Journal of Big Data

RESEARCH Open Access

An ensemble method for estimating the number of clusters in a big data set using multiple random samples

Mohammad Sultan Mahmud[1,2], Joshua Zhexue Huang[1,2*], Rukhsana Ruby[3] and Kaishun Wu[1,2]

*Correspondence:
zx.huang@szu.edu.cn

[1] Big Data Institute, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China
[2] National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, Shenzhen 518060, China
[3] Guangdong Laboratory of Artificial Intelligence and Digital Economy, Shenzhen 518107, China

Abstract

Clustering a big dataset without knowing the number of clusters presents a big challenge to many existing clustering algorithms. In this paper, we propose a Random Sample Partition-based Centers Ensemble (RSPCE) algorithm to identify the number of clusters in a big dataset. In this algorithm, a set of disjoint random samples is selected from the big dataset, and the I-niceDP algorithm is used to identify the number of clusters and initial centers in each sample. Subsequently, a cluster ball model is proposed to merge two clusters in the random samples that are likely sampled from the same cluster in the big dataset. Finally, based on the ball model, the RSPCE ensemble method is used to ensemble the results of all samples into the final result as a set of initial cluster centers in the big dataset. Intensive experiments were conducted on both synthetic and real datasets to validate the feasibility and effectiveness of the proposed RSPCE algorithm. The experimental results show that the ensemble result from multiple random samples is a reliable approximation of the actual number of clusters, and the RSPCE algorithm is scalable to big data.

Keywords: Ensemble learning, Number of clusters, Random sample partition, Cluster ball model, Approximate computing

Introduction

In this paper, we propose an ensemble method for estimating the number of clusters in a big dataset, the most important parameter in many clustering algorithms like *k*-means. Because this parameter, often unknown in unlabeled data, is often guessed by the user, incorrect guesses result in inaccurate clustering results. Therefore, finding this number can improve the clustering result. However, automatic identification of the number of clusters in a big dataset is a challenge to the classical methods, e.g., Elbow [1], Silhouette [2], Gap statistic [3], and I-nice [4], due to the data size and the complexity of the inherent clusters in the data. A strategy we take here is to use multiple samples of the big dataset to estimate several possible values of the number of clusters and then ensemble the multiple results to improve the final estimate.

Since it is hard or impractical to investigate a big dataset entirely, using a random sample to compute an approximate result for the whole big dataset often becomes imperative

Springer Open

© The Author(s) 2023. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

[5, 6]. Random sampling is also a popular technique widely used by data scientists to quickly gain insights from a big dataset, despite theoretical and empirical evidence of the benefits of other sampling techniques [7]. However, sampling a big dataset is an error-prone and inefficient process when the dataset cannot be held in memory. Furthermore, to obtain accurate approximations, it is essential to efficiently select random samples from a big dataset, and in the meantime, to guarantee the quality of selected samples. As such, we adopt the random sample partition (RSP) [8] data model to represent a big dataset, which allows the block-level sampling methods to be used to efficiently select multiple random samples from the big dataset. The critical technical challenges of partitioning and sampling big datasets are highlighted in [9].

Ensemble approach is widely used to tackle the problems of clustering complex data. For clustering big complex datasets, developing scalable and appropriate ensemble methods is the main challenge. The classical clustering ensemble methods combine the outcomes from different models or algorithms on the same dataset to produce an ensemble result, but they are only appropriate for small or moderate-sized datasets. For ensemble clustering of a big dataset, it is required to ensemble the results obtained from different disjointed random samples of the big dataset. In this case, each object can only appear in one clustering result, and the object identifications in different clustering results are lost. The classical clustering ensemble methods are no longer applicable. Therefore, it is necessary to investigate an appropriate ensemble model with new integration functions to ensemble the clustering results from the disjoint random samples.

In response to this, in this paper, we investigate a new clustering ensemble method that is data-adaptive and approximate in estimating the number of clusters. For large-scale data clustering, we are aimed to developing a feasible distributed clustering algorithm that (i) incorporates with a scalable serial algorithm effectively, (ii) runs efficiently on the distributed platform, and (iii) does not require processing the entire dataset. To achieve this goal, we propose a new ensemble algorithm for estimating the number of clusters in a big dataset using multiple random samples. We name our algorithm RSPCE, representing the abbreviation of RSP-based Centers Ensemble. The RSPCE algorithm includes the following steps: (1) division of a big dataset into subsets of random samples, called RSP data blocks, which form the RSP data model; (2) random selection of a subset of RSP data blocks and identification of the number of clusters and the initial cluster centers in each RSP data block by the I-niceDP [10] algorithm; and (3) generation of the final ensemble result as a set of initial centers of K clusters by the RSPCE ensemble method that uses the cluster ball model to merge clusters in random samples which are likely sampled from the same cluster in the big dataset. Unlike the classical ensemble clustering methods, the RSPCE algorithm does not depend on common object ids to ensemble the component clustering results.

We conducted experiments on both synthetic and real-world datasets. The experiment results have shown that the new method is computationally effective and efficient in finding the number of clusters in a big dataset and their initial cluster centers. The experiment results also show that the RSPCE algorithm produces good approximations of the actual numbers of clusters in both synthetic and real-world datasets.

The rest of the paper is organized as follows: In Sect. Related work, we briefly discuss existing work on large-scale data clustering processes. In Sect. Ensemble method for estimating the number of clusters, we introduce preliminaries in the context of this

work. In Sect. The proposed RSPCE algorithm, we present the proposed RSPCE scheme. In Sect. Experiments, we evaluate the performance of the proposed scheme through experimental results. Finally, Sect. Conclusions concludes the paper.

Related work

The number of clusters is an important parameter in many well-established clustering algorithms (e.g., *k*-means, *k*-medoids). This number is unknown in unlabeled datasets, and it is often guessed by the user. The Elbow [1], Silhouette coefficient [2], and Gap statistic [3] are well-known methods for finding the "right" number of clusters in a dataset. These methods identify the number of clusters in a dataset by measuring the quality of several clustering results with different numbers of clusters. However, these methods do not work well on big datasets with a large number of clusters, and they are computationally expensive to use because multiple clustering results must be generated.

I-nice [4] is eminent among the density-based clustering algorithms for estimating the number of clusters that can produce high-quality initial seeds on small datasets. In the I-nice algorithm, the observation points are assigned to the data space for observing the dense regions of clusters in data through the distance distributions between the observation points and objects. Then, to find the number of peaks in a distance distribution, multiple gamma mixture models (GMMs) are built with different components, and the GMM is solved via the EM algorithm. The minimum Akaike information criterion (AICc) is used to select the best-fitted model and the observed largest number of components as the number of clusters.

Automatic clustering algorithms are attracting more attention from the academic community, e.g., density-based clustering and data depth clustering [11, 12]. Density-based algorithms, such as DBSCAN, can cluster datasets with convex shapes and noisy objects, but it is difficult to determine the density threshold [13]. The depth difference method [14] estimates the depth within clusters, the depth between clusters, and the depth difference to finalize the optimal value of K. However, for datasets with complex decision graphs, it is difficult to correctly identify clustering centers.

In practice, according to data preprocessing strategies, big data clustering processes can be classified as sampling-based, incremental, condensation-based, and divide-and-conquer strategy-based. *Sampling-based techniques* typically select a subset of a given dataset, employ only the sampled subsets to find the number of clusters, and then allow the remaining data to obtain the final outcome [15, 16]. The success of sample-based methods depends on the premise that the chosen representative samples have significant information about the dataset. *Incremental approaches*, on the other hand, reduce the computation time by scanning the data points only once [17, 18]. For fast clustering, modified global *k*-means [19] and multiple medoids-based fuzzy clustering [17] methods have been developed based on the idea of incremental clustering. *Condensation-based methods* speed up the performance by encapsulating the data into a special data structure, such as trees and graphs [20, 21]. *Divide-and-conquer strategies* split the big dataset into several subsets or sub-spaces that can fit into the memory. Later, the clustering algorithms are applied to these subsets or sub-spaces independently (please see [22–24]). The final clustering results are obtained by merging the partial clusters of subsets or sub-spaces.

For big data clustering, a bootstrap method [25] was proposed to estimate the number of clusters, which minimizes the corresponding estimated clustering instability. The kluster [26] procedure takes randomly selected clusters as the initial seeds to determine the final number of clusters in a dataset in an iterative manner, which confirms the most frequent mean of the resulting clusters from the iterations as the optimal number of clusters. On the other hand, the X-means method [27] automatically determines the number of clusters based on Bayesian information criterion (BIC) scores. At each iteration, this method executes local decisions about the selection of the current center split to better fit the data. In the same vein, coresets [28] have been constructed to scale to massive datasets with the distributed clustering idea.

Although many clustering ensemble techniques have been developed (such as [22, 23, 29–31]), they either produce incorrect results or are inefficient for use in big data applications. The traditional methods [1–3] for finding the K value run the clustering algorithm several times with a different K value for each run, which is not suitable for big data analysis. We observe that there are two limitations to the traditional clustering approaches. First, in a typical scenario, the algorithms require the number of clusters in advance for the clustering process. Second, traditional algorithms operate at the object level and are incapable of dealing with clustering ensembles with big data and large ensemble sizes. Furthermore, the classical ensemble technique combines the results of different models or algorithms on the same dataset to produce a robust result, where a scalable method is required to identify the number of clusters in a big dataset. We focus on the data-subset clustering ensemble technique, which is an approximate computing method to estimate the correct clustering outcome from the subsets of a big dataset.

Ensemble method for estimating the number of clusters

In this section, we propose a new method that uses multiple random samples of a big dataset to identify the number of clusters. We first give the definition of a random sample from a big dataset and define the random sample partition data model to represent a big dataset as a partition of random samples. Then, we present the I-niceDP method for finding the number of clusters in a random sample. Finally, we propose a ball model for integrating the results of multiple random samples into the ensemble result as the number of clusters in the big dataset.

Multiple random samples of a big dataset

The existing methods are computationally infeasible for finding the number of clusters in a big dataset. Instead, using a random sample to estimate is an acceptable choice. However, a large sample could possibly result in an accurate estimate but will be computationally expensive. An alternative is to use multiple random samples of smaller size for this purpose. In this case, we need to solve the problems of drawing multiple random samples from a big dataset effectively and efficiently, and ensemble the results of those multiple random samples into the final result.

Definition 1 *(Random sample of a big dataset)* Let D be a subset of big dataset $\mathbb{D}$, i.e., $D \subset \mathbb{D}$. D is a random sample of $\mathbb{D}$ if

$$F(D) \approx F(\mathbb{D}) \tag{1}$$

where $F()$ is the cumulative distribution function.

The simple random sampling process can be used on $\mathbb{D}$ to generate D, which satisfies this definition. However, sampling a distributed big data file to generate multiple independent random samples is a time-consuming process. In this work, we use the random sample partition data model to represent a big dataset as a set of random sample data blocks, so the block-level sampling method is used to efficiently select multiple random samples.

The random sample partition (RSP) [8] is defined as follows:

Definition 2 *(Random sample partition of a big dataset)* Let $\mathbb{D}$ be a big dataset and $\{D_1, D_2, ..., D_m\}$ be a set of m random samples of $\mathbb{D}$. $\{D_1, D_2, ..., D_m\}$ is a random sample partition of $\mathbb{D}$ if

- $D_i \neq \emptyset$
- $D_i \cap D_j = \emptyset$
- $\bigcup_{i=1}^{m} D_i = \mathbb{D}$
- $F(D_i) \approx F(\mathbb{D}), \qquad 1 \leq i \leq m$

The first three conditions define a partition of $\mathbb{D}$, whereas the last condition categorizes a random sample partition.

In the random sample partition, all RSP data blocks $\{D_1, D_2, ..., D_m\}$ satisfy the definition of a random sample of $\mathbb{D}$. To generate multiple random samples, we simply randomly select a few RSP data blocks from the RSP data model without going through all the records of $\mathbb{D}$ several times. As a result, the sampling process of multiple random samples is improved significantly.

Finding the number of clusters in a random sample

Given a set of random samples as a set of RSP data blocks $\{D_1, D_2, ..., D_b\}$ where $(b < m)$, one important step is to find the number of clusters in each sample D_i. In this work, we are not only interested in the number of clusters K, but also the initial centers of the K clusters in each subset D_i. For this reason, we first use the density peak-based algorithm I-niceDP [10] to compute K and the initial centers of clusters in a random sample. Then, we use the k-means algorithm to cluster the random sample and refine the cluster centers. The I-niceDP operator is defined as follows:

$$(k_i, C_i) = \text{ I-niceDP } (D_i), \qquad 1 \leq i \leq b \tag{2}$$

where I-niceDP is an operator on random sample D_i, and (k_i, C_i) are the two return values of the function. The first term is the number of clusters in D_i, and the second term $C_i = \{c_i, c_2, ..., c_{k_i}\}$ is the set of centers of the k clusters.

After (k_i, C_i) are obtained from D_i, they are used as the input parameters to the k-means algorithm to compute k refined cluster centers in D_i as follows:

$$C_i^* = k\text{-means}\ (k_i, C_i, D_i), \qquad 1 \leq i \leq b \tag{3}$$

where $C_i^* = \{c_i^*, c_2^*, ..., c_{k_i}^*\}$ is the set of the refined centers of k_i clusters in D_i.

Applying the two operators I-niceDP and k-means to all b random samples $\{D_1, D_2, ..., D_b\}$, we obtain b sets of refined centers and make union of these sets to form a new set as

$$C^* = C_1^* \cup C_2^* \cup ... \cup C_b^*, \tag{4}$$

where C_i^* is the result of k-means (k_i, C_i, D_i).

The set C^* contains totally $K = \sum_{i=1}^{b} k_i$ cluster centers from b random samples. Since the random samples are taken from the same big dataset, they should have similar inherent clusters. Therefore, the numbers of clusters in them should be very close to each other, and the centers of clusters in different random samples should also be located closely. Considering these properties, in the next subsection, we propose a method, called the ball model, to be used to aggregate the nearby centers in C^* into an ensemble set of centers as the initial cluster centers in the big dataset.

Ball model for representing clusters

Given a random sample D_i, the k-means operator of (3) generates a set of k_i clusters. The centers of clusters in the same random sample should be separate, but the centers of clusters in different random samples can be very close to each other. In this case, the two clusters in the two random samples may represent the same cluster in the big dataset. Therefore, the two centers should be merged into one, indicating the same cluster of the big dataset. In this subsection, we will determine whether two clusters in two different random samples represent the same cluster in the big dataset.

Since the k-means clustering process produces spherical clusters, we propose a ball model to represent a spherical cluster as a ball. The main features of this ball model are defined below.

Definition 3 *(Radius of a cluster ball)* Let C_i be a cluster of n points, and c_i the center of the cluster. The radius of the cluster ball, r_i, is defined as the average of the distances between all points in C_i and its center c_i below

$$r_i = \frac{1}{n} \sum_{i=1}^{n} \|d(c_i - x_i)\|, \tag{5}$$

where x_i is an object in C_i.

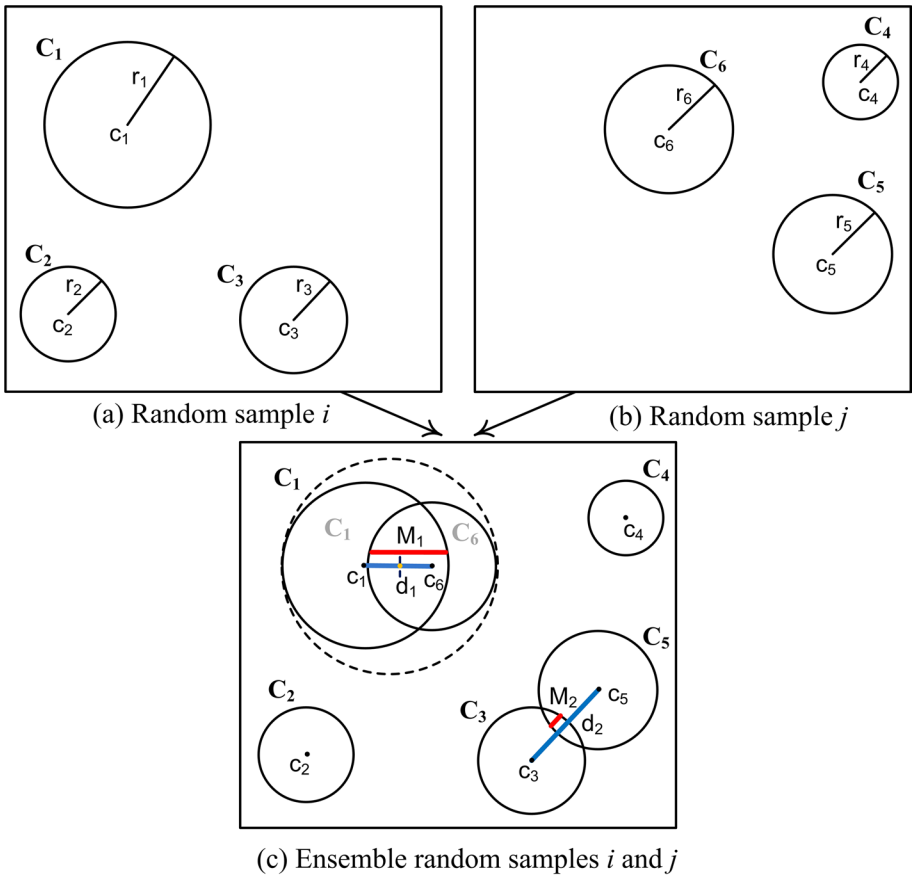


Fig. 1 Illustration of separate and overlapping cluster balls. **a** and **b** show the cluster balls of two RSP blocks, three cluster balls each; **c** the disjoint and intersection cluster balls from the two RSP blocks

We use the average distance to define the radius of cluster ball to reduce the impact of outliers, i.e., few points faraway from the cluster center.

Definition 4 *(Cluster ball)* Let C_i be a cluster. Its cluster ball, denoted as CB_i, is defined as a 3-tuple

$$\text{CB}_i = (C_i, c_i, r_i), \tag{6}$$

where c_i and r_i are the center and radius of cluster C_i, respectively. Note that, in the literature, "centers" and "centroids" are used, alternatively and represent the same meaning.

We can use cluster balls to determine whether two clusters are well separated or overlapping.

Definition 5 *(Well-separate and overlapping clusters)* Let CB_i and CB_j be two balls of clusters C_i and C_j, respectively. We say that C_i and C_j are well-separated if CB_i and CB_j are disjoint. If CB_i and CB_j intersect, we say that C_i and C_j are overlapping.

Figure 1 illustrates an example of merging the clusters from two random samples, RSP block i in Fig. 1a and RSP block j in Fig. 1b. Three clusters are found in each random

sample and represented as three cluster balls. Figure 1c shows clusters C_1 and C_6 intersect, and clusters C_3 and C_5 intersect, whereas clusters C_2 and C_4 are separated from others. Two overlapping clusters are likely to be the same cluster of the big dataset. The definition below defines a property to merge two overlapping clusters.

Definition 6 ($\frac{1}{2}$*-ball intersection property)* Let CB_i and CB_j be two cluster balls. We say that CB_i and CB_j have $\frac{1}{2}$-ball intersection property if $\|c_i - c_j\| \leq \frac{1}{2}(r_i + r_j)$, where $(r_i, r_j > 0)$. If two cluster balls have a $\frac{1}{2}$-ball property, they are *strongly proximal*.

This property is used to determine whether two clusters found from different random samples indicate the same cluster of the big dataset. If so, they can be merged into one cluster. If two clusters indicate the same cluster in the big dataset, their cluster balls must intersect and satisfy the $\frac{1}{2}$-ball intersection property. In this case, the intersected cluster balls in random samples are merged to obtain the optimal number of clusters in the big dataset. For example, in Fig. 1 (c), cluster balls CB_1 and CB_6 satisfy this property, so they are likely sampled from the same cluster of the big dataset and need to merge into one cluster.

Ensemble method for merging clusters with ball model

Using Definition 6, we can integrate the set of cluster centers in C^* into the ensemble set of centers as the initial cluster centers of the big dataset. This process is carried out as follows:

1. Randomly select a center c_p^* from C^*. Make the center c_p^* as a candidate of the final centers in the set of CF, i.e., final set of centers. Compute the cluster ball CB_p^* and remove c_p^* from C^*.
2. Randomly select a center c_q^* from C^* and compute the cluster ball CB_q^*.
3. Compute the $\frac{1}{2}$-ball intersection property of the two cluster balls CB_p^* and CB_q^*.
4. If the two balls are disjoint, ignore the second ball CB_q^* and go to Step 2; otherwise, if the two balls do not satisfy the $\frac{1}{2}$-ball intersection property of Definition 6, ignore the second ball CB_q^* and go to Step 2; otherwise, add the cluster ball in the set CF. If all centers in C^* have been tested, go to next step; otherwise, go to Step 2.
5. Merge the centers in CF by computing the mean of the centers as the center of a cluster in the big datset and go to Step 1 until the centers of all clusters in the big dataset are found.

Since we have included the radius of the cluster ball in the ball model, it is straightforward to test whether two cluster balls are disjoint or not. However, when two cluster balls intersect, Definition 6 plays an important role in determining the merge of two clusters. The intersection of two cluster balls satisfying the $\frac{1}{2}$-ball intersection property is a much stronger indication that the two clusters could be the same cluster in the big dataset.

The proposed RSPCE algorithm

In this section, we present the algorithms used in the basic steps of the RSPCE algorithm for estimating the number of clusters in a big dataset and finding the initial cluster centers. The basic steps are summarized as follows:

1 Given a big dataset, generate its RSP data model for efficiently selecting multiple random samples.
2 For each random sample, find the number of clusters and the centers of the clusters.
3 For given two clusters, use the ball model to determine whether two clusters could be the same cluster in the big dataset.
4 Use the ball model to integrate the clusters from multiple random samples into an ensemble set of clusters and initial cluster centers.

In the following, we present the algorithms in each step and give a complexity analysis of the RSPCE algorithm.

Algorithm for generating multiple samples

In this work, we use the random sample partition (RSP) data model to convert a big dataset into a set of disjoint random sample data blocks, so that each data block is used as a random sample of the big dataset. Therefore, to identify the number of clusters in a big dataset, we use *Algorithm 1* to convert it to a set of RSP data block files for random sample selection.

The inputs of the algorithm are a big dataset $\mathbb{D}$ and the size of each RSP data block n. The output is a set of m RSP data blocks, where $m = N/n$, which are saved as a set of RSP data block files $\{D_1, D_2, ..., D_m\}$.

Algorithm 1 is executed as follows: Lines 2-3 compute the number of objects N and the number of RSP blocks m. Line 4 generates a sequence of N unique random numbers following a uniform distribution. Line 5 appends the sequence of random numbers as one additional id in $\mathbb{D}$. Line 6 randomizes the records of $\mathbb{D}$ by sorting the records on the random number id. Lines 8-11 cut the sequence of the randomized records of $\mathbb{D}$ sequentially into m sub-sequences, each being written as an RSP data block file.

Algorithm 1: Generating random samples of a big dataset.

Input : $\mathbb{D}$: A big dataset.
n: Number of objects in each sample.

```
1  begin
2      N = getNumberOfObjects(𝔻);
3      m = getNumberOfPartitions(N, n);
4      rand = generateRandomNumbers(N);
5      rand.append(𝔻);
6      J = getOrder(𝔻);
7      /* ascending or decending order of records sort by rand.        */
8      for i = 1 to m do
9          D_i = getRS(J, m);
10         /* sequentially cut from sorted data                        */
11     end
12 end
```

Output: $\{D_1, D_2, ..., D_m\} = \mathbb{D}$, a set of random sample.

Algorithm 2: I-niceDP algorithm.

Input : $\mathbb{D} = \{D_1, D_2, ..., D_m\}$; A big dataset.

```
1  begin
2      Randomly select few RSP samples, D^b_{i=1}, b < m.
3      forall D^b_{i=1} do
4          Generate a random observation point, p;
5          Compute the Euclidean distance vector X_p between the data points of D_i and p;
6          Calculate the number M_max of GMM components using the KDE method;
7          for M = M_max − Δ1; M ≤ M_max + Δ2; M++ do
8              Model X_p to GMM(p, M);
9              Use the EM algorithm to solve GMM(p, M);
10             Calculate AICc(M) of GMM(p, M);
11         end
12         Select the best-fitted model GMM(p) corresponding to the minimum AICc value with
             K components;
13         Set the GMM(p) as the final model GMM_final and K as the number of clusters;
14         Separate the IDs of data points for each GMM component from the final model GMM_final;
15         Calculate the local density of each data point in the k-th GMM component;
16         Find the data points with the top 50% local density values;
17         Use the density peaks mechanism to determine the high-density points in the k-th GMM
             component as initial cluster centers;
18         Operate the k-means with assigning K and the initial cluster centers;
19         Determine the final clusters number K and centers;
20     end
21 end
```

Output: Numbers of clusters K_i and initial centers $\{c_1^i, c_2^i, ..., c_{k_i}^i\}$ where $i = 1, 2, ..., b$.

Algorithm for finding the number of clusters in a random sample

We model clusters in a big dataset as normal distributions where each cluster has a high density area, which is reflected as the density peak of the normal distribution. Therefore, the number of clusters in a dataset is corresponding to the number of density peaks in the dataset. The I-niceDP algorithm [10] (i.e., an improved version of I-nice) was designed for identifying the number of clusters in a dataset by finding the number of density peaks in the distance distribution of objects in the dataset with respect to an observation point. Therefore, I-niceDP is chosen as the operator to identify the number of clusters in each random sample. The pseudo-code of the I-niceDP algorithm is presented in *Algorithm 2*. The input to the algorithm is an RSP data representation of a big dataset and the number of RSP data blocks *b*. The output is a set of *b* values indicating the numbers of clusters found in the *b* random samples and *b* sets of cluster centers.

The I-niceDP algorithm is explained below. First, Line 2 randomly selects *b* RSP data blocks. Starting from Line 3, each RSP data block is computed separately as follows:

1 Line 4 generates an observation point as a reference for computing the distance distribution of objects.
2 Line 5 computes a set of distances between the observation point and the data points of an RSP sample to form the distance vector.
3 Line 6 computes the maximal number of GMM components $\mathcal{M}_{\max}$ using the kernel density estimation (KDE) method, where Δ_1 and Δ_2 are two thresholds that control the potential number of components.
4 Lines 7-10 compute a set of GMMs from the distance vector with the number of components smaller than or equal to the maximal number $\mathcal{M}_{\max}$, and each GMM model is built using the EM algorithm.

5 Lines 12-14 select the most fitted model based on the AICc criterion.
6 Lines 15–17 determine the high-density data points for each GMM component using the density peak mechanism, and these high-density data points are used as the initial cluster centers.
7 Finally, lines 18-19 assign the initial cluster centers to the *k*-means algorithm to cluster the input data and refine the cluster centers as the output result of the random sample.

After all RSP data blocks are computed, the set of refined cluster centers as defined in Eq. (4) is generated.

Algorithm for identifying two clusters being one using ball model

The I-niceDP algorithm generates a set of clusters from b random samples. Some of these clusters are likely sampled from the same cluster of the big dataset, so they have to be merged into one cluster as an approximation of the true cluster in the big dataset. *Algorithm 3* is designed to use the ball model to identify the two clusters which are likely to be one cluster in the big dataset.

The inputs to the algorithm are two clusters C_i and C_j and their cluster centers c_i and c_j. Line 2 computes the radii of the two clusters r_i and r_j. Lines 3-4 build two cluster balls CB_i and CB_j. Line 6 checks if the two balls are disjoint, set Merge = false. Lines 7-8 check if the two balls overlap, set Merge = false; otherwise, set Merge = true in Line 10. Output CB_j if Merge = true; otherwise, output nothing.

Algorithm 3: Algorithm for identifying two clusters being one using ball model.

Input : C_i, C_j; the ith and jth clusters.
c_i, c_j; centers of clusters C_i, C_j.

```
1  begin
2      Compute the radii of the two clusters r_i and r_j using Eq. (5);
3      CB_i = (C_i, c_i, r_i);
4      CB_j = (C_j, c_j, r_j);
5      The following conditions are satisfied while mapping the centers.
6      if CB_i ∩ CB_j = ∅ is disjoint then Merge=false;
7      else if CB_i ∩ CB_j ≠ ∅ then
8          if CB_i ∩ CB_j do not satisfy 1/2-ball intersection property then Merge=false;
9          else
10             Merge=true;
11         end
12     end
13 end
```

Output: CB_j If Merge = true, else nothing

Algorithm for ensembling the numbers of clusters in multiple samples

Finally, the pseudo code of the RSPCE algorithm is illustrated in *Algorithm 4*. The inputs are a big dataset $\mathbb{D}$ and the sample size n. First, in Line 2, *Algorithm 1*, shown as the operator RSP(), is called to convert $\mathbb{D}$ into a set of m RSP data blocks. Line 3 randomly selects b RSP blocks. Lines 4–8 call *Algorithm 2* as operator I-niceDP () and operator k-means () to compute the set of initial cluster centers C^*. Lines 11–13 call *Algorithm 3* to find the clusters which are likely to be the same cluster. Line 14 merges the clusters as one and adds it to the set of the final cluster centers. Line 16 counts the number of the

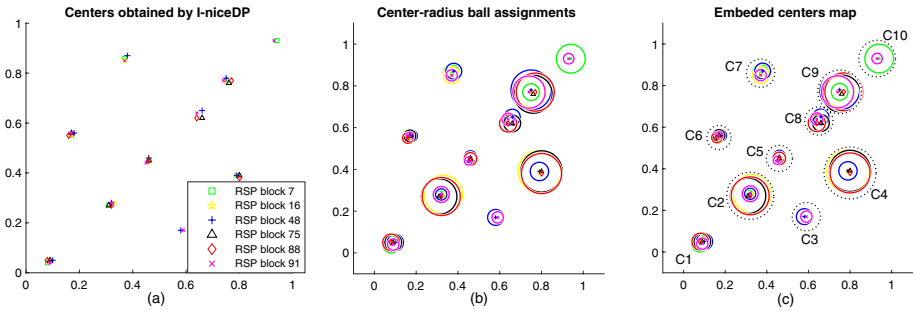


Fig. 2 Illustration of the results of three steps of the RSPCE algorithm from six random samples of dataset DS1. **a** Individual centers were obtained from the 6 randomly chosen samples by I-niceDP, **b** Cluster balls of the centers, **c** Ensemble centers of the 6 samples by the RSPCE algorithm

final cluster centers, CF. Finally, the algorithm outputs the number of clusters and the set of cluster centers.

Figure 2 illustrates the results of the three steps of the RSPCE algorithm. Figure 2a shows all refined centers found from 6 random samples of dataset DS1. We can see the sets of clusters from the 6 random samples are very similar. Figure 2b plots all cluster balls, and Fig. 2c shows the final set of cluster centers which are close to the true centers.

Algorithm 4: The RSPCE algorithm.

Input : $\mathbb{D}$; A big dataset.
n; Number of objects in each random sample.

begin
  $(D_1, D_2, ..., D_m) = \mathbf{RSP}(\mathbb{D}, \mathbf{n})$;
  $(D_1, D_2, ..., D_b) \subset (D_1, D_2, ..., D_m)$;
  forall $D_i \in (D_1, D_2, ..., D_b)$ **do**
    $(k_i, C_i) = \mathbf{I\text{-}niceDP}(D_i)$;
    $C_i^* = \mathbf{k\text{-}means}(k_i, D_i, C_i)$;
    $C^* = C^* + C_i^*$;
  end
  forall $C_p \in C^*$ **do**
    $\mathsf{CF}_p = \emptyset$;
    forall $C_q \in C^*$ **do**
      $\mathsf{CF}_p = \mathbf{Algorithm\_3}(C_p, C_q, c_p, c_q)$;
    end
    $\mathsf{CF} = \mathsf{CF} + \mathbf{merge}(\mathsf{CF}_p)$;
  end
  $K = \mathbf{count}(\mathsf{CF})$;
end

Output: K, CF

Complexity analysis

Given a big dataset with N objects, we have the following major parts that need to be considered: generating an RSP data representation, randomly selecting a subset of random samples; finding the number of clusters of each random sample, using the k-means algorithm to refine the initial cluster centers of each random sample, and finally, using the ball model to ensemble the results of the multiple random samples.

Suppose the number of objects in each random sample is n, and b samples are randomly selected from m, where $b < m$. The random sample generation operation has a complexity of $\mathcal{O}(n \log(N/n))$. I-niceDP algorithm generates O one-dimensional data and density peaks, and hence the time complexity of this algorithm is $\mathcal{O}(bnOK)$, where O is

the number of observation points. The complexity of k-means algorithm is $\mathcal{O}(bnTdK)$, where T is the maximum number of iterations. The time complexity of cluster ball learning process is $\mathcal{O}(bK)$. Therefore, the overall complexity of the RSPCE algorithm is $\mathcal{O}(n\log(N/n) + bnOK + bnTdK + bK) = \mathcal{O}(n\log(N/n) + (nO + nTd + 1)bK)$, which is linear to the number of data blocks b.

The RSPCE algorithm is implemented in a distributed platform with Q nodes, the computational complexity of the RSPCE algorithm can be reduced to $(\mathcal{O}(n\log(N/n) + (nO + nTd + 1)bK))/Q$. Therefore, the proposed RSPCE algorithm is efficient and scalable.

Experiments

A series of experiments were conducted on both synthetic and real-world datasets to demonstrate the performance of the proposed RSPCE algorithm and show its practical efficiency. In this section, the datasets and the experiment settings are presented. Evaluation measures are defined. The experiment results are analyzed, and the homogeneity of the results is discussed. Finally, the computational efficiency and scalability of the algorithm are demonstrated.

Datasets

The characteristics of the synthetic and real-world datasets used in the experiments are summarized in Table 1 and described below:

- Synthetic datasets. Five synthetic datasets, named DS1 to DS5, were generated in dimensions of 2 and 10 with different numbers of clusters in multivariate normal distributions. The numbers of clusters, the sizes of each cluster, the dimensions and total objects in these datasets are given in Table 1.
- Real-world datasets. Four real-world datasets used in the experiments are the following: *Covertype*[1] dataset with 581,012 objects describes 7 forest cover types in 54 different geographic measurements. There are 84% of objects in 2 types (type-1 36.5% and type-2 48.7%). The rest 16% of the objects are in other 5 types. *KDD'99ID*[2] dataset with about 5 million objects describes the connections of sequences of network intrusion detection. It has 23 classes, and 98.3% of the dataset belong to 3 classes (normal 19.6%, neptune 21.6%, and smurf 56.8%). *PokerHand*[3] dataset has more than 1 million objects, each being an example of a hand consisting of five playing cards drawn from a standard deck of 52. The dataset has 10 predictive features and 10 classes with two dominant classes accounting for over 90% of the samples (nothing in hand 49.9% and one pair 42.4%). *SUSY*[4] dataset was generated with Monte Carlo simulations. It has 5 million objects, 18 features and 2 classes.

[1] https://archive.ics.uci.edu/ml/datasets/covertype.
[2] https://www.kdd.org/kdd-cup/view/kdd-cup-1999/Data.
[3] https://archive.ics.uci.edu/ml/datasets/Poker+Hand.
[4] https://archive.ics.uci.edu/ml/datasets/SUSY.

Table 1 Characteristics of the datasets (*d*: dimensions, *N*: number of objects, *K*: number of clusters or classes)

Dataset	*d*	Cluster/class sizes	*N*	*K*
DS1	2	3 clusters; each has 50,000 4 clusters; each has 100,000 3 clusters; each has 150,000	1,000,000	10
DS2	2	7 clusters; each has 25,000 1 cluster; has 30,000 1 cluster; has 45,000 6 clusters; each has 50,000 3 clusters; each has 75,000 1 cluster; has 100,000 1 cluster; has 125,000	1,000,000	20
DS3	10	3 clusters; each has 15,000 3 clusters; each has 20,000 11 clusters; each has 25,000 1 cluster; has 30,000 2 clusters; each has 35,000 4 clusters; each has 40,000 3 clusters; each has 50,000 1 cluster; has 60,000 2 clusters; each has 75,000	1,000,000	30
DS4	10	5 clusters; each has 5000 2 clusters; each has 10,000 4 clusters; each has 15,000 4 clusters; each has 20,000 16 clusters; each has 25,000 1 cluster; has 35,000 1 cluster; has 40,000 5 clusters; each has 50,000 1 cluster; has 60,000 1 cluster; has 75,000	1,000,000	40
DS5	10	1 cluster; has 10,000 19 clusters; each has 15,000 14 clusters; each has 20,000 13 clusters; each has 25,000 1 cluster; has 30,000 2 clusters; each has 35,000	1,000,000	50
Covertype	54	211,840 : 283,301 : 35,754 : 27,747 : 8483 : 17,367 : 20,510	581,012	7
KDD'99ID	41	972,781 : 2,807,886 : 1,072,017 : 53 : 979 : 264 : 21 : 2203 : 8 : 12 : 2 : 1020 : 20 : 7 : 4 : 30 : 9 : 10 : 3 : 12,481 : 15,892 : 2316 : 10,413	4,940,000	23
PokerHand	10	513,702 : 433,097 : 48,828: 3978 : 21,634 : 2050 : 1460 : 236 : 17 : 8	1,025,010	10
SUSY	18	2,712,173 : 2,287,827	5,000,000	2

In the four real-world datasets, the *K* values in the middle column are the numbers of objects in *K* classes

The small classes are underlined

Table 2 Parameter settings used in the experiments

Ensemble sizes (*es*)	Sample sizes (*n*)
{5, 10, 20, 30, 40, 50}%	A = 5000; B = 10,000

Experiment settings

In the experiments, all datasets were converted to RSP data representations, i.e., each dataset being transformed into a set of RSP data blocks. The left column of Table 2 shows the percentages of the total RSP blocks used to estimate the number of clusters. In the experiments, six different sizes of subsets of RSP blocks were used. The right column shows the two block sizes used to partition the synthetic datasets and the real-world dataset *Covertype*. The other three real-world datasets were partitioned with the block sizes of {1, 2, 5, 10, 15, and 20}% of the whole datasets. Therefore, each dataset is transformed into more than one RSP representation.

Six existing methods were selected for comparison of the performance of the proposed RSPCE algorithm. They are nselectboot [25], kluster [26], X-means [27], Elbow [1], Silhouette [2], and Gap statistics [3]. The number of clusters K in the last four methods was assigned to $K = 2$ to 100. For the bootstrap method of kluster, the number of the bootstrap samples was set to 20.

The experiments were performed on three local nodes equipped with x64-based processor, Intel(R) core i7–7700, CPU 3.60Hz, 8 GB of memory, and 1 TB of storage. The RSPCE algorithm was implemented in Python−3.7.3. with py2r, fpc, densityClust and clvalid R packages. Three observation points were used in the step of I-niceDP of the RSPCE algorithm.

Evaluation metrics

The internal and stability measures below were used to evaluate the results of the comparison methods and the RSPCE algorithm.

Internal measures

The following internal measures were used to evaluate the compactness, connectivity, and separation of the cluster partitions.

Inertia or within-cluster sum-of-squares (SSE) measures the internal coherence of objects in a cluster [32]. The lower the inertia value, the better the cluster. Zero is optimal.

The *silhouette coefficient (SC)* [2] evaluates the clustering quality by combining the ideas on how well the clusters are separated (i.e., separation) and how compact are the clusters (i.e., tightness). The SC of clusterings is computed as follows:

$$\mathrm{SC} = \frac{b - a}{\max(a, b)} \tag{7}$$

where a is the average distance between a cluster and all other data points in the same cluster, and b is the average distance between a cluster and all other data points in the nearest cluster. We can calculate the average SC as the mean of the SC for all samples. A higher score closer to 1 is related to a model with better-defined clusters.

Davies-Bouldin index (DBI) [33] is an internal evaluation metric, which is used to validate the clustering process using quantities and data points residing in the dataset. The DBI for K clusters is defined as

$$\text{DBI}\,(K) = \frac{1}{K}\sum_{i=1}^{K}\max_{i\neq j}\frac{\Delta(C_i)+\Delta(C_j)}{\delta(C_i,C_j)} \tag{8}$$

where $\delta(C_i, C_j)$ is the inter-cluster distance, i.e., the distance between clusters C_i and C_j, $\Delta(C_i)$ is the intra-cluster distance of cluster C_i, i.e., distance within the cluster C_i. The lower the DBI value, the better the clustering result.

It is reasonable to define some intuitive metrics using conditional entropy analysis under the ground truth class assignments information. *V-measure* [34] is an entropy-based measure that explicitly measures how successfully the criteria of homogeneity and completeness are satisfied. V-measure is computed as the harmonic mean of distinct homogeneity and completeness scores. *Homogeneity* captures only the information of the members in a single class for each cluster, whereas *completeness* captures the information of all members of a given class assigned to the same cluster. V-measure is equivalent to normalized mutual information (NMI) metric.

The adjusted rand index (ARI) [35] and the adjusted mutual information (AMI) [36] are also used to evaluate the performance of the RSPCE algorithm. These two measures are defined as follows:

Given the ground truth result $P = \{C_1, C_2, ..., C_k\}$ with K clusters and the predicted result $P' = \{C'_1, C'_2, ..., C'_{k'}\}$ with K' clusters, the *adjusted rand index (ARI)* [35] measures the similarity of the two assignments defined as

$$\text{ARI}\,(P,P') = \frac{\sum_{i=1}^{K}\sum_{j=1}^{K'}\binom{|C_i\cap C'_j|}{2} - X_3}{\frac{1}{2}(X_1+X_2)-X_3} \tag{9}$$

where

$$X_1 = \sum_{i=1}^{K}\binom{|C_i|}{2}, X_2 = \sum_{j=1}^{K'}\binom{|C'_j|}{2}, X_3 = \frac{2X_1X_2}{n(n-1)} \tag{10}$$

where $i \in \{1, ..., K\}, j \in \{1, ..., K'\}, n$ is the total number of data samples, and $|.|$ denotes the cardinality of the cluster. ARI value varies between zero and one. The higher value indicates that the resulted clustering outcome is more close to the actual one.

AMI [36] is defined as

$$\text{AMI}\,(P,P') = \frac{\text{NMI}_{max}(P,P') - \mathbf{E}\{\,\text{NMI}_{\max}(P,P')\}}{1-\mathbf{E}\{\,\text{NMI}_{\max}(P,P')\}} \tag{11}$$

where

$$\text{NMI}\,(P,P') = \frac{2\varphi(P;P')}{\phi(P)+\phi(P')} \tag{12}$$

$$\varphi(P; P') = \sum_i \sum_j \frac{|C_i \cap C'_j|}{n} \log \frac{n\left|C_i \cap C'_j\right|}{|C_i||C'_j|} \tag{13}$$

$$\phi(P) = -\sum_i \frac{|C_i|}{n} \log \frac{|C_i|}{n} \tag{14}$$

$$\phi(P') = -\sum_j \frac{|C'_j|}{n} \log \frac{|C'_j|}{n} \tag{15}$$

AMI values are between 0 and 1. The higher the AMI value, the better the quality of clusters.

Stability measures

The average proportion of nonoverlap (APN) and the average distance between means (ADM) [32] were used to measure the stability and consistency of the results by comparing the ground truth of the entire dataset with the obtained number of clusters in a sample.

Let C_i represent the true clusters via an ideal clustering process, and C'_j be the clusters on the b random samples. Given the total number of clusters K, the APN measure is defined as

$$\text{APN}\ (P, P') = \frac{1}{KK'} \sum_{i=1}^{K} \sum_{j=1}^{K'} \left(1 - \frac{n(C'_j \cap C_i)}{n(C_i)}\right). \tag{16}$$

The APN resides in [0, 1], and the value close to zero corresponds to highly consistent clustering results.

The *ADM* computes the average distance between cluster centers determined based on the entire dataset and the random samples. It is defined as

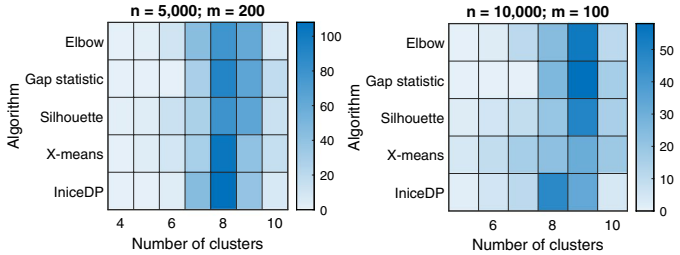
$$\text{ADM}\ (P, P') = \frac{1}{KK'} \sum_{i=1}^{K} \sum_{j=1}^{K'} \text{dist}\ (C_i, C'_j). \tag{17}$$

where C_i is the mean of the objects in a cluster which contains object i on the entire dataset, and C'_j is the predicted one defined on the random samples. This metric is based on the Euclidean distance. It also has a value between 0 and 1, and smaller values are preferred.

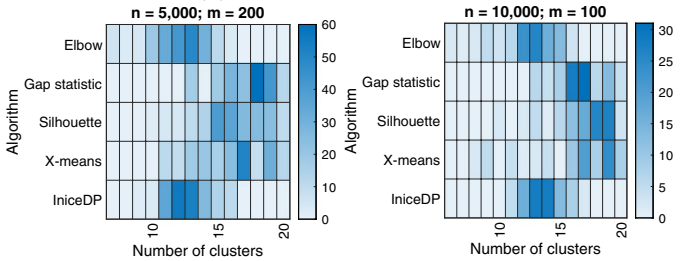
Experiment results and analysis

Results of the number of clusters

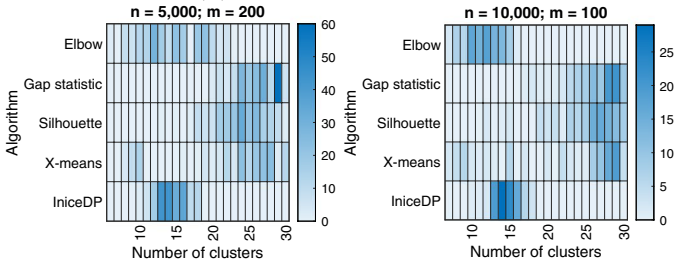
The first set of experiments was to use five existing methods to identify the number of clusters from random samples of the five synthetic datasets in Table 1. Two sample sizes of 5000 points and 10,000 points were used. For each random sample in a synthetic dataset, the number of clusters in the sample was discovered by the five methods. Since there are m random samples in one synthetic dataset for each sample size, m results of the



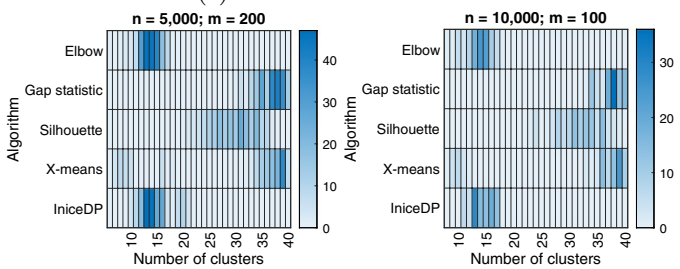
(a) DS1 with 10 clusters.



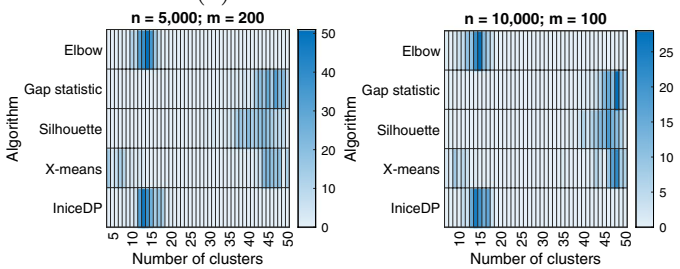
(b) DS2 with 20 clusters.



(c) DS3 with 30 clusters.



(d) DS4 with 40 clusters.



(e) DS5 with 50 clusters.

Fig. 3 Heatmaps of the results of the five methods on *m* random samples in the five synthetic datasets with two RSP representations *n*: 5000 and 10,000. The columns of each figure are the numbers of clusters, and the rows are the five methods. The dark color in a cell indicates that a high percentage of the *m* results were identified as the number of clusters by the column with the corresponding method

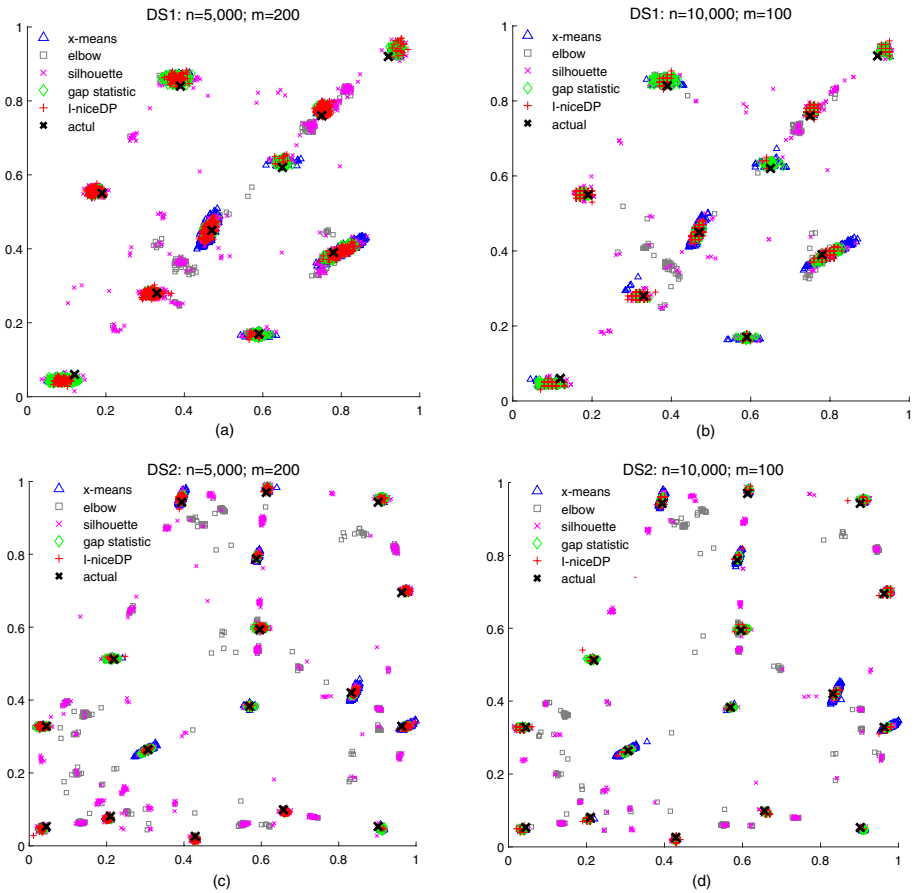
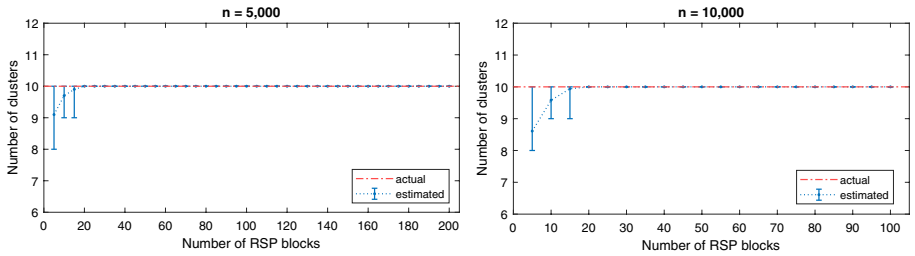


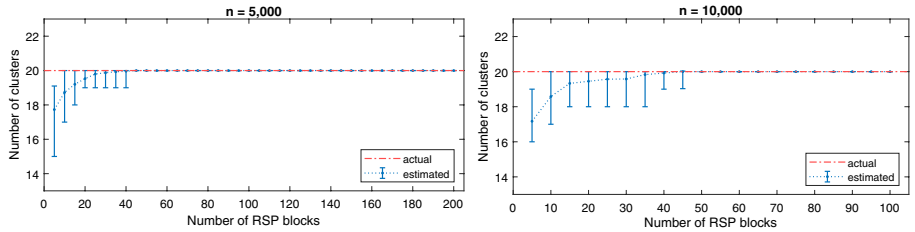
Fig. 4 Scatter plots of identified centers in random samples of different sizes, achieved by different algorithms from DS1 (**a** and **b**) and DS2 (**c** and **d**) datasets. The centers estimated by the algorithms are indicated by distinct symbols and colors

number of clusters were found by each method. The heatmaps of the results of the five methods on m random samples in the five synthetic datasets with two RSP representations each are shown in Fig. 3. The columns of each figure are the numbers of clusters, and the rows are the five methods. The dark color in a cell indicates that a high percentage of the m results was identified as the number of clusters by the column with the corresponding method. For example, in the right figure of Fig. 3a, the dark cell of the second column from the right in the row of Gap statistic implies that the majority number of clusters identified by Gap statistic from 100 random samples of data DS1 is 9. From Fig. 3, we can see that Gap statistic, Silhouette and X-means performed better than Elbow and I-niceDP in identifying the number of clusters from the random samples of a big dataset. Another observation is that a bigger sample size results in a more accurate result.

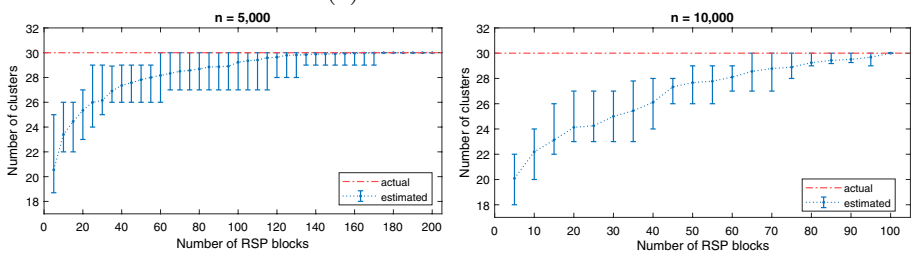
Using the two-dimensional synthetic datasets of DS1 and DS2, we compared the true cluster centers with the cluster centers identified by the five methods from the random samples of the two datasets. The results are plotted in Fig. 4. We can see that the cluster centers identified by I-niceDP are closer to the true centers than those identified by the other four methods. These results indicate that I-niceDP is more capable in identifying



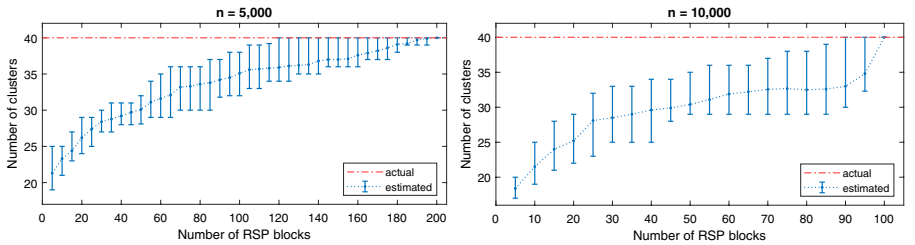
(a) DS1 with 10 clusters.



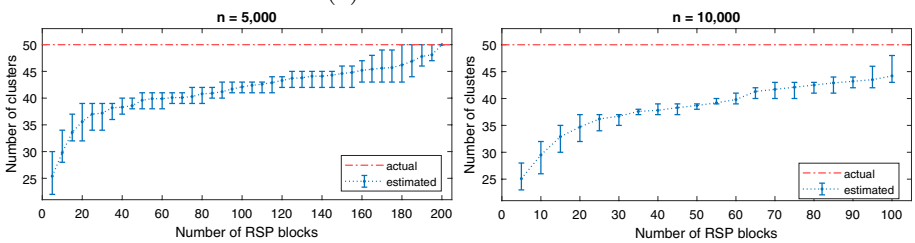
(b) DS2 with 20 clusters.



(c) DS3 with 30 clusters.



(d) DS4 with 40 clusters.



(e) DS5 with 50 clusters.

Fig. 5 Performance of the RSPCE algorithm on five synthetic datasets on the maximal, average and minimal numbers of clusters over 20 runs against the number of RSP blocks used

better initial cluster centers than other existing methods, so it is chosen in the operator to identify the number of clusters from a random sample in the algorithm.

Figure 5 shows the performance of the RSPCE algorithm in identifying the number of clusters in the five synthetic datasets with two different sample sizes. The horizontal

axis in each plot is the number of random samples used by the algorithm. The vertical axis shows the number of clusters identified. The horizontal straight line indicates the true number of clusters in each dataset. For the same number of random samples, the RSPCE algorithm was run 20 times on each RSP representation of a synthetic dataset. From Fig. 5, we can see that for the two-dimensional datasets DS1 and DS2 with fewer clusters, the RSPCE algorithm can easily identify the true number of clusters with a few random samples. For the high-dimensional dataset DS3 with fewer clusters, the RSPCE algorithm can also converge to the true number of clusters as the number of random samples increased. However, for the two high-dimensional datasets DS4 and DS5, the RSPCE algorithm needs more random samples to converge to the true number of clusters. Another observation in the ensemble method is that smaller samples gave better results than the bigger samples. The reason may be that more smaller samples generate more diverse results, which can improve the final ensemble result. However, more investigations are required to give a firm conclusion on this observation.

Improvements of clustering results

Having obtained the number of clusters in each synthetic dataset and the initial cluster centers by the RSPCE algorithm, we used the *k*-means algorithm to cluster each random sample used in the RSPCE algorithm with the number of clusters and the initial cluster centers as input parameters. For each set of random samples, we used 8 internal measures to validate the clustering results. Table 3 shows all validation results of 5 synthetic datasets with 2 sample sizes and 6 different subsets of random samples listed in column *es*. We can see clearly that the clustering result was improved as more random samples were used by the RSPCE algorithm. Again, smaller random sample sizes resulted in better clustering results. This observation is consistent with the one from Fig. 5.

We also investigated the stability and consistency performance of the RSPCE algorithm in detecting cluster centers. The APN and ADM measures were used to evaluate the clustering consistency by comparing the results obtained in different numbers of random samples. The results of APN and ADM scores are shown in Table 4. It appears that the APN and ADM scores tend to decrease as the number of random samples increases. Again, the RSPCE algorithm performed significantly better on the datasets with smaller numbers of clusters. The set of random samples containing $10 \sim 20\%$ of the big dataset gave the better results. These results also show that the RSPCE algorithm can generate stable cluster centers which are close to the centers of true clusters in the entire dataset.

In the experiments, we observed that small samples ($n < 2000$) often miss mini-clusters in the sample, or they do not have enough points to categorize small clusters. Increasing the random sample size can solve this problem, but the computing cost also increases. A tradeoff on the sample size needs considerations in practice.

Statistical homogeneity test

Homogeneity tests were conducted to verify the cluster centers discovered from random samples. The distribution of the distances between these centers should be similar to the distribution of the distances of the true centers in the big dataset. We conducted

Table 3 Validations of 6 internal and 2 external measures on the clustering results of 5 synthetic datasets with 2 sample sizes (A = 5000; B = 10,000) and 6 different subsets of random samples listed in column *es*

Dataset	*n*	*es*	SSE	Silhouette	DB index	Homogeneity	Completeness	V-measure	AMI	ARI
DS1	A	5%	7.398 ± 3.013	0.849 ± 0.016	0.231 ± 0.037	0.971 ± 0.025	1.000 ± 0.000	0.985 ± 0.013	0.971 ± 0.025	0.967 ± 0.028
		10%	5.716 ± 3.128	0.859 ± 0.016	0.209 ± 0.038	0.986 ± 0.025	1.000 ± 0.000	0.993 ± 0.013	0.986 ± 0.025	0.984 ± 0.028
		20%	3.993 ± 0.116	0.867 ± 0.001	0.190 ± 0.003	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		30%	3.939 ± 0.053	0.868 ± 0.001	0.188 ± 0.002	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		40%	3.910 ± 0.024	0.868 ± 0.000	0.188 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		50%	3.894 ± 0.033	0.868 ± 0.000	0.188 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	B	5%	22.386 ± 15.800	0.830 ± 0.036	0.270 ± 0.080	0.954 ± 0.035	0.999 ± 0.002	0.976 ± 0.019	0.954 ± 0.035	0.948 ± 0.041
		10%	14.165 ± 5.580	0.850 ± 0.015	0.227 ± 0.035	0.974 ± 0.024	1.000 ± 0.000	0.987 ± 0.012	0.974 ± 0.024	0.971 ± 0.026
		20%	10.542 ± 5.089	0.860 ± 0.014	0.206 ± 0.031	0.989 ± 0.022	1.000 ± 0.000	0.995 ± 0.011	0.989 ± 0.022	0.988 ± 0.024
		30%	7.900 ± 0.214	0.867 ± 0.001	0.189 ± 0.004	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		40%	7.635 ± 0.229	0.868 ± 0.002	0.188 ± 0.004	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		50%	7.514 ± 0.129	0.869 ± 0.001	0.186 ± 0.002	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
DS2	A	5%	10.928 ± 5.194	0.855 ± 0.031	0.309 ± 0.036	0.939 ± 0.029	1.000 ± 0.000	0.968 ± 0.016	0.938 ± 0.029	0.896 ± 0.052
		10%	3.754 ± 3.128	0.908 ± 0.027	0.182 ± 0.093	0.981 ± 0.018	1.000 ± 0.000	0.991 ± 0.009	0.981 ± 0.019	0.971 ± 0.028
		20%	1.667 ± 1.788	0.926 ± 0.015	0.120 ± 0.045	0.993 ± 0.012	1.000 ± 0.000	0.997 ± 0.006	0.993 ± 0.012	0.989 ± 0.019
		30%	0.638 ± 0.010	0.935 ± 0.001	0.093 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		40%	0.635 ± 0.011	0.935 ± 0.000	0.093 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		50%	0.632 ± 0.006	0.935 ± 0.001	0.093 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	B	5%	18.394 ± 10.015	0.865 ± 0.037	0.287 ± 0.051	0.948 ± 0.032	1.000 ± 0.000	0.974 ± 0.017	0.948 ± 0.032	0.911 ± 0.061
		10%	11.054 ± 8.737	0.894 ± 0.036	0.226 ± 0.096	0.971 ± 0.028	1.000 ± 0.000	0.985 ± 0.015	0.971 ± 0.028	0.951 ± 0.051
		20%	7.470 ± 4.989	0.908 ± 0.022	0.191 ± 0.078	0.982 ± 0.015	1.000 ± 0.000	0.991 ± 0.007	0.982 ± 0.015	0.972 ± 0.022
		30%	7.518 ± 4.322	0.907 ± 0.019	0.197 ± 0.069	0.982 ± 0.013	1.000 ± 0.000	0.991 ± 0.006	0.982 ± 0.013	0.972 ± 0.019

Table 3 (continued)

Dataset	*n*	*es*	SSE	Silhouette	DB index	Homogeneity	Completeness	V-measure	AMI	ARI
		40%	6.027 ± 3.173	0.914 ± 0.014	0.176 ± 0.060	0.987 ± 0.009	0.987 ± 0.009	0.993 ± 0.005	0.987 ± 0.009	0.979 ± 0.015
		50%	1.264 ± 0.008	0.935 ± 0.000	0.093 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
DS3	A	5%	14.204 ± 8.207	0.761 ± 0.075	0.411 ± 0.096	0.907 ± 0.050	1.000 ± 0.000	0.951 ± 0.027	0.905 ± 0.050	0.846 ± 0.083
		10%	9.462 ± 3.442	0.803 ± 0.034	0.342 ± 0.028	0.933 ± 0.023	1.000 ± 0.000	0.965 ± 0.012	0.932 ± 0.023	0.885 ± 0.043
		20%	4.252 ± 1.560	0.861 ± 0.021	0.263 ± 0.044	0.968 ± 0.012	1.000 ± 0.000	0.984 ± 0.007	0.968 ± 0.013	0.947 ± 0.022
		30%	3.357 ± 2.198	0.873 ± 0.029	0.229 ± 0.077	0.976 ± 0.019	1.000 ± 0.000	0.988 ± 0.010	0.976 ± 0.019	0.961 ± 0.032
		40%	2.531 ± 1.776	0.884 ± 0.025	0.201 ± 0.066	0.983 ± 0.017	1.000 ± 0.000	0.991 ± 0.009	0.982 ± 0.017	0.971 ± 0.030
		50%	2.095 ± 1.392	0.889 ± 0.019	0.196 ± 0.047	0.986 ± 0.013	1.000 ± 0.000	0.993 ± 0.007	0.986 ± 0.013	0.979 ± 0.024
	B	5%	39.314 ± 10.666	0.715 ± 0.038	0.478 ± 0.077	0.875 ± 0.026	1.000 ± 0.000	0.933 ± 0.015	0.874 ± 0.026	0.796 ± 0.040
		10%	29.491 ± 10.190	0.752 ± 0.041	0.410 ± 0.053	0.900 ± 0.029	1.000 ± 0.000	0.947 ± 0.016	0.899 ± 0.029	0.832 ± 0.047
		20%	17.259 ± 8.164	0.810 ± 0.045	0.327 ± 0.070	0.936 ± 0.027	1.000 ± 0.000	0.967 ± 0.014	0.936 ± 0.028	0.890 ± 0.046
		30%	15.913 ± 7.190	0.818 ± 0.039	0.323 ± 0.058	0.941 ± 0.024	1.000 ± 0.000	0.970 ± 0.013	0.940 ± 0.025	0.898 ± 0.041
		40%	11.339 ± 6.358	0.843 ± 0.039	0.285 ± 0.059	0.958 ± 0.023	1.000 ± 0.000	0.978 ± 0.012	0.957 ± 0.023	0.927 ± 0.042
		50%	6.694 ± 3.438	0.873 ± 0.022	0.238 ± 0.046	0.975 ± 0.015	1.000 ± 0.000	0.988 ± 0.008	0.975 ± 0.015	0.960 ± 0.028
DS4	A	5%	17.860 ± 5.959	0.687 ± 0.051	0.543 ± 0.075	0.855 ± 0.040	0.999 ± 0.001	0.921 ± 0.023	0.852 ± 0.040	0.785 ± 0.070
		10%	9.654 ± 2.088	0.767 ± 0.027	0.423 ± 0.034	0.913 ± 0.017	0.999 ± 0.001	0.954 ± 0.009	0.910 ± 0.017	0.877 ± 0.025
		20%	8.015 ± 2.516	0.792 ± 0.037	0.384 ± 0.059	0.928 ± 0.022	0.999 ± 0.001	0.962 ± 0.012	0.925 ± 0.022	0.899 ± 0.033
		30%	4.050 ± 2.283	0.847 ± 0.033	0.295 ± 0.064	0.963 ± 0.022	1.000 ± 0.001	0.981 ± 0.012	0.962 ± 0.023	0.955 ± 0.037
		40%	3.149 ± 1.398	0.861 ± 0.017	0.267 ± 0.027	0.972 ± 0.013	1.000 ± 0.000	0.986 ± 0.006	0.971 ± 0.014	0.970 ± 0.022
		50%	1.984 ± 0.834	0.875 ± 0.012	0.233 ± 0.036	0.984 ± 0.009	1.000 ± 0.000	0.992 ± 0.004	0.983 ± 0.009	0.985 ± 0.010
	B	5%	45.941 ± 3.544	0.650 ± 0.016	0.557 ± 0.045	0.819 ± 0.009	0.996 ± 0.005	0.900 ± 0.005	0.817 ± 0.009	0.722 ± 0.006
		10%	36.157 ± 9.622	0.688 ± 0.040	0.509 ± 0.050	0.852 ± 0.034	0.997 ± 0.004	0.919 ± 0.020	0.850 ± 0.035	0.776 ± 0.060
		20%	22.751 ± 7.440	0.748 ± 0.039	0.452 ± 0.054	0.902 ± 0.028	1.000 ± 0.001	0.948 ± 0.016	0.900 ± 0.029	0.860 ± 0.043

Table 3 (continued)

Dataset	*n*	*es*	SSE	Silhouette	DB index	Homogeneity	Completeness	V-measure	AMI	ARI
		30%	14.541 ± 7.407	0.798 ± 0.050	0.373 ± 0.077	0.934 ± 0.030	1.000 ± 0.000	0.966 ± 0.016	0.933 ± 0.030	0.909 ± 0.048
		40%	8.740 ± 4.905	0.841 ± 0.035	0.306 ± 0.063	0.959 ± 0.022	1.000 ± 0.000	0.979 ± 0.011	0.958 ± 0.022	0.948 ± 0.034
		50%	5.240 ± 1.944	0.866 ± 0.016	0.257 ± 0.036	0.976 ± 0.011	1.000 ± 0.000	0.988 ± 0.006	0.975 ± 0.012	0.974 ± 0.017
DS5	A	5%	10.964 ± 1.646	0.694 ± 0.026	0.499 ± 0.046	0.879 ± 0.015	1.000 ± 0.000	0.936 ± 0.008	0.873 ± 0.015	0.748 ± 0.022
		10%	8.272 ± 2.108	0.738 ± 0.035	0.444 ± 0.037	0.906 ± 0.023	1.000 ± 0.000	0.951 ± 0.013	0.902 ± 0.024	0.800 ± 0.048
		20%	6.011 ± 0.872	0.778 ± 0.018	0.401 ± 0.024	0.929 ± 0.008	1.000 ± 0.000	0.963 ± 0.004	0.925 ± 0.008	0.847 ± 0.016
		30%	5.904 ± 0.743	0.779 ± 0.015	0.398 ± 0.020	0.930 ± 0.006	1.000 ± 0.000	0.964 ± 0.003	0.926 ± 0.006	0.848 ± 0.013
		40%	5.086 ± 0.483	0.796 ± 0.012	0.373 ± 0.017	0.936 ± 0.006	1.000 ± 0.000	0.967 ± 0.003	0.932 ± 0.006	0.860 ± 0.011
		50%	4.118 ± 0.516	0.819 ± 0.011	0.342 ± 0.014	0.949 ± 0.006	1.000 ± 0.000	0.974 ± 0.004	0.946 ± 0.007	0.883 ± 0.012
	B	5%	33.551 ± 4.208	0.623 ± 0.019	0.583 ± 0.058	0.828 ± 0.012	1.000 ± 0.000	0.906 ± 0.007	0.825 ± 0.012	0.654 ± 0.020
		10%	25.541 ± 3.411	0.665 ± 0.021	0.505 ± 0.011	0.857 ± 0.019	0.999 ± 0.002	0.922 ± 0.011	0.854 ± 0.020	0.699 ± 0.038
		20%	18.505 ± 2.550	0.720 ± 0.014	0.469 ± 0.023	0.897 ± 0.009	1.000 ± 0.001	0.946 ± 0.005	0.895 ± 0.009	0.786 ± 0.017
		30%	17.424 ± 1.054	0.725 ± 0.009	0.458 ± 0.019	0.900 ± 0.005	1.000 ± 0.000	0.948 ± 0.003	0.898 ± 0.005	0.789 ± 0.013
		40%	15.016 ± 1.346	0.751 ± 0.016	0.425 ± 0.010	0.914 ± 0.009	1.000 ± 0.000	0.955 ± 0.005	0.912 ± 0.009	0.814 ± 0.022
		50%	12.226 ± 1.438	0.778 ± 0.014	0.400 ± 0.024	0.929 ± 0.007	1.000 ± 0.000	0.963 ± 0.004	0.927 ± 0.007	0.850 ± 0.019

Table 4 Stability validations on the results of the RSPCE algorithm on synthetic datasets with 2 sample sizes, A = 5000 and B = 10,000 (lower value is better)

Dataset	*n\es*	APN						ADM					
		5%	10%	20%	30%	40%	50%	5%	10%	20%	30%	40%	50%
DS1	A	0.0070	0	0	0	0	0	0.4648	0.4736	0.4666	0.4660	0.4697	0.4603
	B	0.0140	0.0050	0	0	0	0	0.4768	0.4751	0.4640	0.4628	0.4618	0.4599
DS2	A	0.0042	0.0009	0	0	0	0	0.5765	0.5762	0.5802	0.5802	0.5804	0.5800
	B	0.0045	0.0021	0.0011	0.0006	0.0001	0	0.5837	0.5785	0.5810	0.5805	0.5803	0.5801
DS3	A	0.0110	0.0077	0.0044	0.0031	0.0022	0.0013	0.5206	0.5236	0.5274	0.5262	0.5285	0.5304
	B	0.0300	0.0240	0.0215	0.0185	0.0144	0.0086	0.5211	0.5171	0.5208	0.5209	0.5220	0.5253
DS4	A	0.0417	0.0345	0.0270	0.0210	0.0160	0.0123	0.5065	0.5059	0.5089	0.5084	0.5071	0.5073
	B	0.0540	0.0462	0.0370	0.0288	0.0260	0.0240	0.5045	0.5079	0.5082	0.5039	0.5063	0.5027
DS5	A	0.0396	0.0288	0.0234	0.0190	0.0169	0.0124	0.4999	0.5004	0.5015	0.5026	0.5023	0.5022
	B	0.0498	0.0410	0.0306	0.0266	0.0244	0.0226	0.4980	0.4953	0.4977	0.4982	0.4987	0.4994

two-samples Kolmogorov-Smirnov (KS)-test and Z-test to compare the distance distributions of cluster centers between the entire dataset $\mathcal{G}$ and random samples $\mathcal{A}$.

The null-hypothesis is that $\mathcal{G}$ and $\mathcal{A}$ have the same distribution, whereas the alternative hypothesis implies that they have different distributions. We set $h = 1$ if we reject the null-hypothesis (i.e., the distributions are not the same); otherwise, we set $h = 0$ in the case of accepting the null hypothesis. We tested at a significant level of 5%. The p-value is the probability of having a false rejection in the case of a null hypothesis. The corresponding test results are presented in Table 5. We can see that the null-hypothesis is accepted in all cases. Figure 6 illustrates that the test CDFs (green, blue, cyan, magenta, yellow, and black) of random samples match the empirical CDF (red) of the whole dataset closely, and the highest difference is small.

Comparisons of RSPCE with other methods

We compared the results of the RSPCE algorithm in identifying the number of clusters from multiple random samples with the results of other methods in identify the number of clusters from one random sample. Table 6 shows the results of the synthetic datasets, and Table 7 shows the results of the real-world datasets. The sample size is 5,000 points. Different numbers of random samples, as shown in column *es*, were used in the RSPCE algorithm. For the same number of random samples, the RSPCE algorithm ran 20 times on different sets of random samples. Other methods were applied to each random sample to generate one result. The average value and the standard deviations from the multiple runs were calculated. We can see that the RSPCE algorithm performed the best in general for both synthetic datasets and real-world datasets.

Specifically, Silhouette, Gap statistic and the RSPCE algorithm are more accurate in identifying the number of clusters in all five synthetic datasets. Among them, the RSPCE algorithm performed best. kluster and Elbow performed well in DS1 but not well in other datasets. Neither nselectboot nor X-means performed well in all datasets. It seems that the advantage of the bootstrap method in nselectboot did not work well. nselectboot, X-means, and Gap statistics all overestimated the number of clusters. Elbow and kluster, on the other hand, underestimated the number of clusters in the last four datasets.

Moreover, RSPCE was able to identify the cluster centers more accurately. Except for the Silhouette and Gap statistic methods, none of them were able to identify the cluster centers of five synthetic datasets. The Silhouette and Gap statistic algorithms are Euclidean distance-based, and hence computationally expensive.

We examined the effectiveness of the RSPCE algorithm on four real-world datasets. The number of classes in these datasets were used as the "true" number of clusters. The corresponding results are displayed in Table 7. The original numbers of classes and their estimated clusters are well correlated with the results obtained by the RSPCE algorithm.

Computational efficiency

In this section, we compare the computation efficiency of seven methods for identifying the number of clusters against different data sizes. The results are plotted in Fig. 7, with the execution time measured in minutes. We can see that comparatively, Gap statistic and Silhouette methods were inefficient. Other methods performed on these datasets similarly in execution time.

Table 5 Results of two-samples KS-test and Z-test on two distance distributions among the actual cluster centers and the estimated cluster centers of the synthetic datasets by the RSPCE algorithm

Dataset	*n*		KS-test			Z-test		
		es	*h*	*k*	*p*	*h*	*z*	*p*
DS1	A	5%	0	0.083	0.998	0	0.407	0.684
		10%	0	0.111	0.930	0	− 0.712	0.476
		20%	0	0.089	0.992	0	− 0.381	0.703
		30%	0	0.089	0.992	0	− 0.461	0.645
		40%	0	0.089	0.992	0	− 0.586	0.558
		50%	0	0.111	0.930	0	− 0.518	0.605
	B	5%	0	0.172	0.552	0	− 0.764	0.445
		10%	0	0.144	0.766	0	− 0.932	0.352
		20%	0	0.089	0.992	0	− 0.439	0.661
		30%	0	0.089	0.992	0	− 0.419	0.675
		40%	0	0.067	1.000	0	− 0.178	0.859
		50%	0	0.067	1.000	0	− 0.090	0.928
DS2	A	5%	0	0.027	1.000	0	0.130	0.897
		10%	0	0.047	0.981	0	− 0.422	0.673
		20%	0	0.047	0.981	0	− 0.422	0.673
		30%	0	0.047	0.981	0	− 0.427	0.670
		40%	0	0.047	0.981	0	− 0.427	0.670
		50%	0	0.037	0.999	0	− 0.356	0.722
	B	5%	0	0.068	0.841	0	− 0.816	0.414
		10%	0	0.036	1.000	0	0.126	0.900
		20%	0	0.047	0.981	0	− 0.422	0.673
		30%	0	0.053	0.950	0	− 0.445	0.656
		40%	0	0.047	0.981	0	− 0.445	0.656
		50%	0	0.042	0.995	0	− 0.424	0.672
DS3	A	5%	0	0.037	0.975	0	0.675	0.499
		10%	0	0.029	0.998	0	− 0.250	0.802
		20%	0	0.027	0.998	0	0.597	0.551
		30%	0	0.027	0.998	0	0.597	0.551
		40%	0	0.012	1.000	0	0.029	0.977
		50%	0	0.018	1.000	0	0.088	0.930
	B	5%	0	0.027	1.000	0	0.387	0.699
		10%	0	0.043	0.935	0	0.793	0.428
		20%	0	0.044	0.901	0	0.635	0.525
		30%	0	0.057	0.606	0	1.277	0.202
		40%	0	0.033	0.988	0	0.583	0.560
		50%	0	0.022	1.000	0	0.092	0.927

Table 5 (continued)

Dataset	*n*		KS-test			Z-test		
		es	*h*	*k*	*p*	*h*	*z*	*p*
DS4	A	5%	0	0.044	0.811	0	− 0.986	0.324
		10%	0	0.031	0.986	0	0.239	0.811
		20%	0	0.038	0.795	0	0.928	0.353
		30%	0	0.039	0.750	0	0.884	0.377
		40%	0	0.022	0.998	0	0.117	0.907
		50%	0	0.030	0.930	0	0.419	0.675
	B	5%	0	0.073	0.382	0	1.347	0.178
		10%	0	0.031	0.995	0	− 0.066	0.947
		20%	0	0.046	0.833	0	− 0.788	0.431
		30%	0	0.050	0.707	0	− 1.024	0.306
		40%	0	0.036	0.939	0	− 0.243	0.808
		50%	0	0.036	0.853	0	0.933	0.351
DS5	A	5%	0	0.017	1.000	0	0.002	0.999
		10%	0	0.018	0.999	0	0.182	0.856
		20%	0	0.013	1.000	0	− 0.140	0.889
		30%	0	0.042	0.381	0	− 0.547	0.584
		40%	0	0.024	0.947	0	− 1.639	0.101
		50%	0	0.020	0.987	0	− 0.291	0.771
	B	5%	0	0.022	1.000	0	0.086	0.931
		10%	0	0.029	0.959	0	0.149	0.882
		20%	0	0.022	0.988	0	0.481	0.631
		30%	0	0.012	1.000	0	− 0.023	0.981
		40%	0	0.024	0.965	0	0.354	0.723
		50%	0	0.018	0.998	0	− 0.006	0.995

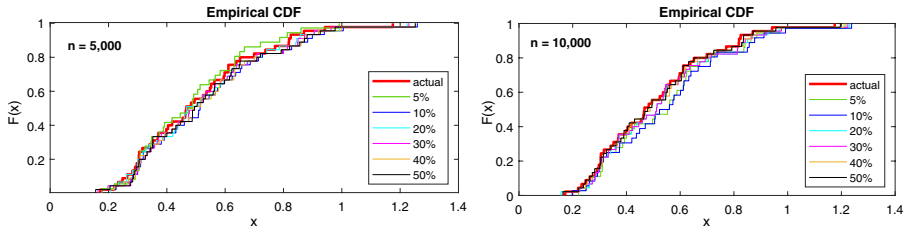
Here, *k* and *z* refer to the test statistics for KS-test and Z-test, respectively. $h = 0$ indicates that the test does not reject the null hypothesis at the 5% significance level, and *p*-values are probabilities of the positive results

It is noteworthy that we adopt a subset of random samples from the big dataset to approximate results as the estimation of the entire dataset. Thus, the proposed RSPCE approach does not require to analyze the entire dataset altogether.

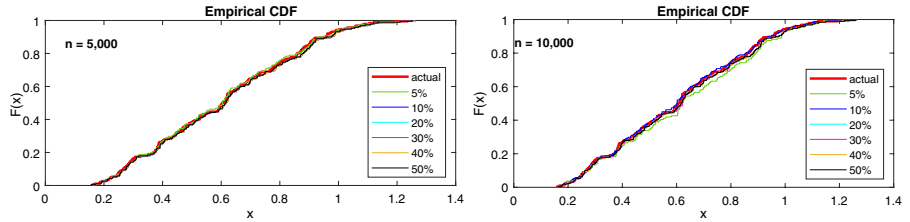
Conclusions

In this paper, we proposed a multiple random sample-based ensemble method to estimate the number of clusters in a large dataset. We partitioned a big dataset into a set of RSP data blocks as random samples of the big dataset. Then, we randomly select a subset of data blocks and identify the number of clusters independently. Finally, we ensemble the results of the multiple random samples as an estimate of the entire dataset. Moreover, a cluster ball model was introduced to ensemble the clusters of the random samples that are likely sampled from the same cluster in the big dataset.

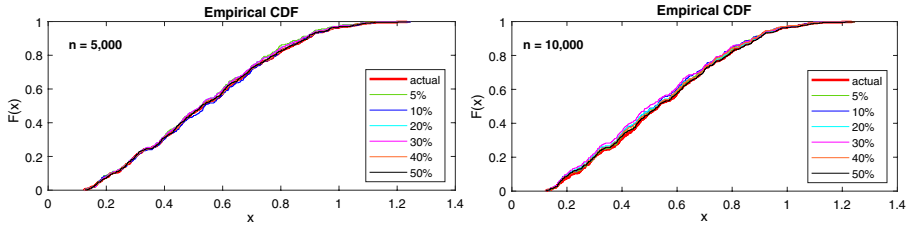
We conducted extensive experiments to investigate the effectiveness and stability of the RSPCE algorithm and further analyzed the impact of the sample size and the ensemble size. The experimental results demonstrated that the proposed algorithm was capable of generating good approximations of the actual cluster centers in the big dataset



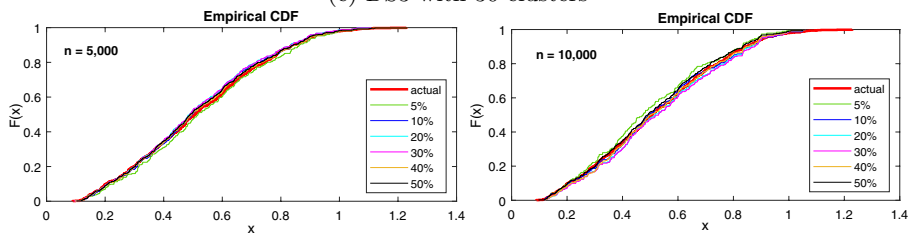
(a) DS1 with 10 clusters



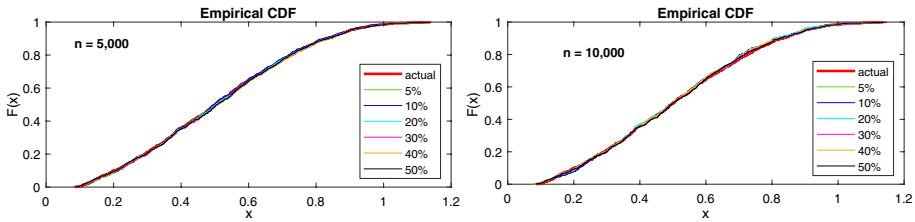
(b) DS2 with 20 clusters



(c) DS3 with 30 clusters



(d) DS4 with 40 clusters



(e) DS5 with 50 clusters

Fig. 6 The KS-test plots for comparison of actual centers' distance distribution versus the distance distribution of the centers by the RSPCE algorithm. The left-plot uses sample size $n = 5000$, and the right-plot uses $n = 10{,}000$

from a few random samples. The experiment results also demonstrated that the RSPCE algorithm is scalable to big data and flexible for clustering large-scale data on single machines or a cluster.

One should note that our cluster ball model is only suitable for merging clusters in spherical shapes. This is a limit of the RSPCE algorithm when it is applied to the dataset

Table 6 Comparison of the estimated number of clusters produced by different methods on the synthetic datasets

Dataset	*K*	*es*	nselectboot	kluster	X-means	Elbow	Silhouette	Gap statistic	RPSCE
DS1	10	5%	13.6 ± 1.6 (0.36)	**9.8** ± 0.4 (0.03)	18.8 ± 3.3 (0.83)	9.3 ± 1.2 (0.09)	9.6 ± 0.6 (0.05)	12.6 ± 1.2 (0.26)	9.7 ± 0.8 (0.07)
		10%	13.3 ± 1.3 (0.33)	9.8 ± 0.4 (0.02)	22.2 ± 5.9 (1.22)	9.8 ± 1.5 (0.12)	9.8 ± 0.4 (0.02)	12.3 ± 0.9 (0.23)	**10.0** ± 0.0 (0.00)
		20%	14.0 ± 1.4 (0.40)	**10.0** ± 0.0 (0.00)	20.2 ± 4.8 (1.02)	9.7 ± 1.6 (0.13)	9.8 ± 0.5 (0.03)	12.2 ± 1.1 (0.22)	**10.0** ± 0.0 (0.00)
		30%	13.8 ± 1.1 (0.38)	**10.0** ± 0.0 (0.00)	23.2 ± 2.5 (1.32)	8.5 ± 0.8 (0.15)	9.8 ± 0.4 (0.02)	11.7 ± 1.2 (0.17)	**10.0** ± 0.0 (0.00)
		40%	13.4 ± 1.5 (0.34)	**10.0** ± 0.0 (0.00)	22.8 ± 3.8 (1.28)	9.0 ± 1.3 (0.10)	NA	12.8 ± 0.5 (0.28)	**10.0** ± 0.0 (0.00)
		50%	14.4 ± 0.9 (0.44)	**10.0** ± 0.0 (0.00)	19.4 ± 1.9 (0.94)	10.2 ± 0.8 (0.05)	NA	12.5 ± 1.0 (0.25)	**10.0** ± 0.0 (0.00)
DS2	20	5%	28.7 ± 1.1 (0.47)	14.6 ± 0.5 (0.28)	43.0 ± 2.5 (1.15)	8.3 ± 1.3 (0.59)	**20.2** ± 0.7 (0.02)	21.0 ± 0.7 (0.06)	18.7 ± 1.3 (0.11)
		10%	29.3 ± 0.8 (0.47)	14.8 ± 0.4 (0.26)	43.8 ± 1.9 (1.20)	8.0 ± 1.2 (0.60)	**19.8** ± 0.4 (0.02)	20.6 ± 0.5 (0.04)	19.6 ± 0.7 (0.07)
		20%	28.7 ± 0.8 (0.44)	14.8 ± 0.4 (0.26)	42.8 ± 3.7 (1.14)	8.6 ± 1.8 (0.57)	**20.2** ± 0.4 (0.02)	21.0 ± 0.7 (0.07)	**19.9** ± 0.8 (0.02)
		30%	29.5 ± 0.5 (0.48)	15.0 ± 0.0 (0.25)	42.6 ± 3.9 (1.13)	9.0 ± 2.1 (0.55)	NA	20.8 ± 0.4 (0.06)	**20.0** ± 0.0 (0.01)
		40%	29.0 ± 1.5 (0.45)	15.0 ± 0.0 (0.25)	46.8 ± 1.6 (1.14)	8.5 ± 1.0 (0.58)	NA	21.0 ± 0.7 (0.07)	**20.0** ± 0.0 (0.00)
		50%	29.2 ± 0.1 (0.46)	15.0 ± 0.0 (0.25)	40.8 ± 6.6 (1.04)	9.2 ± 1.0 (0.54)	NA	20.8 ± 0.4 (0.06)	**20.0** ± 0.0 (0.00)
DS3	30	5%	39.6 ± 0.8 (0.32)	14.3 ± 0.8 (0.52)	40.1 ± 19.4 (0.68)	7.3 ± 1.2 (0.76)	21.3 ± 1.6 (0.41)	**31.7** ± 0.5 (0.06)	23.1 ± 1.3 (0.24)
		10%	39.3 ± 0.7 (0.31)	14.5 ± 0.5 (0.51)	40.2 ± 19.1 (0.66)	6.4 ± 1.3 (0.79)	**27.7** ± 0.9 (0.08)	32.8 ± 1.6 (0.09)	25.2 ± 1.2 (0.16)
		20%	38.8 ± 1.6 (0.29)	14.7 ± 0.5 (0.51)	40.4 ± 20.4 (0.70)	6.9 ± 1.3 (0.77)	25.5 ± 1.7 (0.18)	**31.2** ± 1.4 (0.05)	27.7 ± 1.7 (0.09)
		30%	39.6 ± 0.5 (0.32)	14.8 ± 0.4 (0.51)	39.0 ± 20.4 (0.70)	7.2 ± 0.4 (0.76)	NA	32.6 ± 1.7 (0.09)	**28.1** ± 1.8 (0.06)
		40%	39.4 ± 0.9 (0.31)	14.8 ± 0.5 (0.51)	40.4 ± 20.4 (0.70)	7.2 ± 1.3 (0.76)	NA	31.8 ± 0.8 (0.06)	**28.4** ± 0.9 (0.05)
		50%	38.0 ± 1.2 (0.27)	14.8 ± 0.4 (0.51)	48.2 ± 3.5 (0.61)	7.1 ± 1.4 (0.75)	NA	NA	**29.3** ± 1.8 (0.03)
DS4	40	5%	49.5 ± 0.7 (0.24)	13.3 ± 1.9 (0.66)	37.0 ± 24.2 (0.38)	8.1 ± 0.9 (0.80)	29.5 ± 4.1 (0.36)	**45.5** ± 3.5 (0.12)	25.4 ± 1.1 (0.37)
		10%	48.6 ± 1.6 (0.22)	13.8 ± 1.5 (0.65)	36.7 ± 27.5 (0.58)	8.4 ± 1.1 (0.79)	**42.3** ± 2.8 (0.05)	43.8 ± 2.4 (0.10)	27.2 ± 1.9 (0.30)
		20%	48.2 ± 1.6 (0.21)	14.0 ± 1.3 (0.65)	37.7 ± 25.7 (0.61)	8.5 ± 0.5 (0.79)	34.6 ± 2.7 (0.15)	**44.9** ± 1.8 (0.11)	31.6 ± 2.2 (0.22)
		30%	47.8 ± 1.6 (0.20)	14.5 ± 0.8 (0.63)	41.3 ± 28.9 (0.63)	9.5 ± 0.8 (0.76)	**36.7** ± 2.5 (0.09)	45.3 ± 1.6 (0.12)	34.2 ± 2.8 (0.11)
		40%	48.8 ± 1.1 (0.22)	14.3 ± 1.2 (0.64)	51.7 ± 21.1 (0.55)	8.8 ± 1.3 (0.78)	NA	45.3 ± 1.8 (0.12)	**36.6** ± 2.5 (0.09)
		50%	47.8 ± 1.5 (0.20)	14.7 ± 0.5 (0.63)	42.9 ± 26.7 (0.64)	8.5 ± 1.0 (0.79)	NA	NA	**38.1** ± 1.3 (0.05)

Table 6 (continued)

Dataset	*K*	*es*	nselectboot	kluster	X-means	Elbow	Silhouette	Gap statistic	RPSCE
DS5	50	5%	58.4 ± 2.2 (0.17)	14.3 ± 1.0 (0.71)	35.9 ± 33.6 (0.63)	7.3 ± 0.7 (0.86)	41.3 ± 3.6 (0.21)	**53.6** ± 1.4 (0.07)	37.2 ± 2.8 (0.26)
		10%	58.7 ± 1.5 (0.17)	14.8 ± 0.4 (0.70)	36.7 ± 30.2 (0.59)	7.5 ± 0.8 (0.85)	41.7 ± 4.5 (0.20)	**52.8** ± 2.5 (0.05)	41.4 ± 1.9 (0.21)
		20%	58.6 ± 2.1 (0.18)	14.5 ± 0.8 (0.71)	30.4 ± 36.1 (0.70)	7.5 ± 1.2 (0.85)	**52.1** ± 1.8 (0.04)	54.4 ± 1.3 (0.08)	43.8 ± 0.8 (0.14)
		30%	58.2 ± 2.2 (0.16)	14.3 ± 1.0 (0.71)	37.2 ± 36.0 (0.69)	8.8 ± 0.9 (0.83)	**52.5** ± 4.2 (0.04)	54.1 ± 0.9 (0.05)	45.6 ± 1.5 (0.10)
		40%	59.2 ± 0.8 (0.18)	14.7 ± 0.5 (0.71)	32.3 ± 35.3 (0.71)	8.8 ± 0.8 (0.82)	NA	NA	**45.9** ± 0.8 (0.09)
		50%	57.4 ± 1.8 (0.15)	14.8 ± 0.4 (0.70)	20.6 ± 30.5 (0.75)	7.3 ± 1.4 (0.85)	NA	NA	**46.5** ± 1.1 (0.07)

The average value of 20 runs is displayed together with " ± " standard deviation. The best and second best results are shown in bold and underlined, respectively. The values in parenthesis indicate the mean relative error

Sample size for RSPCE is $n = 5000$; K is the true number of clusters in the dataset; and *es* is the ensemble size. The percentage was selected randomly

NA indicates not available, i.e., the value cannot be computed

Table 7 Comparison of the estimated number of clusters produced by different methods on the real-world datasets

Dataset	*K*	*es*	nselectboot	kluster	X-means	Elbow	Silhouette	Gap statistic	RSPCE
Covertype	7	5%	2.8 ± 1.2	11.3 ± 0.5	2.1 ± 0.3	7.4 ± 1.2	2.1 ± 0.3	2.2 ± 0.3	4.6 ± 0.9
		10%	4.0 ± 1.9	10.3 ± 1.0	2.3 ± 0.7	7.3 ± 1.3	2.3 ± 1.5	2.3 ± 0.3	6.0 ± 0.7
		20%	6.6 ± 3.5	10.7 ± 0.5	2.2 ± 0.4	7.7 ± 0.8	2.2 ± 0.2	2.2 ± 0.2	7.1 ± 0.5
		30%	8.4 ± 5.5	10.7 ± 1.0	2.2 ± 0.4	8.0 ± 0.7	2.4 ± 0.4	2.2 ± 0.1	7.8 ± 0.8
		40%	9.4 ± 5.5	11.3 ± 1.0	2.2 ± 0.4	8.4 ± 1.4	NA	2.1 ± 0.1	8.2 ± 0.8
		50%	9.8 ± 5.9	11.0 ± 0.6	2.1 ± 0.3	7.3 ± 0.9	NA	2.1 ± 0.1	8.2 ± 0.6
KDD'99ID	23	1%	6.8 ± 6.7	10.6 ± 0.9	53.6 ± 5.9	6.8 ± 0.9	4.3 ± 0.5	48.7 ± 0.6	2.9 ± 0.4
		2%	5.8 ± 6.3	10.4 ± 0.5	51.2 ± 5.2	4.8 ± 0.7	3.6 ± 0.4	46.5 ± 1.4	3.0 ± 0.0
		5%	6.8 ± 7.5	10.4 ± 0.5	53.0 ± 4.5	4.8 ± 0.6	3.6 ± 0.5	48.7 ± 0.6	3.0 ± 0.0
		10%	7.3 ± 8.5	10.2 ± 0.4	53.0 ± 4.5	5.2 ± 0.8	3.4 ± 0.5	46.5 ± 1.7	3.0 ± 0.0
		15%	8.0 ± 8.7	10.5 ± 0.6	53.5 ± 4.8	4.2 ± 0.8	NA	NA	3.0 ± 0.0
		20%	7.7 ± 8.1	10.4 ± 0.5	53.0 ± 4.5	4.4 ± 0.5	NA	NA	3.0 ± 0.0
PokerHand	10	1%	3.2 ± 0.6	2.9 ± 0.3	17.8 ± 2.4	6.7 ± 1.7	4.5 ± 0.9	11.4 ± 1.3	3.5 ± 0.5
		2%	3.4 ± 0.7	3.6 ± 0.4	18.3 ± 2.5	7.2 ± 1.9	4.7 ± 0.5	12.2 ± 0.9	3.7 ± 0.6
		5%	2.7 ± 0.5	3.5 ± 0.6	17.9 ± 1.9	7.3 ± 1.2	3.9 ± 0.8	11.4 ± 2.3	3.9 ± 0.8
		10%	2.3 ± 0.6	3.6 ± 0.5	19.4 ± 1.5	6.9 ± 1.8	4.8 ± 0.8	11.7 ± 1.2	3.9 ± 0.4
		15%	2.4 ± 0.6	3.6 ± 0.4	18.1 ± 1.7	6.6 ± 1.3	NA	NA	4.0 ± 0.0
		20%	2.1 ± 0.8	3.8 ± 0.3	17.4 ± 1.4	6.8 ± 1.5	NA	NA	4.0 ± 0.0
SUSY	2	1%	2.9 ± 0.4	8.3 ± 0.7	13.4 ± 1.3	4.3 ± 0.8	7.7 ± 0.8	4.5 ± 0.9	2.6 ± 0.6
		2%	3.2 ± 0.6	8.4 ± 0.5	11.9 ± 2.5	4.6 ± 1.2	6.5 ± 2.1	5.0 ± 1.6	3.2 ± 0.4
		5%	2.6 ± 0.7	9.2 ± 1.1	13.3 ± 2.1	5.1 ± 0.7	8.1 ± 1.5	5.2 ± 0.8	3.4 ± 0.5
		10%	2.5 ± 0.5	8.6 ± 0.5	15.1 ± 0.9	6.7 ± 0.9	9.3 ± 1.2	6.1 ± 1.2	3.3 ± 0.5
		15%	2.9 ± 1.1	9.2 ± 0.8	16.9 ± 1.2	7.7 ± 1.3	NA	NA	3.7 ± 0.4
		20%	2.8 ± 0.7	9.4 ± 0.6	15.4 ± 1.8	6.5 ± 0.8	NA	NA	3.8 ± 0.4

The average value of 20 runs is displayed together with "±" standard deviation

Sample size for RSPCE is $n = 5000$; K is the true number of classes in the dataset; and *es* is the ensemble size. The percentage was selected randomly

NA indicates not available, i.e., the value cannot be computed

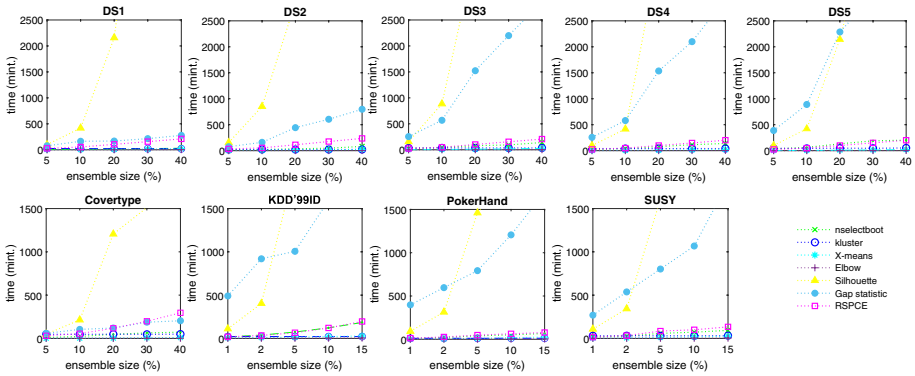


Fig. 7 Execution time (in minutes) of different methods against different data sizes. The plot is the average value of 20 random runs with the same data size

with clusters of irregular shapes. In future work, we will address this issue by adopting the graph ensemble for high-dimensional and complex non-linear manifold structure datasets, such as moon-shaped and Swiss-roll data. Besides, we will investigate a statistical framework to design an ensemble for distributed clustering that exercises both the weight information and the efficiency of multiple random samples.

Acknowledgements
Not applicable.

Author contributions
MS Mahmud: Conceptualization, Investigation, Software, Methodology, Validation, Writing - original draft; JZ Huang: Supervision, Methodology, Funding acquisition, Writing - review & editing; RR: Project administration, Validation, Writing - review & editing; KW: Funding acquisition, Resources. All authors read and approved the final manuscript.

Funding
This research has been supported by the National Natural Science Foundation of China Grant-61972261.

Availability of data and materials
The code and experimental data are available at https://github.com/sultanszu/RSPCE.git.

Declaration

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Compeing interests
The authors declare that they have no competing interests.

Received: 15 July 2022 Accepted: 28 February 2023
Published online: 01 April 2023

References

1. Thorndike RL. Who belongs in the family. Psychometrika. 1953. https://doi.org/10.1007/BF02289263.
2. Rousseeuw PJ. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math. 1987;20:53–65. https://doi.org/10.1016/0377-0427(87)90125-7.
3. Tibshirani R, Walther G, Hastie T. Estimating the number of clusters in a data set via the gap statistic. J R Stat Soc Series B. 2001;63(2):411–23. https://doi.org/10.1111/1467-9868.00293.
4. Masud MA, Huang JZ, Wei C, Wang J, Khan I, Zhong M. I-nice: a new approach for identifying the number of clusters and initial cluster centres. Inf Sci. 2018;466:129–51. https://doi.org/10.1016/j.ins.2018.07.034.
5. Nair R. Big data needs approximate computing: technical perspective. Commun ACM. 2014;58(1):104–104. https://doi.org/10.1145/2688072.
6. Meng X-L. Statistical paradises and paradoxes in big data (i): law of large populations, big data paradox, and the 2016 US presidential election. Ann Appl Stat. 2018;12(2):685–726. https://doi.org/10.1214/18-AOAS1161SF.

7. Rojas, J.A.R., Beth Kery, M., Rosenthal, S., Dey, A.: Sampling techniques to improve big data exploration. In: 2017 IEEE 7th Symp. Large Data Analy Vis. 2017. 10.1109/LDAV.2017.8231848
8. Salloum S, Huang JZ, He Y. Random sample partition: a distributed data model for big data analysis. IEEE Trans Ind Informat. 2019;15(11):5846–54. https://doi.org/10.1109/TII.2019.2912723.
9. Mahmud MS, Huang JZ, Salloum S, Emara TZ, Sadatdiynov K. A survey of data partitioning and sampling methods to support big data analysis. Big Data Mining Anal. 2020;3(2):85–101.
10. He Y, Wu Y, Qin H, Huang JZ, Jin Y. Improved i-nice clustering algorithm based on density peaks mechanism. Inf Sci. 2021;548:177–90. https://doi.org/10.1016/j.ins.2020.09.068.
11. Xu X, Ding S, Wang Y, Wang L, Jia W. A fast density peaks clustering algorithm with sparse search. Inform Sci. 2021;554:61–83. https://doi.org/10.1016/j.ins.2020.11.050.
12. Rodriguez A, Laio A. Clustering by fast search and find of density peaks. Science. 2014;344(6191):1492–6. https://doi.org/10.1126/science.1242072.
13. Schubert E, Sander J, Ester M, Kriegel HP, Xu X. Dbscan revisited, revisited: why and how you should (still) use dbscan. ACM Trans Database Syst. 2017. https://doi.org/10.1145/3068335.
14. Patil C, Baidari I. Estimating the optimal number of clusters k in a dataset using data depth. Data Sci Eng. 2019;4:132–40.
15. Zhao X, Liang J, Dang C. A stratified sampling based clustering algorithm for large-scale data. Know Based Syst. 2019;163:416–28. https://doi.org/10.1016/j.knosys.2018.09.007.
16. Jia J, Xiao X, Liu B, Jiao L. Bagging-based spectral clustering ensemble selection. Pattern Recognit Lett. 2011;32(10):1456–67. https://doi.org/10.1016/j.patrec.2011.04.008.
17. Wang Y, Chen L, Mei J. Incremental fuzzy clustering with multiple medoids for large data. IEEE Trans Fuzzy Syst. 2014;22(6):1557–68. https://doi.org/10.1109/TFUZZ.2014.2298244.
18. Hu J, Li T, Luo C, Fujita H, Yang Y. Incremental fuzzy cluster ensemble learning based on rough set theory. Know Based Syst. 2017;132:144–55. https://doi.org/10.1016/j.knosys.2017.06.020.
19. Bagirov AM, Ugon J, Webb D. Fast modified global k-means algorithm for incremental cluster construction. Pattern Recognit. 2011;44(4):866–76. https://doi.org/10.1016/j.patcog.2010.10.018.
20. Mimaroglu S, Erdil E. Combining multiple clusterings using similarity graph. Pattern Recogn. 2011. https://doi.org/10.1016/j.patcog.2010.09.008.
21. Huang D, Lai J, Wang CD. Ensemble clustering using factor graph. Pattern Recognit. 2016;50(C):131–42. https://doi.org/10.1016/j.patcog.2015.08.015.
22. Ayad HG, Kamel MS. On voting-based consensus of cluster ensembles. Pattern Recognit. 2010;43(5):1943–53. https://doi.org/10.1016/j.patcog.2009.11.012.
23. Iam-On N, Boongoen T, Garrett S, Price C. A link-based approach to the cluster ensemble problem. IEEE Trans Pattern Anal Mach Intell. 2011;33(12):2396–409. https://doi.org/10.1109/TPAMI.2011.84.
24. Yang J, Liang J, Wang K, Rosin PL, Yang M. Subspace clustering via good neighbors. IEEE Trans Pattern Anal. 2020;42(6):1537–44. https://doi.org/10.1109/TPAMI.2019.2913863.
25. Fang Y, Wang J. Selection of the number of clusters via the bootstrap method. Comput Stat Data Anal. 2012;56(3):468–77. https://doi.org/10.1016/j.csda.2011.09.003.
26. Estiri H, Abounia Omran B, Murphy SN. kluster: an efficient scalable procedure for approximating the number of clusters in unsupervised learning. Big Data Res. 2018;13:38–51. https://doi.org/10.1016/j.bdr.2018.05.003.
27. Pelleg, D., Moore, A.W.: X-means: Extending k-means with efficient estimation of the number of clusters. In: Proc. 17th Int. Conf. Mach. Learn. ICML '00, pp. 727–734. Morgan Kaufmann Publishers Inc., CA, USA 2000.
28. Bachem, O., Lucic, M., Krause, A.: Scalable k-means clustering via lightweight coresets. In: Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD'18), NY, USA, pp. 1119–1127 (2018). 10.1145/3219819.3219973.
29. Wu J, Liu H, Xiong H, Cao J, Chen J. K-means-based consensus clustering: a unified view. IEEE Trans Knowl Data Eng. 2015;27(1):155–69. https://doi.org/10.1109/TKDE.2014.2316512.
30. Iam-On N, Boongeon T, Garrett S, Price C. A link-based cluster ensemble approach for categorical data clustering. IEEE Trans Knowl Data Eng. 2012;24(3):413–25.
31. Ren Y, Domeniconi C, Zhang G, Yu G. Weighted-object ensemble clustering: Methods and analysis. Knowl Inf Syst. 2017;51(2):661–89. https://doi.org/10.1007/s10115-016-0988-y.
32. Brock G, Pihur V, Datta S, Datta S. clvalid: an r package for cluster validation. J Stat Softw. 2008;25(4):1–22. https://doi.org/10.18637/jss.v025.i04.
33. Davies DL, Bouldin DW. A cluster separation measure. IEEE Trans Pattern Anal Mach Intell. 1979;1(2):224–7. https://doi.org/10.1109/TPAMI.1979.4766909.
34. Rosenberg, A., Hirschberg, J.: V-measure: A conditional entropy-based external cluster evaluation measure. In: Proc. 2007 Joint Conf. Empir. Methods Nat. Lang. Process. Comput. Nat. Lang. Learn. (EMNLP-CoNLL), pp. 410–420. Association for Computational Linguistics, Prague, Czech Republic 2007. 10.1109/10.7916/D80V8N84.
35. Lawrence H, Phipps A. Comparing partitions. J Classif. 1985;2(1):193–218. https://doi.org/10.1007/BF01908075.
36. Vinh NX, Epps J, Bailey J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. J Mach Learn Res. 2010;11:2837–54. https://doi.org/10.5555/1756006.1953024.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.