Journal of Systems Engineering and Electronics
Vol. 33, No. 3, June 2022, pp.647 – 664

Adaptive spectral affinity propagation clustering

TANG Lin[1], SUN Leilei[1,2], GUO Chonghui[1,*], and ZHANG Zhen[1]

1. Institute of Systems Engineering, Dalian University of Technology, Dalian 116024, China; 2. School of Computer Science and Engineering, Beihang University, Beijing 100191, China

Abstract: Affinity propagation (AP) is a classic clustering algorithm. To improve the classical AP algorithms, we propose a clustering algorithm namely, adaptive spectral affinity propagation (AdaSAP). In particular, we discuss why AP is not suitable for non-spherical clusters and present a unifying view of nine famous arbitrary-shaped clustering algorithms. We propose a strategy of extending AP in non-spherical clustering by constructing category similarity of objects. Leveraging the monotonicity that the clusters' number increases with the self-similarity in AP, we propose a model selection procedure that can determine the number of clusters adaptively. For the parameters introduced by extending AP in non-spherical clustering, we provide a grid-evolving strategy to optimize them automatically. The effectiveness of AdaSAP is evaluated by experiments on both synthetic datasets and real-world clustering tasks. Experimental results validate that the superiority of AdaSAP over benchmark algorithms like the classical AP and spectral clustering algorithms.

Keywords: affinity propagation (AP), Laplacian eigenmap (LE), arbitrary-shaped cluster, model selection.

DOI: 10.23919/JSEE.2022.000060

1. Introduction

Much data has been collected with the rapid development of sensing and storage technology in the past years. How to extract knowledge from these data is an essential topic. To understand these data, we usually organize them into meaningful groups. Clustering is a technique that makes objects inside a cluster more similar than those between clusters [1]. As an efficient unsupervised learning method, clustering has been used in numerous fields, e.g., image segmentation, face recognition, video summary, text mining, web analysis, genome analysis, social community detection, and marketing segmentation [2,3].

Different kinds of clustering algorithms [4,5] have been proposed. These algorithms can be divided into two categories: spherical clustering and non-spherical clustering algorithms, according to the shape of the distribution of data within a cluster. Spherical clustering means that these data are ball shaped in space in a cluster. Non-spherical clustering means the shape of data within the cluster is uncertain. If a clustering algorithm can handle both spherical and non-spherical distribution data, it is called an arbitrary-shape clustering algorithm. Affinity propagation (AP) clustering is an extremely successful one [6] as a spherical clustering algorithm. AP is also suitable for both structured and unstructured clustering data. It takes a collection of pairwise similarities as input. Unlike traditional exemplar-based algorithms, for example, the *k*-centers algorithm [7], the AP algorithm treats all objects as potential exemplars and usually results in a better clustering result [6,8]. Considering the problems caused by the parameter preference in the AP clustering algorithm. Li et al. [9] proposed the adjustable preference AP (APAP) algorithm, which can automatically adjust the value of each element preference during the iteration process. For solving the non-spherical cluster problem, Fan et al. [10] used the density distributions and structures of data to define an adaptive AP clustering algorithm, which adopts a nearest neighbor searching strategy. Now AP clustering has been widely applied in text mining [11,12], gene expression analysis [13,14], social network analysis [15,16], etc. Meanwhile, many improved AP algorithms have been developed, such as the semi-supervised AP [17,18], the hierarchical AP [19], the incremental AP [20], and the fast AP [21,22].

Though AP has achieved remarkable success, it also suffers from some bottleneck problems in practical applications:

(i) AP cannot be used to discover clusters with non-spherical shapes, which limits the applications of AP in many real-world clustering tasks, such as image segmentation, location-based search, graph partition, and community detection. The shapes of underlying clusters in these tasks are often much more complicated than what

Manuscript received November 30, 2020.
*Corresponding author.
This work was supported by the National Natural Science Foundation of China (71771034; 71901011; 71971039) and the Scientific and Technological Innovation Foundation of Dalian (2018J11CY009).

AP can deal with.

(ii) It is difficult for users to control the clusters' number. A parameter called preference p needs to be specified by users, which controls the clusters' number. The only prior knowledge of preference p is that the clusters' number increases the value of p. However, users usually need to tune p many times to get the desired number of clusters. On the other hand, p cannot be determined automatically. In our opinion, p should be able to vary its value automatically and stop at a suitable value. The clustering result can be better.

(iii) Our extension of AP from the above two perspectives may introduce other parameters. The proposed AP clustering method in this paper should be able to find the appropriate parameter values adaptive.

This paper proposes a clustering algorithm named adaptive spectral AP (AdaSAP) to improve the classical AP clustering algorithms from the following aspects:

(i) We study how to extend AP clustering in the non-spherical clustering problem.

(ii) Leverage the monotonicity of preference p, which is able to determine the number of clusters automatically. The clusters' number (or the proper value of p) found by our procedure can well reflect the structure of the studied dataset.

(iii) An evolutionary process is proposed, which can tune the new parameters AdaSAP adaptively.

This paper also has two main theoretical contributions. First, it provides an overview of existing famous arbitrary-shaped clustering algorithms and points out the essential of arbitrary-shaped clustering it; second, it provides a general strategy to determine the clusters' number, which is a long-standing problem in the clustering field. From the practical perspective, this paper improves the state-of-the-art AP clustering algorithms, and many practical clustering tasks can benefit from the proposed AdaSAP algorithm.

The rest of the paper is organized as follows: Section 2 briefly describes the classical AP clustering algorithm and Laplacian eigenmaps (LE), and provides a unifying view of the existing famous arbitrary-shaped clustering algorithms. In Section 3, we propose the AdaSAP algorithm. In Section 4, we discuss how to determine the number of clusters automatically in AdaSAP. In Section 5, experimental results are presented. In Section 6, we conclude our work and future work.

2. Related work

This section presents foundations and related work for further discussions. We also present a unifying view of the existing arbitrary-shaped clustering algorithms.

2.1 AP clustering

AP is an exemplar-based clustering algorithm. The algorithm's objective is to maximize the sum of all the similarities of the cluster's inner objects and the cluster's exemplar. The exemplar set is the microcosm of the entire dataset [23]. How to find good exemplars is a formidable combination's optimization problem.

The AP algorithm considers all data points as potential exemplars initially. It treats each data point as a network node. Two kinds of messages, responsibilities and availabilities, pass on each edge of the graph.

There are $N=\{x_1,x_2,\cdots,x_n\}$ sample points clustering, and point x_i and point x_j are two of the sample points. Responsibility $r(i,j)$ indicates how strong point x_i wants to choose candidate exemplar x_j as its exemplar. According to [6], $r(i,j)$ is computed as

$$r(i,j)\leftarrow s(i,j)-\max_{j',j'\neq j}\{\alpha(i,j')+s(i,j')\},\quad i,j,j'\in\{1,2,\cdots,n\}. \tag{1}$$

Availability $\alpha(i,j)$ means how well-suited it is for object x_i to choose object x_j as its exemplar. It is computed as

$$\alpha(i,j)\leftarrow \min\{0,r(j,j)+\sum_{i',i'\notin\{i,j\}}\max\{0,r(i',j)\}\},\quad i,j,j'\in\{1,2,\cdots,n\}. \tag{2}$$

According to (1) and (2), the responsibilitiy and availability values are continuously updated through an iterative process until convergence. The clustering result $\hat{e}=(\hat{e}_1,\hat{e}_2,\cdots,\hat{e}_n)$ is

$$\hat{e}_i=\arg\max_j\{\alpha(i,j)+r(i,j)\},\quad i,j,\in\{1,2,\cdots,n\}. \tag{3}$$

2.2 LEs

The generic problem of manifold learning is as follows: For a set $\{x_1,x_2,\cdots,x_n\}$ of n points in $\boldsymbol{X}^l$ in dimension l, the goal of manifold learning is to find the intrinsic dimension m of the manifold M. Then a set of points $\{y_1,y_2,\cdots,y_n\}$ in $\boldsymbol{X}^m$ ($m\ll l$) can represent them well in term of the local relationships.

In order to preserve local relationships between points, LEs first construct an adjacency graph with weights $\boldsymbol{G}=(\boldsymbol{V},\boldsymbol{E},\boldsymbol{W})$. $\boldsymbol{G}$ represents the completely connected graph. $\boldsymbol{V}$ corresponds to objects X is the set of points. $\boldsymbol{E}$ is the set of edges, the element e_{ij} is the edge of objects i and j. And $\boldsymbol{W}$ is the set of weights on the edges, the weight of the edge e_{ij} is w_{ij}, which is equivalent to (or increases with) s_{ij}. And then map the graph $\boldsymbol{G}$ into a low-

dimensional manifold [24]. Many methods have been proposed to construct graph $\boldsymbol{G}$. They guarantee that the closer the two points are, the greater their edge weight is.

When mapping $\boldsymbol{G}$ to a low-dimension space, we need to make the connection points as close as possible. $\boldsymbol{Y}=[y_1,y_2,\cdots,y_n]^{\mathrm{T}}$ is the low-dimensional graph representation. In this process, we need to minimize the objective function with appropriate constraints.

$$\min \sum_{i,j}(\boldsymbol{y}_i-\boldsymbol{y}_j)^2 w_{ij}.$$

By derivation and transformation [24], we need to minimize

$$\sum_{i,j}\left\|\boldsymbol{y}_i-\boldsymbol{y}_j\right\|^2 w_{ij}=\mathrm{tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{Y})$$

where $\boldsymbol{y}_i=[y_i^1,y_i^2,\cdots,y_i^m]$ is the m-dimensional representation of the ith vertex. $\boldsymbol{L}=\boldsymbol{D}-\boldsymbol{W}$ is defined as the Laplacian matrix. Furthermore, it is reduced to find

$$\begin{aligned}&\arg\min\ \mathrm{tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{Y})\\&\text{s.t. }\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{Y}=\boldsymbol{I}.\end{aligned}$$

2.3 A unifying view of the existing arbitrary-shaped clustering algorithms

Particularly for unstructured data, Belkin et al. thought those arbitrary-shaped clusters cannot be separated [24]. They discussed the intrinsic consistency of spectral embedding methods and kernel principal component analysis in 2004 [25]. Their work bridges three main fields, which are spectral clustering, kernel methods, and nonlinear dimension reduction together. Fig. 1 presents the relationships between the five studied algorithms.

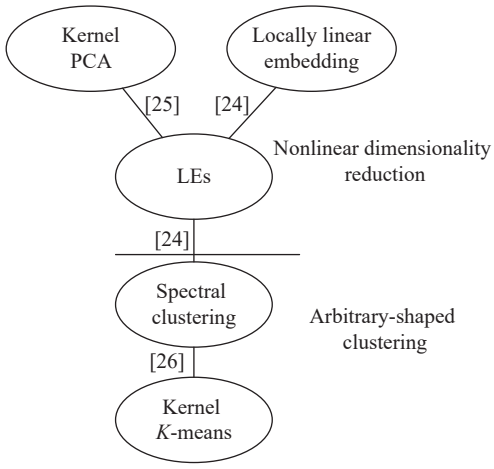


Fig. 1 Relationships among the five studied algorithms

Another important category of arbitrary-shaped clustering algorithm is a density-based one. We discuss the mathematical similarity of three typical density-based clustering algorithms as follows:

First, densinty-based spatial clustering of applications with noise (DBSCAN) [27] describes the density based on the data point neighborhood. To find arbitrary-shaped clusters, DBSCAN uses the iterative method to cluster. Mean-shift was proposed in 1975 [28] for non-parametric density gradient estimation. Then Cheng applied mean-shift on clustering [29] in 1995. Comaniciu et al. [30] applied mean-shift as a clustering method, and the work makes the method well-known.

Density peaks clustering (DPC) is a recently proposed density-based clustering algorithm published in Science in 2014 [31]. At first, it finds some particular objects called density peaks. It then assigns each remaining object to a density peak along a density gradient descent direction. A density peak in DPC is the object with the highest density in a relatively large scope. Such an object is called a density peak. Two indicators are the local density ϱ and the minimum distance δ.

We know that ϱ in DPC equals $|N_\epsilon(x_i)|$ in DBSCAN. In addition, when we use the kernel function to estimate ϱ of an object, the indicator ϱ is in fact the density $f(x)$ in mean-shift [32]. That is, the three definitions can be alternatively used in the three density-based arbitrary-clustering algorithms. Based on the estimated density of an object, DBSCAN defines a relationship of two objects as density-connected, which can be explained from our unifying view that, if two objects as density-connected, then category similarity $s'(i,j)=1$; otherwise, $s'(i,j)=0$. In mean-shift and DPC, an object i is assigned to its nearest neighbor with even higher density. If two points have the same ending point (or local density maximum point), $s'(i,j)=1$; otherwise, $s'(i,j)=0$. Therefore, the main difference between DBSCAN with mean-shift and DPC lies in the label propagation manner. From the perspective of density estimation and category similarity construction, they are essentially very similar. Table 1 compares the three density-based arbitrary clustering algorithms, where ε is a similarity threshold (or cutoff distance), $\eta(\cdot)$ can be an activation function or a kernel function.

Table 1 Comparisons of three density-based arbitrary clustering algorithms

Method	Density estimator	$s'(i,j)=1$
DBSCAN	$\sum_j \eta(s_{ij}-\varepsilon)$	Density-connected
Mean-shift	$\sum_j \eta(s_{ij}-\varepsilon)$	Same ending point
DPC	$\sum_j \eta(s_{ij}-\varepsilon)$	Same density peak

The last arbitrary-shaped clustering algorithm studied in this section is the Chameleon algorithm, which is an agglomerative hierarchical clustering algorithm using dynamic modeling [33]. It consists of three phases: (i) using K-nearest neighbors to generate a sparse adjacency graph $\boldsymbol{G}'$ according to the similarity matrix $\boldsymbol{S}$; (ii) dividing the sparse adjacencygraph into many sub-graphs by the graph partitioning algorithm, a sub-graph can be seen as an initial sub-cluster in agglomerative clustering; (iii) recombining sub-clusters agglomerative to generate the deprogram. Chameleon is a heuristic clustering algorithm, so it is difficult to reason the mathematical equivalence of Chameleon with other arbitrary-shaped clustering algorithms. Even though the mechanism behind Chameleon is still strictly with our unifying view, assume that $\boldsymbol{C}_i$, $\boldsymbol{C}_j$, and $\boldsymbol{C}_k$ are three sub-graphs at the leaf level. In the procedure of agglomeration, $\boldsymbol{C}_i$ is first combined with $\boldsymbol{C}_j$, and then the three sub-clusters are merged together. From the viewpoint of the category similarity, the dynamic modeling process in Chameleon is, in fact, a procedure of category similarity construction, while the category similarity of two objects is also determined by the local (or key) feature similarity.

Fig. 2 summarizes all the above discussions. It can be known that kernel principle component analysis (PCA), LE, locally linear embedding (LLE), kernel K-means [34], and spectral clustering [35] are equivalent from the mathematics perspective, which were presented by [24−26]. DBSCAN, mean-shift, and DPC are consistent both in terms of density estimation and label propagation. The only difference lies in the label propagation manner. Such consistency has not been discussed ever before. This presents the three on the same ground. From an even higher level, though some algorithms are proposed as heuristic clustering algorithms, it is difficult to get the consistency of these algorithms with others. We can still find two key principles: (i) in whichever algorithm, a category similarity matrix $\boldsymbol{S}^c$ is constructed either explicitly or implicitly; (ii) larger feature similarities s_{ij} play a significant role during the construction of $\boldsymbol{S}^c$, which means that the distant similarity information is not necessary for arbitrary-shaped clustering.

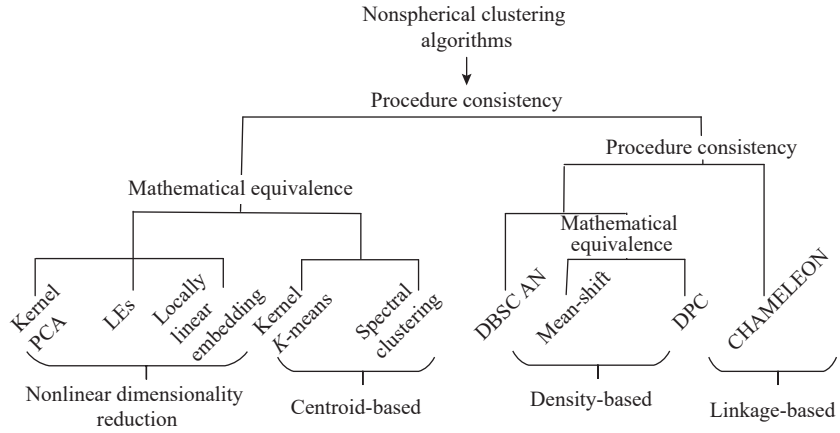


Fig. 2 Hierarchy of nine studied algorithms

Based on the unifying view of arbitrary-shaped clustering algorithms, a general framework of arbitrary-shaped clustering is proposed in Fig. 3. All the studied nine arbitrary-shaped clustering algorithms can be well explained by the framework. Spectral clustering for unstructured data starts with feature similarity matrix $\boldsymbol{S}$ which represents the similarity matrix which also be called affinity matrix, where s_{ij} is the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. A popular similarity measure is computed by negative distances of objects as follows: $s_{ij} = (\max(\boldsymbol{D}) - d_{ij})/(\max(\boldsymbol{D}) - \min(\boldsymbol{D}))$. kernel K-means mapping collected data $\boldsymbol{X}$ to $\boldsymbol{X}'$ by kernels if we select proper kernels, the mapping in kernel K-means is equivalent to that in the spectral method. Density-based clustering algorithms first compute the density of each object by its neighborhood, which is essentially an operation on sparse graph $\boldsymbol{G}'$, as $\boldsymbol{G}'$ reflects the relationship of neighbor objects. We usually obtain $\boldsymbol{G}'$ from the complete graph $\boldsymbol{G}$, usually using the methods include the K-nearest neighbor (KNN) or similarity greater than ε (ε-NN) as a condition, where ε is the minimum value of similarity. Then the three density-based arbitrary-shaped clustering algorithms implement label propagation, where two objects having larger category similarity shares the same cluster label. Chameleon constructs $\boldsymbol{G}'$ according to feature similarity $\boldsymbol{S}$ and constructs category similarity matrix $\boldsymbol{S}^c$ using a dynamic process. Recently, Wang et al. [36] proposed a spectral cluster algorithm based on message passing (MPSC) with a density sensitive similarity measurement. After that Wang et al. [37] proposed an improved density-based adaptive p-spectral clustering algorithm. To the best of our knowledge, this is the first effort that

adopts label propagation by spectral clustering to extend AP in arbitrary-shaped clustering.

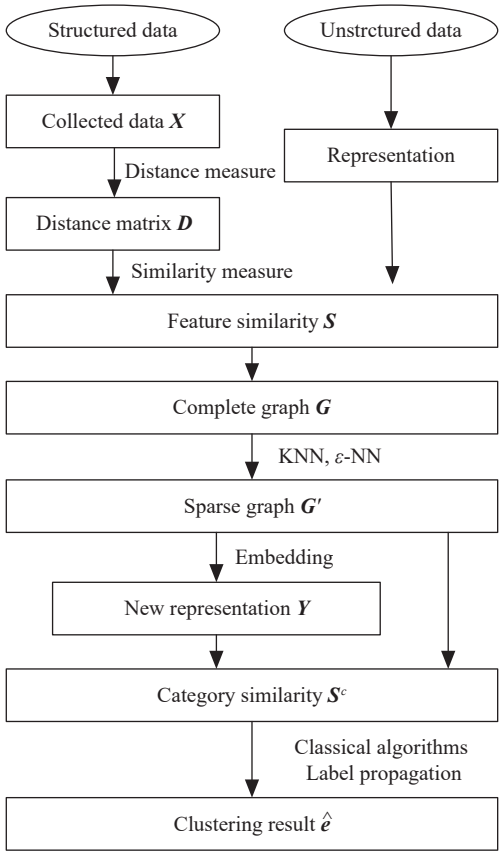


Fig. 3 A general framework of arbitrary-shaped clustering algorithms

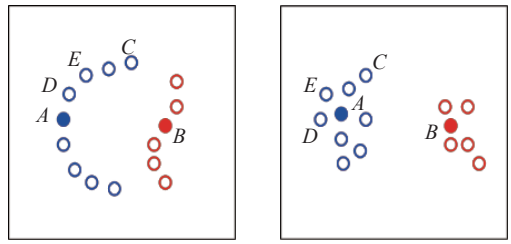
3. Methodology

As a centroid-based clustering algorithm, the classical AP clustering algorithm can only find spherical clusters. This section discusses how to extend AP in the discovery of clusters with complex shapes. Firstly, a toy example illustrates this paper's core idea; then we point out the essentials of arbitrary-shaped clustering.

After that, we propose an algorithm named arbitrary-shaped AP, which combines the spherical clustering with the AP clustering algorithm based on message passing of the category label. Furthermore, we also solve the problem of how to set the parameters in AP clustering. First, we introduce a method for determining the parameter p of AP to obtain the better clusters' number. Second, we use a parameter grid to optimize the parameters both k (the number of nearest neighbors) and d (the dimensionality in the lower dimensions) while constructing the sparse graph $\boldsymbol{G}'$.

3.1 Motivation

Fig. 4 is a toy example, which illustrates the motivation of this paper. In Fig. 4(a), the 15 points in a two-dimensional space can be divided into two clusters. Points A and B are exemplars of the two clusters. Point C should be assigned to exemplar B since point C is closer to point B than to point A, denoted by $s(B,C) > s(A,C)$ if similarities are computed by spatial distance. However, such an assignment does not take the propagation of the cluster label into account. Intuitively, we often infer that if (i) points A and D are with the same cluster label and (ii) points D and E also belong to the same cluster, then points A and E should have the same cluster label. That is, whether points A and E should be divided into the same cluster is determined by $\min(s(A,D), s(D,E))$ rather than $s(A,E)$. According to such inference, point C should be assigned to exemplar A finally, which is accordant with our intuition.



(a) Dataset A (b) Dataset B

Fig. 4 Motivation of nearby propagation

In most of the existing literature, the clustering algorithm is directly implemented on a computed or provided similarity matrix, where the similarities used mainly reflect the distances of objects in the original feature space. However, what is desired in clustering is how properly two objects should be divided into the same cluster. This measure is called category similarity in this paper. For the convenience of distinguishing, the original similarity is called feature similarity. Take Fig. 1(b) for example, the feature similarity is $s(A,C) < s(B,C)$, while the category similarity is $s^c(A,C) > s^c(B,C)$, if the propagation of the cluster label is taken into account. It can also be observed that local feature similarity plays an important part in computing category similarity. For example, $s^c(A,C)$ is mainly determined by $\min(s(A,D), s(D,E))$ rather than $s(A,E)$. Therefore, the key of arbitrary-shaped clustering is to use category similarity instead of feature similarity, where the category similarity of two objects is constructed according to local feature similarities of the two objects, which can reflect not only the closeness of two objects, but also the connectivity of them. Based on the idea of category similarity, we extend the classical AP algorithm with the spectral algorithm.

3.2 Arbitrary-shaped AP clustering

The extension of the classical AP algorithm in arbitrary-

shaped clustering is guided by the framework shown in Fig. 3. In the proposed arbitrary-shaped AP clustering algorithm, we construct a sparse adjacency graph $\boldsymbol{G}'$ according to feature similarity matrix $\boldsymbol{S}$; then we map the sparse graph into a low dimensional space by LEs; third, we construct the similarity matrix of objects in new space, which is category similarity $\boldsymbol{S}^c$ that reflects both closeness and connectivity of two objects; last, classical AP is implemented on $\boldsymbol{S}^c$ to get a non-spherical clustering result c. Algorithm 1 presents the proposed spectral affinity propagation (SAP) clustering algorithm.

Algorithm 1 SAP

Input: $\{x_1, x_2, \cdots, x_n\}$
Output: $c = \{c_1, c_2, \cdots, c_n\}$
Steps:
1: Compute feature similarity matrix $\boldsymbol{S}$;
2: Construct the adjacency graph $\boldsymbol{G}'$ according to $\boldsymbol{S}$;
3: Choose the weights of the edges between nearby vertices;
4: Map the weighted graph $\boldsymbol{G}'$ into a low-dimensional space;
5: Compute category similarity matrix $\boldsymbol{S}^c$ in the new space;
6: Passing messages by (1) and (2) according to $\boldsymbol{S}^c$;
7: Repeat Step 6 till convergence, and get the final clustering result.

On the one hand, the proposed SAP is an extension of classical AP in non-spherical clustering. On the other hand, it can also be seen as an improvement of existing spectral clustering algorithms by replacing K-means partition with AP partition. A methodology-level significance of SAP is that category similarity, which is constructed by larger feature similarities, is essential to arbitrary-shaped clustering, which can reflect not only the closeness but also the connectivity of two objects. Based on the constructed category similarity matrix, most of the spherical clustering algorithms can be extended to handle clusters with complex shapes.

3.3 Model selection and parameter optimization

The proposed SAP extends the classical AP in arbitrary-shaped clustering problems, making AP able to be used to discover clusters with complex shapes. This section discusses how to determine the clusters' number and optimize the parameters in SAP.

3.3.1 Determining number of clusters

Determining the number of clusters is a long-standing problem. Essentially, it is a balance of model accuracy and model complexity. The best balance may be different for different datasets, which means there is no generic solution for such kind of problem. In supervised learning, the balance can be adjusted by model validation. However, such a validation procedure cannot be implemented due to a lack of teaching signals in unsupervised problems like clustering. Because of the difficulty analyzed above, there is still no convincing method of determining the number of clusters in current literature. In most cases, the clusters' number k is specified by users when we use K-means and K-medoids.

In AP clustering, the clusters' number k is not required any longer. Alternatively, a parameter called preference p needs to be specified in advance, which is used as a shared self-similarity $s(i,i) = p$. An observation is that the clusters' number k increases with the value of p. In general, preference p is set as the median of the input similarities [6]. In fact, the influence of p on k is also related to the objects' number N. Therefore, we use p^c (preference coefficient) as an input to avoid the influence of the objects' number.

$$p = \text{mean}\ (\boldsymbol{S}) - p^c N. \tag{4}$$

where $\boldsymbol{S}$ is the normalized similarity matrix, which ranges from 0 to 1.

The automatic selection of parameter p^c in this paper is motivated by the following common sense: Imagine we are observing a number of collected data objects $N = \{x_1, x_2, \cdots, x_n\}$. Move the objects from near to far, and our observations go from clear to increasingly blurred. We call this going from a fine granularity to a coarse granularity. Clustering results also vary with observation distance. The number of clusters varies from n to 1, where n means each object is treated as a cluster, and 1 means all the objects are viewed as one cluster. Though different users may prefer different observation granularity, the most stable number of clusters reflects the most probably observed structure of the studied datasets. Algorithm 2 presents the proposed adptive affinity propagation (AdaAP) algorithm, which can determine the value of p^c adaptively.

Algorithm 2 AdaAP

Input: $\boldsymbol{S}$, α
Output: $c = \{c_1, c_2, \cdots, c_n\}$
Steps:
1: Initialize $p^c = 0, p = \text{mean}(\boldsymbol{S}) - p^c N, p_0^c = [\cdot], K_0 = [\cdot]$
2: Run classical AP on $\boldsymbol{S}$ with preference p;
3: Get clustering result $\hat{e}$, $k = \left|\text{unique}(\hat{e})\right|, P^c = [P_0^c p^c]$, $K = [K_0 k]$;
4: If $\max K < N, p^c = \min(P^c) - \alpha, p = \text{mean}(\boldsymbol{S}) - p^c N$, $P_0^c = P^c, K_0 = K$, go to Step 2;
5: If $\min(K) > 1, p^c = \max(P^c) + \alpha, p = \text{mean}(\boldsymbol{S}) - p^c N$,

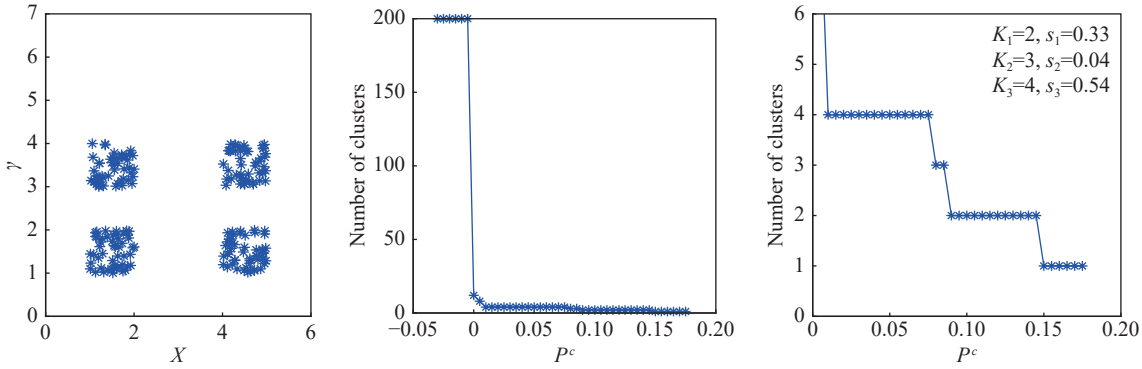
$P_0^c = P^c, K_0 = K$, go to Step 2;
6: $(k, \boldsymbol{I}) = \text{mode}\ K, p^c = \text{mean}\ P^c\boldsymbol{I}$
7: Get final clustering result C with p=mean(s)$-\overline{P^c}N$

We improve the classical AP clustering to make it able to capture non-spherical clusters and to determine the number of clusters adaptively. In the transformed space, message passing is implemented to identify the exemplars. Similar to the classical AP algorithm, the computational complexity is $O(N^2)$, where N is the number of data objects. Our method also supports sparse message passing: if we reduce the number of similarity pairs from N^2 to M and retain only the key similarity pairs, the complexity of the proposed method is $O(M)$.

In Algorithm 2, α is the step length of p^c. We first decrease p^c from a moderate value to a minimum value, which leads to the maximum number of clusters equal to N; then increase p^c from the moderate value to a maximum value, which leads to a clustering result with only one cluster. During this process, $\bar{k}$ is the most stable k value, which is obtained by mode$(\cdot)$ operation on K. So k is viewed as the true number of clusters. $\overline{P^c}$ is The mean of $P^c(\boldsymbol{I})$ is used as the most proper p^c value for this dataset. $\boldsymbol{I}$ stands for the serial numbers of K=$\bar{k}$.

Fig. 5 is a toy example to illustrate how AdaAP determines the clusters' number automatically. The synthetic dataset is shown as Fig. 5(a), which consists of 200 points from four clusters. According to AdaAP, we change the value of p^c from minimum to maximum; the clusters' number decreases with p^c, as shown in Fig. 5(b). Fig. 5(c) is a part of Fig. 5(b). It can be known that the most stable number of clusters is four, the second stable number of clusters is two, which means the most suitable number of clusters for this dataset is four, the second choice of the clusters' number is two. It can be known that AdaAP can find the correct number automatically. The result is very intuitive.

Algorithm 2 presents how to run AP clustering without predefined parameters, which can find the most stable number of clusters automatically. In fact, Algorithm 2 varies the clusters' number from 1 to N by changing p^c. If a user specifies the clusters' number, Algorithm 2 can also find the corresponding p^c value by a bisection procedure.



(a) Four clusters (b) P^c values (c) Results of clusters

Fig. 5 AdaAP determining number of clusters

3.3.2 Optimization of parameters

In Algorithm 1, there are two other parameters that need to be specified by users. Step 2 uses two types of adjacency graphs constructed by ε-neighborhood and KNN separately. We need to specify either ε or K for constructing an adjacency graph. In Step 4, we embed the con-structured adjacency graph into a low-dimensional space. The dimension of the new space d needs to be determined by users. What's more, the two parameters have a great impact on the final clustering result. Therefore, we propose a grid-evolving strategy to select the most suitable parameters. Assume that the KNN graph is employed, combine Algorithms 1 and 2, and we get the final AdaSAP as shown in Algorithm 3. Algorithm 3 first uses a parameter grid to generate many parameter pairs and finds the best parameter pair (k*,d*) by evaluating the clustering result; then it shrinks the parameter grid toward (k*,d*), and finds a new optimal parameter pair around (k*,d*). It repeats the above procedure until the nearby grid's margin is equal to or smaller than 1. Initially, the ranges of k and d are set [1,N], and g can be 2, 5, or 10. It can be viewed as a bisection grid when g=2. η controls the extent of shrinking.

Algorithm 3 AdaSAP

Input: $\{x_1, x_2, \cdots, x_n,\}$
Output: $c = \{c_1, c_2, \cdots, c_n\}$
Steps:
1: $k_{\min} = 1, \beta = \dfrac{N-1}{g}, d_{\min} = 1, r = \dfrac{N-1}{g}$;
2: Define $g \times g$ parameter grid $\boldsymbol{G}$ with $\boldsymbol{G}(i,j) = (k_{\min} + (i-1)\beta, d_{\min} + (j-1)\gamma)$;
3: For each parameter grid, implement Steps 1−5 of Algorithm 1;
4: Get clustering result by Algorithm 2 on the new

similarity matrix;

5: Evaluate the $g \times g$ clustering results, and get the parameter pair (k^*, d^*) with the best clustering result;

6: Redefine parameter grid $\boldsymbol{G}'$ with (k^*, d^*) as grid center, $k'_{\max} - k'_{\min} = (k_{\max} - k_{\min})/\eta$; $d'_{\max} - d'_{\min} = (d_{\max} - d_{\min})/\eta$;

7: Repeat Steps 3−6 till $k'_{\max} - k'_{\min} < \eta \times g$ or $d'_{\max} - d'_{\min} < \eta \times g$, output the clustering result with optimal k and d.

4. Computational experiments

We first evaluate the performance of the adaptive spectral AP clustering algorithms by computational experiments. Then compared with the classical AP clustering, the AdaSAP method has at least two advantages: (i) It can deal with clusters with complex shapes, while the original AP method can only discover spherical clusters; (ii) The number of clusters can be determined automatically by AdaSAP or specified by users. Thus, the experimental part is organized as follows: (i) testing of AdaSAP on the discovery of non-spherical clusters; (ii) testing of AdaSAP on automatic determination of the clusters' number; (iii) comparison of AdaSAP with baseline algorithms by real-world clustering tasks.

4.1 Evaluation criteria

The performance of the clustering algorithm is usually evaluated by the internal dispersity, which means the sum of similarities between objects and their exemplars. We use it as an evaluating criterion in this paper.

However, this paper aims to discover complex-shaped clusters. In such a clustering problem, one should not only take the closeness of objects into account but also consider the connectivity between objects, which cannot be reflected by the internal dispersity. We also use category-label based measures to evaluate the algorithms' effectiveness. By comparing the clustering result with the actual category label, these measures can evaluate different clustering algorithms' performance scientifically and reasonably.

The first category-label based evaluation criterion is normalized mutual information (NMI) [38] defined as

$$\mathrm{NMI} = \frac{I(c_t, \hat{c}_t)}{\sqrt{H(c_t)H(\hat{c}_t)}} \tag{5}$$

where c_t stands for the real cluster tag and $\hat{c}_t$ stands for the tag of the result clustering. $I(c_t, \hat{c}_t)$ denotes the mutual information between c_t and $\hat{c}_t$. $H(\cdot)$ represents information entropy.

Accuracy is a more direct category-label based criteria defined as follows:

$$\mathrm{Accu} = \frac{\sum_{i=1}^{n} \alpha(x_i^c, f(\hat{x}_i^c))}{n} \tag{6}$$

where for the object x_i, x_i^c is the real label, and $\hat{x}_i^c$ is the result clustering label. If $i=j$, $\alpha(i,j)=1$; otherwise, $\alpha(i,j)=0$. Function $f(\cdot)$ labels the clustering tag by comparing the actual label and marking it as the most faithful label or not.

The third evaluation criterion is covering, which is mainly used to evaluate clustering algorithms in image segmentation tasks. According to [39], it is defined as

$$\mathrm{convering}(S'_{\mathrm{result}} \to S_{\mathrm{result}}) = \frac{1}{N} \sum_{R \in S_{\mathrm{result}}} |R| \cdot \max_{R' \in S'_{\mathrm{result}}} O(R, R') \tag{7}$$

where the standard segmentation is S'_{result}, and the segmentation result is S_{result}. $O(R, R')$ is the overlap between two regions R and R', $|R|$ indicates the number of pixels in the region R, and N stands for the number of pixels in the whole image.

4.2 Experiments on the synthetic dataset

We validate the effectiveness of the AdaSAP algorithm by experiments on the synthetic datasets. We first test the ability of SAP in discovering non-spherical clusters by four synthetic datasets, then illustrate how AdaAP can determine the clusters' number automatically.

4.2.1 Testing SAP in discovering non-spherical clusters

A lot of synthetic datasets have been generated to test the ability of clustering algorithms in discovering clusters with complex shapes. Among these datasets, the concentric circles in [2,40], two moons in [41,42], and spirals in [2,40] are the most frequently used synthetic datasets.

We combine the synthetic datasets mentioned above together and generate four new datasets with even more complex cluster shapes: (i) dataset includes 900 data points distributed around three concentric circles (3C); (ii) dataset includes data points distributed on a circle and two rectangles (1C2R); (iii) the dataset is made up of a circle and two moons (1C2M); (iv) the dataset is made up of a circle and two spirals (1C2S), and they are knotted in a three-dimensional space.

The clustering results of four studied algorithms are shown in Fig. 6. AP is just suitable for discovering spherical clusters. The sub-figures in the left column show that the original AP cannot cluster these datasets correctly. In AdaSAP, the data points are first mapped into a new space by LEs, and then AP clusters the data points under the new space. The distances between two objects in the new space can reflect the closeness and the connectivity of objects in the original space. Therefore, AdaSAP can discover clusters with complex shapes. A similar method is spectral shapes, as shown in the sub-figures on the right clustering [34]. However, the objects in the new space are clustered by K-means, which makes spectral clustering fail to get the correct clustering result if the initial centers

are not correctly set. The sub-figures named spectral K-means (SKM) in the second column illustrate the incorrect clustering results suffered by spectral clustering. Different from spectral clustering, AdaSAP uses AP to cluster data points under the new space. The initial exemplar set is not required in AdaSAP. Therefore, it can always get the correct clustering result. Although the DPC algorithm can find the cluster center automatically and realize the efficient clustering of arbitrary shape data, the results also are not good in these datasets. The sub-figures named DPC in the third column illustrate the clustering results.

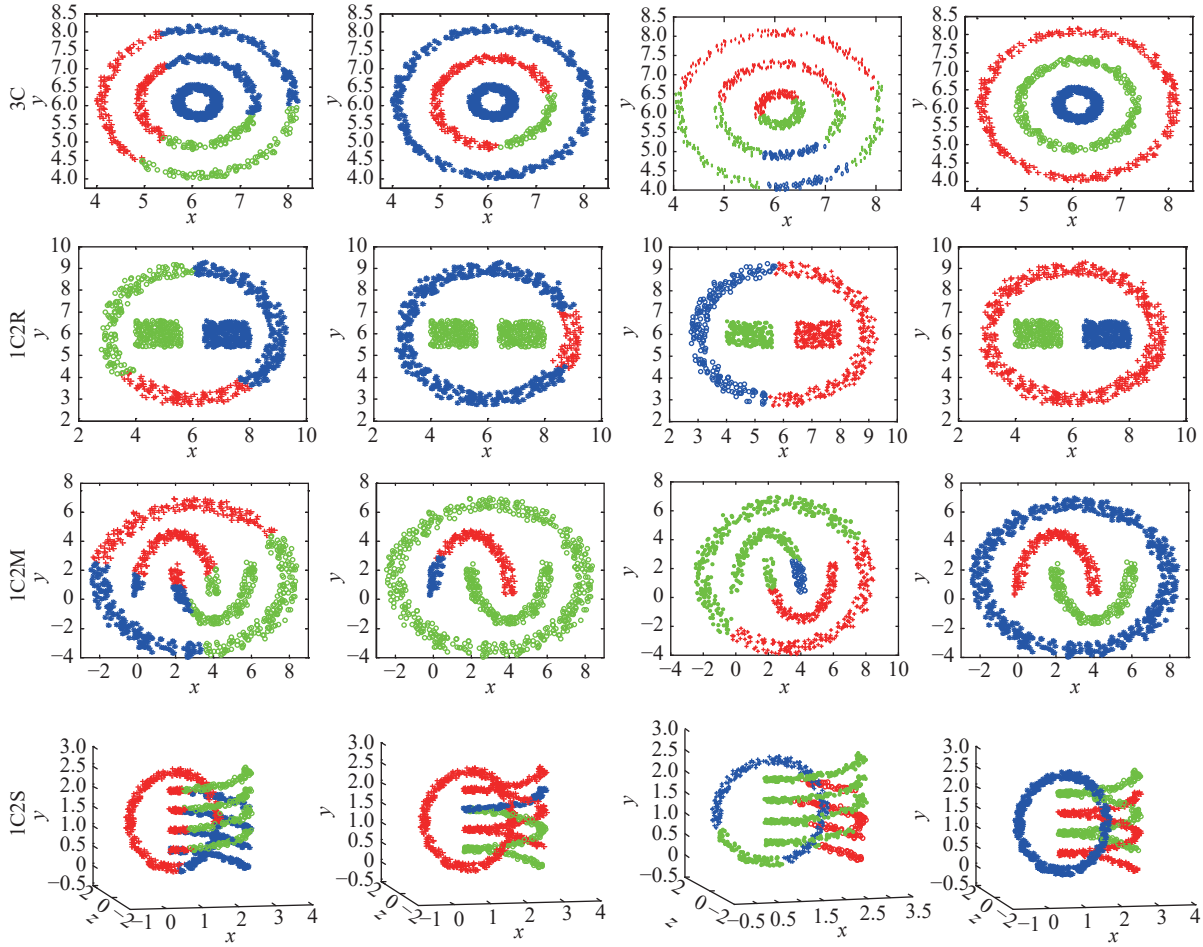


Fig. 6 Computational experiments on four synthetic datasets

To test the robustness of the algorithm, we add random noise to the experimental data 1C2S. In the following experiment, we add 1%, 3%, and 5% random noise to the experimental data, respectively. Then we use the AdaSAP algorithm to cluster the datasets containing noise, and the experimental results are shown in Fig. 7. Experimental results show that there is no effect of the experimental results even the data contain different noise. It also proves that the algorithm has strong robustness.

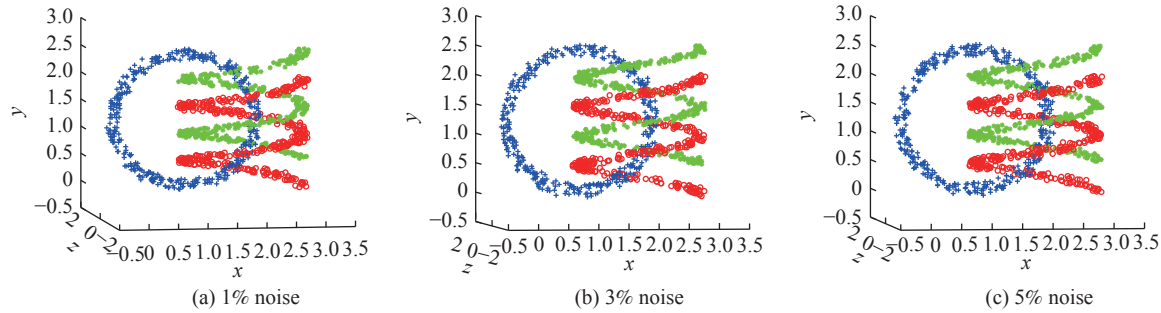


Fig. 7 Clustering results by the noise datasets

4.2.2 Testing AdaAP in determining number of clusters

Fig. 5 presents a toy example to illustrate how AdaAP works. This section validates the effectiveness of AdaAP by more synthetic datasets. Fig. 8(a) shows the original dataset. In Fig. 8(b) and Fig. 8(c), we move the left two clusters leftward gradually. Fig. 8(d)−Fig. 8(f) show how the clusters' number varies with p^c in this case. Fig. 8(d) demonstrates that the most appropriate number of clusters for the original dataset is four. According to Fig. 8(e), the probability of K=4 decreases, while the probability of K=2 increases, but the most proper number of clusters is still four. Fig. 8(c) moves the two left clusters apart from the two right clusters in a further step, so it is very likely for us to view the left two clusters as one cluster and the right two clusters as the other one. Fig. 8(f) suggests that K=2 is the first choice of the clusters' number, while K=4 becomes the second choice. It is unlikely to choose another value (rather than two and four) as the true number of clusters.

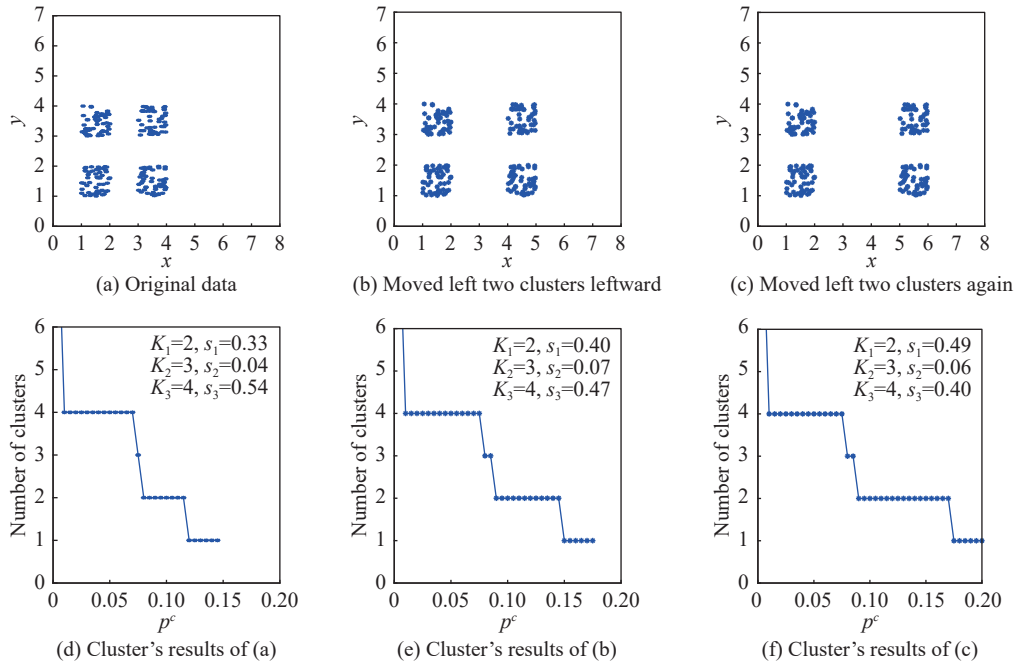
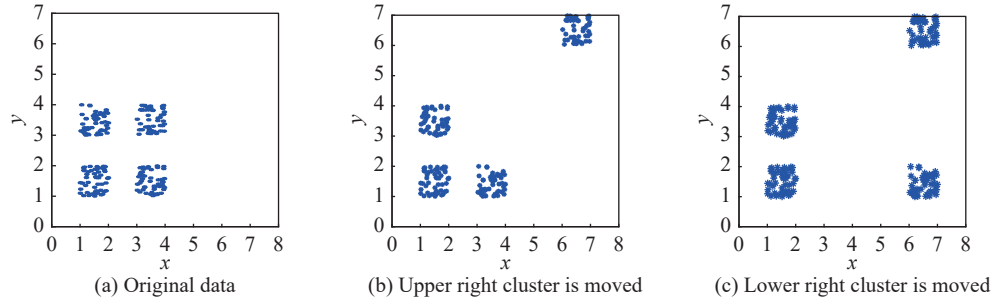
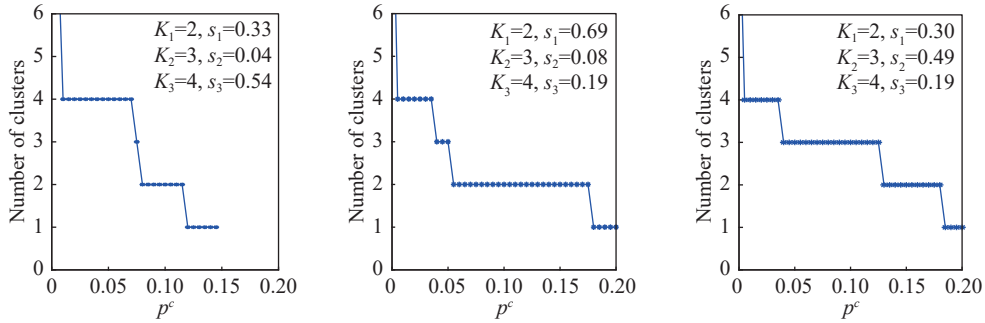


Fig. 8 Computational experiments on four synthetic datasets

In Fig. 9, we generate synthetic datasets in another way. The upper right cluster is moved far apart from the other three clusters in Fig. 9(b), while the lower right cluster is further moved away in Fig. 9(c). Therefore, the most proper number of clusters for dataset in Fig. 9(b) is two, while K=4 is the second choice; the most suitable number of clusters for dataset in Fig. 9(c) is three. According to Fig. 9(e) and Fig. 9(f), AdaAP selects K=2 and K=3 as numbers of clusters for the two datasets, which are accordant with our observations.



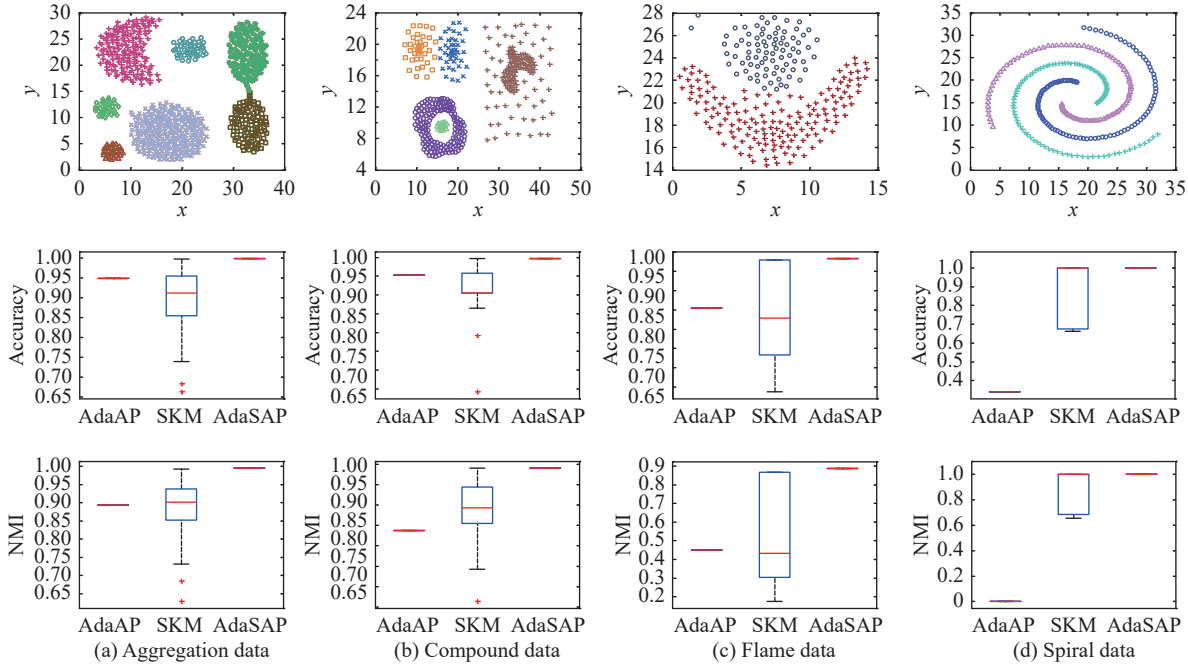


(d) Cluster's results of (a) (e) Cluster's results of (b) (f) Cluster's results of (c)

Fig. 9 AdaAP determining number of clusters automatically in another way to modify datasets

4.3 Experiments on public datasets

The performance of AdaSAP was evaluated by experiments on publicly available datasets. The website in [43] provides several kinds of clustering datasets. We choose the four most popularly used shape sets from this website. Fig. 10 presents the four studied datasets named Aggregation, Compound, Flame, and Spiral.



(a) Aggregation data (b) Compound data (c) Flame data (d) Spiral data

Fig. 10 Computational experiments on four public datasets

Aggregation consists of seven clusters. Two of them are connected. The compound has five clusters in total. One cluster is surrounded by another. The Flame dataset has two clusters. The Spiral dataset consists of data points scattered along three separate spirals. It can be known that most of the clusters have complex shapes, so arbitrary-shaped clustering algorithms usually test their performance with the four public datasets. Additionally, the data points' real class label is also provided to compare different clustering algorithms by Accuracy and NMI.

On the one hand, the AdaSAP algorithm can be seen as an extension of AP in arbitrary-shaped clustering, so we select classical AP as a benchmark algorithm. On the other hand, AdaSAP can be seen as an improvement of spectral clustering as both the two algorithms construct a new similarity matrix leveraging LEs, so we choose spectral clustering SKM algorithm as the other benchmark method. On each dataset, the three algorithms are first compared by Accuracy, and then by NMI, the clustering result of SKM is affected by the initial exemplar set. Therefore, we repeat SKM clustering on each dataset 100 times. Fig. 10 presents the box-plots of Accuracy and NMI achieved by each clustering algorithms. It can be known that AdaSAP achieves the highest Accuracy and NMI on the four datasets. The performance of AdaAP is unanimously inferior to AdaSAP on all the datasets, which suggests that AdaSAP can achieve better clustering results than AdaAP when there exist non-spherical clusters. However, the highest clustering performance achieved by SKM is equal to that achieved by AdaSAP.

In most cases, the clustering performance of SKM is lower than that of AdaSAP. Therefore, replacing *K*-means by AP to cluster objects in a new space is a feasible way to improve the existing spectral clustering algorithms' performance.

4.4 Experiments on image segmentation task

Image segmentation plays a vital role in computer vision. It can be formulated as a data clustering problem [44,45]. In this kind of method, every pixel is treated as an object. The clustering algorithm is employed to divide the pixels into a certain number of clusters. Pixels with the same category label are segmented in the same region. In image segmentation, the shapes of clusters are usually very complex.

Some images from the Berkeley segmentation dataset [46] (BSDS) are used in this paper. The dataset contains so many color images, including humans, sceneries, animals, buildings, flowers, etc. All the images are in a size of 321×481 (or 481×321). Besides the images, five ground-truth segmentation of each image is provided.

In this paper, the segmentation of an image is accomplished by three main steps: (i) Convert an image into many objects. For example, an image in a size of 321×481 can be converted into 1 536 (32×48) objects, each object is a patch with 10×10. (ii) Cluster these patches into several groups. All three clustering algorithms mentioned are implemented. (iii) Refine the clustering result from the patch level to the pixel level, which is realized by assigning each pixel a category label according to its most similar patch.

In the second step, we have to compute similarities between patches. The similarity is computed based on both color and spatial features. The feature of the ith patch is $F_i=(F_i^s,F_i^c)$, where λ is a coefficient which balances the importance of the two different features. $F_i^s=(f_i^h,f_i^v)$ is the location of the ith patch on the image. For the sake of simplicity, we use a relatively simple color feature scheme in this paper. $F_i^c=(f_i^r,f_i^g,f_i^b)$ is the average of each patch under the RGB space. At last, the similarity between a patch and a pixel is computed based on the same mechanism as a pixel can also be represented by a quintuple vector.

Finally, segmentation of an image can be represented by a two-dimensional label matrix $\boldsymbol{S}$. Five ground-truth segmentation of each image is provided in BSDS, and the best-matched ground-truth segmentation $\boldsymbol{S}'$ is used as the standard segmentation to evaluate the quality of segmentation $\boldsymbol{S}$. Two indicators NMI and covering are used to measure the concordance between a segmentation $\boldsymbol{S}$ and the standard segmentation $\boldsymbol{S}'$. The experiments are implemented on 20 randomly selected images from BSDS, and the computational results are presented in Fig. 11 and Fig. 12.

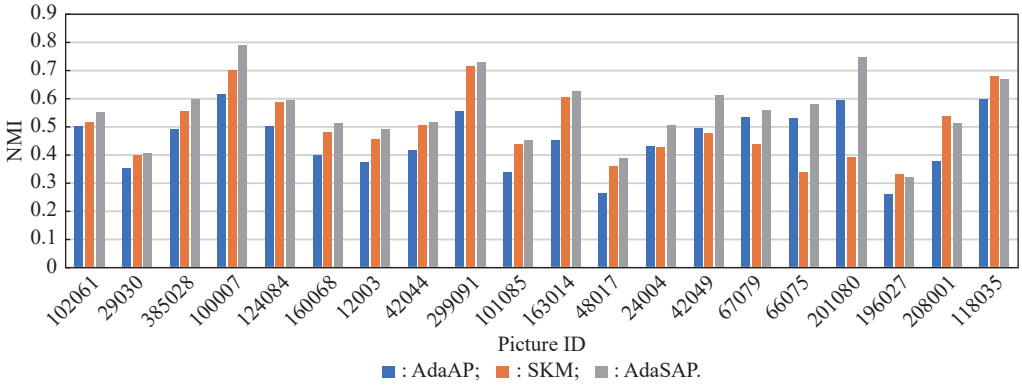


Fig. 11 Comparisons of three clustering algorithms in image segmentation by NMI

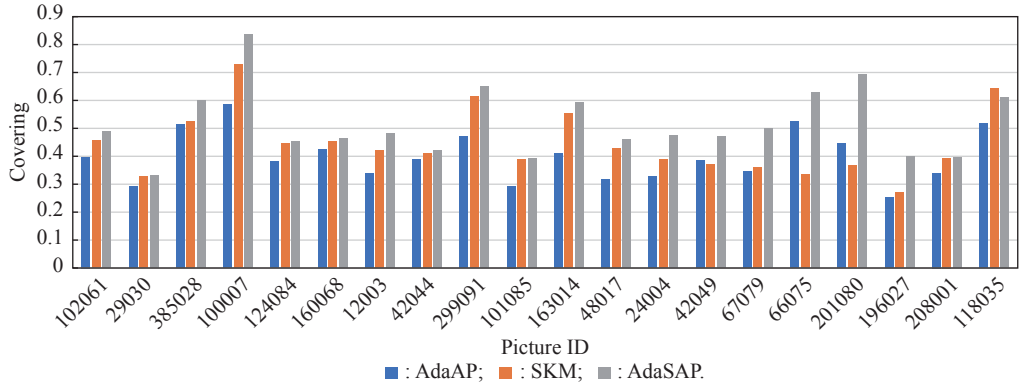


Fig. 12 Comparisons of three clustering algorithms in image segmentation by Covering

Fig. 11 demonstrates that AdaSAP achieves 17 highest NMI values and three median NMI values among experiments on 20 images. NMI achieved by AdaSAP is always higher than that achieved by AdaAP. From the perspective of Covering, as shown in Fig. 12 the superiority of AdaSAP over the other algorithms is even more significant. AdaSAP over the other algorithms is even more significant. It wins out with 19 highest Covering values, and with only one median Covering value. The runner-up algorithm SKM has only three highest values in NMI and one highest value in Covering. These experimental results indicate that AdaSAP can well competent the image segmentation task. Though the segmentation performance of an algorithm varies on different images, which makes an algorithm difficult to achieve consistent highest performance on all images, AdaSAP has the best segmentation performance on a larger proportion of the images. Moreover, it rarely gets the worst segmentation result.

4.5 Experiments on face recognition task

Three of the most popular datasets, the Olivetti Research Laboratory (ORL) face dataset [47], the University of Manchester Institute of Science and Technology (UMIST) database [48], and the Yale database [49], are considered in this subsection. Table 2 shows the description of the three datasets.

Table 2 Description of the three datasets

Dataset	Sample	Image	Size	Class
ORL	400	Grey	112×92	40
UMLST	575	Grey	112×92	20
Yale	165	Grey	243×320	15

The ORL dataset. Four hundred images were taken of 40 humans at different times, varying the lighting, facial expressions (with open or closed eyes, smiling or not), and facial details (glasses or no glasses). A dark homogeneous background took ten images per person with the subjects in an upright, frontal position.

The UMIST dataset. There are 575 images of 20 person. Each person shows in some poses, from profile to frontal views.

The Yale dataset. It contains 11 images for each of 15 individuals. The images show in lighting conditions (left-light, center-light, and right-light), facial expression (normal, happy, sad, sleepy, surprised, and a wink), with or without glasses.

The appearance-based method [50] is used to extract the features of face images, which is a popular method in face recognition. In this method, a two-dimensional face image of size w by h pixels is expressed by a vector in a $w\times h$ space. According to [51], grey level normalization is also implemented, then the negative Euclidian distance is used to compute similarities between objects.

AP is implemented on the obtained similarity matrix to provide benchmark performance. In AdaSAP and spectral clustering, we first construct an adjacent graph according to the similarity matrix and then map the adjacent graph onto a low-dimensional manifold. In this paper, the dimension of the newly constructed low-dimensional space is set five, which means that we reduce the dimension of the feature space from 112×92 (or 243×320 to five before implementing the clustering algorithms.

The experimental results of three clustering algorithms on different face datasets are presented in Table 3 and Table 4. AdaSAP achieves the highest performance in both NMI and accuracy. Compared with AdaAP, AdaSAP can find clusters with complex shapes; while compared with SKM, AdaSAP can avoid the impact of the initial exemplar set on clustering performance. This experiment validates the superiority of AdaSAP over two classical algorithms, AP, and spectral clustering.

Table 3 Comparisons of three clustering algorithms in terms of NMI

Dataset	AdaAP	SKM	AdaSAP
ORL	0.913	0.863	0.971
UMLST	0.565	0.729	0.774
Yale	0.637	0.618	0.680

Table 4 Comparisons of three clustering algorithms in terms of accuracy

Dataset	AdaAP	SKM	AdaSAP
ORL	93.00	80.26	98.00
UMLST	53.21	65.69	69.43
Yale	64.55	59.87	66.36

4.6 Experiments on real medical data

Mechanical ventilation is the most critical life support method for intensive cate unit (ICU) physicians. Retrospective studies based on patient care have essential significance for ICU physicians. The AdaSAP algorithm was used to subdivide the patients into different subgroups. The real dataset of our experiment comes from multiparameter intelligent monitoring in intensive care III (MIMIC-III) [52], a publicly available critical care medicine database developed by the MIT Computational Physiology Laboratory.

The experimental procedure is shown in Fig. 13. It consists of three stages, namely data filtration, feature matrix construction, and clustering. Firstly, during the

data filtering stage, we select adult neoplastic patients from the MIMIC-III database who are not in the perinatal period. Secondly, we construct the feature matrix in the feature matrix construction stage. At the clustering stage, based on the constructed feature matrix, the AdaSAP algorithm was used to cluster.

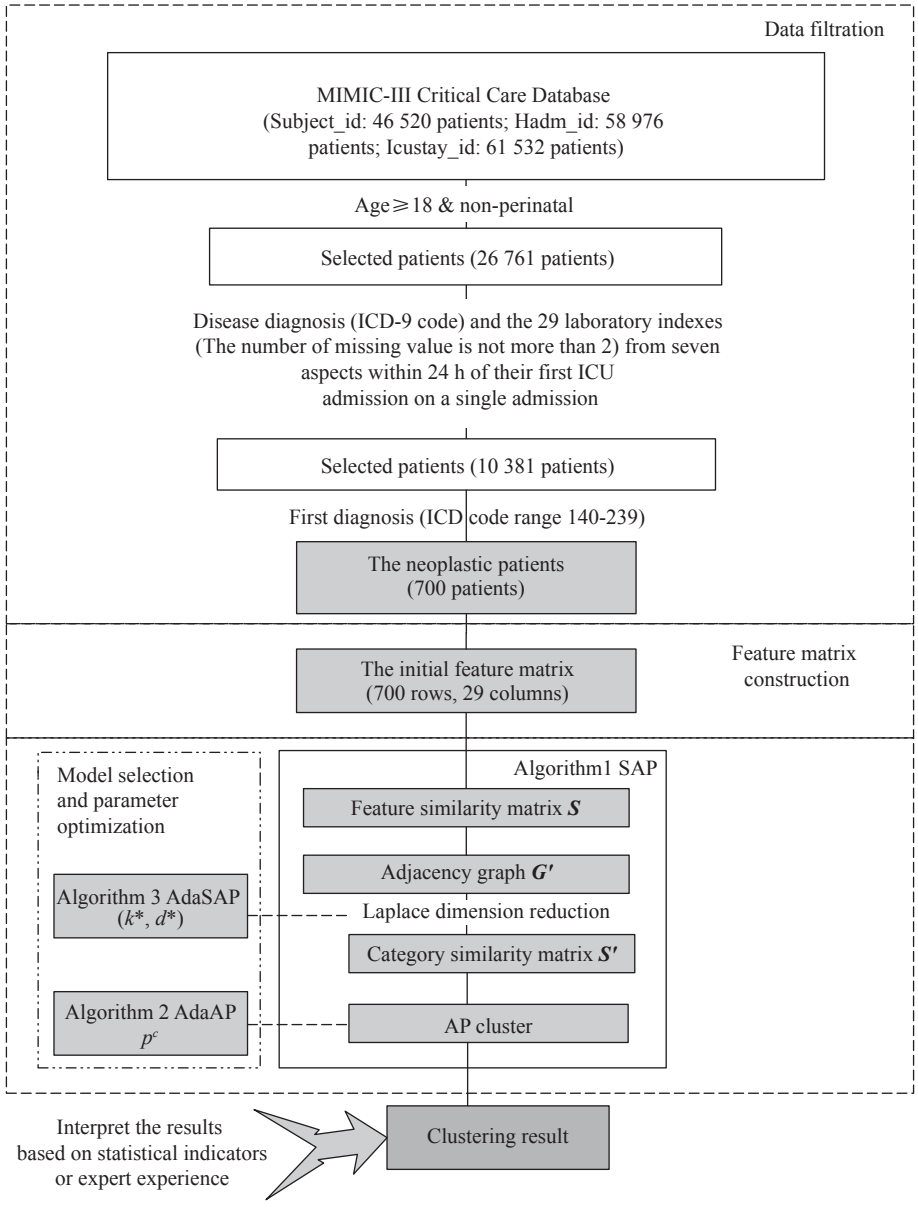


Fig. 13 Experimental procedure on real medical data

4.6.1 Data filtration

MIMIC-III is a large, freely-available database comprising deidentified health-related data associated with the patients who stayed in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012. The dataset includes "subject_id", which identifies a unique ID for each patient; "hadm_id", which identifies a unique ID for each patient admission to the hospital; "icustay_id" which identifies a unique ID for each patient admission to ICU. subject_id tells us that the database contains 46 520 patients; hadm_id tells us that the database contains 58 976 patients; icustay_id tells us that the database contains 61 532 patients. The experimental dataset was selected from adult patients (age⩾18, non-perinatal patients) within 24 h of their first ICU admission (disregard multi ICU admissions afterwards). The total number of cases is 26 761. Referring to the data provided by the MIMIC-III database, the experts finally selected disease diagnosis and 29 laboratory indicators from seven aspects[53]. These laboratory indicators are shown in Table 5.

Table 5 29 laboratory indicators from seven aspects

Index	Aspect	Indicator
1	Demographic	Age, Gender
2	Vital signs (24-hour average values on admission to ICU)	Heart rate, arterial pressure, body temperature, oxygen saturation, respiratory rate, central venous pressure
3	Blood gas	Base excess, buffer base, hemoglobin, lactate, PCO_2, PH, PO_2, white blood cells
4	Incoming and outgoing	Incoming, outgoing, and balance volumes
5	Patient diagnosis	Simplified acute physioligy score (SAPS), sofa
6	Parameters related to mechanical ventilation	Positive end expiratory pressure (PEEP), fraction of inspiration O_2 (FiO_2), tidal volume (VT), peak pressure, average pressure, inspiratory plateau pressure
7	Patient end results	Whether or not dead in the ICU, alive days from the admission into the ICU

Due to the problem of missing important attributes in some patient data, the cases are filtered again, the filtering condition is that the number of missing attributes is no more than two, and finally 10 381 patient cases. According the ICD codes of the patients' first disease diagnosis, the patients are divided into subgroups as shown in Fig. 14.

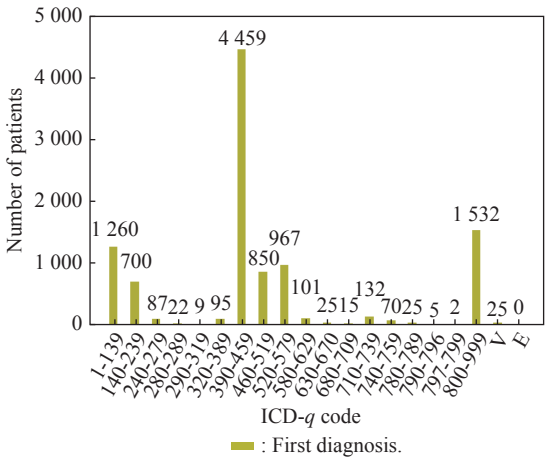


Fig. 14 Subgroups of the patients according to the first disease diagnosis

However, each subgroup of patients can be further subdivided by related laboratory indexes. This task is a typical non-spherical cluster task. Fine clustering of patients is needed. The AdaSAP algorithm not only clusters arbitrary shapes but also helps to find the optimal parameters. The following experiment is carried out with the neoplastic patients (ICD code range 140−239).

4.6.2 Feature matrix construction

In the feature matrix of our experiment, each row represents the data of one patient and one column of the matrix represents an indicator. Therefore, the dimension of the feature matrix used in this experiment is 700×29. Since all the indicators in this experiment are equally weighted, we normalize each indicator.

4.6.3 Clustering by AdasAP

First of all, we calculate the similarity matrix $\boldsymbol{S}$ based on the feature matrix by negative Euclidean distance. Second, we construct the adjacency graph $\boldsymbol{G}'$ using the KNN algorithm. In the experiment, the number of nearest neighbors k=11. Thirdly, Laplace feature mapping for dimensionality reduction. We got the category similarity matrix $\boldsymbol{S}'$. the dimension d is finally set to 10. The parameters (k,d) are auxiliary selected by Algorithm 3. Because the dataset has not been labeled, we use the contour coefficient to evaluate the result of the cluster, and larger clustering contour coefficients indicate better clustering. At last, the typical AP algorithm is used to cluster the category similarity matrix $\boldsymbol{S}'$. the p^c value is set to 0.05 according to Algorithm 2. The patients are divided into three clusters, as shown in Fig. 15.

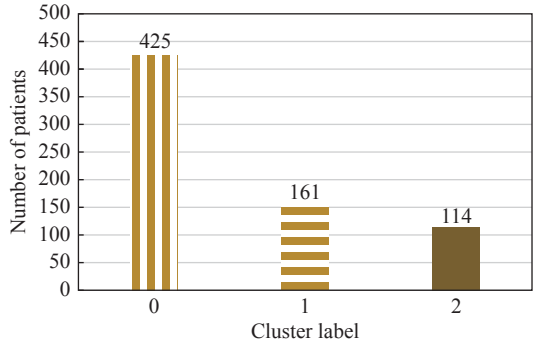


Fig. 15 Distribution of the patients among the clusters

Furthermore, we do a one-way ANOVA between groups for each indicator. The results show that four of the indicators are significant (P<0.05). The names of the indicators are SAPSII, incoming, temperature, and VT. We use the boxplots to visualize the indicators under different clusters, as shown in Fig. 16. The experts analyzed the result that since the first diagnosis was neoplasm, the patient was mechanically ventilated purely for respiratory support. Patients with cluster 0 have the highest SAPSII scores among the three clusters but do not have obvious symptoms of fever. During the treatment, the incomings of the patients are large but the tidal volumes of mechanical ventilation are not high. The Cluster 1 patients generally have high body temperatures, but the SPASII scores are relatively not high. This type of treatment protocol has

the characteristic of the lower input volume and higher tidal volume. Cluster 2 patients have neither high SPASII scores nor significant abnormalities in body temperature using the usual treatment strategy.

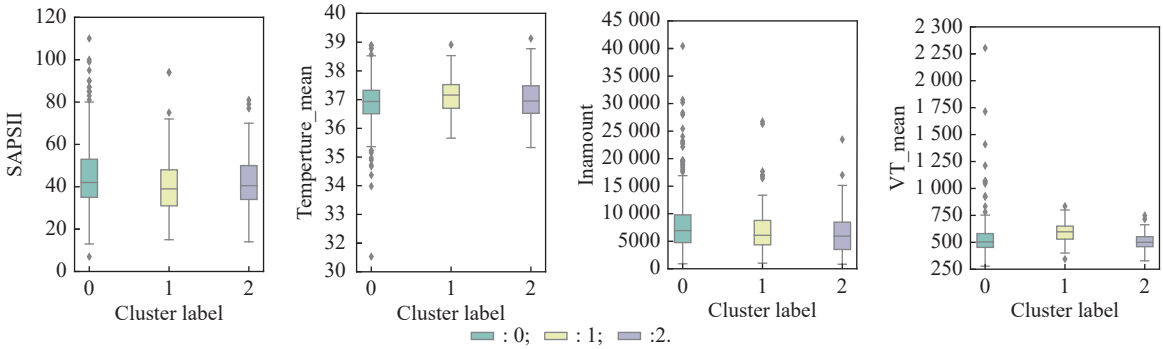


Fig. 16 Distribution of the indicators among the clusters

5. Conclusions and future works

In this paper, we propose an AdaSAP, which improves the classical AP clustering from the following two perspectives. On the one hand, we design an arbitrary-shaped AP clustering algorithm by introducing spectral embedding with category similarity. On the other hand, we propose a procedure to solve a long-standing model selection problem in the clustering field. Leveraging the monotonicity of the number of exemplars with preference, the proposed adaptive AP can determine the clusters' number automatically. Both synthetic datasets and practical clustering tasks are used to test the performance of AdaSAP. Experimental results validate the effectiveness of the proposed AdaSAP.

In the future, we will explore the following directions:

(i) The AdaSAP algorithm has good support for both structured and unstructured data, and we will explore how our model should be applied to specific domains, such as text clustering.

(ii) We will focus on deep clustering algorithms. Combine our work with deep learning to further improve the effect.

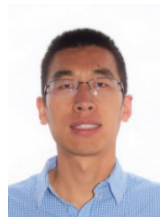
References

[1] HAN J W, KAMBER M, PEI J. Data mining: concepts and techniques. 3rd ed. California: Morgan Kaufmann, 2011.
[2] JAIN A K. Data clustering: 50 years beyond *K*-means. Pattern Recognition Letters, 2010, 31(8): 651–666.
[3] CHEN X L, XIE H R, WANG F L, et al. A bibliometric analysis of natural language processing in medical research. BMC Medical Informatics and Decision Making, 2018, 18(1): 14.
[4] ABDOLREZA H. Black hole: a new heuristic optimization approach for data clustering. Information Sciences, 2013, 222(3): 175–184.
[5] XU D K, TIAN Y J. A comprehensive survey of clustering algorithms. Annals of Data Science. 2015, 2(2): 165–193.
[6] FREY B J, DUECK D. Clustering by passing messages between data points. Science, 2007, 315(5814): 927–976.
[7] KARIV O, HAKIMI S. An algorithmic approach to network location problems. I: the p-centers. SIAM Journal on Applied Mathematics, 1979, 37(3): 513–538.
[8] FREY B J, DUECK D. Response to comment on "clustering by passing messages between data points". Science, 2008, 319(5864): 726.
[9] LI P, JI H F, WANG B L, et al. Adjustable preference affinity propagation clustering. Pattern Recognition Letters, 2017, 85(C): 72–78.
[10] FAN Z Y, JIANG J, WENG S Q, et al. Adaptive density distribution inspired affinity propagation clustering. Neural Computing and Applications, 2019, 31(1): 435–445.
[11] GUAN R C, SHI X H, MARCHESE M, et al. Text clustering with seeds affinity propagation. IEEE Trans. on Knowledge and Data Engineering, 2011, 23(4): 627–637.
[12] KAZANTSEVA A, SZPAKOWICZ S. Linear text segmentation using affinity propagation. Proc. of the Conference on Empirical Methods in Natural Language Processing, 2011: 284–293.
[13] LEONE M, SUMEDHA, WEIGT M. Clustering by soft-constraint affinity propagation: applications to gene-expression data. Bioinformatics, 2007, 23(20): 2708–2715.
[14] FARINELLI A, DENITTO M, BICEGO M. Biclustering of expression microarray data using affinity propagation. Proc. of the IAPR International Conference on Pattern Recognition in Bioinformatics, 2011: 13–24.
[15] JIA C Y, JIANG Y W, YU J. Affinity propagation on identifying communities in social and biological networks. Proc. of the International Conference on Knowledge Science, 2010: 597–602.
[16] LAI D R, NARDINI C, LU H T. Partitioning networks into communities by message passing. Physics Review E, 2011, 83(1): 16115.
[17] ARZENO N M, VIKALO H. Semi-supervised affinity propagation with soft instance-level constraints. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2015, 37(5): 1041–1052.
[18] ZHOU R H, LIU Q M, WANG J, et al. Modified semi-supervised affinity propagation clustering with fuzzy density fruit fly optimization. Neural Computing and Applications, 2021, 33: 4695–4712.
[19] ANNA K, STAN S. Hierarchical topical segmentation with affinity propagation. Proc. of the 25th International Conference on Computational Linguistics, 2014: 37–47.
[20] SUN L L, GUO C H. Incremental affinity propagation clustering based on message passing. IEEE Trans. on Knowledge and Data Engineering, 2014, 26(11): 2731–2744.
[21] SUN L L, GUO C H, LIU C R, et al. Fast affinity propagation clustering based on incomplete similarity matrix. Knowledge and Information Systems, 2017, 51(3): 941–963.
[22] LI Y, GUO C H, SUN L L. Fast clustering by affinity propagation based on density peaks. IEEE Access, 2020, 8:

138884–138897.
[23] MEZARD M. Where are the exemplars. Science, 2007, 315(5814): 949–951.
[24] BELKIN M, NIYOGI P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation, 2003, 15(6): 1373–1396.
[25] BENGIO Y, DELALLEAU O, ROUX N L, et al. Learning eigenfunctions links spectral embedding and kernel PCA. Neural Computation, 2004, 16(10): 2197–2219.
[26] DHILLON I S, GUAN Y Q, KULIS B. Kernel *K*-means: spectral clustering and normalized cuts. Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004: 551–556.
[27] ESTER M, KRIEGEL H P, SANDER J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. Proc. of the Second International Conference on Knowledge Discovery and Data Mining, 1996: 226–231.
[28] FUKUNAGA K, HOSTETLER L D. The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans. on Information Theory, 1975, 21(1): 32–40.
[29] CHENG Y Z. Mean shift, mode seeking, and clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1995, 17(8): 790–799.
[30] COMANICIU D, MEER P. Mean shift: a robust approach toward feature space analysis. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2002, 24(5): 603–619.
[31] RODRIGUEZ A, LAIO A. Clustering by fast search and find of density peaks. Science, 2014, 344(6191): 1492–1496.
[32] SUN L L, CHEN G Q, XIONG H, et al. Cluster analysis in data-driven management and decisions. Journal of Management Science and Engineering, 2017, 2(4): 227.
[33] KARYPIS G, HAN E, KUMAR V. Chameleon: hierarchical clustering using dynamic modeling. IEEE Computer, 1999, 32(8): 68–75.
[34] SCHOLKOPF B, SMOLA A, MULLER K. Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation, 1998, 10(5): 1299–1319.
[35] NG A Y, JORDAN M I, WEISS Y. On spectral clustering: analysis and an algorithm. Proc. of the 14th International Conference on Neural Information Processing Systems, 2001: 849–856.
[36] WANG L J, DING S F, JIA H J. An improvement of spectral clustering an improvement of spectral clustering via message passing and density sensitive similarity. IEEE Access, 2019, 7: 101054–101062.
[37] WANG Y R, DING S F, WANG L J, et al. An improved density-based adaptive p-spectral clustering algorithm. International Journal of Machine Learning and Cybernetics, 2020: 3691304.
[38] VINH N X, EPPS J, BAILEY J. Information theoretic measures for clustering comparison: is a correction for chance necessary? Journal of Machine Learning Research, 2010, 11: 2837–2854.
[39] ARBELAEZ P, MAIRE M, FOWLKES C, et al. Contour detection and hierarchical image segmentation. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2011, 33(5): 898–916.
[40] PASI F, SAMI S. *K* -means properties on six clustering benchmark datasets. Applied Intelligence, 2018, 48(12): 4743–4759.
[41] BAGHSHAH M S, SHOURAKI S B. Kernel-based metric learning for semi-supervised clustering. Neurocomputing, 2010, 73(7/9): 1352–1361.
[42] CHEN W F, FENG G C. Spectral clustering with discriminant cuts. Knowledge-Based Systems, 2012, 28: 27–37.
[43] FANTI P, SIERANOJA S. K-means properties on six clustering benchmark datasets. Applied Intelligence, 48(12), 4743–4759.
[44] FISCHER B, BUHMANN J M. Path-based clustering for grouping of smooth curves and texture segmentation. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2003, 25(4): 513–518.
[45] CHANG H, YEUNG D Y. Robust path-based spectral clustering. Pattern Recognition, 2008, 41(1): 191–203.
[46] MARTIN D, FOWLKES C, TAL D, et al. Berkeley segmentation dataset. https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/.
[47] AT&T Laboratories Cambridge. The ORL database of faces. https://cam-orl.co.uk/facedatabase.html.
[48] GRAHAM D. The UMIST dataset. https://www.visioneng.org.uk/datasets/.
[49] RECOGNITION F. Yale face database. http://vision.ucsd.edu/content/yale-face-database.
[50] HUANG P, TANG Z M, CHEN C K, et al. Local maximal margin discriminant embedding for face recognition. Journal of Visual Communication and Image Representation, 2014, 25(2): 296–305.
[51] RADUCANU B, DORNAIKA F. A supervised non-linear dimensionality reduction approach for manifold learning. Pattern Recognition, 2012, 45(6): 2432–2444.
[52] JOHNSON A, POLLARD T J, SHEN L, et al. MIMIC-III, a freely accessible critical care database. Scientific Data, 2016, 3(1): 160035.
[53] SU L X, ZHANG R M, ZHANG Q, et al. The effect of mechanical ventilation on peripheral perfusion index and its association with the prognosis of critically ill patients. Critical Care Medicine, 2019, 47(5): 685–690.

Biographies

TANG Lin was born in 1980. She received her B.S. degree in computer science and technology from Liaoning Technology University in 2003, and M.S. degree in computer application from Dalian University of Technology in 2008. She is currently pursuing her Ph.D. degree on management science and engineering at Dalian University of Technology. Her research interests include text data mining, machine learning, and deep learning.
E-mail: tanglin@dlut.edu.cn



SUN Leilei was born in 1985. He received his B.S. and M.S. degrees in control theory and control engineering from Dalian University of Technology in 2009 and 2012. He received his Ph.D. degree from Institute of Systems Engineering, Dalian University of Technology, in 2017. He was a postdoctoral research fellow from 2017 to 2019 in School of Economics and Management, Tsinghua University. Currently he is an assistant professor of the State Key Laboratory of Software Development Environment and Big Data Brain Computing Lab, Beihang University. His research interests include machine learning and data mining. He has published several papers on IEEE Trans. on Data and Knowledge Engineering, Knowledge and Information Systems, and ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
E-mail: leileisun@buaa.edu.cn

GUO Chonghui was born in 1973. He received his B.S. degree in Mathematics from Liaoning University in 1995, M.S. degree in Operational Research and Control Theory in 1999, and Ph.D. degree in Management Science and Engineering from Dalian University of Technology in 2002. He was a postdoctoral research fellow in the Department of Computer Science in Tsinghua University. Currently he is a professor of the Institute of Systems Engineering, Dalian University of Technology. His research concentrate on data mining and knowledge discovery. He has published over 150 peer-reviewed papers in academic journals and conferences, in addition to five text-books and two monographs. He has been the Principal Investigator on over 10 research projects from the Government and the Industry.

E-mail: dlutguo@dlut.edu.cn

ZHANG Zhen was born in 1986. He received his B.S. in engineering management from China University of Petroleum (Eastern China) and Ph.D. degree in management science and engineering from the Dalian University of Technology in 2014. Currently he is an associate professor with the Institute of Systems Engineering, Dalian University of Technology. His current research interests include group decision making, computing with words and big data analysis. He is an associate editor of Kybernetes and Journal of Intelligent & Fuzzy Systems, and an editorial board member of the International Journal of Computational Intelligence Systems.

E-mail: zhen.zhang@dlut.edu.cn