entropy

MDPI

Article

Fast Fusion Clustering via Double Random Projection

Hongni Wang [1,†], Na Li [1,†], Yanqiu Zhou [2], Jingxin Yan [3], Bei Jiang [4], Linglong Kong [4,*] and Xiaodong Yan [5,*]

[1] School of Statistics and Mathematics, Shandong University of Finance and Economics, Jinan 250014, China; wanghongni@mail.sdufe.edu.cn (H.W.); lina@sdufe.edu.cn (N.L.)
[2] School of Science, Guangxi University of Science and Technology, Liuzhou 545006, China; 100002146@gxust.edu.cn
[3] Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China; yanjingxin22@mails.ucas.ac.cn
[4] Department of Mathematical and Statistical Sciences, University of Alberta, Edmonton, AB T6G 2G1, Canada; bei1@ualberta.ca
[5] Zhongtai Securities Institute for Financial Studies, Shandong University, Jinan 250100, China
* Correspondence: lkong@ualberta.ca (L.K.); yanxiaodong@sdu.edu.cn (X.Y.)
† These authors contributed equally to this work.

Abstract: In unsupervised learning, clustering is a common starting point for data processing. The convex or concave fusion clustering method is a novel approach that is more stable and accurate than traditional methods such as *k*-means and hierarchical clustering. However, the optimization algorithm used with this method can be slowed down significantly by the complexity of the fusion penalty, which increases the computational burden. This paper introduces a random projection ADMM algorithm based on the Bernoulli distribution and develops a double random projection ADMM method for high-dimensional fusion clustering. These new approaches significantly outperform the classical ADMM algorithm due to their ability to significantly increase computational speed by reducing complexity and improving clustering accuracy by using multiple random projections under a new evaluation criterion. We also demonstrate the convergence of our new algorithm and test its performance on both simulated and real data examples.

Keywords: unsupervised learning; random projection; ADMM algorithm; fusion clustering

Citation: Wang, H.; Li, N.; Zhou, Y.; Yan, J.; Jiang, B.; Kong, L.; Yan, X. Fast Fusion Clustering via Double Random Projection. *Entropy* **2024**, *26*, 376. https://doi.org/10.3390/e26050376

Academic Editor: Sotiris Kotsiantis

Received: 12 March 2024
Revised: 25 April 2024
Accepted: 25 April 2024
Published: 28 April 2024

1. Introduction

Clustering is a pivotal technique in unsupervised learning, applied extensively across various scientific and technological fields that handle large datasets. Clustering also plays a crucial role in data labelling, which sets the stage for the application of artificial intelligence and machine learning models [1,2] on the organized data to perform predictive analytics and classification tasks. Traditional clustering algorithms like k-means, Gaussian mixture models, and hierarchical clustering often face stability challenges due to their concave optimization formulations, which can lead to variability in results due to factors such as initial conditions or data outliers [3–5]. Recent advancements in convex or concave fusion methods have shown promise in enhancing stability, achieving more consistent global or local optimality and reliable estimation of cluster centers and counts through sparse-inducing penalties on pairwise centers [6–9]. For clustering high-dimensional data, the data can be mapped into a high-dimensional feature space (kernel space) for processing [10], or clustering can be achieved by optimizing a smooth and continuous objective function that is based on robust statistics [11]. This paper introduces a comprehensive empirical validation of these methods across simulation studies and real data analysis, detailing their improved stability over traditional methods and the practical implications of these advancements.

In fusion clustering, p-dimensional observations $\boldsymbol{X}_i$, $i = 1, \ldots, n$ are each parameterized by their own centroid $\boldsymbol{\mu}_i$. These centroids are estimated under the assumption that

all observations can be grouped into K clusters $\mathcal{G}_1, \ldots, \mathcal{G}_K$, such that for $i \in \mathcal{G}_k$, $\boldsymbol{\mu}_i = \boldsymbol{\rho}_k$, where $\boldsymbol{\rho}_k$ represents the cluster center for observations in cluster $\mathcal{G}_k$. Fusion clustering aims to concurrently estimate the cluster centroids $\boldsymbol{\rho}_k$ and the partitions $\mathcal{G}_k$ by minimizing the following objectives

$$\frac{1}{2}\sum_{i=1}^{n}\|\boldsymbol{X}_i - \boldsymbol{\mu}_i\|^2 + \sum_{i<j} p_\lambda(\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|_\tau). \tag{1}$$

The penalty function $p_\lambda(\|\cdot\|\tau)$ is used to control the complexity of the model, and it is determined by the tuning parameter λ. The form of the norm used is represented by $\|\cdot\|\tau$. This penalty function is typically used in fusion clustering to encourage sparsity in the estimated cluster centroids.

The penalty function $p_\lambda(\|\cdot\|\tau)$ controls the complexity of the model and is determined by the tuning parameter λ. The norm used is $\|\cdot\|\tau$. The penalty function is typically used in fusion clustering to promote sparsity in cluster centroids.

Convex fusion clustering methods have been widely studied due to their computational simplicity and ability to find global optima. These methods often employ ℓ_1, ℓ_2, or ℓ_∞ penalties as the penalty function $p_\lambda(\|\cdot\|_\tau)$ [12–17]. However, convex fusion can lead to biased estimates of the individual centroids, resulting in solutions with a large number of dense clusters [18,19]. To address this issue, researchers have proposed using concave fusion clustering methods, such as those using minimax concave penalties (MCPs) [20], truncated Lasso penalties (TLPs) [8], and arbitrary concave penalties.

While robust, convex and concave fusion clustering methods are computationally demanding with a $\mathcal{O}(n^2 p)$ complexity, which can limit their practicality in scenarios involving large sample sizes n and high-dimensional datasets p. This article proposes a strategy for overcoming this limitation using random projection techniques [21–24]. The approach involves the construction of a random diagonal matrix whose diagonal elements are sourced from a binary distribution. This matrix is then projected onto the pairwise component of the fusion method. By doing so, the number of pairwise differences between individual centroids, $\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|$, is substantially reduced. This reduction not only decreases the computational load but also maintains the integrity of the clustering process, enhancing the algorithm's scalability without excessively increasing the operational overhead. We provide empirical evidence demonstrating that this method significantly reduces the computational time while preserving the clustering quality, as shown in our simulation section.

In unsupervised learning, rapid clustering processes are crucial for handling large datasets efficiently. Our study introduces a novel approach to fusion clustering to enhance computational speed without compromising accuracy. Our contributions are summarized as follows: (1) We propose using random projection techniques to simplify the fusion aspect of clustering, effectively diminishing the pairwise centroids discrepancies and significantly boosting computational efficiency by minimizing the fusion step's complexity. (2) We have developed a novel double recursive random projection ADMM method designed for efficient high-dimensional fusion clustering, improving the accuracy of clustering.

In the remainder of this paper, the proposed new ADMM algorithm will be described in Section 2. This section will also include an analysis of the computational complexity and convergence of the algorithm. It will also include a strategy for improving cluster accuracy. The finite-sample properties of the proposed new ADMM algorithm will be evaluated through simulation studies in Section 3, and the method will be demonstrated using a real data example in Section 4. Concluding remarks will be presented in Section 5, and technical proofs will be provided in the Appendices A and B.

2. Methodology

To improve convex or concave fusion clustering efficiency, we propose an extension of the classical ADMM algorithm based on a random projection called RP-ADMM. A random projection can significantly reduce the time and computational resources needed to analyze

high-dimensional data, making it suitable for large datasets and real-time processing. In this section, we will discuss the RP-ADMM algorithm's computational complexity and convergence.

2.1. Random Projection Based ADMM

Previous ADMM algorithms for convex or concave fusion clustering [6,8] have suffered from a high computational burden due to the need to consider all $n(n-1)/2$ pairwise differences between individual centroids. This is represented by the fusion matrix $\mathcal{E} = \{(\boldsymbol{e}_i - \boldsymbol{e}_j), i < j\}^{\mathrm{T}}_{\frac{n(n-1)}{2} \times n}$, where $\boldsymbol{e}_i$ is the ith unit vector with a 1 in the ith position and 0s elsewhere, and $\boldsymbol{e}_i - \boldsymbol{e}_j$ can be interpreted as the difference between the ith and jth individual centroids. The computational complexity of this approach is $\mathcal{O}(n^2)$, which becomes infeasible for large sample sizes n.

Bernoulli distribution-based random projections ADMM

It is worth noting that pairwise differences between individual centroids can be deduced from other differences. For example, if we know that $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2 = \mathbf{0}$ and $\boldsymbol{\mu}_2 - \boldsymbol{\mu}_3 = \mathbf{0}$, we can conclude that $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_3 = \mathbf{0}$. This means that it may be unnecessary to consider the row $\boldsymbol{e}_1 - \boldsymbol{e}_3$ in $\mathcal{E}$. To reduce the computational burden of convex or concave fusion clustering, we propose a random projection approach. This only considers a small subset of the $n(n-1)/2$ pairwise differences between individual centroids. This is achieved by generating indicators π_{ij} from a Bernoulli distribution with probability α. We then form a random matrix $\boldsymbol{\Pi}$, which is a diagonal matrix with diagonal elements $(\pi_{12}, \cdots, \pi_{1n}, \pi_{23}, \cdots, \pi_{2n}, \cdots, \pi_{(n-1)n})^{\mathrm{T}}$. If $\pi_{ij} = 1$, the difference between $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_j$ is taken into account; if $\pi_{ij} = 0$, it is not considered. The probability α controls the size of the subset of pairwise differences considered. The matrix $\boldsymbol{\Pi}\mathcal{E}$ can be seen as a projection of $\mathcal{E}$ onto a sparse matrix. This is with about $n(n-1)(1-\alpha)/2$ rows being zero vectors and about $n(n-1)\alpha/2$ ones being nonzero vectors. This projection is based on a Bernoulli distribution. Finally, we form a new fusion matrix $\boldsymbol{\Omega}$ by deleting the rows of zero vectors in $\boldsymbol{\Pi}\mathcal{E}$. The new fusion matrix is given by $\boldsymbol{\Omega} = (\boldsymbol{\Omega}_1, \cdots, \boldsymbol{\Omega}_\kappa)^{\mathrm{T}}$, where $\boldsymbol{\Omega}_j$, $j = 1, \cdots, \kappa$, denotes jth row vector of $\boldsymbol{\Omega}$.

We just consider $\tau = 2$ in (1) for simplicity and propose a random projection-based fusion criterion by

$$\ell_p(\boldsymbol{\mu}; \lambda) = \frac{1}{2} \sum_{i=1}^{n} \|\boldsymbol{X}_i - \boldsymbol{\mu}_i\|^2 + \sum_{i<j} \pi_{ij} p_\lambda(\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|), \tag{2}$$

where $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_n)^{\mathrm{T}}_{n \times p}$. Furthermore, the objective function in (2) is equivalent to

$$\begin{aligned} \widetilde{\ell}_p(\boldsymbol{\mu}, \boldsymbol{\phi}; \lambda) &= \frac{1}{2} \|\boldsymbol{X} - \boldsymbol{\mu}\|_F^2 + \sum_{j=1}^{\kappa} p_\lambda(||\boldsymbol{\phi}_j||), \\ \text{subject to } \boldsymbol{\Omega}\boldsymbol{\mu} - \boldsymbol{\phi} &= 0, \end{aligned} \tag{3}$$

where $\boldsymbol{X} = (\boldsymbol{X}_1, \cdots, \boldsymbol{X}_n)^{\mathrm{T}}$, $\boldsymbol{\phi} = (\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_\kappa)^{\mathrm{T}}_{\kappa \times p}$. Under the constraints in (3), the augmented Lagrangian $Q(\boldsymbol{\mu}, \boldsymbol{\phi}, \boldsymbol{\eta}; \lambda)$ has the form

$$\widetilde{\ell}_p(\boldsymbol{\mu}, \boldsymbol{\phi}; \lambda) + \sum_{j=1}^{\kappa} \boldsymbol{\eta}_j^{\mathrm{T}}(\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Omega}_j - \boldsymbol{\phi}_j) + \frac{\varphi}{2} \|\boldsymbol{\Omega}\boldsymbol{\mu} - \boldsymbol{\phi}\|_F^2, \tag{4}$$

where the dual variables $\boldsymbol{\eta} = (\boldsymbol{\eta}_1, \cdots, \boldsymbol{\eta}_\kappa)^{\mathrm{T}}_{\kappa \times p}$ are Lagrange multipliers, and φ is a tuning parameter. Under the iterative value $\boldsymbol{\mu}^{(m)}$ and $\boldsymbol{\eta}^{(m)}$ at the mth step, we conduct the

Bernoulli distribution-based random projection ADMM (RP-ADMM) iterative algorithm and compute the estimates of $(\boldsymbol{\phi}, \boldsymbol{\eta}, \boldsymbol{\mu})$ as follows:

$$\boldsymbol{\phi}^{(m+1)} = \arg\min_{\boldsymbol{\phi}} L(\boldsymbol{\phi}, \boldsymbol{\mu}^{(m)}, \boldsymbol{\eta}^{(m)}; \lambda), \tag{5}$$

$$\boldsymbol{\eta}^{(m+1)} = \boldsymbol{\eta}^{(m)} + \varphi(\boldsymbol{\Omega}\boldsymbol{\mu}^{(m)} - \boldsymbol{\phi}^{(m+1)}), \tag{6}$$

$$\boldsymbol{\mu}^{(m+1)} = \arg\min_{\boldsymbol{\mu}} Q(\boldsymbol{\mu}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m+1)}; \lambda), \tag{7}$$

where $L(\boldsymbol{\phi}, \boldsymbol{\mu}^{(m)}, \boldsymbol{\eta}^{(m)}; \lambda)$ equals

$$\frac{\varphi}{2}\|\boldsymbol{\Omega}\boldsymbol{\mu}^{(m)} - \boldsymbol{\phi} + \varphi^{-1}\boldsymbol{\eta}^{(m)}\|_F^2 + \sum_{j=1}^{\kappa} p_\lambda(\left\|\boldsymbol{\phi}_j\right\|), \tag{8}$$

and $Q(\boldsymbol{\mu}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m+1)}; \lambda)$ equals

$$\begin{aligned} &\widetilde{\ell}_p(\boldsymbol{\mu}, \boldsymbol{\phi}^{(m+1)}) + \frac{\varphi}{2}\|\Omega\boldsymbol{\mu} - \boldsymbol{\phi}^{(m+1)}\|_F^2 \\ &+ \sum_{j=1}^{\kappa} \boldsymbol{\eta}_j^{T(m+1)}(\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Omega}_j - \boldsymbol{\phi}_j^{(m+1)}). \end{aligned} \tag{9}$$

Ma and Huang (2017) [18] have argued that under (8), the element $\boldsymbol{\phi}_j^{(m+1)}$ of $\boldsymbol{\phi}^{(m+1)}$ is the minimizer of $\frac{\varphi}{2}||\boldsymbol{\zeta}_j^{(m)} - \boldsymbol{\phi}_j||^2 + p_\lambda(||\boldsymbol{\phi}_j||)$, where $\boldsymbol{\zeta}_j^{(m)} = \boldsymbol{\Omega}_j^{\mathrm{T}}\boldsymbol{\mu}^{(m)} + \varphi^{-1}\boldsymbol{\eta}_j^{(m)}$. For different thresholding operator $p_\lambda(\cdot)$, the estimate $\boldsymbol{\phi}_j^{(m+1)}$ has different results. Such as,

- For the Lasso penalty [25],

$$\begin{aligned} &\boldsymbol{\phi}_j^{(m+1)} = S(\boldsymbol{\zeta}_j^{(m)}, \lambda/\varphi); \\ &S(\boldsymbol{w}, t) = \begin{cases} (1 - t/||\boldsymbol{w}||)\boldsymbol{w}, & \text{if } t/||\boldsymbol{w}|| < 1; \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

- For SCAD penalty [26] with $a > 1/\varphi + 1$,

$$\boldsymbol{\phi}_j^{(m+1)} = \begin{cases} S(\boldsymbol{\zeta}_j^{(m)}, \lambda/\varphi), & \text{if } ||\boldsymbol{\zeta}_j^{(m)}|| \le \lambda + \lambda/\varphi; \\ \boldsymbol{\zeta}_j^{(m)}, & \text{if } ||\boldsymbol{\zeta}_j^{(m)}|| > a\lambda; \\ \frac{S(\boldsymbol{\zeta}_j^{(m)}, a\lambda/((a-1)\varphi))}{1-1/((a-1)\varphi)}, & \text{otherwise.} \end{cases}$$

- For the MCP [27] with $a > 1/\varphi$,

$$\boldsymbol{\phi}_j^{(m+1)} = \begin{cases} \frac{S(\boldsymbol{\zeta}_j^{(m)}, \lambda/\varphi)}{1-1/(a\varphi)}, & \text{if } ||\boldsymbol{\zeta}_j^{(m)}|| \le a\lambda; \\ \boldsymbol{\zeta}_j^{(m)}, & \text{otherwise.} \end{cases}$$

- For the TLP [8] with $a > 1$,

$$\boldsymbol{\phi}_j^{(m+1)} = \begin{cases} S(\boldsymbol{\zeta}_j^{(m)}, \lambda/\varphi), & \text{if } ||\boldsymbol{\zeta}_j^{(m)}|| \le a\lambda; \\ \boldsymbol{\zeta}_j^{(m)}, & \text{otherwise.} \end{cases}$$

Through some algebra, the problem of (9) is equivalent to the minimization of the function $h(\boldsymbol{\mu}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m+1)})$, which has the from

$$\frac{1}{2}\|\boldsymbol{X}-\boldsymbol{\mu}\|_F^2+\frac{\varphi}{2}||\boldsymbol{\Omega}\boldsymbol{\mu}-\boldsymbol{\phi}^{(m+1)}+\varphi^{-1}\boldsymbol{\eta}^{(m+1)}||_F^2.$$

Under the given value of $\boldsymbol{\phi}^{(m+1)}$, $\boldsymbol{\eta}^{(m+1)}$, the updated $\boldsymbol{\mu}^{(m+1)}$ are

$$\boldsymbol{\mu}^{(m+1)} \quad = \quad (\varphi\boldsymbol{\Omega}^{\mathrm{T}}\boldsymbol{\Omega}+\boldsymbol{I}_n)^{-1}\Big(\boldsymbol{X}+\varphi\boldsymbol{\Omega}^{\mathrm{T}}(\boldsymbol{\phi}^{(m+1)}-\varphi^{-1}\boldsymbol{\eta}^{(m+1)})\Big)$$

where $\boldsymbol{I}_n$ is $n\times n$ identity matrix. $\boldsymbol{\mu}^{(m+1)}$ and $\boldsymbol{\phi}^{(m+1)}$ are updated according to the random projection ADMM iterative algorithm (5)–(7) until the input of some convergence criteria, such as both dual and primal residuals being close to zero [28] in our practice. The convergence time of ADMM is highly related to the penalty parameter φ. A poor selection of φ can result in a slow convergence for the ADMM algorithm [29] and thus RP-ADMM. In this paper, we fix $\varphi=1$ throughout for simplicity.

To facilitate the updates of $(\boldsymbol{\phi}^{(m+1)},\boldsymbol{\eta}^{(m+1)},\boldsymbol{\mu}^{(m+1)})$ at the $(m+1)$th step in (5) to (7) of the RP-ADMM iterative algorithm, we need to specify a proper initial value (warm start). Here, we set $\boldsymbol{\eta}^{(0)}=\boldsymbol{0}$, $\boldsymbol{\phi}^{(0)}=\boldsymbol{\Omega}\boldsymbol{\mu}^{(0)}$ and obtain the initial estimators $\boldsymbol{\mu}^{(0)}=(\lambda^{\star}\boldsymbol{\Omega}^{\mathrm{T}}\boldsymbol{\Omega}+\boldsymbol{I}_n)^{-1}\boldsymbol{X}$ as the minimizer of a ridge fusion criterion

$$\frac{1}{2}\|\boldsymbol{X}-\boldsymbol{\mu}\|_F^2+\frac{\lambda^{\star}}{2}\|\boldsymbol{\Omega}\boldsymbol{\mu}\|^2. \tag{10}$$

We summarize the above analysis in Algorithm 1.

Algorithm 1 RP-ADMM for fusion clustering

Input: data $\boldsymbol{X}_1,\cdots,\boldsymbol{X}_n$; Initialize $\boldsymbol{\mu}^{(0)}$, $\boldsymbol{\eta}^{(0)}$; tuning parameter, λ
Output: an estimate of $\boldsymbol{\mu}$

```
for m = 0, 1, 2, ··· do
  compute φ^(m+1) using (5)
  compute η^(m+1) using (6)
  compute μ^(m+1) using (7)
  if convergence criterion is met, then
    Stop and denote the last iteration by μ̂(λ),
  else
    m = m + 1
  end if
end for
```

Practically, we would not want to conduct the RP-ADMM updates comprehensively until convergence to save computing time in the first iterations. Another trick is to adopt the initial values of subsequent convex relaxations as optimal values from the previous relaxed convex problem, which significantly reduces the number of RP-ADMM iterations.

2.2. Selection of Optimal Tuning Parameter

For a given λ, the converging value $\widehat{\boldsymbol{\mu}}(\lambda)$ of the above RP-ADMM procedure is defined as

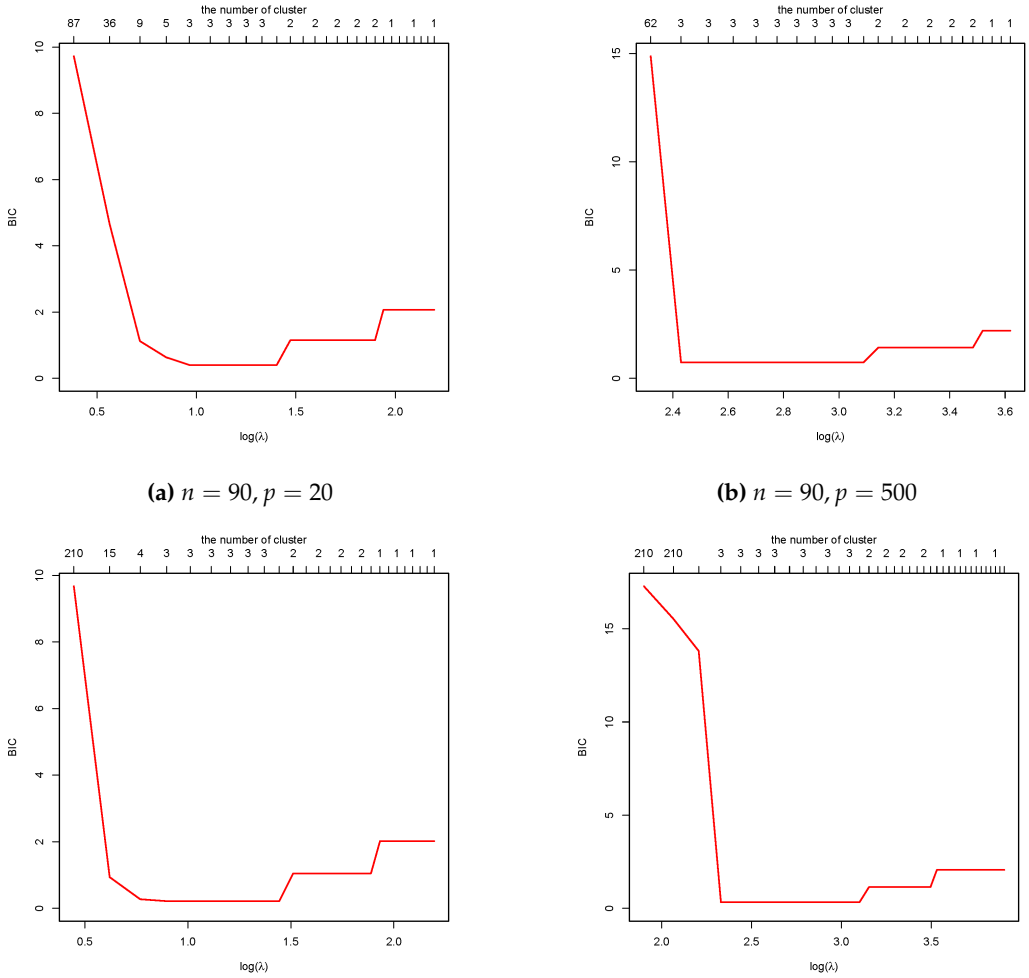
$$\widehat{\boldsymbol{\mu}}(\lambda)=\mathrm{argmin}_{\boldsymbol{\mu}}\ell_p(\boldsymbol{\mu};\lambda), \tag{11}$$

where $\ell_p(\boldsymbol{\mu};\lambda)$ is defined in (2) and the optimal value of λ can be selected via a properly constructed data-driven criterion. In particular, we partition the support of λ into a grid of $\lambda_{\min}=\lambda_0<\lambda_1<\cdots<\lambda_J=\lambda_{\max}$, and for each λ_j, we compute a solution path of $\widehat{\boldsymbol{\mu}}(\lambda_j)$ and obtain $\widehat{K}(\lambda_j)$ distinct cluster centroids $\{\widehat{\rho}_1(\lambda_j),\ldots,\widehat{\rho}_{\widehat{K}(\lambda_j)}(\lambda_j)\}$, The optimal $\widehat{\lambda}$ is selected by minimizing a data-driven BIC, i.e., $\widehat{\lambda}=\mathrm{argmin}_{\lambda_j;j=1,\ldots,J}\mathrm{BIC}(\lambda_j)$, where

$$\mathrm{BIC}(\lambda) = \log\Big\{\frac{1}{np}\|\boldsymbol{X} - \widehat{\boldsymbol{\mu}}(\lambda)\|_F^2\Big\} + (\log(np) + 2\log(p))\widehat{K}(\lambda)/n. \tag{12}$$

Subsequently, we obtain the estimator $\widehat{\boldsymbol{\mu}} = \widehat{\boldsymbol{\mu}}(\widehat{\lambda})$, and the individuals can be separated into $\widehat{K} = \widehat{K}(\widehat{\lambda})$ clusters accordingly, i.e., $\widehat{\mathcal{G}}_k = \{i : \widehat{\boldsymbol{\mu}}_i = \widehat{\boldsymbol{\rho}}_k, i = 1, \ldots n\}, k = 1, \ldots, \widehat{K}$.

Other methods for tuning parameters in clustering, such as generalized degrees of freedom with generalized cross-validation [8] and stability-based cross validation [25,30] can provide good results but may require extensive computation or the specification of a hyperparameter perturbation size [8]. In contrast, the proposed BIC is easy to compute and performs well in estimating cluster centroids and the true number of clusters (K). Figure 1 shows the change in BIC values against $\log(\lambda)$ and the cluster number of the simulation. Across all cases with different values of n and p, we observe that BIC(λ) decreases as the value of $\log(\lambda)$ increases. With recovering the true cluster number $K = 3$, BIC($\widehat{\lambda}$) reaches a minimum at the optimal $\widehat{\lambda}$. Moreover, when $\log(\lambda)$ keeps increasing, the cluster centroids are continuously integrated, and BIC(λ) is enlarged. However, further research is needed to fully prove the consistency of the BIC in combination with the objective function (2).



(a) $n = 90, p = 20$ **(b)** $n = 90, p = 500$

(c) $n = 210, p = 20$ **(d)** $n = 210, p = 500$

Figure 1. Plots of BIC values against $\log(\lambda)$ and the estimated cluster number of simulation with different n, p and true cluster number $K = 3$.

2.3. Recursive RP-ADMM and Cluster Matrix

In the above cluster analysis, the effect of randomness on the clustering results was not considered. However, empirical analysis has shown that the impact of this randomness on the estimated cluster centers and numbers is minimal (i.e., $\widehat{\rho}_k$'s and $\widehat{K}$'s). However, the impact on the final partitioning results (i.e., which observations are grouped into a single cluster) can be significant. In response to this, we propose the Recursive RP-ADMM (RRP-ADMM) procedure, which performs multiple RP-ADMM cluster analyses by generating M random matrices (i.e., $\mathbf{\Omega}_m$'s, $m = 1, \cdots, M$) and repeatedly conducting the analysis.

Once the multiple RP-ADMM cluster analyses have been completed, we must summarize the results. We define a $n \times n$ symmetric cluster matrix $\mathcal{C}$ where $\mathcal{C}_{ij} = 1$ denotes that the ith and jth observations belong to the same cluster; otherwise, $\mathcal{C}_{ij} = 0$. Another $n \times n$ symmetric matrix $\widehat{\mathcal{D}}$ is introduced, with element $\widehat{\mathcal{D}}_{ij}$ representing the relative frequency of the ith and jth observations belonging to the same cluster over the M independent RP-ADMM clustering procedures. The decision of whether the ith and jth observations should be grouped into a single cluster or not can then be treated as a classification problem, with the two possible class labels being 1 (belong to the same cluster) or 0 (do not belong to the same cluster). We can use an indicator function to transform the relative frequency into class labels and generate an estimator for the cluster matrix $\widehat{\mathcal{C}}$, i.e.,

$$\widehat{\mathcal{C}} = \{\widehat{\mathcal{C}}_{ij} : \widehat{\mathcal{C}}_{ij} = \mathbb{1}_{(\widehat{\mathcal{D}}_{ij} \geq 0.5)}\}, \tag{13}$$

where $\mathbb{1}_{(\cdot)}$ denotes the indicator function. We summarize the above procedure in Algorithm 2. This transformation can be understood as a voting-based aggregation strategy, similar to the one proposed by [31], which aims to reduce misclassification errors and improve the accuracy of the clustering. To evaluate the accuracy of the clustering results, we define a new measure called the similarity index (SI) between two data clusterings:

$$\text{SI} = \frac{1}{n^2 - n} \|\widehat{\mathcal{C}} - \mathcal{C}\|_1 = \frac{1}{n^2 - n} \sum_{i=1}^{n} \sum_{j=1}^{n} |\widehat{\mathcal{C}}_{ij} - \mathcal{C}_{ij}|. \tag{14}$$

Like the Rand Index (RI) measure [32], the newly introduced evaluation criterion can be seen as a measure of the percentage of correct decisions made by some algorithm. The SI values also range from 0 to 1, with lower values indicating better algorithm performance.

Algorithm 2 RRP-ADMM for fusion clustering

Input: data $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_n$; M; Initialize $\boldsymbol{\mu}^{(0)}$, $\boldsymbol{\eta}^{(0)}$; tuning parameter, λ
Output: an estimate of $\boldsymbol{\mu}$

```
for m = 0, 1, ..., M do
    compute μ̂^(m) using RP-ADMM
end for
while 1 ≤ i ≤ n do
    compute D̂_ij and Ĉ_ij from (13)
end while
```

The classical convex or concave fusion clustering procedure in (1) requires $\mathcal{O}(n^2 p)$ operations and $\mathcal{O}(n^2 p + np)$ of storage for a single round of ADMM updates with primal and dual residual calculations, because all pairs of centroids are shrunk together in this method.

The RP-ADMM algorithm significantly improves computational efficiency compared to classical ADMM algorithm. It requires only $\mathcal{O}(\kappa p + np)$ of storage, compared to $\mathcal{O}(n^2 p + np)$ for the classical ADMM algorithm, because the variables $\boldsymbol{\eta}$ and $\boldsymbol{\phi}$ have only κ columns rather than $n(n-1)/2$. Additionally, the RP-ADMM algorithm requires only $\mathcal{O}(\kappa p)$ operations for its most computationally demanding step, in comparison to $\mathcal{O}(n^2 p)$ for the classical ADMM algorithm. The RP-ADMM algorithm also requires $\mathcal{O}(\kappa n)$ operations to conduct Cholesky factorization in every iteration, in comparison to $\mathcal{O}(n^3)$ for

the classical ADMM algorithm. This efficient Cholesky factorization is computed only once and reused across repeated RP-ADMM updates.

At the end of this subsection, we will demonstrate the convergence of the RP-ADMM algorithm by showing that the sequence generated by the algorithm contains a subsequence that converges to a stationary point.

Lemma 1. *Let* $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ *be the sequence generated by Algorithm 1, then for some constant* $c > 0$,

$$\begin{aligned} &Q(\boldsymbol{\mu}^{(m+1)}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m+1)}) - Q(\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}) \\ &\leq -\frac{c}{2}\|\boldsymbol{\mu}^{(m+1)} - \boldsymbol{\mu}^{(m)}\|^2 + \psi\|\boldsymbol{\eta}^{(m+1)} - \boldsymbol{\eta}^{(m)}\|^2 \end{aligned} \tag{15}$$

In order to prove that the sequence $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ is convergent, we need to assume that $\boldsymbol{\phi}^{(m)}$ is bounded and $\psi\|\boldsymbol{\eta}^{(m+1)-\boldsymbol{\eta}^{(m)}}\| \to 0$ which are often observed in numerical tests.

Theorem 1. *If* $\{\boldsymbol{\phi}^{(m)}\}_{k=1}^{\infty}$ *are bounded and* $\psi_2\|\boldsymbol{\nu}^{(m+1)} - \boldsymbol{\nu}^{(m)}\|_F + \psi_1\|\boldsymbol{\eta}^{(m+1)} - \boldsymbol{\eta}^{(m)}\|_F \to 0$, *then* $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ *is bounded. Moreover, there exist a subsequence* $\{\boldsymbol{\mu}^{(k_j)}, \boldsymbol{\phi}^{(k_j)}, \boldsymbol{\eta}^{(k_j)}\}_{k_j=1}^{\infty}$, *such that*

$$\begin{aligned} \lim_{k_j \to \infty} (&\|\boldsymbol{\mu}^{(k_j+1)} - \boldsymbol{\mu}^{(k_j)}\| + \|\boldsymbol{\phi}^{(k_j+1)} - \boldsymbol{\phi}^{(k_j)}\| \\ &+ \|\boldsymbol{\eta}^{(k_j+1)} - \boldsymbol{\eta}^{(k_j)}\|) = 0, \end{aligned}$$

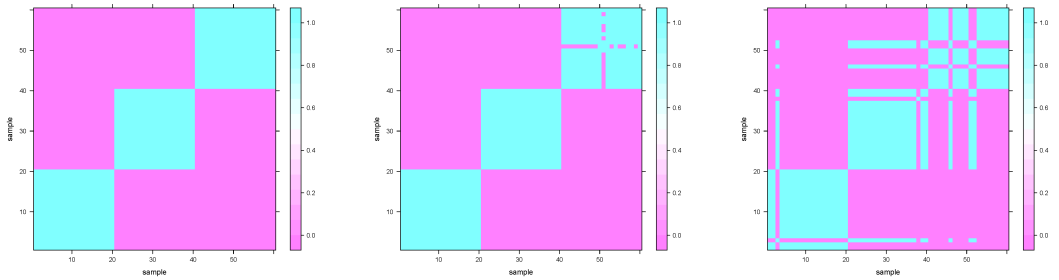
and thus, $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ *has a subsequence which converges to the stationary point.*

3. Simulation

In this part of the study, simulation experiments were conducted to compare the performance of the extended and classical ADMM clustering algorithms in terms of computational time and clustering accuracy, using the evaluation criterion in (14). The Lasso-based fusion method often leads to the formation of dense clusters with a minor penalty for small differences in $\|\boldsymbol{\phi}_j\|$, which can result in the formation of many spurious clusters with very small differences among them [6]. In contrast, the concave penalty method tends to produce a clear cluster structure and a well-defined number of clusters [8]. Therefore, in this study, we focus on the MCP-based fusion method [27] which compares the conventional ADMM's clustering performance and the proposed new ADMM algorithm.

3.1. Low-Dimensional Setting

In this part, we evaluated the clustering performance of the classical ADMM, RP-ADMM, and RRP-ADMM algorithms on low-dimensional synthetic data generated from three overlapping convex clusters with the same spherical shape in some number of dimensions p and sample size n. The synthetic data were generated from three populations $\mathcal{P}_k = \mathcal{N}(\rho_k, \boldsymbol{\Sigma})$, $k = 1, \cdots, K$ with $K = 3$, $\rho_1 = \mathbf{3}_p$, $\rho_2 = \mathbf{0}_p$, $\rho_3 = -\mathbf{3}_p$ and $\boldsymbol{\Sigma} = (\sigma_{kj})_{p\times p}$ with $\sigma_{jj} = 1$ and $\sigma_{kj} = 0.1^{|k-j|}$ for $k \neq j$. This setting was chosen deliberately to allow overlap in the sample sets generated from clusters proximal to each other, thereby increasing the complexity of the clustering task. As illustrated in Figure 2c, the clustering performance using a single random projection (RP-ADMM) was suboptimal, indicating challenges with cluster separability under this setup. Conversely, Figure 2b demonstrates that recursive random projection (RRP-ADMM) significantly improved clustering results. The recursive times for the RP-ADMM and RRP-ADMM algorithms were set to $M = 10$.



(a) True cluster matrix

(b) Estimated cluster matrix by RRP-ADMM

(c) Estimated cluster matrix by RP-ADMM

Figure 2. The level plots of cluster matrix including the true one in the left panel, estimators calculated from RRP-ADMM and RP-ADMM in the middle and right panels, respectively.

To evaluate the accuracy of the RP-ADMM, relax-and-split approach [33] (RS-ADMM) and RRP-ADMM algorithms in recovering the true cluster matrix, we generated a random sample of $n = 60$ observations with 1–20 drawn from $\mathcal{P}_1$, 21–40 drawn from $\mathcal{P}_2$, and 41–60 drawn from $\mathcal{P}_3$, and set the number of dimensions to $p = 5$. The probability α of generating a 1 in the random matrix was set to $\alpha = c\frac{\log(n)}{n}$, where c controls the probability size. The level plots in Figure 2 use colour to visualize the values of 1's and 0's in the cluster matrix. The results show that both RP-ADMM and RRP-ADMM can accurately recover the true cluster matrix, with RRP-ADMM showing more accurate gradation than the true cluster matrix. Single random projection (RP-ADMM) can cause high variance in clustering outcomes due to the randomness of the sampling process. To mitigate this issue, we have adopted the voting-based pooling technique [31], which reduces variance by averaging results from recursive random projection (RRP-ADMM).

To further evaluate the performance of the algorithms, we calculated the values of the index SI defined in (14) after 100 replicates under different c choices. We depicted the results as boxplots in Figure 3. These results show that RRP-ADMM consistently improves clustering accuracy compared to RP-ADMM, as evidenced by the smaller median and standard error of SI values.

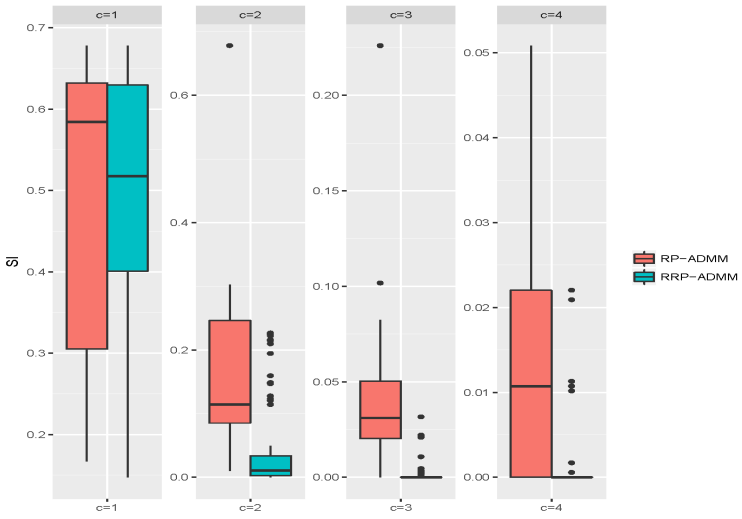


Figure 3. Boxplots of SI values through RP-ADMM and RRP-ADMM algorithms, respectively, under four choices of c after 100 replicates.

Next, we will compare the performance of classical ADMM and RRP-ADMM in terms of computation time per iteration and the SI after 100 trials. The sample size is varied with

$n = 60, 150, 240, 360$ points and $\alpha = 4\frac{\log(n)}{n}$, while $p = 2$ is kept constant. In this study, we have limited the number of points to 360, as the classical ADMM algorithm requires a significant amount of computation time for a single realization with more points. We will also compare the performance of the Similarity Index (SI) and Rand Index (RI) in evaluating the clustering results. Therefore, we should calculate the partitioning structure of all points based on the estimated cluster matrix graph. This process involves first identifying the point a_1 with the most neighbors and aggregating the connected points with point a_1 as cluster 1, then finding the second point a_2 with the most edges to form cluster 2, and repeating this process until there are no more points remaining.

Table 1 shows the mean values of the SI, RI, and the consumed time in seconds for different sample sizes under different methods after 100 replicates. Based on the data in Table 1, we can observe the following: (i) The proposed RRP-ADMM significantly reduces the time required for convex or concave fusion clustering, especially when the sample size increases. (ii) RRP-ADMM produces smaller SI and larger RI values, possibly due to the voting-based pooling technique improving cluster accuracy. (iii) As the sample size increases, the SI and RI values decrease. The boxplots in Figures 4 and 5 demonstrate the superiority of the RRP-ADMM algorithm over the classical ADMM algorithm in terms of both the SI values and the square root of run time, as seen in the results obtained from 100 replicates with four different sample sizes. These results further reinforce our belief in the effectiveness of the RRP-ADMM algorithm.

Table 1. The mean values of Similarity index (SI), Rand Index (RI) and run time in seconds against different sample sizes and different methods after 100 replicates.

	ADMM			**RRP-ADMM**			**RS-ADMM**		
Sample Size	**SI**	**RI**	**Time**	**SI**	**RI**	**Time**	**SI**	**RI**	**Time**
$n = 60$	0.081	0.921	7	0.059	0.933	2	0.080	0.925	10
$n = 150$	0.058	0.945	88	0.046	0.957	7	0.056	0.947	121
$n = 240$	0.049	0.962	352	0.045	0.974	17	0.047	0.966	551
$n = 360$	0.042	0.973	1582	0.040	0.986	41	0.042	0.978	1864

Note: 'SI' represents the similarity index defined in (14), 'RI' denotes Rand Index [32]. TIME is the required time in seconds in a single round of ADMM.

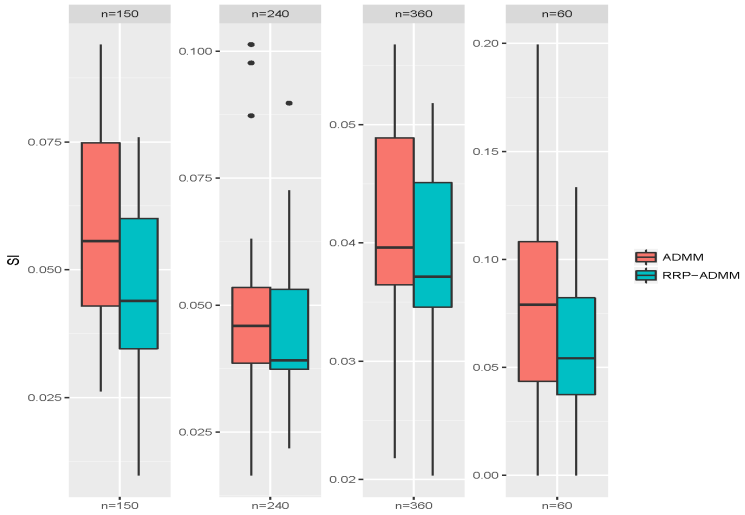


Figure 4. Boxplots of SI values through classical ADMM and RRP-ADMM algorithms, respectively, under four choices of sample sizes n after 100 replicates.

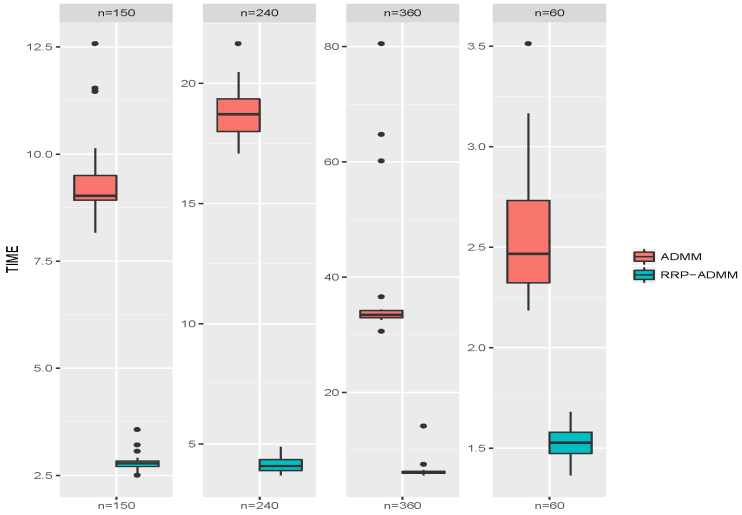


Figure 5. Boxplots of the square root of the run time through classical ADMM and RRP-ADMM algorithms, respectively, under four choices of sample sizes n after 100 replicates.

3.2. High-Dimensional Setting

In this part, we investigate using the double random projection-based alternating direction method of multiplier (DRP-ADMM and DRRP-ADMM) algorithms for clustering high-dimensional data sets. We employ a recursive Gaussian distribution-based random projection strategy in the first step to mitigate the impact of randomness on cluster results. Since the classical ADMM algorithm is computationally intensive in high-dimensional settings, we focus on evaluating the performance of the DRP-ADMM and DRRP-ADMM algorithms with recursive times $M = 9$, using three Gaussian random projections in the outer layer and three binary random projections in the inner layer. The simulated data sets consist of two overlapping convex clusters with the same spherical shape. They are generated using a population $\mathcal{P}_k = \mathcal{N}(\rho_k, \mathbf{\Sigma})$, $k = 1, 2$ with $\rho_1 = \mathbf{1}_p$, $\rho_2 = -\mathbf{1}_p$. Furthermore, $\mathbf{\Sigma} = (\sigma_{kj})_{p\times p}$ with $\sigma_{jj} = 1$ and $\sigma_{kj} = 0.1^{|k-j|}$ for $k \neq j$. We consider four high-dimensional cases with $p = 1000, 2000, 3000, 5000$ and a fixed sample size of $n = 100$.

We evaluate the accuracy of the DRP-ADMM and DRRP-ADMM algorithms in recovering the true cluster matrix. To do this, we first generate a Gaussian random matrix $\boldsymbol{R}$ with dimensions $p \times q$ in the first projection. The elements of $\boldsymbol{R}$ correspond to $\mathcal{N}(0, 1/\sqrt{q})$. We set $q = \lceil \frac{\kappa}{\varepsilon^2/2 - \varepsilon^3/3} \log(n) \rceil$ with $\varepsilon = 1$ and $\kappa = \frac{5}{6}$. See [21,23] for the number of projections. In the second step, we generate a diagonal binary random matrix with probability $\alpha = 4\frac{\log(n)}{n}$ of equaling one. Then, we calculate the values of the SI index defined in Equation (14) and plot the results as boxplots in Figure 6 after 100 replicates for different values of p. The results show that the DRRP-ADMM algorithm consistently outperforms the DRP-ADMM algorithm regarding the median and standard error of the SI values for all values of p, indicating that the DRRP-ADMM algorithm improves clustering accuracy.

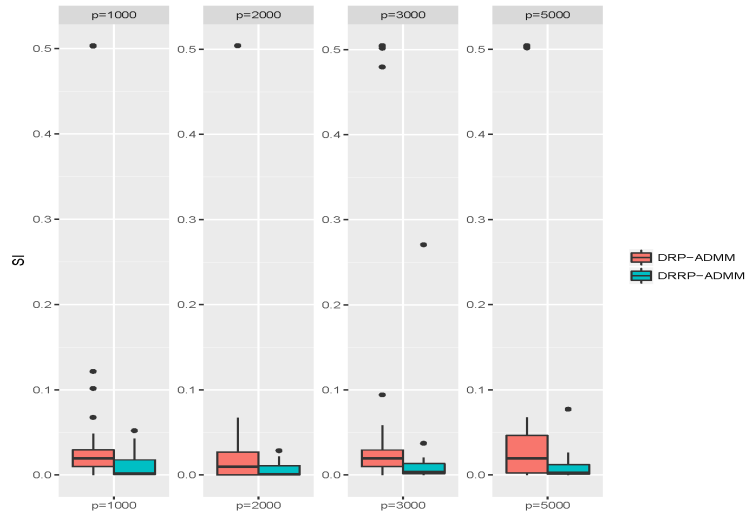
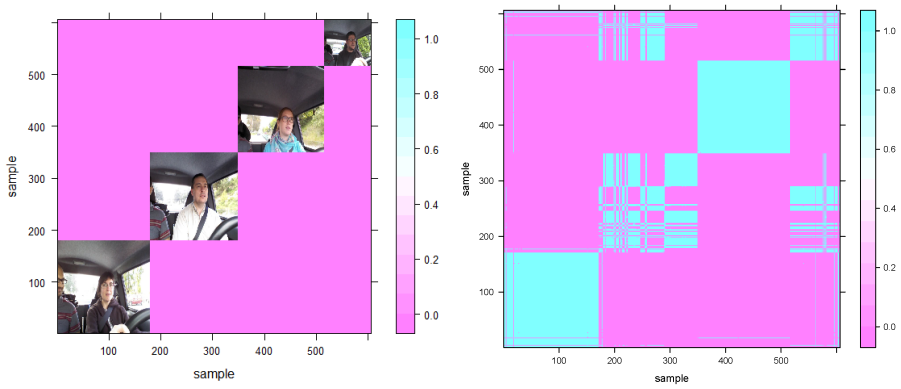


Figure 6. Boxplots of SI values through DRP-ADMM and DRRP-ADMM algorithms, respectively, under four choices of dimensions p after 100 replicates.

4. Real Data Analysis

In this study, we use the DrivFace dataset to demonstrate the effectiveness of our proposed clustering procedure. The DrivFace database consists of $n = 606$ images of 640,480 pixels each, captured from four drivers (two women and two men) over different days and containing $p = 17$ facial features such as glasses and beards. Each driver's images containing similar facial features can be grouped into one cluster, resulting in a total of $K = 4$ clusters as shown in Figure 7a. Firstly, we know the true labels of the dataset; that is, there are four clusters, and we also know which observations belong to the common cluster. Secondly, because the similarity among observations in the pictures is very high across different clusters, it is challenging to separate them. Therefore, we can use this dataset to evaluate our proposed clustering method.

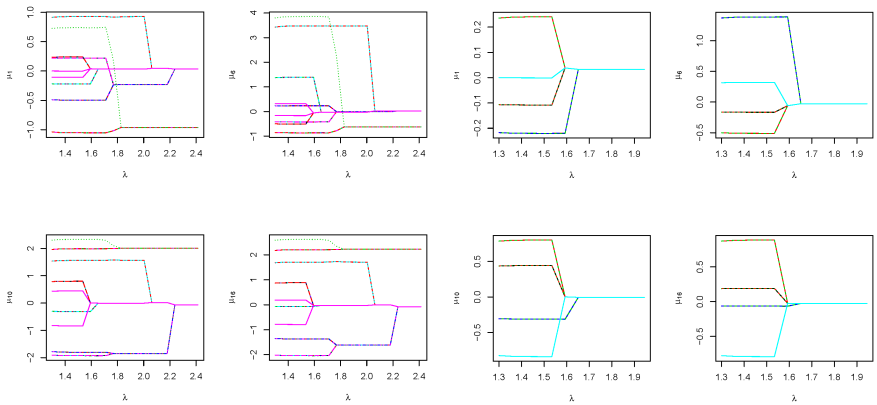


(a) True cluster matrix **(b)** Estimated cluster matrix

Figure 7. True (**a**) and estimated (**b**) cluster matrix in DrivFace data.

Due to the large sample size of the DrivFace dataset, we do not use the classical ADMM algorithm, which would require $606 \times (606 - 1) \times 17/2$ operations in a single ADMM iteration. Instead, we first scale the samples by each feature and apply the RP-ADMM procedure to estimate individual centers using a grid of λ values. We plot the *fusiongrams* of four selected variables in Figure 8, and the scrutiny of Figure 8a implies that some outlying points (influential points) cause the clusters to be dense. We then remove these 55 points and plot a new *fusiongram* in Figure 8b. The optimal λ value, as determined by the developed BIC criterion in Equation (12), is 1.38, indicating that the

estimated number of clusters is four, the same as the number of drivers. We apply the proposed RRP-ADMM algorithm with a Bernoulli-distribution-based random projection procedure to further improve the cluster accuracy using $\alpha = 10\log(n)/n$ and a recursive number $M = 20$. Using the estimated optimal tuning parameter of 1.38, we obtain the estimated cluster matrix in Figure 7b, which closely resembles the true cluster matrix in Figure 7a. The calculated similarity index (SI) value is 0.098. Moreover, the value of Adjusted Rand Index (ARI) is 0.672.



(a) Before deleting the influence points **(b)** After deleting the influence points

Figure 8. The above *fusiongrams* are plotted from 4 selected variables in DrivFace data before (left panel) and after (right panel) deleting the influence points, respectively.

5. Conclusions

We propose using the recursive random projection-based ADMM (RRP-ADMM) method to improve the speed and accuracy of convex and nonconvex fusion clustering. In simulations and real data examples, the RRP-ADMM method demonstrates superior performance in fast calculation and accurate clustering results. The RRP-ADMM algorithm is scalable and can be applied to deal with heterogeneous issues in any setting that involves fusion techniques.

However, some challenges still need to be addressed in this field. One challenge is efficiently transforming the cluster matrix graph into the target partitioning structure and determining the optimal number of clusters. Another challenge is using prior information about which points are more likely to be integrated into a single cluster to reduce the number of pairwise comparisons. Additionally, a further study is needed to determine the theoretical probability of achieving a probability of one in binary random projection. Another future research direction involves performing clustering simultaneously with feature selection, using techniques such as incorporating feature weights [34] or introducing sparsity [14].

Author Contributions: Conceptualization, Y.Z.; Methodology, L.K.; Software, H.W.; Formal analysis, J.Y.; Writing—original draft, X.Y.; Writing—review & editing, B.J.; Supervision, N.L. All authors have read and agreed to the published version of the manuscript.

Funding: Xiaodong Yan was supported by National Key R&D Program of China (No. 2023YFA1008701), the National Natural Science Foundation of China (No. 12371292), the National Statistical Science Research Project (No. 2022LY080) and Jinan Science and Technology Bureau (No. 2021GXRC056). Na Li was supported by grants from the National Natural Science Foundation of China (No. 12171279), and the China Academy of Engineering Science and Technology Development Strategy Shandong Research Institute Consulting Research Project (No. 202302SDZD04). Hongni Wang was supported by the State Scholarship Fund from China Scholarship Council (No. 202208370132). Bei Jiang and Linglong Kong were partially supported by grants from the Canada CIFAR AI Chairs program, the Alberta Machine Intelligence Institute (AMII), and Natural Sciences and Engineering Council

of Canada (NSERC), and Linglong Kong was also partially supported by grants from the Canada Research Chair program from NSERC.

Data Availability Statement: The DrivFace dataset is publicly available at UC Irvine Machine Learning Repository https://archive.ics.uci.edu/dataset/378/drivface, accessed on 11 March 2024.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Proof of Lemma 1

By the objection function,

$$Q(\boldsymbol{\mu}^{(m+1)}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m+1)}) - Q(\boldsymbol{\mu}^{(m+1)}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m)}) = \psi \|\boldsymbol{\eta}^{(m+1)} - \boldsymbol{\eta}^{(m)}\|^2 \tag{A1}$$

and

$$Q(\boldsymbol{\mu}^{(m+1)}, \boldsymbol{\phi}^{(m+1)}, \boldsymbol{\eta}^{(m)}) - Q(\boldsymbol{\mu}^{(m+1)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}) \leq 0 \tag{A2}$$

Moreover, $\boldsymbol{\mu} \mapsto Q(\boldsymbol{\mu}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)})$ is strongly convex, as the Hessian matrix $(\psi \boldsymbol{\Omega}^T \boldsymbol{\Omega} + \boldsymbol{I}_{np})$ is positive definite, and there exists a constant $c > 0$ such that the following inequality holds:

$$Q(\boldsymbol{\mu}^{(m+1)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}) - Q(\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}) \leq -\frac{c}{2} \|\boldsymbol{\mu}^{(m+1)} - \boldsymbol{\mu}^{(m)}\|^2 \tag{A3}$$

Summing (A1)–(A3), we have the result of the above Lemma. In order to prove that the sequence $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ is convergent, we need to assume that $\boldsymbol{\phi}^{(m)}$ is bounded and $\psi \|\boldsymbol{\eta}^{(m+1)} - \boldsymbol{\eta}^{(m)}\| \to 0$, which are often observed in numerical tests.

Appendix B. Proof of Theorem 1

Since $\{\boldsymbol{\phi}^{(m)}\}_{k=1}^{\infty}$ are bounded, $\boldsymbol{\mu}^{(m)}$ is also bounded. So $Q(\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)})$ and $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ are bounded. For convenience, we note

$$\begin{aligned} L^{(m)} &:= Q(\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}), \\ y^{(m)} &:= \frac{c}{2} \|\boldsymbol{\mu}^{(m+1)} - \boldsymbol{\mu}^{(m)}\|^2, \\ z^{(m)} &:= \|\boldsymbol{\eta}^{(m+1)} - \boldsymbol{\eta}^{(m)}\|^2. \end{aligned}$$

Since $L^{(m)}$ is bounded, there exist a subsequence $\{L^{(k_j)}\}$, such that

$$\lim_{k_j \to \infty} L^{(k_j)} = \liminf_{k \to \infty} L^{(m)}$$

By Lemma 1 and $\lim_{k \to \infty} z^{(m)} \to 0$, we have

$$\begin{aligned} \liminf_{k_j \to \infty} y^{(k_j)} &\leq \liminf_{k_j \to \infty} (L^{(k_j)} - L^{(k_j+1)} + z^{(k_j)}) \\ &= \liminf_{k \to \infty} L^{(m)} - \liminf_{k_j \to \infty} L^{(k_j+1)} \leq 0. \end{aligned}$$

As $y^{(k_j)} \geq 0$, $\liminf_{k_j \to \infty} y^{(k_j)} = 0$, which means

$$\liminf_{k_j \to \infty} \|\boldsymbol{\mu}^{(k_j+1)} - \boldsymbol{\mu}^{(k_j)}\| = 0,$$

together with $\|\boldsymbol{\eta}^{(m+1)} - \boldsymbol{\eta}^{(m)}\| \to 0$, we have

$$\liminf_{k_j \to \infty} \|\boldsymbol{\phi}^{(k_j+1)} - \boldsymbol{\phi}^{(k_j)}\| = 0.$$

The sequence $\{\boldsymbol{\mu}^{(m)}, \boldsymbol{\phi}^{(m)}, \boldsymbol{\eta}^{(m)}\}_{k=1}^{\infty}$ have a subsequence $\{\boldsymbol{\mu}^{(k_j)}, \boldsymbol{\phi}^{(k_j)}, \boldsymbol{\eta}^{(k_j)}\}_{k_j=1}^{\infty}$ which converges to a point $\{\boldsymbol{\mu}^*, \boldsymbol{\phi}^*, \boldsymbol{\eta}^*\}$. Then, we have

$$\boldsymbol{\Omega}_j \boldsymbol{\mu}^* - \phi_j^* = 0, 1 \leq j \leq \kappa.$$

Moreover, the procedure to solve the objective function satisfies the following optimality system:

$$\begin{cases} \boldsymbol{\mu}^{(m+1)} - \boldsymbol{X} + \psi \boldsymbol{\Omega}^{\mathrm{T}}(\boldsymbol{\Omega}\boldsymbol{\mu}^{(m+1)} - \boldsymbol{\phi}^{(m)} + \frac{\boldsymbol{\eta}^{(m)}}{\psi}) = 0, \\ 0 \in -\psi(\boldsymbol{\Omega}_j \boldsymbol{\mu}^{(m+1)} - \boldsymbol{\phi}_j^{(m+1)} + \frac{\eta_j^{(m+1)}}{\psi}) + \frac{\partial p_\lambda(\|\boldsymbol{\phi}_j\|)}{\partial \boldsymbol{\phi}_j}|_{\boldsymbol{\phi}_j = \boldsymbol{\phi}_j^{(m+1)}}. \end{cases}$$

So,

$$\begin{cases} \boldsymbol{\mu}^* - \boldsymbol{X} - \boldsymbol{\Omega}^T \boldsymbol{\eta}^* = 0, \\ 0 \in -\boldsymbol{\eta}_j^* + \frac{\partial P_\lambda(\|\boldsymbol{\phi}_j\|)}{\partial \boldsymbol{\phi}_j}|_{\boldsymbol{\phi}_j = \boldsymbol{\phi}_j^*}. \end{cases}$$

Therefore, $\{\boldsymbol{\mu}^*, \boldsymbol{\phi}^*, \boldsymbol{\eta}^*\}$ is a KKT point of objective function. We complete the proof.

References

1. Haq, M.A. CDLSTM: A novel model for climate change forecasting. *Comput. Mater. Contin.* **2022**, *71*, 2. [CrossRef]
2. Haq, M.A. SMOTEDNN: A novel model for air pollution forecasting and AQI classification. *Comput. Mater. Contin.* **2022**, *71*, 1. [CrossRef]
3. Van Der Kloot, W.A.; Spaans, A.M.J.; Heiser, W.J. Instability of hierarchical cluster analysis due to input order of the data: The PermuCLUSTER solution. *Psychol. Methods* **2005**, *10*, 468. [CrossRef] [PubMed]
4. Xu, R.; Wunsch, D. Survey of clustering algorithms. *IEEE Trans. Neural Netw.* **2005**, *16*, 645–678. [CrossRef] [PubMed]
5. Yang, X.; Yan, X.; Huang, J. High-dimensional integrative analysis with homogeneity and sparsity recovery. *J. Multivar. Anal.* **2019**, *174*, 104529. [CrossRef]
6. Chi, E.C.; Lange, K. Splitting methods for convex clustering. *J. Comput. Graph. Stat.* **2015**, *24*, 994–1013. [CrossRef] [PubMed]
7. Lindsten, F.; Ohlsson, H.; Ljung, L. Clustering using sum-of-norms regularization: With application to particle filter output computation. In Proceedings of the 2011 IEEE Statistical Signal Processing Workshop (SSP), Nice, France, 28–30 June 2011; pp. 201–204. [CrossRef]
8. Pan, W.; Shen, X.; Liu, B. Cluster Analysis: Unsupervised Learning via Supervised Learning with a Non-convex Penalty. *J. Mach. Learn. Res.* **2013**, *14*, 1865.
9. Yang, X.; Yan, X. Mechanism and a new algorithm for nonconvex clustering. *J. Stat. Comput. Sim.* **2020**, *90*, 719–746. [CrossRef]
10. Paul, D.; Chakraborty, S.; Das, S.; Xu, J. Implicit annealing in kernel spaces: A strongly consistent clustering approach. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 5862–5871. [CrossRef]
11. Shah, S.A.; Koltun, V. Robust continuous clustering. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 9814–9819. [CrossRef]
12. Hocking, T.D.; Joulin, A.; Bach, F.; Vert, J.P. Clusterpath an algorithm for clustering using convex fusion penalties. In Proceedings of the 28th International Conference on Machine Learning, Washington, DC, USA, 28 June–2 July 2011; p. 1.
13. Radchenko, P.; Mukherjee, G. Convex clustering via l1 fusion penalization. *J. R. Stat. Soc. B.* **2017**, *79*, 1527–1546. [CrossRef]
14. Wang, B.; Zhang, Y.; Sun, W.W.; Fang, Y. Sparse convex clustering. *J. Comput. Graph. Stat.* **2018**, *27*, 393–403. [CrossRef]
15. Yan, X.; Yin, G.; Zhao, X. Subgroup analysis in censored linear regression. *Stat. Sinica* **2021**, *31*, 1027–1054. [CrossRef]
16. Yan, X.; Wang, H.; Zhou, Y.; Yan, J.; Wang, Y.; Wang, W.; Xie, J.; Yang, S.; Zeng, Z.; Chen, X. Heterogeneous logistic regression for estimation of subgroup effects on hypertension. *J. Biopharm. Stat.* **2022**, *32*, 969–985. [CrossRef] [PubMed]
17. Zhu, C.; Xu, H.; Leng, C.; Yan, S. Convex optimization procedure for clustering: Theoretical revisit. *Adv. Neural Inf. Process. Syst.* **2014**, 1619–1627. [CrossRef]
18. Ma, S.; Huang, J. A concave pairwise fusion approach to subgroup analysis. *J. Am. Stat. Assoc.* **2017**, *112*, 410–423. [CrossRef]
19. Ma, S.; Huang, J. Estimating subgroup-specific treatment effects via concave fusion. *arXiv* **2016**, arXiv:1607.03717.
20. Marchetti, Y.; Zhou, Q. Iterative subsampling in solution path clustering of noisy big data. *arXiv* **2014**, arXiv:1412.1559.
21. Achlioptas, D. Database-Friendly Random Projections. In Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, Santa Barbara, CA, USA, 21–23 May 2001; Association for Computing Machinery: New York, NY, USA, 2001; pp. 274–281. [CrossRef]
22. Ailon, N.; Chazelle, B. The Fast Johnson–Lindenstrauss Transform and Approximate Nearest Neighbors. *SIAM J. Comput.* **2009**, *39*, 302–322. [CrossRef]
23. Bingham, E.; Mannila, H. Random Projection in Dimensionality Reduction: Applications to Image and Text Data. In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 26–29 August 2001; Association for Computing Machinery: New York, NY, USA, 2001; pp. 245–250. [CrossRef]

24. Kane, D.M.; Nelson, J. Sparser johnson-lindenstrauss transforms. *J. ACM* **2014**, *61*, 1–23. [CrossRef]
25. Tibshirani, R.; Walther, G. Cluster validation by prediction strength. *J. Comput. Graph. Stat.* **2005**, *14*, 511–528. [CrossRef]
26. Fan, J.; Lv, J. Nonconcave penalized likelihood with NP-dimensionality. *IEEE T. Inform. Theory* **2011**, *57*, 5467–5484. [CrossRef]
27. Zhang, C.-H. Nearly unbiased variable selection under minimax concave penalty. *Ann. Stat.* **2010**, *38*, 894–942. [CrossRef] [PubMed]
28. Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.* **2011**, *3*, 1–122. [CrossRef]
29. Ghadimi, E.; Teixeira, A.; Shames, I.; Johansson, M. Optimal parameter selection for the alternating direction method of multipliers (ADMM): Quadratic problems. *IEEE Trans. Autom. Control* **2014**, *60*, 644–658. [CrossRef]
30. Liu, B.; Shen, X.; Pan, W. Integrative and regularized principal component analysis of multiple sources of data. *Stat. Med.* **2016**, *35*, 2235–2250. [CrossRef]
31. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
32. Rand, W.M. Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850. [CrossRef]
33. Zheng, P.; Aravkin, A. Relax-and-split method for nonconvex inverse problems. *Inverse Probl.* **2020**, *36*, 095013. [CrossRef]
34. Chakraborty, S.; Xu, J. Biconvex clustering. *J. Comput. Graph. Stat.* **2023**, *32*, 1524–1536. [CrossRef]