta_c = 0.005$	0.5386	0.0029	0.0027	2
CC	$q = 5; \gamma = 14$	**1.0000**	**1.0000**	**1.0000**	**2**
MCKM	$q = 2; \gamma = 10$	**1.0000**	**1.0000**	**1.0000**	**2**
	D4 ($n = 5000, p = 3, k = 2$)				
K-Means	$k = 2$	0.8338	0.4701	0.4318	–
SMKM	$k = 2$	0.8338	0.4701	0.4318	–
AP	$\Lambda = -200$	0.8527	0.5087	0.4877	**2**
SMCL	$\alpha_c = \eta_c = 0.001$	0.9988	0.9878	0.9952	**2**
CC	$q = 5; \gamma = 20$	**1.0000**	**1.0000**	**1.0000**	**2**
MCKM	$q = 2; \gamma = 1.5$	**1.0000**	**1.0000**	**1.0000**	**2**
	D5 ($n = 5000, p = 2, k = 15$)				
K-Means	$k = 15$	0.8600	0.8823	0.7798	–
SMKM	$k = 15$	0.9700	0.9465	0.9379	–
AP	$\Lambda = -10$	**0.9712**	**0.9479**	**0.9402**	**15**
SMCL	$\alpha_c = \eta_c = 0.005$	0.9312	0.9306	0.8395	14
CC	$q = 5; \gamma = 9.5$	0.8314	0.8879	0.7311	**15**
MCKM	$q = 1; \gamma = 0.1$	0.9580	0.9326	0.9148	**15**
	D6 ($n = 5250, p = 2, k = 35$)				
K-Means	$k = 35$	0.9068	0.9489	0.8652	–
SMKM	$k = 35$	0.9890	0.9838	0.9775	–
AP	$\Lambda = -5$	**0.9893**	0.9840	0.9782	**35**
SMCL	$\alpha_c = \eta_c = 0.0001$	0.3058	0.6329	0.0366	7
CC	$q = 5; \gamma = 2.1$	0.8992	0.9544	0.9819	**35**
MCKM	$q = 1; \gamma = 0.05$	**0.9893**	**0.9841**	**0.9783**	**35**

MCKM achieves a well trade-off between clustering quality and efficiency. In terms of quality, MCKM is an outstanding clustering algorithm for recovering a better local minima without given cluster number. In terms of efficiency, for AP, the computation of pairwise similarity between samples and the updates of the three matrices (Responsibility, Availability, and Similarity) is extremely time-consuming, particularly when dealing with a large number of samples, such as D4, D5, D6, HTRU2, and Statlog; For SMCL, a significant amount of computation is required for Eq. (7), (8), (9), (10); Compared with CC, we note that n samples are involved in the CC convex optimization model (11) whereas s^* prototypes are involved in the CM convex optimization model (16), where the number of the multi-prototypes by MPS $s^* \ll n$. Overall, the running time of MCKM is less than that of CC. This is consistent with the complexity analysis in Subsections 4.1 and 4.2. Therefore, both MPS and CM in MCKM are very efficient.

5.4 Performance of the Approximation to the Global Minima of K-Means Problem

In the third set of experiments, we verify the approximation capability of MCKM on the global minima of K-Means problem. The corresponding K-Means errors of the chosen algorithms on all data sets are compared with the optimal errors of the corresponding data sets. Referring to [16], K-Means cost function can equivalently be rewritten as:

$$J_{\mathbf{X}} = \sum_{i=1}^{k} \frac{1}{2|\mathcal{C}_i|} \sum_{j,j' \in \mathcal{C}_i} \|\mathbf{x}_j - \mathbf{x}_{j'}\|^2. \tag{20}$$

For a data set $\mathbf{X}$, the optimal error $J^*_{\mathbf{X}}$ can be calculated using the partition of the cluster from the true label. The corresponding error $J_{\mathbf{X}}$ for an algorithm can be calculated based on the partition of the cluster determined by the algorithm. Table 5 shows the approximation capability of the algorithms by using $|J_{\mathbf{X}} - J^*_{\mathbf{X}}|$. The best results are shown in boldface.

From Table 5, we conclude that MCKM approximate better than the other algorithms in almost all data sets. This is attributed to MPS's better adaptation to the arbitrary shape data sets and CM's superior merging mechanism for the multi-prototypes.

6 Conclusion

In this paper, multi-prototypes convex merging based K-Means clustering algorithm (MCKM) is proposed to recover a better local minima of K-Means problem without the cluster number given first. In the proposed algorithm, a multi-prototypes sampling (MPS) is used to select the appropriate number of multi-prototypes with better adaptation to data distribution. Then, a convex merging (CM) technique is developed to formulate the merging of the multi-prototypes task as a convex optimization problem. Specifically, CM obtains the optimal merging and estimate the correct cluster number. Two theoretical proofs are given to guarantee that the cost of MCKM (MPS+CM) can achieve a constant factor approximation to the optimal cost of the K-Means problem. Experimental results have verified MCKM's effectiveness and efficiency on synthetic and real-world data sets.

It is noteworthy that the results of MCKM are highly sensitive to parameter selection (e.g. ρ, q, and γ). Therefore, for future work, an interesting possibility is to implement the adaptive parameter selection technique for recovering better local minima. Another possibility is to design a new version of MPS with the improved initial point and sampling probabilities for better upper bound approximation.

References

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[2] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 809–821, 2016.

TABLE 2
Running time in seconds on the synthetic data sets. The first two algorithms require the number of clusters be given a priori while the last three algorithms do not. The timings are averaged over 20 trials. The standard deviations are given after the means.

Algorithms	D1	D2	D3	D4	D5	D6
K-Means	0.007±0.001	0.006±0.001	0.010±0.001	0.006±0.001	0.016±0.001	0.025±0.001
SMKM	0.021±0.001	0.016±0.001	0.022±0.001	0.025±0.001	0.131±0.001	0.280±0.002
AP	14.330±0.552	5.315±0.497	1.309±0.172	43.564±3.775	43.427±0.246	48.844±2.681
SMCL	10.485±3.121	5.081±1.930	8.564±0.026	12.354±1.218	14.495±0.115	27.590±3.448
CC	0.634±0.136	0.305±0.001	0.185±0.001	0.568±0.002	3.448±0.227	1.880±0.021
MCKM	0.166±0.001	0.057±0.001	0.115±0.001	0.075±0.001	0.130±0.001	0.109±0.001

TABLE 3
The evaluation of the clustering results of the different algorithms on the five real-world data sets. The best results are shown in boldface.

Algorithms	Parameter	F*	NMI	ARI	k^*
	HTRU2 ($n = 17898, p = 8, k = 2$)				
K-Means	$k = 2$	0.9121	0.3396	0.5318	–
SMKM	$k = 2$	0.9121	0.3395	0.5317	–
AP	$\Lambda = -500$	0.8684	0.2878	0.4344	**2**
SMCL	$\alpha_c = \eta_c = 0.001$	0.9050	0.2754	0.4753	**2**
CC	$q = 8; \gamma = 5$	0.9228	0.3571	0.5484	**2**
MCKM	$q = 2; \gamma = 2$	**0.9637**	**0.5195**	**0.6784**	**2**
	Iris ($n = 150, p = 4, k = 3$)				
K-Means	$k = 3$	0.8227	0.6873	0.6255	–
SMKM	$k = 3$	0.8873	0.7392	0.7148	–
AP	$\Lambda = -5$	0.9007	**0.7777**	0.7405	**3**
SMCL	$\alpha_c = \eta_c = 0.001$	0.7778	0.7337	0.3705	2
CC	$q = 5; \gamma = 1$	0.8955	0.7701	0.7312	**3**
MCKM	$q = 2; \gamma = 0.5$	**0.9008**	0.7578	**0.7430**	**3**
	Wine ($n = 178, p = 13, k = 3$)				
K-Means	$k = 3$	0.9509	0.8356	0.8545	–
SMKM	$k = 3$	0.9495	0.8357	0.8484	–
AP	$\Lambda = -5$	0.9022	0.7325	0.7134	**3**
SMCL	$\alpha_c = \eta_c = 0.001$	0.8179	0.6787	0.4804	2
CC	$q = 5; \gamma = 1.5$	0.9444	0.8252	0.8368	**3**
MCKM	$q = 2; \gamma = 2$	**0.9721**	**0.8926**	**0.9149**	**3**
	X8D5K ($n = 1000, p = 8, k = 5$)				
K-Means	$k = 5$	0.9401	0.9587	0.9152	–
SMKM	$k = 5$	**1.0000**	**1.0000**	**1.0000**	–
AP	$\Lambda = -10$	**1.0000**	**1.0000**	**1.0000**	**5**
SMCL	$\alpha_c = \eta_c = 0.001$	**1.0000**	**1.0000**	**1.0000**	**5**
CC	$q = 5; \gamma = 1$	**1.0000**	**1.0000**	**1.0000**	**5**
MCKM	$q = 2; \gamma = 1$	**1.0000**	**1.0000**	**1.0000**	**5**
	Statlog ($n = 4435, p = 36, k = 6$)				
K-Means	$k = 6$	0.6911	0.6147	0.5305	–
SMKM	$k = 6$	0.6910	0.6148	0.5307	–
AP	$\Lambda = -180$	0.6998	0.5956	0.5206	**6**
SMCL	$\alpha_c = \eta_c = 0.001$	0.4674	0.2610	0.0227	2
CC	$q = 5; \gamma = 13.15$	0.6955	0.5532	0.4294	**6**
MCKM	$q = 2; \gamma = 4$	**0.8279**	**0.6477**	**0.6175**	**6**

[3] N. J. Nilsson, "Artificial intelligence: A modern approach," *Applied Mechanics & Materials*, vol. 263, no. 5, pp. 2829–2833, 2003.

[4] A. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[5] M. R. Rezaee, P. M. J. van der Zwet, B. P. E. Lelieveldt, R. J. van der Geest, and J. H. C. Reiber, "A multiresolution image segmentation technique based on pyramidal segmentation and fuzzy clustering," *IEEE Transactions on Image Processing*, vol. 9, no. 7, pp. 1238–1248, 2000.

[6] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding, and C.-T. Lin, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, 2017.

[7] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[8] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010, award winning papers from the 19th International Conference on Pattern Recognition (ICPR).

[9] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," *Applied Intelligence*, vol. 48, no. 12, pp. 4743–4759, 2018.

[10] F. Nie, J. Xue, D. Wu, R. Wang, H. Li, and X. Li, "Coordinate descent method for k-means," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 5, pp. 2371–2385, 2022.

[11] J. Peña, J. Lozano, and P. Larrañaga, "An empirical comparison of four initialization methods for the k-means algorithm," *Pattern Recognition Letters*, vol. 20, no. 10, pp. 1027–1040, 1999.

[12] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, vol. 8, 2007, pp. 1027–1035.

[13] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, "Scalable k-means++," *Proc. VLDB Endow.*, vol. 5, 03 2012.

[14] S. Lattanzi and C. Sohler, "A better k-means++ algorithm via local search," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3662–3671.

[15] J. Peng and Y. Wei, "Approximating k-means-type clustering via semidefinite programming," *SIAM journal on optimization*, vol. 18, no. 1, pp. 186–205, 2007.

[16] S. Dasgupta, *The hardness of k-means clustering*. Department of Computer Science and Engineering, University of California , 2008.

[17] P. Fränti and O. Virmajoki, "Iterative shrinking method for clustering problems," *Pattern Recognition*, vol. 39, no. 5, pp. 761–775, 2006.

[18] M. Muhr and M. Granitzer, "Automatic cluster number selection using a split and merge k-means approach," in *2009 20th International Workshop on Database and Expert Systems Application*. IEEE, 2009, pp. 363–367.

[19] J. Lei, T. Jiang, K. Wu, H. Du, G. Zhu, and Z. Wang, "Robust k-means algorithm with automatically splitting and merging clusters and its applications for surveillance data," *Multimedia Tools and Applications*, vol. 75, no. 19, pp. 12 043–12 059, 2016.

[20] H. Ismkhan, "Ik-means-+: An iterative clustering algorithm based on an enhanced version of the k-means," *Pattern Recognition*, vol. 79, pp. 402–413, 2018.

[21] M. Capó, A. Pérez, and J. A. Lozano, "An efficient split-merge restart for the k-means algorithm," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 4, pp. 1618–1627, 2022.

[22] H. Xiong, J. Wu, and J. Chen, "K-means clustering versus validation measures: a data-distribution perspective," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 318–331, 2008.

[23] S. Dasgupta and L. J. Schulman, "A probabilistic analysis of em for mixtures of separated, spherical gaussians," *Journal of Machine Learning Research*, vol. 8, pp. 203–226, 2007.

[24] R.-D. Buhai, Y. Halpern, Y. Kim, A. Risteski, and D. Sontag, "Empirical study of the benefits of overparameterization in learning latent variable models," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1211–1219.

[25] J. Liang, L. Bai, C. Dang, and F. Cao, "The k-means-type algorithms versus imbalanced data distributions," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 728–745, 2012.

TABLE 4
Running time in seconds on the real-world data sets. The first two algorithms require the number of clusters be given a priori while the last three algorithms do not. The timings are averaged over 20 trials. The standard deviations are given after the means.

Algorithms	HTRU2	Iris	Wine	X8D5K	Statlog
K-Means	0.035±0.002	0.004±0.001	0.003±0.001	0.005±0.001	0.019±0.001
SMKM	0.085±0.001	0.020±0.001	0.016±0.001	0.029±0.001	0.049±0.001
AP	1506.3±7.8	0.056±0.020	0.069±0.017	1.055±0.051	36.112±0.485
SMCL	131.727±5.836	0.110±0.001	0.170±0.001	0.863±0.001	70.035±8.425
CC	202.409±1.582	0.180±0.001	0.116±0.001	0.190±0.001	55.175±0.667
MCKM	0.402±0.004	0.150±0.010	0.107±0.001	0.100±0.001	0.279±0.001

TABLE 5
Performance of the approximation of the chosen algorithms on all data sets measured by $|J_{\mathbf{X}} - J^*_{\mathbf{X}}|$. The best results are shown in boldface.

Data sets	D1	D2	D3	D4	D5	D6	HTRU2	Iris	Wine	X8D5K	Statlog
$J^*_{\mathbf{X}}$	60.5422	51.2649	54.8162	221.6124	8.0299	3.8056	778.7111	3.9087	24.9993	28.2419	974.7959
K-Means	6.9862	5.5864	21.8904	60.6594	3.6511	1.7755	172.2072	5.0353	4.9308	5.2891	343.7963
SMKM	6.9862	5.5864	21.8904	60.6594	0.5653	0.0337	177.6846	0.4096	0.5192	**0**	348.9786
AP	5.0812	5.5300	21.8594	59.9449	0.5579	0.0315	150.9821	0.3911	0.3951	**0**	303.6403
SMCL	2.0607	0.2956	2.8287	0.0697	1.6028	93.1209	162.1329	2.1631	39.7478	**0**	1113.925
CC	**1.4758**	**0.1841**	**0**	**0**	8.9249	2.5693	174.2149	0.3648	0.3584	**0**	359.1041
MCKM	2.0926	0.2956	**0**	**0**	**0.1814**	**0.0313**	**41.3547**	**0.3037**	**0.3316**	**0**	**138.0041**

[26] C.-D. Wang, J.-H. Lai, C. Y. Suen, and J.-Y. Zhu, "Multi-exemplar affinity propagation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 9, pp. 2223–2237, 2013.

[27] L. Zhang and A. Amini, "Label consistency in overfitted generalized k-means," in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 7965–7977.

[28] C.-D. Wang, J.-H. Lai, and J.-Y. Zhu, "Graph-based multiprototype competitive learning and its applications," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 934–946, 2011.

[29] Y. Lu, Y.-M. Cheung, and Y. Y. Tang, "Self-adaptive multiprototype-based competitive learning approach: A k-means-type algorithm for imbalanced data clustering," *IEEE transactions on cybernetics*, vol. 51, no. 3, pp. 1598–1612, 2019.

[30] M. Chen and X. Li, "Concept factorization with local centroids," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 11, pp. 5247–5253, 2021.

[31] F. Lindsten, H. Ohlsson, and L. Ljung, "Clustering using sum-of-norms regularization: With application to particle filter output computation," in *2011 IEEE Statistical Signal Processing Workshop (SSP)*, 2011, pp. 201–204.

[32] C. Zhu, H. Xu, C. Leng, and S. Yan, "Convex optimization procedure for clustering: Theoretical revisit," *Advances in Neural Information Processing Systems*, vol. 2, pp. 1619–1627, 01 2014.

[33] A. Panahi, D. Dubhashi, F. D. Johansson, and C. Bhattacharyya, "Clustering by sum of norms: Stochastic incremental algorithm, convergence and cluster recovery," in *International conference on machine learning*. PMLR, 2017, pp. 2769–2777.

[34] E. C. Chi and K. Lange, "Splitting methods for convex clustering," *Journal of Computational and Graphical Statistics*, vol. 24, no. 4, pp. 994–1013, 2015.

[35] D. Sun, K.-C. Toh, and Y. Yuan, "Convex clustering: Model, theoretical guarantee and efficient algorithm," *Journal of Machine Learning Research*, vol. 22, no. 9, pp. 1–32, 2021.

[36] W. Qian, Y. Zhang, and Y. Chen, "Structures of spurious local minima in k-means," *IEEE Transactions on Information Theory*, vol. 68, no. 1, pp. 395–422, 2022.

[37] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[38] K. P. Sinaga and M.-S. Yang, "Unsupervised k-means clustering algorithm," *IEEE Access*, vol. 8, pp. 80 716–80 727, 2020.

[39] M. B. Gorzałczany and F. Rudziński, "Generalized self-organizing maps for automatic determination of the number of clusters and their multiprototypes in cluster analysis," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 2833–2845, 2017.

[40] L. Kaufmann and P. Rousseeuw, "Clustering by means of medoids," *Data Analysis based on the L1-Norm and Related Methods*, pp. 405–416, 1987.

[41] O. F. Reyes-Galaviz and W. Pedrycz, "Enhancement of the classification and reconstruction performance of fuzzy c-means with refinements of prototypes," *Fuzzy Sets and Systems*, vol. 318, pp. 80–99, 2017.

[42] T. Ouyang, W. Pedrycz, O. F. Reyes-Galaviz, and N. J. Pizzi, "Granular description of data structures: A two-phase design," *IEEE Transactions on Cybernetics*, vol. 51, no. 4, pp. 1902–1912, 2021.

[43] J. K. Parker and L. O. Hall, "Accelerating fuzzy-c means using an estimated subsample size," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 5, pp. 1229–1244, 2013.

[44] J.-P. Mei, Y. Wang, L. Chen, and C. Miao, "Large scale document categorization with fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 5, pp. 1239–1251, 2016.

[45] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.

Appendix A
Proof of Theorem 3

Here, we prove that the multi-prototypes obtained by MPS can achieve a constant factor approximation to the optimal cost of K-Means problem.

Proof. Let $\mathbf{v}(\mathbf{x})$ be the prototype to which $\mathbf{x}$ belongs and $\mathbf{v}^*(\mathbf{x})$ be the optimal prototype to which $\mathbf{x}$ belongs. Assume that MPS has chosen s samples, $1 \leq s \leq n$, as the prototypes $\mathbf{V}$, and we continue to choose the next prototype $\mathbf{x}^{(s+1)}$ from $\mathbf{X}$. The probability of being selected is precisely $D(\mathbf{x}^{(s+1)})^2 / \sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2$. After adding the prototype $\mathbf{x}^{(s+1)}$, any sample $\mathbf{x}$ will contribute $\min(D(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^{(s+1)}\|)^2$ to the objective function. Therefore,

$$E[J_{\mathbf{X}}^{\text{MPS}}] = \sum_{\mathbf{x}^{(s+1)}\in\mathbf{X}} \frac{D(\mathbf{x}^{(s+1)})^2}{\sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2} \sum_{\mathbf{x}\in\mathbf{X}} \min(D(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^{(s+1)}\|)^2.$$

According to the termination condition of MPS, since $\mathbf{x}^{(s+1)}$ is selected, we have:

$$\frac{R(s) - R(s+1)}{R(s)} \geq \varepsilon \Leftrightarrow (1-\varepsilon)R(s) \geq R(s+1).$$

Based on (13), we have $E[J_{\mathbf{X}}^{\text{MPS}}] \leq (1-\varepsilon)\sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2$. By the power-mean inequality $\|\mathbf{x} - \mathbf{v}(\mathbf{x})\|^2 \leq 2\|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\|^2 + 2\|\mathbf{v}^*(\mathbf{x}) - \mathbf{v}(\mathbf{x})\|^2$, we have

$$\begin{aligned} E[J_{\mathbf{X}}^{\text{MPS}}] &\leq (1-\varepsilon)\sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2, \\ &\leq 2(1-\varepsilon)\sum_{\mathbf{x}\in\mathbf{X}} (\|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\|^2 + \|\mathbf{v}^*(\mathbf{x}) - \mathbf{v}(\mathbf{x})\|^2), \\ &= 2(1-\varepsilon)(J_{\mathbf{X}}{}^{\text{opt}} + \sum_{\mathbf{x}\in\mathbf{X}} \|\mathbf{v}^*(\mathbf{x}) - \mathbf{v}(\mathbf{x})\|^2). \end{aligned}$$

Assume that MPS continues to run and terminates after the algorithm has sampled s^* prototypes. Because MPS adopts D^2 sampling method, we have for any $\mathbf{x}$, $D(\mathbf{x}^{(s^*+1)})^2 \geq D(\mathbf{x})^2$. Accordingly, we have:

$$D(\mathbf{x}^{(s^*+1)}) \geq D(\mathbf{x}) \geq \left| \|\mathbf{v}(\mathbf{x}) - \mathbf{v}^*(\mathbf{x})\| - \|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\| \right|.$$

Then, define $\mathbf{X}_a = \{\mathbf{x} | \|\mathbf{v}(\mathbf{x}) - \mathbf{v}^*(\mathbf{x})\| \geq \|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\|\}$ and $\mathbf{X}_b = \{\mathbf{X} \setminus \mathbf{X}_a\}$. Let $n_a = |\mathbf{X}_a|$ represents the cardinality of the set $\mathbf{X}_a$. Combining the above derivations, we have:

$$\begin{aligned} E[J_{\mathbf{X}}^{\text{MPS}}] &\leq 2(1-\varepsilon)(J_{\mathbf{X}}{}^{\text{opt}} + \sum_{\mathbf{x}\in\mathbf{X}} \|\mathbf{v}^*(\mathbf{x}) - \mathbf{v}(\mathbf{x})\|^2) \\ &\leq 2(1-\varepsilon)(J_{\mathbf{X}}{}^{\text{opt}} + \sum_{\mathbf{x}\in\mathbf{X}_b} \|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\|^2 \\ &\quad + \sum_{\mathbf{x}\in\mathbf{X}_a} \left[\|\mathbf{x} - \mathbf{v}^*(\mathbf{x})\| + D(\mathbf{x}^{(s^*+1)})\right]^2). \end{aligned}$$

Since MPS terminates at s^* steps, we have:

$$\frac{R(s^*) - R(s^*+1)}{R(s^*)} \leq \varepsilon \Leftrightarrow (1-\varepsilon)R(s^*) \leq R(s^*+1)$$

which is equivalent to:

$$\sum_{\mathbf{x}\in\mathbf{X}} \{D(\mathbf{x})^2 - \min(D(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^{(s^*+1)}\|)^2\} \leq \varepsilon \sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2.$$

Here, $\mathbf{X}$ is divided into three parts according to the following rules: 1) $\mathbf{x} \in \mathbf{X}_1$, if $\min(D(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^{(s^*+1)}\|)^2 = D(\mathbf{x})^2$; 2) $\mathbf{x} \in \mathbf{X}_2$, if $\min(D(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^{(s^*+1)}\|)^2 = \|\mathbf{x} - \mathbf{x}^{(s^*+1)}\|)^2$; 3) $\mathbf{X}_3 = \{\mathbf{x}^{(s^*+1)}\}$. Evidentially,

$$\begin{aligned} &\sum_{\mathbf{x}\in\mathbf{X}} \{D(\mathbf{x})^2 - \min(D(\mathbf{x}), \|\mathbf{x} - \mathbf{x}^{(s^*+1)}\|)^2\} \\ &= 0 + \sum_{\mathbf{x}\in\mathbf{X_2}} \{D(\mathbf{x})^2 - \|\mathbf{x} - \mathbf{x}^{(s^*+1)}\|^2\} + D(\mathbf{x}^{(s^*+1)})^2. \end{aligned}$$

Therefore, we have:

$$D(\mathbf{x}^{(s^*+1)})^2 \leq \varepsilon \sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2.$$

The last step is summarized as follows:

$$E[J_{\mathbf{X}}^{\text{MPS}}] \leq 2(1-\varepsilon)(3J_{\mathbf{X}}{}^{\text{opt}} + 2\varepsilon n_a \sum_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x})^2).$$

The proof process is complete. □

Appendix B
The ADMM for solving CM.

Algorithm 3 ADMM for Solving (16)

```
Input: The multi-prototypes $\mathbf{V}_{\text{MPS}}$, the number of the multi-prototypes $s^*$, the number of neighboring samples $q$, a positive constant $\kappa$, the tuning parameter $\gamma$, the termination $\eta$, $\boldsymbol{\lambda}^0$, and new variables $\boldsymbol{y}^0$;
Output: The optimal solutions, $\boldsymbol{\mu}_1^*, ..., \boldsymbol{\mu}_{s^*}^*$.
set the iteration number $t = 0$
$\bar{\mathbf{V}}_{\text{MPS}}$ is the average column of $\mathbf{V}_{\text{MPS}}$;
compute $E$ and $W$ based on Eq. (12);
$\sigma_l = \frac{\gamma w_l}{\nu}, l \in E$;
repeat
    compute $\boldsymbol{Z}$ with $\boldsymbol{z}_i = \mathbf{v}_i + \sum_{l_1=i}(\boldsymbol{\lambda}_l^m + \nu\boldsymbol{y}_l^m) - \sum_{l_2=i}(\boldsymbol{\lambda}_l^m + \nu\boldsymbol{y}_l^m), i = 1, 2, ..., s^*$;
    update $\boldsymbol{\mu}^{t+1} = \frac{1}{1+c^*\nu}\boldsymbol{Z} + \frac{c^*\nu}{1+c^*\nu}\bar{\mathbf{V}}_{\text{MPS}}$;
    update $\boldsymbol{y}^{t+1}$ with $\boldsymbol{y}_l^{t+1} = \text{prox}_{\sigma_l\|\cdot\|}(\boldsymbol{\mu}_{l_1}^{t+1} - \boldsymbol{\mu}_{l_2}^{t+1} - \nu^{-1}\boldsymbol{\lambda}_l^t), l \in E$;
    update $\boldsymbol{\lambda}^{t+1}$ with $\boldsymbol{\lambda}_l^{t+1} = \boldsymbol{\lambda}_l^t + \nu(\boldsymbol{y}_l^{t+1} - \boldsymbol{\mu}_{l_1}^{t+1} + \boldsymbol{\mu}_{l_2}^{t+1}), l \in E$;
    $t = t + 1$;
until Stopping criterion is met
Obtain the optimal solutions, $\boldsymbol{\mu}_1^*, ..., \boldsymbol{\mu}_{s^*}^*$.
```

The objective of CM (16) is recast as the equivalent constrained problem:

$$\begin{aligned} \min_{\boldsymbol{\mu}_1,...,\boldsymbol{\mu}_{s^*}\in\mathbb{R}^p} \frac{1}{2}\sum_{i=1}^{s^*} \|\boldsymbol{\mu}_i - \mathbf{v}_i\|^2 + \gamma \sum_{l\in E} w_l \|\boldsymbol{y}_l\|_2, \\ s.t. \quad \boldsymbol{\mu}_{l_1} - \boldsymbol{\mu}_{l_2} - \boldsymbol{y}_l = 0, \end{aligned} \tag{21}$$

where $l = (l_1, l_2)$ with $l_1 < l_2$, and $\boldsymbol{y}_l = \boldsymbol{\mu}_{l_1} - \boldsymbol{\mu}_{l_2}$ is introduced to simplify the penalty terms. For the constrained optimization problem (21), the augmented Lagrangian is given by:

$$\mathcal{L}_\nu(\boldsymbol{\mu}, \boldsymbol{y}, \boldsymbol{\lambda}) = \frac{1}{2}\sum_{i=1}^{s^*}\|\boldsymbol{\mu}_i - \mathbf{v}_i\|^2 + \gamma\sum_{l\in E} w_l\|\boldsymbol{y}_l\|_2 + \sum_{l\in E}\langle\boldsymbol{\lambda}_l, \boldsymbol{y}_l - \boldsymbol{\mu}_{l_1} + \boldsymbol{\mu}_{l_2}\rangle + \frac{\nu}{2}\sum_{l\in E}\|\boldsymbol{y}_l - \boldsymbol{\mu}_{l_1} + \boldsymbol{\mu}_{l_2}\|_2^2. \tag{22}$$

ADMM minimizes the augmented Lagrangian by the following iterative process:

$$\begin{aligned}\boldsymbol{\mu}^{t+1} &= \arg\min_{\boldsymbol{\mu}} \mathcal{L}_\nu(\boldsymbol{\mu}, \boldsymbol{y}^t, \boldsymbol{\lambda}^t);\\ \boldsymbol{y}^{t+1} &= \arg\min_{\boldsymbol{y}} \mathcal{L}_\nu(\boldsymbol{\mu}^{t+1}, \boldsymbol{y}, \boldsymbol{\lambda}^t);\\ \boldsymbol{\lambda}_l^{t+1} &= \boldsymbol{\lambda}_l^t + \nu(\boldsymbol{y}_l^{t+1} - \boldsymbol{\mu}_{l_1}^{t+1} + \boldsymbol{\mu}_{l_2}^{t+1}), l \in E,\end{aligned} \tag{23}$$

where t is the iteration number. ADMM for solving (16) is summarized in Algorithm 3. For other solvers, please refer to [34], [35],

Appendix C
Proof of Theorem 4

Here, we prove that the final clustering result of MCKM can achieve a constant factor approximation to the optimal cost of K-Means problem.

Proof. Let $\mathbf{v}_{\text{MPS}}(\mathbf{x})$ be the prototype to which $\mathbf{x}$ belongs in the multi-prototypes obtained by MPS, and $\mathcal{C}^{\text{MPS}} = \{\mathcal{C}_1^{\text{MPS}}, \mathcal{C}_2^{\text{MPS}}, ..., \mathcal{C}_{s^*}^{\text{MPS}}\}$ is the corresponding clusters. Let $|\mathcal{C}_i^{\text{MPS}}|$ represents the cardinality of $\mathcal{C}_i^{\text{MPS}}$ for $i = 1, 2, ..., s^*$. $\mathbf{v}^*(\mathbf{x})$ be the optimal prototype to which $\mathbf{x}$ belongs. $\boldsymbol{\mu}^* = \{\boldsymbol{\mu}_1^*, ..., \boldsymbol{\mu}_{s^*}^*\}$ is the optimal solution obtained by CM. Assume that the merge result of multi-prototypes of MPS is obtained by CM for a given suitable γ, and CM evaluates the correct number of clusters, k^*

After the optimal solutions $\boldsymbol{\mu}_1^*, ..., \boldsymbol{\mu}_{s^*}^*$ of (11) are obtained, the multi-prototypes of MPS are assigned to be in one cluster if and only if their optimal solutions $\boldsymbol{\mu}^*$ are the same. Finally, the samples are assigned to its corresponding solution $\boldsymbol{\mu}_{\mathbf{x}}^*$. In detail, the solution of $\mathbf{x}_j$ is $\boldsymbol{\mu}_i^*$, if the solution of $\mathbf{v}_{\text{MPS}}(\mathbf{x}_j)$ is $\boldsymbol{\mu}_i^*$ for $i = 1, 2, ..., s^*, j = 1, 2, ..., n$. Therefore,

$$J_{\mathbf{X}}^{\text{opt}} = \sum_{j=1}^{n}\|\mathbf{x}_j - \mathbf{v}^*(\mathbf{x}_j)\|^2 \leq \sum_{j=1}^{n}\|\mathbf{x}_j - \boldsymbol{\mu}_{\mathbf{x}_j}^*\|^2$$

As analyzed above, the objective function of MCKM is clearly given by:

$$J_{\mathbf{X}}^{\text{MCKM}} = \sum_{j=1}^{n}\|\mathbf{x}_j - \boldsymbol{\mu}_{\mathbf{x}_j}^*\|^2.$$

By the power-mean inequality $\|\mathbf{x}_j - \boldsymbol{\mu}_{\mathbf{x}_j}^*\|^2 \leq 2\|\mathbf{x}_j - \mathbf{v}_{\text{MPS}}(\mathbf{x}_j)\|^2 + 2\|\mathbf{v}_{\text{MPS}}(\mathbf{x}_j) - \boldsymbol{\mu}_{\mathbf{x}_j}^*\|^2$ for $j = 1, 2, ..., n$, we have:

$$\begin{aligned}J_{\mathbf{X}}^{\text{opt}} &= \sum_{j=1}^{n}\|\mathbf{x}_j - \mathbf{v}^*(\mathbf{x}_j)\|^2 \leq J_{\mathbf{X}}^{\text{MCKM}}\\ &\leq 2(\sum_{j=1}^{n}\|\mathbf{x}_j - \mathbf{v}_{\text{MPS}}(\mathbf{x}_j)\|^2 + \|\mathbf{v}_{\text{MPS}}(\mathbf{x}_j) - \boldsymbol{\mu}_{\mathbf{x}_j}^*\|^2)\\ &\leq 2(\sum_{j=1}^{n}\|\mathbf{x}_j - \mathbf{v}_{\text{MPS}}(\mathbf{x}_j)\|^2) + 2n_b(\sum_{i=1}^{s^*}\|\mathbf{v}_i - \boldsymbol{\mu}_i^*\|^2)\end{aligned}$$

where $n_b = \max_{1\leq i\leq s^*}|\mathcal{C}_i^{\text{MPS}}|$. Then, based on the CM model (16), we add a term to the inequality above, and we have:

$$\begin{aligned}J_{\mathbf{X}}^{\text{MCKM}} &\leq 2(\sum_{j=1}^{n}\|\mathbf{x}_j - \mathbf{v}_{\text{MPS}}(\mathbf{x}_j)\|^2) + 2n_b(\sum_{i=1}^{s^*}\|\mathbf{v}_i - \boldsymbol{\mu}_i^*\|^2)\\ &\leq 2(\sum_{j=1}^{n}\|\mathbf{x}_j - \mathbf{v}_{\text{MPS}}(\mathbf{x}_j)\|^2) + 2n_b(\sum_{i=1}^{s^*}\|\mathbf{v}_i - \boldsymbol{\mu}_i^*\|^2\\ &\quad + \gamma\sum_{i<z} w_{iz}\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_z\|)\\ &= 2J_{\mathbf{X}}^{\text{MPS}} + 2n_b J_{\mathbf{V}_{\text{MPS}}}^{\text{CM}}.\end{aligned}$$

The last step is summarized as follows:

$$J_{\mathbf{X}}^{\text{opt}} \leq J_{\mathbf{X}}^{\text{MCKM}} \leq 2J_{\mathbf{X}}^{\text{MPS}} + 2n_b J_{\mathbf{V}_{\text{MPS}}}^{\text{CM}}.$$

The proof process is complete. □