Pattern Analysis and Applications (2021) 24:1231–1248
https://doi.org/10.1007/s10044-021-00981-1

THEORETICAL ADVANCES

A novel clustering algorithm by adaptively merging sub-clusters based on the Normal-neighbor and Merging force

Guan Junyi[1] · Sheng li[1] · He Xiongxiong[1] · Chen Jiajia[1]

Received: 18 May 2020 / Accepted: 29 April 2021 / Published online: 12 May 2021

Abstract
Clustering by fast search and find of density peaks (DPC) is a popular clustering method based on density and distance. In DPC, each non-center point's cluster label is led by its nearest point with higher density, which may cause some misclassifications of non-center points and interfere with the choice of correct cluster centers in the decision graph. To avoid these defects, we propose a novel clustering algorithm that automatically generates clusters without using the decision graph based on the Normal-neighbor and Merging force (NM-DPC). We conduct a series of experiments on various challenging synthetic datasets. Experimental results demonstrate that NM-DPC can better identify clusters of complex shapes and automatically recognize the number of clusters.

Keywords Data clustering · Density peaks · Decision graph

1 Introduction

Clustering, a process of dividing a collection of objects into multiple classes with similar characteristics, is an important tool in data mining and has been widely applied to scientific and engineering applications [1–5] such as in computer vision, image mining [6], image segmentation [7], text mining [8]. Since clustering is a problem without a unique solution, numerous clustering method is proposed based on their special definitions of a cluster [9].

For example, K-Means [10, 11] as one of the most popular clustering algorithms defines a cluster as a group of data points with a small distance from a cluster center. Due to its simplicity and efficiency, K-Means has been widely used in various disciplines. However, K-Means still has some limitations: It cannot detect clusters with arbitrary shapes; it can easily get into local minima [12]; it requires the number of cluster centers as an input parameter. Despite various algorithms have been developed to remedy these limitations [13–15], they all fail to detect clusters with arbitrary shapes due to the fact that data points are always assigned to the nearest center. The classic graph-based spectral clustering [16] algorithm can recognize arbitrary-shaped clusters by considering a cluster as a set of closely connected points in a graph structure. Nevertheless, like K-Means, spectral clustering also requires the number of cluster centers as input.

Density-based clustering method is outstanding in automatically identifying clusters of arbitrary shapes without setting cluster centers. Density-based spatial clustering of applications with noise (DBSCAN) [17] as a typical density-based method can detect any arbitrary shape clusters with specified density thresholds, such as ϵ, the neighborhood radius and MinPts, the minimum number of points included in the neighborhood with radius ϵ [18, 19]. However, DBSCAN may merge two or more clusters that are in close proximity.

Density peak clustering (DPC) [20] proposed by Rodriguez and Laio can effectively partition closely connected clusters by initially finding density peaks. DPC assumes that a cluster center should have a higher density ρ than its surrounding neighbors and have a relatively

✉ Sheng li
shengli@zjut.edu.cn

Guan Junyi
jonnyguan73@163.com

He Xiongxiong
hxx@zjut.edu.cn

Chen Jiajia
fl_katrina@163.com

[1] College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China

large distance δ from the nearest data point a with higher density. Based on this assumption, cluster centers as large density-distance points can be easily detected in the decision graph (i.e., a density-distance plot). After the cluster centers are determined, DPC's allocation strategy assigns each non-center point into the cluster of its leading point (i.e., the nearest data point a with higher density) to complete clustering without iterating. Although generic allocation strategy is applicable to any cluster shape, it has two conditions: First, all the selected cluster centers are correct; second, each non-center point's cluster label is actually the same as its leading point's. In other words, a wrong selection of cluster centers or any inconsistency between points and their leading points' actual cluster labels both can cause the misclassification of DPC. However, when dealing with clusters of arbitrary and heterogeneous structure, it is difficult to ensure the consistency of each point's actual cluster label and the cluster label of its leading point, which means DPC's allocation strategy is not robust [21].

An example is presented in the following part to better explain the limitation of DPC's allocation strategy.

As shown in Fig. 1a, the dataset is composed of two clusters: the right-side cluster and the left-side cluster, where the number indicates the density ρ of each point, and point C and D are the cluster centers of the two clusters. As shown in Fig. 1b, except for the highest density point (as point C), each point has an arrow pointing to its leading point, and its distance value δ is the Euclidean distance toward its leading point. Although cluster centers C and D can be found intuitively in the decision graph (i.e., a plot of δ_i as a function of ρ_i for each point i) (Fig. 1c), DPC's allocation strategy misclassifies point $D1$ into the left-side cluster due to point C (the leading point of $D1$) and point $D1$ are not really in the same cluster. At the same time, all points affected by point $D1$ that should belong to the right-side cluster are also misclassified to the left-side cluster.

Numerous methods have been proposed to reform the allocation strategy of DPC. In [22], the cluster labels of neighbors play an important role in assigning the non-center points. Pizzagalli et al. [21] assign non-center points based on the shortest path and train a path classifier by providing examples of valid and invalid paths to further eliminate the wrong allocation paths. Non-center points are assigned robust, but the selection of cluster centers still relies on the decision graph. In other words, incorrect selection of cluster centers in the decision graph will directly lead to bad clustering results. Thus, it is critical to select the correct cluster centers in the decision graph. However, in some cases, the decision graph may show some large density-distance points that cannot represent real cluster centers to mislead the correct selection of cluster centers. An example is presented in Fig. 2 to better explain the abovementioned limitation of DPC's decision graph.

In Fig. 2a, the dataset is composed of two clusters: the right-side cluster with center F and the left-side cluster

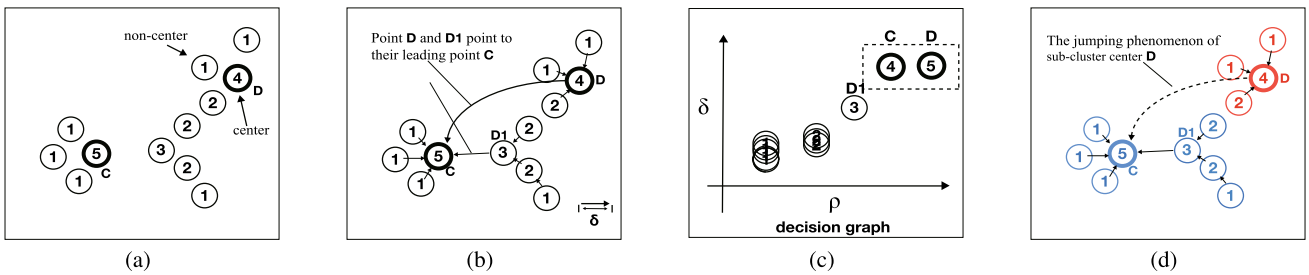


Fig. 1 Allocation strategy leads to the misclassification in DPC

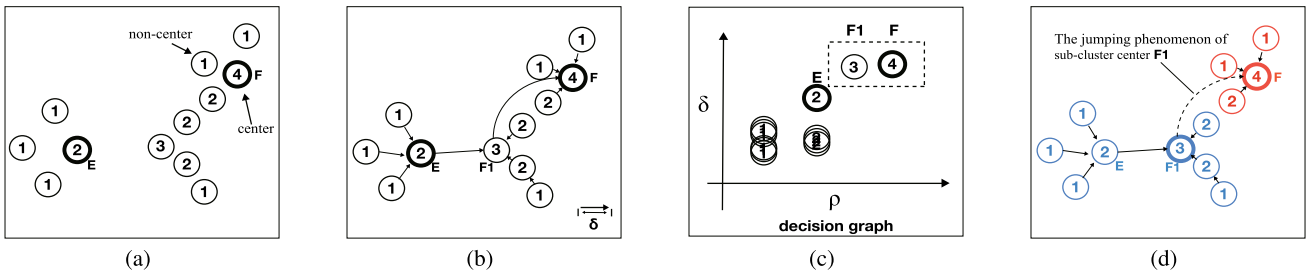


Fig. 2 Leading relationship leads to decision graph misleading the correct choice of centers in DPC

with center E. Figure 2b shows points' leading relationships, where point $F1$ has a high density and a leading point F that far away from it, as a result, point $F1$ has a high ρ value and large δ value. Thus, in the decision graph Fig. 2c, we find that point $F1$'s ρ_{F1} and δ_{F1} are even larger than cluster center E's, thus point $F1$ may more easily be selected as the cluster center of the left-side cluster. This leads to the misclassification of point $F1$ and all points led by it as shown in Fig. 2d.

To obtain a decision graph that can better display the correct cluster centers, some methods have proposed to change the evaluation method of density. [23] evaluates the density of each point based on KNN (K-nearest neighbors) method, which makes the detection of low-density cluster centers in the decision graph become easier. In [24], a shared-nearest-neighbor-based method is used to evaluate the density of each point. Although these methods make the inconspicuous cluster centers in the decision graph clearer, for some complex datasets, it is still challenging to select the correct cluster centers. In addition, the method of selecting cluster centers is manual, which means that the execution of clustering is semi-automated.

In this work, we present a novel automatic clustering algorithm that adaptively merging sub-clusters based on the Normal-neighbor (see Sect. 3.1) and Merging force (see Sect. 3.2), called as NM-DPC. It not only can effectively overcome the limitation of DPC's allocation strategy but also gets rid of the manual selection of cluster centers in the decision graph.

Herein, a definition is introduced to further summarize the defects of DPC, that is, the **jumping phenomenon** of sub-cluster centers. From the above two examples, it can be noted that the point with the highest density (except for the highest density point in the dataset) in a density area that composed of points led by it will jump out of the area to find its leading point. We consider this density area as a sub-cluster where the highest density point is viewed as the sub-cluster center. In this paper, we attribute the limitations of DPC's allocation strategy and decision graph to the unstable allocation of sub-clusters: a sub-cluster center jumps to the wrong area (i.e., an area of points with another cluster label) will directly lead to the misallocation of the entire sub-cluster (see point $D1$ in Fig. 1); a sub-cluster center's jumping behavior may make it have a large density-distance value that may lead to a confusing decision graph (see point $F1$ in Fig. 2). Thus, this jump phenomenon causes DPC to have the following disadvantages:

1. Sub-cluster allocation is unstable.
2. The sub-cluster centers may interfere with the correct selection of cluster centers in the decision graph.

To avoid the jumping phenomenon of sub-cluster centers caused by the instability of the sub-cluster center in finding its leading point, we only allow each point to find its leading point in its normal neighbors (i.e., neighbors with real adjacent relationships) that are obtained based on our normal-neighbor method. As a result, sub-cluster centers (i.e., the points without a leading point in their normal neighbors) are emerged automatically and are no longer divided by their unstable leading points (namely the jumping phenomenon will not occur). Then, sub-clusters are formed according to the sub-cluster points and the allocation strategy that each point is assigned into the cluster of its leading point in its normal neighbors. These sub-clusters need to be merged into clusters to complete clustering. To analyze the possibility of merging two sub-clusters, we propose a concept of Merging force based on the structural characteristics of sub-clusters. By adding a fixed merge threshold, the intersecting sub-clusters are merged into clusters spontaneously according to the merging force between them to complete the clustering.

Our method is fully automatic without applying the decision graph, which, as a result, ensures that our algorithm never misclassifies clusters. Figure 3 simply shows the process of our algorithm. As shown, for the dataset (Fig. 3a), our algorithm first automatically forms three sub-clusters based on Normal-neighbor, then the two sub-clusters on the right-side are merged according to

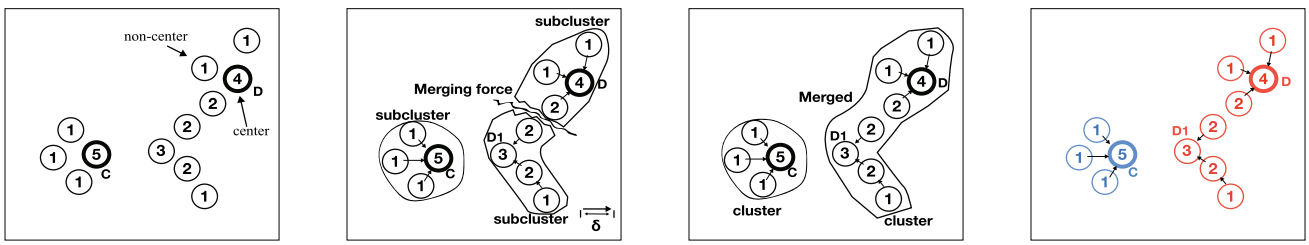


Fig. 3 NM-DPC's clustering process

the Merging force (Fig. 3b, c). Finally, the clustering result is shown in Fig. 3d. Thus, it can be seen that the new features of NM-DPC are:

1. Sub-clusters are generated by each point finding its leading point based on Normal-neighbor, in other words, sub-cluster centers do not need to search for leading points beyond its cluster, which means there will be no jumping phenomenon in NM-DPC.
2. Sub-clusters are merged automatically according to the Merging force between sub-clusters, which means clusters can naturally emerge without using the decision graph.

The rest of this paper is composed as follows: Sect. 2 gives a brief introduction to DPC algorithm and its analysis. Sections 3 and 4 are mainly focused on introducing and analyzing NM-DPC algorithm, while Sect. 5 tests our proposed algorithm by experiments on synthetic and real-world datasets. Finally, Sect. 6 is a general conclusion to this paper.

2 DPC algorithm and analysis

2.1 Notations

The major symbols and notations used in the following parts are presented in Table 1.

2.2 DPC algorithm

DPC defines the local density ρ_i for each data point i as in Eq. 1, and the distance δ_i to the nearest data point with a higher density is defined as Eq. 2. Where d_{ij} is the Euclidean distance between point i and point j, while d_c is the cutoff distance which was proposed in [20]. $\chi(x) = 1$ if $x < 0$, otherwise, $\chi(x) = 0$, basically, ρ_i is equal to the total number of points in the d_c range of data point i. In addition, for some sparse datasets, DPC estimates the local density by a Gaussian kernel with a pre-specified cutoff distance d_c, as in Eq. 3.

$$\rho_i = \sum_{j \neq i} \chi\left(d_{ij} - d_c\right) \tag{1}$$

$$\delta_i = \min_{j:\rho_j > \rho_i} \left(d_{ij}\right) \tag{2}$$

Table 1 Notations in DPC and NM-DPC

Symbol	Meaning
N	The total number of data points in the data set
d_{ij}	The Euclidean distance between point i and j
$\rho = (\rho_1, \rho_2, \ldots \rho_N)$	The local density value of data points
$\delta = (\delta_1, \delta_2, \ldots \delta_N)$	The distance value of data points
i_T	The Tth ($T \leqslant K$) nearset neighbor of point i
i_T^{in}	The minimum inner neighbor distance of i_T
ζ_{i_T}	The jumping coefficient of point i's neighbor i_T that indicates the jump amplitude of the neighbor
$\varepsilon \in \{1, 2, 3, 4, 5\}$	The anti-jump threshold that used to determine the abnormal neighbors
i^{leading}	The leading point of point i
K	The parameters used to set the number of nearest neighbors are considered
KNN_i	The K nearest neighbors of point i
NN_i	The normal nearest neighbors among K nearest neighbors of point i
NAN_i	The nearest abnormal neighbor of point i
$\text{SC} = (\text{SC}_1, \text{SC}_2, \ldots)$	The sub-clusters
$C = (C_1, C_2, \ldots)$	The clusters
$M = (M_{\text{SC}_1}, M_{\text{SC}_2}, \ldots)$	The merging ability of sub-clusters
$\kappa = (\kappa_{\text{SC}_1}, \kappa_{SC_2}, \ldots)$	The sharpness of sub-clusters' density peaks
$B_{\text{SC}_p\text{SC}_q}$	The boundary point set of two sub-clusters
$S_{\text{SC}_p\text{SC}_q}$	The highest density point on the boundary of two intersecting sub-clusters called a saddle point.
$O_{\text{SC}_p\text{SC}_q}$	The overlapping thickness coefficient between two sub-clusters
$\text{MF}_{\text{SC}_p\text{SC}_q}$	The Merging force between two sub-clusters
$\lambda \in [0, 1]$	The merge threshold parameter

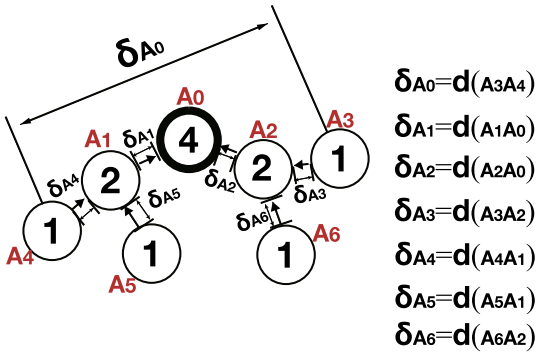


Fig. 4 Distance value δ's illustration

$$\rho_i = \sum_{j \neq i} \exp\left(-\left(\frac{d_{ij}}{d_c}\right)^2\right) \tag{3}$$

δ_i is the minimum distance between point i and any other point j with higher density, and the highest density point $i_{\max}$ has the largest distance $\delta_{i_{\max}} = \max(d_{ij})$. Figure 4 illustrates the basic principle of the distance δ.

According to DPC, δ_i is much larger than the typical nearest neighbor distance only for points that are local or global maximum in the density, thus, cluster centers can be determined because δ_i has an abnormally large distance value [20].

The selection of cluster centers is a critical step in the clustering analysis of DPC. DPC uses a decision graph, that is, the plot where δ_i as a function of ρ_i for each point i. The cluster centers can be determined by finding the points with large ρ-δ in the decision graph.

DPC does not introduce a noise-signal cutoff, instead, it defines the set of points within a distance d_c from other clusters' data points as the border region of each cluster. DPC finds the highest density point within each cluster's border region and denotes its density as ρ_b. The points in the cluster whose density is lower than ρ_b are considered to be noise.

2.3 Analysis

As mentioned in Sect. 1, although DPC has good clustering performance, it still has some defects caused by the jumping phenomenon of sub-cluster centers.

1. The limitation of the decision graph.
2. The allocation limitation of sub-clusters.

These two limitations of DPC are detailed in the following part.

2.3.1 The limitation of the decision graph

An example is presented in Fig. 5 to show the limitation of the decision graph. Figure 5a shows the clustering result (by selecting 2 cluster centers in the decision graph) of DPC on the Jain dataset [25] which is clearly composed of two crescent-shaped clusters, the left-side branch cluster with center E, and the right-side branch cluster with center A.

However, as shown in Fig. 5a, DPC cannot fully recognize the Jain dataset, since it selects point $A1$ and A as the cluster centers (as shown in Fig. 5b), which is obviously a misselection of centers as point $A1$ and A are all belong to the right-side cluster. As a result, point E (the real left-side cluster center) is missed because it is not conspicuous in the decision graph. The reason behind is that point $A1$ has a larger ρ-δ value than point E, which makes it easier to be regarded as a cluster center candidate in the decision graph.

The above example verifies that points with a large ρ-δ value in the decision graph cannot always represent the real cluster centers but may even mislead the choice of cluster centers. In addition, as mentioned in Sect. 1, if we can ensure that the cluster center selection is always correct, DPC may still not completely accurate allocation. This is because DPC is unstable in sub-cluster allocation.

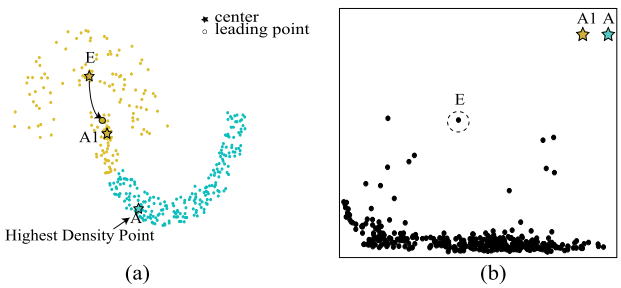


Fig. 5 The clustering result (**a**) and the decision graph (**b**) of DPC with selected Center_A and Center_{A1} as cluster centers on Jain

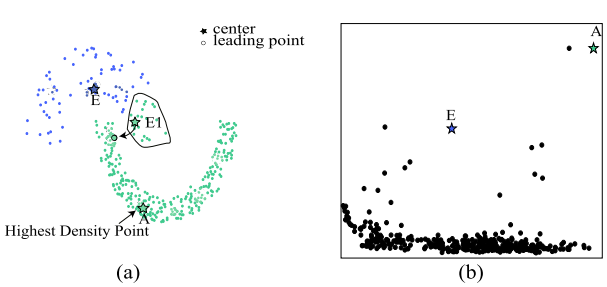


Fig. 6 The clustering result (**a**) and the decision graph (**b**) of DPC with selected Center_A and Center_E as cluster centers on Jain

2.3.2 The allocation limitation of sub-clusters

Due to the jumping phenomenon of sub-cluster centers (mentioned in Sect. 1), even when the choice of cluster centers is correct, some sub-clusters may also be misclassified by DPC.

For example, Fig. 6a shows the clustering result of DPC when it selects the correct cluster centers in the decision graph (Fig. 6b). But, we can still observe that a sub-cluster (circled out by black line) with the sub-cluster center $E1$ is mistakenly divided into the right-side cluster due to the jumping phenomenon of $E1$. The principle behind this is that the labels of points in $E1$'s sub-cluster are all led by the sub-cluster center $E1$, but point $E1$ finds its leading point in the right-side cluster. The labels of all the points in the sub-cluster are led by the sub-cluster center $E1$. But, due to $E1$'s leading point is in the right-side cluster, as a result, the whole sub-cluster of $E1$ is assigned into the right-side cluster.

The above example demonstrates that the jumping phenomenon of sub-cluster centers may cause DPC's misallocation of sub-clusters.

To avoid DPC's limitation of the decision graph and the misallocation of sub-clusters. Herein, a novel algorithm is proposed which performs clustering by adaptively merging sub-clusters based on the Normal-neighbor and Merging force (NM-DPC).

3 The proposed NM-DPC algorithm

NM-DPC algorithm offers a solution that each point searches for its leading point only in its Normal-neighbor which effectively avoids the jumping phenomenon, and the clusters will naturally emerge after the sub-clusters are merged by using the Merging force in between, which breaks the limitation of decision graph.

This section presents the essential details of our proposed clustering algorithm, such as Normal-neighbor, Merging force.

3.1 Normal-neighbor

In order to avoid the jumping phenomenon, based on the idea of KNN (K-Nearest Neighbor), we design the Normal-neighbor method to limit the searching range of each point for its leading point, so that points cannot jump to other clusters to get their leading points (namely the jumping phenomenon is avoided).

In our Normal-neighbor method, we first evaluate the neighbors' distribution characteristic of each point, then use this characteristic to help point in obtaining its neighbors really close to it. In this way, the neighbors of each point can be ensured in a cluster. Unlike in the KNN method, K is a fixed value that may fail to ensure that all K neighbors of a point belong to one cluster. The following part is a detailed introduction of the Normal-neighbor method.

Normal-neighbor method introduces two new definitions: normal neighbors (i.e., neighbors with real adjacent relationships) and abnormal neighbors (i.e., neighbors without adjacent relationships). Herein, for each point i, we view i's nearest abnormal neighbor (NAN_i) as a border between normal and abnormal neighbors of point i. Then, neighbors inside NAN_i (namely inner neighbors of NAN_i) are defined as normal neighbors, and neighbors outside NAN_i are defined as abnormal neighbors. Therefore, as long as NAN_i can be accurately detected, the normal neighbors of i can be identified.

Normal-neighbor method detects the NAN by using the assumption that the NAN has a relatively large distance from its inner neighbors. Thus, the minimum inner neighbor distance of each neighbor needs to be measured for detecting abnormal neighbors, as defined in Eq. 4, where i_T is the Tth neighbor of point i and $0 < t < T$ means i_t is the inner neighbor of i_T.

For example, as shown in Fig. 7, A_1–A_5 are the five nearest neighbors of point A. The distance from A_1–A_5 to A is gradually increased from 1 to 2.2, 3.6, 5.1, 5.2. Compared to A_1–A_4, A_5 is more like an abnormal neighbor to A because it is far from the other neighbors around A. Since the NAN has a relatively large distance from its inner neighbors, it can be intuitively observed that only A_5 is far away from its inner neighbors (A_1–A_4). Then, we calculate the minimum inner distance of A_1–A_5, and get $A_5^{\mathrm{in}} = 5.2$ which is much larger than other neighbors',

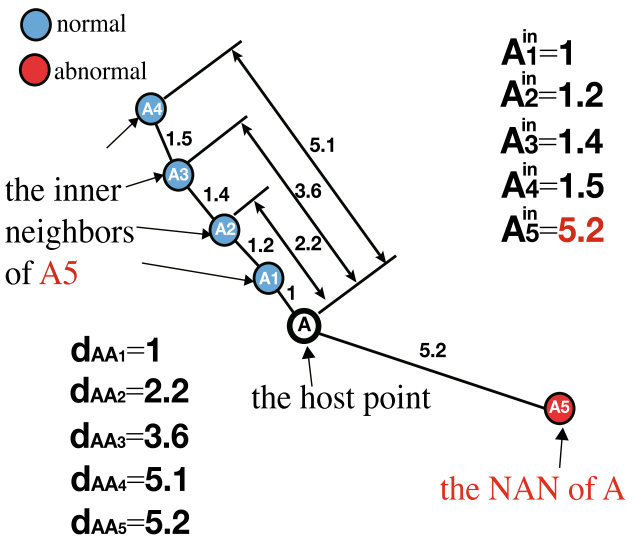


Fig. 7 Minimum inner distance

so we assure that A_5 is an abnormal neighbor of point A and should not be counted as point A's normal neighbor.

To automatically detect the NAN_i (namely point i's nearest abnormal neighbor.), we design a jumping coefficient ζ_{i_T} for each neighbor i_T in KNN_i ($T < K$) of as in Eq. 5. $N_{\min}$ is the minimum number of normal neighbors. After experiment, we find the value of $N_{\min}$ is not sensitive, we generally set $N_{\min} = 5$. Denominator $\frac{1}{T-1} \times \sum_{p=1}^{T-1} i_p^{\mathrm{in}}$ is the average minimum inner distance value of i_T's inner neighbors, which can indicate the compactness of i_T's inner neighbors. A large ζ value of neighbor indicates that the distance between the neighbor and its inner neighbor is large, and vice versa. Since the NAN has a relatively large minimum inner distance, NAN should have a relatively larger ζ than its inner neighbors'. Based on this feature, we use an anti-jump threshold constant ε ($\varepsilon \in [1, 2, 3, 4, 5]$) as an input parameter to detect the abnormal neighbors : if $\zeta_{i_T} > \varepsilon$, point i_T is considered as an abnormal neighbor of point i. Thus, for point i, NAN_i is the nearest neighbor in KNN_i whose ζ vlaue is larger than ε, as defined in Eq. 6. Then, the definition of normal neighbors of point i (NN_i) in KNN_i is shown in Eq. 7.

In a word, our Normal-neighbor method can find the possible homology relationship between K nearest neighbors. When $\varepsilon = \infty$, Normal-neighbor method essentially trend to KNN.

$$i_T^{\mathrm{in}} = \min_{0<t<T}(d_{i_T i_t}), i_T \in \mathrm{KNN}_i \tag{4}$$

$$\zeta_{i_T} = \frac{i_T^{\mathrm{in}}}{\frac{1}{T-1} \times \sum_{t=1}^{T-1} i_t^{\mathrm{in}}}, N_{\min} \leqslant T \leqslant K \tag{5}$$

$$\mathrm{NAN}_i = \left\{ i_T | \min_{\zeta_{i_T > \varepsilon}}(T), i_T \in \mathrm{KNN}_i \right\} \tag{6}$$

$$\mathrm{NN}_i = \left\{ i_T | d_{ii_T} < d_{i\mathrm{NAN}_i}, i_T \in \mathrm{KNN}_i \right\} \tag{7}$$

To demonstrate the performance of our Normal-neighbor, we use Fig. 8 which shows point A's range of obtaining its 5th nearest neighbors based on Normal-neighbor ($\varepsilon = 1, 2, 3$) and KNN. In Fig. 8, normal neighbors are marked in blue, abnormal neighbors are marked in red, and the search range of the 5th nearest neighbor is in the gray area. It can be noted in Fig. 8a–c that Normal-neighbor calculates the average minimum inner distance value of the four inner neighbors ($A1$–$A4$) which is 1.275 and then limits the searching range of A's 5th neighbor according to anti-jump threshold constant ε, so as to exclude the abnormal neighbors as much as possible. While KNN does not limit the search range of the 5th nearest neighbor, as a result, it cannot exclude the abnormal neighbor $A5$ (as in Fig. 8d).

It can be noted that compared to KNN, Normal-neighbor can get appropriate neighbors according to the surrounding distribution characteristics of each point, ensuring that all neighbors are in the same cluster.

3.2 Merging force

In order to merge sub-clusters into clusters automatically, for each sub-cluster, we propose a concept of merging ability (denoted as M). We assume that the stronger the merging ability of two intersecting sub-cluster, the easier they are to be merged, that is, the smaller the overlapping degree required. Based on this idea, we design a Merging force method as in Eq. 8, where $\mathrm{MF}_{\mathrm{SC}_p\mathrm{SC}_q}$ is the Merging force coefficient of intersecting sub-clusters SC_p and SC_q, and $O_{\mathrm{SC}_p\mathrm{SC}_q}$ is the overlapping thickness coefficient of sub-clusters SC_p and SC_q that indicates the overlapping degree of them. The following part is a detailed introduction to our Merging force method.

$$\mathrm{MF}_{\mathrm{SC}_p\mathrm{SC}_q} = \frac{1}{2} \times (M_{\mathrm{SC}_p} + M_{\mathrm{SC}_q}) \times O_{\mathrm{SC}_p\mathrm{SC}_q} \tag{8}$$

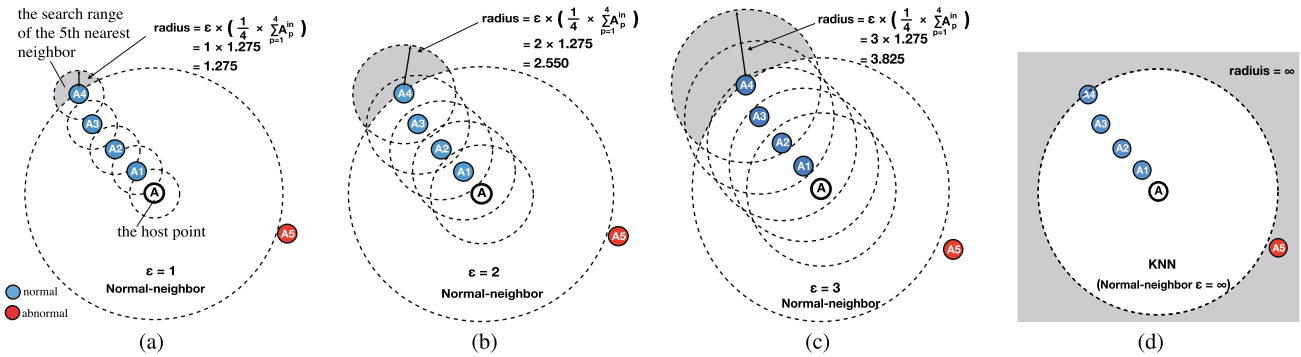


Fig. 8 Point's search range of 5th normal neighbor based on Normal-neighbor ($\varepsilon = 1, 2, 3$) (**a**–**c**) and 5th nearest neighbor based on KNN (**d**)

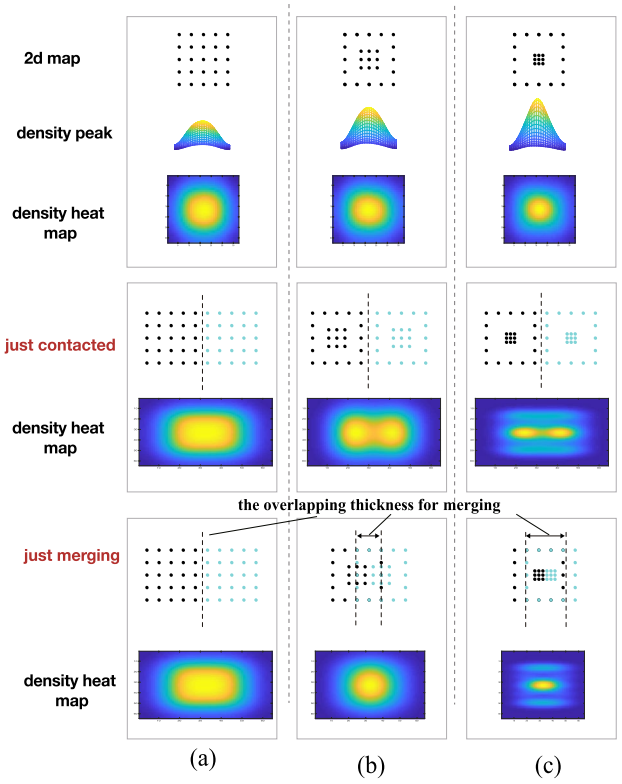


Fig. 9 Three cluster pairs of different distribution types and their merging processes

We assume that the merging ability of a sub-cluster is related to its structure. In order to verify this, we present Fig. 9 to show three clusters of different distribution types and their merging processes. These three types of clusters are all composed of 25 points with different distribution rules: Type_1 (Fig. 9a) is a uniform distribution of cluster, Type_2 (Fig. 9b) is a cluster where the center points are distributed relatively densely, and Type_3 (Fig. 9c) is a cluster where its center points are distributed most densely. To make it more visual, these three types of clusters are converted into three sharp density peaks, and their density heat maps are drawn as shown in Fig. 9. Figure 9 shows two status of $\text{Type}_{1,2,3}$'s merge process: (1) just contacted, (2) just merging. We note that in the same space and with the same quantity of data points, clusters with denser center distribution need a larger overlapping thickness to complete the merging process than clusters with the sparse center distribution. Therefore, to figure out whether two sub-clusters can be merged, except for the overlapping thickness, the sub-cluster structures also need to be considered.

Unlike ISODATA's cluster merging method that based on the distance between centers [26], our method considers the overlapping thickness and the merging ability of sub-clusters based on density, which enables it to deal with clusters of arbitrary shape.

3.2.1 The merging ability coefficient

We notice that sub-cluster with denser center distribution has a sharper density peak tip, and the sharpness of a density peak is related to its merging ability. Thus, we design parameter κ to define the sharpness of density peak, as shown in Eq. 9, where ρ_{SC_p} refers to the density of SC_p's center, $\rho_{\text{SC}_{\text{mean}}}$ refers to the average density of all points in sub-cluster SC_p, and N_{SC_p} refers to the total number of the points in sub-cluster SC_p. Based on the assumption that the merging ability M of sub-cluster is inversely proportional to its sharpness κ, we get Eq 10.

$$\kappa_{\text{SC}_p} = \frac{1}{N_{\text{SC}_p}} \times \sum_{i \in \text{SC}_p} \frac{\left| \rho_i - \rho_{\text{mean}_{\text{SC}_p}} \right|}{\rho_{\text{SC}_p}} \tag{9}$$

$$M_{\text{SC}_p} \times \kappa_{\text{SC}_p} = M_{\text{SC}_q} \times \kappa_{\text{SC}_q}, \text{SC}_p \ldots \text{SC}_q \in \text{dataset}. \tag{10}$$

$$M_{\text{SC}_p} = \frac{\kappa_{\text{SC}_{\text{Gauss}^n}}}{\kappa_{\text{SC}_p}} \times M_{\text{SC}_{\text{Gauss}^n}} \tag{11}$$

Based on Eq. 10, the merging ability M_{SC_p} can be converted to Eq. 11, where $\text{SC}_{\text{Gauss}^n}$ refers to the n-dimensional Gaussian distribution sub-cluster (n is the number of dimensions of the dataset), which functions as a reference body for the merging ability of n-dimensional sub-cluster.

3.2.2 The overlapping thickness coefficient

To define the overlapping thickness coefficient O, we first search for the highest density point on the boundary of two intersecting sub-clusters, which we call, the saddle point.

The saddle point $S_{\text{SC}_p\text{SC}_q}$ between sub-cluster SC_p and SC_q is defined in Eq. 12. $B_{\text{SC}_p\text{SC}_q}$ is the boundary point set of sub-cluster SC_p and SC_q, as in Eq. 13, where $C(i)$ means i's cluster label.

$$S_{\text{SC}_p\text{SC}_q} = \left\{ i | \rho_i = \max_{j \in B_{\text{SC}_p\text{SC}_q}} (\rho_j), i \in B_{\text{SC}_p\text{SC}_q} \right\} \tag{12}$$

$$B_{\text{SC}_p\text{SC}_q} = \left\{ i | C(i) \neq C(j), i,j \in \text{SC}_p \cup \text{SC}_q, j \in \text{KNN}_i, K = 5 \right\} \tag{13}$$

$$O_{\mathrm{SC}_p\mathrm{SC}_q} = \frac{\rho_{S_{\mathrm{SC}_p\mathrm{SC}_q}}}{\rho_{\mathrm{SC}_{ij}^{big}}} \tag{14}$$

We note that the larger the overlapping thickness between sub-clusters, the higher their saddle point density. Thus, we define the overlapping thickness coefficient $O_{\mathrm{SC}_p\mathrm{SC}_q}$ in Eq 14, where $\mathrm{SC}_{ij}^{\mathrm{big}}$ refers to the sub-cluster that contains more points between SC_p and SC_q.

As a result, we transform Eq. 8 into Eq. 15.

$$\mathrm{MF}_{\mathrm{SC}_p\mathrm{SC}_q} = \frac{1}{2} \times \left(\frac{\kappa_{\mathrm{SC}_{\mathrm{Gauss}^n}}}{\kappa_{\mathrm{SC}_p}} + \frac{\kappa_{\mathrm{SC}_{\mathrm{Gauss}^n}}}{\kappa_{\mathrm{SC}_q}} \right) \times \frac{\rho_{S_{\mathrm{SC}_p\mathrm{SC}_q}}}{\rho_{\mathrm{SC}_{ij}^{\mathrm{big}}}} \tag{15}$$

Herein, we set $M_{\mathrm{SC}_{\mathrm{Gauss}^n}} = 1$, and the value of $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^n}}$ is in a fixed range, which will be verified in the following paragraphs.

3.2.3 Derivation of $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^n}}$

Since $\mathrm{SC}_{\mathrm{Gauss}^n}$'s each dimension is an independent normal distribution, $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^n}}$ equals to $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}}$, so the only thing we need to derive is that $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}}$ is a fixed value.

We normalized the distribution of Gauss^1 to $N(0, 1)$ and its probability distribution density function is shown as Eq. 16. In addition, the density estimation can be transformed into a continuous integration method, as in Eq. 17, where x_i is the coordinate of point i on the X-axis. So, the average density $\rho_{\mathrm{mean}_{\mathrm{Gauss}^1}}$ is defined as Eq. 18.

When $x_i = 0$, we can get the center density of Gauss^1 which denoted as $\rho_{\mathrm{Center}_{\mathrm{Gauss}^1}}$, as in Eq. 19.

$$P(x) = \frac{1}{\sqrt{2\pi}} \times \exp\left(\frac{-x^2}{2}\right) \tag{16}$$

$$\begin{aligned} \rho_{i_{\mathrm{Gauss}^1}} &= \int_{-\infty}^{+\infty} N \times P(x_i) \times \exp\left(\frac{-(x-x_i)^2}{d_c^2}\right) \mathrm{d}x \\ &= N \times \frac{d_c}{\sqrt{d_c^2+2}} \times \exp\left(-\frac{x_i^2}{d_c^2+2}\right) \end{aligned} \tag{17}$$

$$\begin{aligned} \rho_{\mathrm{mean}_{\mathrm{Gauss}^1}} &= \int_{-\infty}^{+\infty} N \times P(x_i) \times \rho_{i_{\mathrm{Gauss}^1}} \mathrm{d}x_i \\ &= N \times \frac{d_c}{\sqrt{d_c^2+4}} \end{aligned} \tag{18}$$

$$\rho_{\mathrm{Center}_{\mathrm{Gauss}^1}} = N \times \frac{d_c}{\sqrt{d_c^2+2}}. \tag{19}$$

$$\begin{aligned} \kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}} &= \frac{1}{N} \int_{-\infty}^{+\infty} N \times P(xi) \frac{|\rho_{i_{\mathrm{Gauss}^1}} - \rho_{\mathrm{mean}_{\mathrm{Gauss}^1}}|}{\rho_{\mathrm{Center}_{\mathrm{Gauss}^1}}} \mathrm{d}x_i \\ &2 \times \int_0^{x_{\mathrm{mean}}} P(xi) \frac{\rho_{i_{\mathrm{Gauss}^1}} - \rho_{\mathrm{mean}_{\mathrm{Gauss}^1}}}{\rho_{\mathrm{Center}_{\mathrm{Gauss}^1}}} \mathrm{d}x_i \\ &- 2 \times \int_{x_{\mathrm{mean}}}^{\infty} P(xi) \frac{\rho_{\mathrm{mean}_{\mathrm{Gauss}^1}} - \rho_{i_{\mathrm{Gauss}^1}}}{\rho_{\mathrm{Center}_{\mathrm{Gauss}^1}}} \mathrm{d}x_i \\ &= 2 \times \frac{\mathrm{erf}\left(\sqrt{\frac{d_c^2+4}{2\times(d_c^2+2)}} \times x_i\right) - \mathrm{erf}\left(\frac{1}{\sqrt{2}} \times x_i\right)}{\sqrt{(d_c^2+2)\times(d_c^2+4)}} \Bigg|_0^{x_{\mathrm{mean}}} - \\ &2 \times \frac{\mathrm{erf}\left(\sqrt{\frac{d_c^2+4}{2\times(d_c^2+2)}} \times x_i\right) - \mathrm{erf}\left(\frac{1}{\sqrt{2}} \times x_i\right)}{\sqrt{(d_c^2+2)\times(d_c^2+4)}} \Bigg|_{x_{\mathrm{mean}}}^{+\infty} \\ &= 4 \times \frac{\mathrm{erf}\left(\sqrt{\frac{d_c^2+4}{2\times(d_c^2+2)}} \times x_{\mathrm{mean}}\right) - \mathrm{erf}\left(\frac{1}{\sqrt{2}} \times x_{\mathrm{mean}}\right)}{\sqrt{(d_c^2+2)\times(d_c^2+4)}} \end{aligned} \tag{20}$$

$$x_{\mathrm{mean}} = \frac{\sqrt{(d_c^2+2)\times(\ln(d_c^2+4) - \ln(d_c^2+2))}}{\sqrt{2}} \tag{21}$$

$$\kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}} = \sqrt{2} \times \left(\mathrm{erf}(\sqrt{ln2}) - \mathrm{erf}\left(\frac{\sqrt{\ln 2}}{\sqrt{2}}\right)\right) \approx 0.2349 \tag{22}$$

Therefore, $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}}$ is shown in Eq. 20, where x_{mean} is obtained when$\rho_{x_{\mathrm{mean}}} = \rho_{\mathrm{mean}_{\mathrm{Gauss}^1}}$, as in Eq. 21.

Since $d_c \to 0$, we substitute $d_c = 0$ into Eq. 21 to get $x_{\mathrm{mean}} = \sqrt{\ln 2}$, then substitute $d_c = 0, x_{\mathrm{mean}} = \sqrt{\ln 2}$ into Eq. 20 to obtain $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}}$ as in Eq. 22. Thus, $\kappa_{\mathrm{SC}_{\mathrm{Gauss}^n}} = \kappa_{\mathrm{SC}_{\mathrm{Gauss}^1}} \approx 0.2349$.

4 Clustering process of NM-DPC

This section presents the clustering process of NM-DPC and theoretically analyze the clustering performance of our algorithm.

NM-DPC first generates sub-clusters by generation strategy (Algorithm 1), that is, assigns each point into the same sub-cluster of its nearest normal neighbor with a high density. Then, NM-DPC defines the overlapping thickness coefficient and Merging force between each pair of sub-clusters. Followed, according to a merging threshold $\lambda(\lambda \in [0, 1])$, sub-cluster pairs with overlapping thickness coefficient O larger than λ are directly merged into transition sub-clusters (as Eq. 24); subsequently,

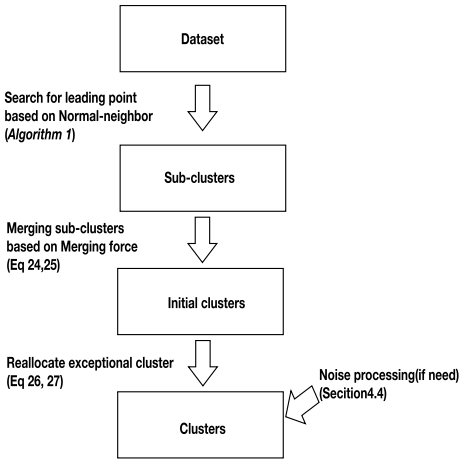


Fig. 10 NM-DPC's process diagram

transition sub-clusters with Merging force *MF* larger than λ are automatically merged into transition sub-clusters into final clusters (as Eq. 25). Figure 10 shows NM-DPC's process diagram.

4.1 Generate sub-clusters based on Normal-neighbor and density

The generation of sub-clusters in our algorithm is based on Normal-neighbor and density. The local density ρ_i of point i is defined by Eq. 3, we replace the DPC method of obtaining the "cutoff distance" d_c with a robust approach that defined in Eq. 23, where k' is the k'th nearest neighbor of point i. In this paper, a density parameter p is used to set the size of k', and its value is generally set to 2% of N (the total number of the data point in the dataset).

$$d_c = \text{mean}\left(\sum_{i\in\text{dataset}} d_{ii_{k'}}\right) \tag{23}$$

The generation steps of sub-clusters based on Normal-neighbor and density are as follows: firstly, each point i obtains its Normal-neighbor NN_i. Followed, each point i finds its leading point in NN_i, denote as i^{leading}, and if ρ_i is the highest in NN_i, point i will be considered as a sub-cluster center. Then, as each non-center point's assignment follows its leading point, the sub-clusters are naturally formed. This is called the generation strategy process, which is described in Algorithm 1.

Algorithm 1 Generation Strategy

procedure GENERATING SUB-CLUSTERS BASED ON DENSITY AND NORMAL-NEIGHBOR
  $Maxdist \leftarrow \infty$
  for $i \in Dataset$ **do**
    $d_{ii^{leading}} = Maxdist$
    for $j \in NN_i$ **do**
      if $\rho_j > \rho_i$ *and* $d_{ij} < d_{ii^{leading}}$ **then**
        $i^{leading} \leftarrow j$
        i's label$\leftarrow i^{leading}$'s label
        $d_{ii^{leading}} \leftarrow d_{ij}$
      end if
    end for
    if $d_{ii^{leading}} = Maxdist$ **then**
      $Subcluster\ center \leftarrow i$
    end if
  end for
  return $Subcluster\ Clusters$
end procedure

As mentioned above, Normal-neighbor is designed to ensure that neighbors of each point all belong to the same cluster, and thus, each sub-cluster generated by our generation strategy is guaranteed to be in the same cluster.

Nevertheless, since the number of sub-clusters always tends to be bigger than the real number of clusters in the dataset, some overlapping sub-clusters should be merged based on the merging relationship in-between.

4.2 The merging of sub-clusters based on the Merging force

After obtaining the Merging force coefficient between each pair of intersecting sub-cluster, we start the merging process of sub-clusters: The first step is to merge sub-clusters into transition sub-clusters, called *merging step 1*, as in Eq. 24; the second step is to merge transition sub-clusters into final clusters, called *merging step 2*, as in Eq. 25, where $\lambda(0 \leqslant \lambda \leqslant 1)$ is a merge threshold. After the two merge steps, the clustering is completed.

$$\text{If} \quad O_{\text{SC}_p\text{SC}_q} > \lambda \quad \text{Merge}, \quad \text{SC}_p, \text{SC}_q \in \text{dataset} \tag{24}$$

$$\text{If} \quad \text{MF}_{\text{SC}_p\text{SC}_q} > \lambda \quad \text{Merge}, \quad \text{SC}_p, \text{SC}_q \in \text{dataset} \tag{25}$$

Figure 11 illustrates the entire process of our method in dealing with the Jain dataset. As shown in Fig. 11a, our method successfully avoids 9 sub-cluster centers' jumping phenomenon (marked by a red cross in Fig. 11a), where two sub-cluster centers (A and B) eager to jump to the other cluster to get their leading points. To better explain how our Normal-neighbor method works, we zoom in detail in Fig. 11a that shows the point B's process of obtaining its normal neighbors. It can be noted

that B_{20} is the nearest abnormal neighbor (namely NAN_B) of point B due to its large B^{in}_{20}. Thus, no matter how large the K value of NN_B (i.e., $K \geqslant 20$ or $K \gg 20$), the NN_B will be restricted to the inner neighbors of B_{20} (within gray area), that is, the number of normal neighbors in NN_B will not larger than 19. So point B cannot jump to sub-cluster C that exceeds the range of NN_B to obtain its leading point B^{leading} (marked by a black circle), as a result, point B without a leading point is considered as a sub-cluster center.

The idea of our method is to initially obtain sub-clusters instead of misallocating sub-clusters due to the jumping phenomenon, and then merge sub-clusters into clusters as shown in Fig. 11b. As shown, we use two merging steps to achieve the final clustering: *merging step 1* that based on the coefficient O; *merging step 2* that based on the coefficient MF. By using $\lambda = 0.8$, *merging step 1* merges 9 sub-cluster into 7 transition sub-clusters which are finally merged into 2 cluster by **merging step 2**. This clustering result is perfect for the Jain dataset.

4.3 Exceptional cluster processing

After sub-clusters have been merged, there may leave some clusters with extremely few data points called exceptional clusters that need to be reprocessed.

To identify exceptional clusters, we design an exceptional cluster filter as in Eq. 26, where N_{C_i} is the total number of points of cluster C_i, and υ is an exceptional cluster filter threshold. If N_{C} is large than υ, we denote cluster C as C_{normal}.

$$\text{If} \quad N_{C_i} < \upsilon \qquad C_i \text{ is an exception cluster} \tag{26}$$

Next, point i in the exceptional clusters is denoted as i^e and assigned to the normal cluster that closest to it, as in Eq. 27.

$$C(i^e) = \left\{ C(j) | d_{ji^e} = \min_{j \in \text{all } C^{\mathrm{normal}}} (d_{ji^e}) \right\} \tag{27}$$

4.4 Noise processing

In terms of noise processing, our algorithm is similar to DPC. We average the density value of each cluster's boundary points, denoted as ρ^b. The point with smaller density than the ρ^b of its cluster is considered as noise.

In summary, the procedure of NM-DPC algorithm is presented as follows:

1. Calculate ρ for each point from Eq. 3;
2. Generate sub-clusters by generation strategy;
3. Calculate the Merging force coefficient MF for each pair of intersecting sub-clusters;
4. First merge the sub-clusters into transition sub-clusters according to Eq. 24, then merge transition sub-clusters into final clusters according to Eq. 25;
5. Exceptional cluster processing;
6. Clustering is accomplished.

4.5 Complexity analysis of NM-DPC

NM-DPC needs space to store the distance from each point to its K-nearest neighbors, and the recognition matrix of sub-clusters. The approximate space complexity of NM-DPC is $O(n^2)$.

The time complexity of NM-DPC depends on the following four parts: (a) the time for computing the distance between points is $O(n^2)$; (b) the time to calculate the local density ρ for each point is $O(n^2)$; (c) the time of obtaining the normal neighbors for each point is $O(n * K^2)$; (d) the time of generating sub-clusters is $O(n^2)$. Thus, the total approximate time complexity of NM-DPC is $O(n^2) + O(n * K^2)$. K is our input parameter that indicates the number of neighbors to obtain for each point and the impact of K value on the overall time complexity is weak since it is usually set as 20 which is much smaller than n.

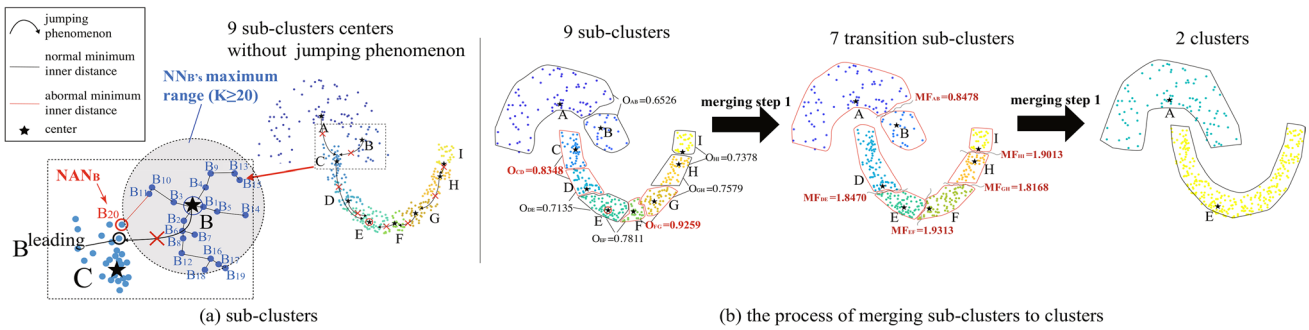


Fig. 11 The clustering process of our algorithm on the Jain dataset

4.6 Some discussions of NM-DPC

NM-DPC follows DPC's assumption that each cluster is regarded as a density peak, but NM-DPC does not use the ρ and δ to search for cluster centers. Compared with DPC's clustering method that first finds the cluster centers before assigning other points, NM-DPC prefers to generate sub-clusters from points, and merge sub-clusters into clusters, in other words, NM-DPC pays more attention to points rather than cluster centers, which enables it to have some outstanding clustering properties that different from DPC.

Generation strategy ensures that the assignment of each point will not be affected by points in other clusters, which helps all the sub-cluster centers in NM-DPC get away from the jumping phenomenon, and naturally avoid the bad influence of jumping phenomenon.

In addition, we propose the Merging force idea into the clustering process, which is beneficial to the automatic emergence of clusters without using the decision graph.

Through the above discussion, we can conclude that NM-DPC does have better clustering results compared to DPC, which will be demonstrated in Sect. 5.

5 Experimental results and analysis

We conducted experiments on synthetic datasets and Olivetti Faces dataset with the purpose of testing the efficiency of our algorithm. These datasets of different characteristics are commonly used to test the performance of clustering algorithms. The synthetic datasets used in experiments are displayed in Table 2. In this section, the performance of NM-DPC is compared with DPC, KNN-DPC [23], DBSCAN, K-Means, Spectral clustering (SC) [16], S-DPC(G) (i.e., the generic method proposed by [21] without training.) and SNN-DPC [24].

The clustering results are evaluated using four evaluation indices: adjusted mutual information (AMI) [34], adjusted Rand index (ARI) [34], Fowlkes–Mallows index (FMI) [35] and clustering accuracy (ACC). The upper bound of the four indicators is 1, where larger values indicate better clustering results.

Before experiments, data are preprocessed by the min-max normalization method in [36]. The parameter requirements of each algorithm are shown as follows: DBSCAN requires two parameters, the maximum radius ε and the minimum point MinPts; The value of cluster number k is indispensable for K-Means; DPC, KNN-DPC and S-DPC(G) need to set the density parameter p, SNN-DPC need to set the K to obtain neighbors. DPC, KNN-DPC, SNN-DPC and S-DPC(G) all need to select the cluster centers in the decision graph manually; our NM-DPC also requires the density parameter p, besides, the K and ε of Normal-neighbor need to be given, in addition, the merge threshold coefficient λ is set as 0.8, the exceptional cluster filter threshold υ is equal to 2% of the total number of the points in the dataset.

Table 2 Synthetic datasets

Dataset	Instances	Attributes	Clusters	Source
Spiral	312	2	3	[27]
Jain	373	2	2	[25]
Flame	240	2	3	[28]
Compound	399	2	6	[29]
Pathbased	300	2	3	[27]
Aggregation	788	2	7	[30]
S3	5000	2	15	[31]
D31	3100	2	31	[32]
R15	600	2	15	[32]
Eyes	238	2	2	[33]

These needed parameters of each algorithm during the experiments are displayed in the subsequent experimental results tables (Tables 3, 4). PAR in Tables 3 and 4 represents the parameter setting of algorithms such as $\mathrm{PAR}_{\text{NM-DPC}} = K/\varepsilon/p$, $\mathrm{PAR}_{\text{DBSCAN}} = \varepsilon/\text{MinPts}$, $\mathrm{PAR}_{\text{DPC}} = p$, $\mathrm{PAR}_{\text{K-Means}} = k$, $\mathrm{PAR}_{\text{KNN-DPC}} = p$, $\mathrm{PAR}_{\text{S-DPC(G)}} = p$, $\mathrm{PAR}_{\text{SNN-DPC}} = K$,and $\mathrm{PAR}_{\text{SC}} = k/\sigma$.

5.1 Synthetic datasets

In this part, a number of synthetic datasets that are widely used to test a variety of clustering algorithms are selected. Table 3 shows the clustering results in terms of the AMI, ARI, FMI and ACC scores on all synthetic datasets listed in Table 2. For K-Means and SC, the best experimental results are selected after multiple experiments.

Next, the clustering results of some synthetic datasets in the experiments will be presented in Fig. 12, where different colors indicate different clusters. Except for DBSCAN and K-Means, the cluster centers obtained from other algorithms are marked with black pentagrams, while black points indicate noise determined by DBSCAN.

Figure 12 and Table 3 show that NM-DPC has optimal clustering performance on almost all datasets

except for the compound dataset, and NM-DPC's performance is merely slightly different from DBSCAN. As shown in Fig. 12, for the compound dataset, we notice that DBSCAN treats sparse points as noise, including the sparse cluster on the right side, while NM-DPC merges two clusters on the right side into one, and that is why NM-DPC's accuracy on compound is lower than DBSCAN. It is worth mentioning that all algorithms' clustering results on the Eyes dataset are not perfect, since they cannot accurately identify the ring cluster. The reason why our algorithm fails to identify the ring cluster of the Eyes dataset is that our Normal-neighbor method cannot prevent the points on the sparse ring from jumping to the square dense clusters to find their leading points. This is because the distribution distance between points in the ring cluster is almost the same as the shortest distance between it and the square cluster, and our method cannot effectively detect abnormal

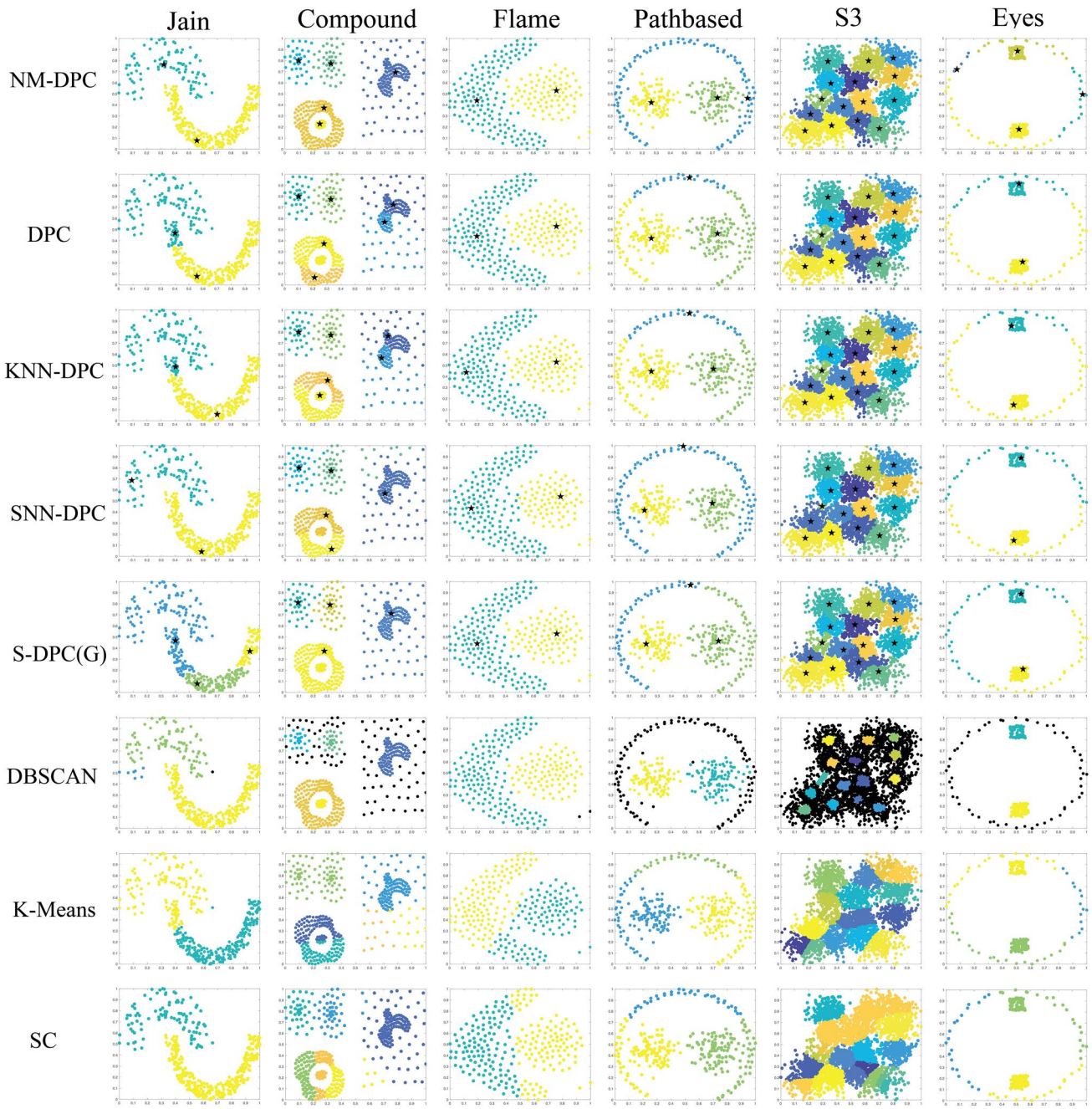


Fig. 12 The clustering results of 8 algorithms on some synthetic datasets

Table 3 The comparison of 8 clustering algorithms on synthetic datasets

Algorithm	AMI	ARI	FMI	ACC	PAR	AMI	ARI	FMI	ACC	PAR
	Spiral						*Jain*			
NM-DPC	**1**	**1**	**1**	**1**	20/2/2%	**1**	**1**	**1**	**1**	20/4/3%
DPC	**1**	**1**	**1**	**1**	5%	0.5396	0.6183	0.8386	0.8954	2%
KNN-DPC	**1**	**1**	**1**	**1**	5%	0.6183	0.7146	0.8819	0.9249	2%
SNN-DPC	**1**	**1**	**1**	**1**	10	**1**	**1**	**1**	**1**	20
S-DPC(G)	**1**	**1**	**1**	**1**	2%	0.2382	0.1277	0.5514	0.7855	2%
DBSCAN	**1**	**1**	**1**	**1**	0.04/2	0.8593	0.9756	0.9905	0.9905	0.08/4
K-Means	-0.006	-0.0055	0.3274	0.3494	3	0.4916	0.5767	0.82	0.882	2
SC	-0.0058	0.0006	0.3510	0.3429	3/2	**1**	**1**	**1**	**1**	2/2
	Compound						*Flame*			
NM-DPC	0.842	0.8531	0.8982	0.8722	20/2/2%	**1**	**1**	**1**	**1**	20/2/2%
DPC	0.6968	0.5461	0.6491	0.8321	2%	**1**	**1**	**1**	**1**	5%
KNN-DPC	0.6913	0.5329	0.6381	0.8321	2%	**1**	**1**	**1**	**1**	2%
SNN-DPC	0.7356	0.5775	0.6791	0.8296	20	0.8165	0.8854	0.9479	0.9708	15
S-DPC(G)	0.7563	0.7825	0.8547	0.8321	2%	**1**	**1**	**1**	**1**	2%
DBSCAN	**0.8714**	**0.9086**	**0.9321**	**0.9321**	0.05/5	0.8732	0.9550	0.9790	0.9917	0.09/8
K-Means	0.6761	0.5598	0.6599	0.8496	6	0.3648	0.4202	0.7201	0.8250	2
SC	0.6170	0.3796	0.5974	0.7393	6/2	0.4420	0.4880	0.8528	0.8500	2/2
	Pathbased						*Aggregation*			
NM-DPC	**0.9579**	**0.9699**	**0.9799**	**0.99**	15/2/2%	0.9892	0.9935	0.9949	0.9962	20/1/2%
DPC	0.4997	0.453	0.6585	0.7333	2%	**0.9922**	**0.9956**	**0.9966**	**0.9975**	4%
KNN-DPC	0.5294	0.4797	0.6703	0.76	3%	**0.9922**	**0.9956**	**0.9966**	**0.9975**	4%
SNN-DPC	0.9001	0.9294	0.9529	0.9767	9	0.9262	0.9272	0.9428	0.9607	20
S-DPC(G)	0.7073	0.6133	0.7511	0.8233	2%	0.9696	0.9749	0.9803	0.9848	2%
DBSCAN	0.871	0.9011	0.934	0.9667	0.08/10	0.9864	0.9913	0.9932	0.9949	0.5%
K-Means	0.5098	0.4613	0.6617	0.7433	3	0.8041	0.7114	0.7724	0.9112	7
SC	0.5607	0.4797	0.7209	0.7600	3/2	0.8015	0.6718	0.8571	0.8617	7/2
	S3						*D31*			
NM-DPC	0.9746	**0.966**	**0.9683**	**0.9832**	20/4/0.5%	0.9545	0.9345	0.9366	0.9674	20/2/0.5%
DPC	**0.9775**	0.9645	0.9669	0.979	1%	0.9539	0.9332	0.9354	**0.9684**	2%
KNN-DPC	0.9628	0.9522	0.9554	0.9738	1%	**0.9554**	**0.9364**	**0.9384**	**0.9684**	2%
SNN-DPC	0.8658	0.8033	0.8166	0.8986	40	0.9589	0.9415	0.9434	0.9710	30
S-DPC(G)	0.8826	0.8302	0.8418	0.9098	2%	0.9552	0.9353	0.9374	0.9677	2%
DBSCAN	0.448	0.0859	0.248	0.496	0.02/30	0.8895	0.8078	0.8186	0.8287	0.04/38
K-Means	0.9001	0.8723	0.8809	0.9344	15	0.9305	0.86	0.8655	0.9152	31
SC	0.8417	0.7100	0.8127	0.8100	15/300	0.9064	0.7012	0.8270	0.8174	31/2
	R15						*Eyes*			
NM-DPC	**0.9938**	**0.9928**	**0.9932**	**0.9967**	20/2/2%	0.6130	0.6698	0.8007	0.8487	20/2/2%
DPC	**0.9938**	**0.9928**	**0.9932**	**0.9967**	2%	0.4933	0.5797	0.7672	0.7647	3%
KNN-DPC	**0.9938**	**0.9928**	**0.9932**	**0.9967**	3%	0.4933	0.5797	0.7672	0.7647	3%
SNN-DPC	**0.9938**	**0.9928**	**0.9932**	**0.9967**	20	0.4905	0.5844	0.7674	0.7647	10
S-DPC(G)	0.9885	0.9857	0.9866	0.9933	2%	0.4926	0.5873	0.7689	0.7647	2%
DBSCAN	0.8755	0.7847	0.8007	0.9150	0.02/3	0.5992	0.6138	0.7979	0.7647	0.04/3
K-Means	0.9329	0.8816	0.8901	0.9217	15	0.5730	0.6434	0.7892	0.8235	3
SC	0.8550	0.5024	0.7307	0.7317	15/2	**0.6711**	**0.6877**	**0.8262**	**0.8529**	3/2

The best values are highlighted

Table 4 The comparison of 8 clustering algorithms on Olivetti faces dataset

Algorithm	AMI	ARI	FMI	ACC	Clusters	PAR
NM-DPC	0.7982	**0.6423**	**0.6593**	**0.775**	40	7/1.47/0.8%
	0.8039	**0.7288**	**0.7355**	**0.875**	48	7/1.1/0.8%
DPC	0.7657	0.6211	0.6356	0.74	40	0.4%
	0.7889	0.6438	0.653	0.7925	48	0.4%
KNN-DPC	0.7287	0.5215	0.5498	0.725	40	1%
	0.7744	0.6127	0.6223	0.805	48	1%
SNN-DPC	0.7650	0.6231	0.6402	0.7375	40	5
	0.7919	0.6422	0.6524	0.7800	48	5
S-DPC(G)	0.7570	0.5758	0.6015	0.7050	40	0.8%
	0.7564	0.5606	0.5797	0.7175	40	0.8%
DBSCAN	0.0714	0.0052	0.1289	0.255	40	0.3/2
K-Means	0.7208	0.5749	0.5888	0.715	40	40
SC	**0.8221**	0.4925	0.6512	0.6650	40	40/10

The best values are highlighted

neighbors (in the square cluster) of points in the ring cluster. This is also the reason why our algorithm merges two clusters on the right side of the compound dataset.

5.2 Olivetti faces dataset

The Olivetti Faces dataset [37], which includes 40 face images of different people, each with 10 different face angles, is a widely used database in the machine learning field.

In this experiment, for NM-DPC, we not only test the performance according to 40 clusters but also the performance of the best result situation (48 clusters). For DPC, KNN-DPC, SNN-DPC and S-DPC(G), we test the performance of selected 40 clusters and the performance of 48 clusters. For DBSCAN, K-Means and SC, we only show their best results of 40 clusters.

The results of all tested algorithms on the Olivetti faces dataset are shown in Table 4. As shown, the ARI, FMI and ACC metrics of NM-DPC are remarkably higher than other algorithms. In the best case where the NM-DPC selects 48 clusters, the AMI, ARI, FMI and ACC values of NM-DPC are 0.875, 0.8039, 0.7288 and 0.7355, respectively, which are still higher than all the comparing algorithms.

5.3 Run time comparison of algorithms

Table 5 shows the run time of our NM-DPC and some other comparison algorithms in seconds on ten tested synthetic datasets. We have analyzed the complexity of NM-DPC in Sect. 4.5, knowing that NM-DPC has the approximate computational complexity of $O(n^2) + O(n * K^2)$. From the experimental results, we get that the calculation time of NM-DPC is not necessarily longer than that of DPC, and for Spiral, Jain, Flame, Pathbased datasets, NM-DPC is even faster than DPC.

Table 5 Run time of NM-DPC and some comparative algorithms on some synthetic datasets (unit: second)

Dataset	NM-DPC (s)	DPC (s)	KNN-DPC (s)	DBSCAN (s)	K-Means (s)
Pathbased (300 instances)	0.355	0.396	0.614	0.180	0.080
Flame (240 instances)	0.182	0.404	0.543	0.207	0.079
Spiral (312 instances)	0.289	0.456	0.783	0.109	0.069
Jain (373 instances)	0.427	0.439	0.749	0.216	0.070
Compound (399 instances)	0.521	0.513	0.651	0.172	0.088
Aggregation (788 instances)	0.571	0.533	0.685	0.178	0.078
D31 (3100 instances)	4.173	2.897	2.865	0.235	0.097
S3 (5000 instances)	9.895	5.249	8.063	0.460	0.111

5.4 The limitation of decision graph

One of NM-DPC's advantages is that clusters can be naturally emerged without using the decision graph. Although the decision graph can help us visually discover cluster centers, it cannot always show clearly.

Herein, we present Fig. 13, the decision graphs of DPC, KNN-DPC, SNN-DPC [24], S-DPC(G) [21] on dataset Jain, where the red point indicates the misselected cluster center, and the green point indicates the true cluster center, to show the limitation of the decision graph. We can easily observe that the points in the upper right corner of the decision graph are most likely to be selected as cluster centers, and the actual left-side cluster center will be missed, which results in poor clustering results on Jain.

The above examples verify that the decision graph cannot always show the correct cluster centers clearly, and may even mislead the selection of correct cluster centers.

5.5 The evaluation of the sensitivity of NM-DPC's parameters

There are four parameters in NM-DPC: the merge threshold coefficient λ, the density parameter p, the K, and ε of Normal-neighbor. Table 6 clearly presents NM-DPC's input parameters and how to set them.

In NM-DPC, λ is fixed to 0.8, thus only p, K, and ε need to be set. As can be observed in the above experiments, p, K and ε are easy to be set, except for parameter ε which needs to be adjusted from 1 to 5, K is basically 20, and p is basically 2%.

Table 6 Input parameters of NM-DPC

Parameter	Meaning	Setting (default)
$\lambda \in [0, 1]$	The merge threshold	$\lambda = 0.8$
$\varepsilon \in \{1, 2, 3, 4, 5\}$	The anti-jump threshold	$\varepsilon = 2$
p	The density parameter	$p = 2\%$
K	The number of neighbors to search	$K = 20$

To demonstrate the robustness of our parameters, Table 7 displays the AMI values of some synthetic datasets using different K or ε or p, respectively.

As shown in the upper table of Table 7, when parameter ε and p are appropriate, changing the size of ε parameter K can hardly affect the clustering results, which benefits from the anti-jump threshold constant ε in Normal-neighbor that essentially determines the upper limit of K. Despite K takes the highest value, N (the total number of data points in the dataset), Normal-neighbor will limit the K size to satisfy normal neighbors.

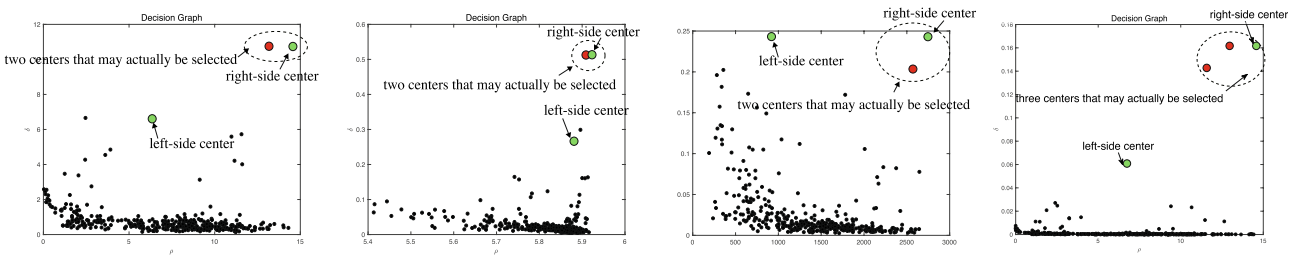
As shown in the bottom left table of Table 7, when parameter K and p are appropriate, changing the size of ε hardly impacts the clustering results.

As shown in the bottom-right table of Table 7, when parameter K and ε are appropriate, changing the size of p also hardly affects the clustering results.

The above experiments verify that the sensitivity of the three parameters of our algorithm is low.

6 Conclusion

NM-DPC inherits DPC's feature that each cluster center is regarded as a density peak. To avoid the jumping phenomenon, NM-DPC first generates sub-clusters by Generation Strategy, which can obtain the local structure information of the point to ensure that the points in the sub-cluster belong to the same cluster. To break the limitation of the decision graph, NM-DPC obtains the clusters by adaptively merging the sub-clusters according to their Merging force that is not involved in DPC. NM-DPC is able to perform clustering completely automatically. Compared to DPC, NM-DPC is more suitable to process multi-peak, multi-density cluster datasets of complex shapes.



(a) DPC on Jain (b) KNN-DPC on Jain (c) SNN-DPC on Jain (d) S-DPC(G) on Jain

Fig. 13 The decision graphs of DPC, KNN-DPC, SNN-DPC and S-DPC(G) on the Jain dataset

Table 7 AMI values of synthetic datasets using different K or ε or p

Dataset	K													
	1%N	5%N	10%N	15%N	20%N	25 %N	30%N	40%N	50%N	60%N	70%N	80%N	100%N	
Spiral ($\varepsilon = 2, p = 2\%$)	-	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	
Compound ($\varepsilon = 2, p = 2\%$)	-	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	**0.842**	
R15 ($\varepsilon = 3, p = 2\%$)	0.9779	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	
Aggregation ($\varepsilon = 1, p = 2\%$)	0.9536	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	

Dataset	ε					Dataset	p				
	1	2	3	4	5		1%	2%	3%	4%	5%
Spiral ($K = 20, p = 2\%$)	0.8692	**1**	**1**	**1**	**1**	Spiral ($K = 20, \varepsilon = 2$)	0.676	**1**	**1**	**1**	**1**
Compound ($K = 20, p = 2\%$)	0.8557	**0.842**	**0.842**	**0.842**	**0.842**	Compound ($K = 20, \varepsilon = 2$)	0.8557	**0.842**	**0.842**	**0.842**	0.7576
R15 ($K = 20, p = 2\%$)	**0.9938**	**0.9938**	**0.9938**	**0.9938**	**0.9938**	R15 ($K = 20, \varepsilon = 3$)	0.9866	**0.9938**	**0.9938**	0.9605	0.8783
Aggregation ($K = 10, p = 2\%$)	**0.9922**	**0.9922**	**0.9922**	**0.9922**	**0.9922**	Aggregation ($K = 20, \varepsilon = 1$)	**0.9922**	**0.9922**	**0.9922**	**0.9922**	0.919

The best values are highlighted

The experimental results on classical synthetic datasets and the Olivetti Faces dataset show that non-center points are assigned to the appropriate cluster by the generation strategy, and NM-DPC can find cluster centers accurately without referring the decision graph. Furthermore, NM-DPC is not sensitive to its parameters, which also makes it a robust clustering algorithm.

However, the clustering performance of NM-DPC in the multidimensional datasets does not show outstanding advantages, and the robustness of the density estimation method needs to be solved. For future work, we still need to find an efficient way to adaptively estimate density and improve clustering performance for a multidimensional dataset.

Acknowledgements This work was supported by the National Science Foundation of P.R. China (Grants: 61873239) and Zhejiang Science Foundation (Grant:2020C03074).

References

1. Jain AK (1999) Data clustering: a review. ACM Comput Surv 31(3):264–323
2. Hansen P, Jaumard B (1997) Cluster analysis and mathematical programming. Math Program 79(1–3):191–215
3. Xu R, Wunsch D II (2007) Computational intelligence in clustering algorithms, with applications
4. Xu R, Wunsch DC (2010) Clustering algorithms in biomedical research: a review. IEEE Rev Biomed Eng 3:120–154
5. Jain AK, Dubes RC (1988) Algorithms for clustering data. Technometrics 32(2):227–229
6. X Qian, Y Wu, M Li, Y Ren, S Jiang, Z Li (2020) LAST: location-appearance-semantic-temporal clustering based POI summarization. IEEE Trans Multimed
7. Wu Z, Leahy Richard M (1993) An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. IEEE Trans Pattern Anal Mach Intell 15(11):1101–1113
8. Berry Michael W, Castellanos Malu (2007) Survey of text mining: clustering, classification, and retrieval. Springer, Berlin
9. Jordan MI, Mitchell TM (2015) Machine learning: trends, perspectives, and prospects. Science 349(6245):255–260
10. Macqueen J (1965) Some methods for classification and analysis of multivariate observations. In: Proceedings of Berkeley symposium on mathematical statistics and probability
11. Jain AK (2008) Data clustering: 50 years beyond k-means. In: Machine learning and knowledge discovery in databases
12. Rahman MA, Islam MZ (2014) A hybrid clustering technique combining a novel genetic algorithm with K-Means. Knowl Based Syst 7:1345–365
13. Tzortzis G, Likas A (2014) The MinMax K-Means clustering algorithm. Pattern Recognit 47(7):2505–2516
14. Likas A, Vlassis N, Verbeek JJ (2003) The global K-Means clustering algorithm. Pattern Recognit 36(2):451–46
15. Xie J, Jiang S, Xie W, Gao X (2011) An efficient global K-Means clustering algorithm. JCP 6(2):27–279
16. Von Luxburg U (2007) A tutorial on spectral clustering. Stat Comput 17(4):395–416
17. Ester M, Kriegel HP, Xu X, Sanders J (1996) A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: International conference on knowledge discovery and data mining
18. Han J, Kamber M (2006) Data mining: concepts and techniques. In: Data mining concepts models methods and, 2nd edn, vol 5, no 4, pp 1–18
19. Xu R, Wunsch DC (2005) Survey of clustering algorithms. IEEE Trans Neural Netw 16(3):645–678
20. Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science 344:1492–1496
21. Pizzagalli Diego Ulisse, Gonzalez Santiago F, Krause Rolf (2019) A trainable clustering algorithm based on shortest paths from density peaks. Sci Adv 5(10):eaax3770
22. Xie J, Gao H, Xie W, Liu X, Grant PW (2016) Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors. Inf Sci 354:19–40
23. Du M, Ding S, Jia H (2016) Study on density peaks clustering based on k-nearest neighbors and principal component analysis. Knowl Based Syst 99:135–145
24. Liu R, Wang H, Yu X (2018) Shared-nearest-neighbor-based clustering by fast search and find of density peaks. Inf Sci 450:200–226
25. Jain AK, Law MH (2005) Data clustering: a user's dilemma. In: International conference on pattern recognition and machine intelligence, pp 1–10
26. Ball GH, Hall DJ (1965) ISODATA, a novel method of data analysis and pattern classification. Stanford Research Iinst, Menlo Park CA
27. Chang H, Yeung D-Y (2008) Robust path-based spectral clustering. Pattern Recognit 41(1):191–203
28. Zahn CT (1971) Graph-theoretical methods for detecting and describing gestalt clusters. IEEE Trans Comput 100(1):68–86
29. Fu L, Medico E (2007) FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data. BMC Bioinform 8(1):1–15
30. Gionis A, Mannila H, Tsaparas P (2007) Clustering aggregation. ACM Trans Knowl Discov Data 1(1):4
31. Frnti P, Virmajoki O (2006) Iterative shrinking method for clustering problems. Pattern Recognit 39(5):761–775
32. Veenman CJ, Reinders MJT, Backer E (2002) A maximum variance cluster algorithm. IEEE Trans Pattern Anal Mach Intell 24(9):1273–1280
33. L Zelnikmanor, P Perona (2004) Self-tuning spectral clustering. Neural Inf Process Syst
34. Vinh HX, Epps J, Bailey J (2010) Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. J Mach Learn Res 11(1):2837–2854
35. Fowlkes EB, Mallows CL (1983) A method for comparing two hierarchical clusterings. J Am Stat Assoc 78(383):553–569
36. Franti Pasi, Virmajoki Olli, Hautamaki Ville (2006) Fast agglomerative clustering using a k-nearest neighbor graph. IEEE Trans Pattern Anal Mach Intell 28(11):1875–1881
37. Samaria FS, Harter AC (1994) Parameterisation of a stochastic model for human face identification. In: Proceedings of the second IEEE workshop on applications of computer vision, pp 138–142