Density Peaks Clustering based on Nature Nearest Neighbor and Multi-cluster Mergers

Hui Ma (✉️ 1175017782@qq.com)  
Huzhou University  https://orcid.org/0000-0003-4603-0099

Ruiqin Wang  
Huzhou University

Shuai Yang  
Huzhou University

Research Article

**Keywords:** Density Peaks, Nature Nearest Neighbor, Shared Nearest Neighbor, Local Density

**Posted Date:** November 22nd, 2021

**DOI:** https://doi.org/10.21203/rs.3.rs-825405/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
Density Peaks Clustering based on Nature Nearest Neighbor and Multi-cluster Mergers

Hui Ma¹ · Ruiqin Wang¹,²,* · Shuai Yang¹

¹School of Information Engineering, Huzhou University, Huzhou, China
²Zhejiang Province Key Laboratory of Smart Management & Application of Modern Agricultural Resource, Huzhou, China

Abstract
Clustering by fast search and find of Density Peaks (DPC) has the advantages of being simple, efficient, and capable of detecting arbitrary shapes, etc. However, there are still some shortcomings: 1) the cutoff distance is specified in advance, and the selection of local density formula will affect the final clustering effect; 2) after the cluster centers are found, the assignment strategy of the remaining points may produce “Domino effect”, that is, once a point is misallocated, more points may be misallocated subsequently. To overcome these shortcomings, we propose a density peaks clustering algorithm based on natural nearest neighbor and multi-cluster mergers. In this algorithm, a weighted local density calculation method is designed by the natural nearest neighbor, which avoids the selection of cutoff distance and the selection of the local density formula. This algorithm uses a new two-stage assignment strategy to assign the remaining points to the most suitable clusters, thus reducing assignment errors. The experiment was carried out on some artificial and real-world datasets. The experimental results show that the clustering effect of this algorithm is better than those other related algorithms.

Keywords: Density Peaks · Nature Nearest Neighbor · Shared Nearest Neighbor · Local Density

1 Introduction
Clustering analysis is a significant technique in data mining (Frey et al. 2007; Han et al. 2011). It is the grouping of similar data objects into the same cluster by attributes or features to find principles, knowledge, or rules that have existed but not been found (Rui et al. 2005; Guyon et al. 2012). It has been widely employed in many areas, such as image processing, business intelligence, and machine learning (Pappas 1992; Guo et al. 2009; Leong et al. 2017).

Because of the extensive application of clustering, scholars have researched clustering and put forward many clustering algorithms. According to the characteristics of different clustering, clustering include partition clustering (Kaufman 2008), hierarchical clustering (Dasgupta et al. 2005), density clustering (Kriegel et al. 2011), grid-based clustering (Deng et al. 2018), and model-based clustering (Fahad et al. 2014). K-means algorithm (Jain 2009) is a well-known partition clustering algorithm. It is brief and efficient, but the number of clusters needs to be fixed in advance. Moreover, the random selection of the initial cluster centers will easily lead the result of clustering into the optimal local solution. DBSCAN algorithm (Esté et al. 1996) is a widely used density clustering, which improves the clustering speed, but the clustering result is sensitive to neighborhood radius and threshold. The proposal of the
The OPTICS algorithm (Ankerst et al. 1999) partly solves the problem of sensitive input parameters of DBSCAN algorithm. Rodríguez et al. (2014) first proposed DPC algorithm. The algorithm is efficient, can detect any shape of clusters, and does not need to specify the number of clusters. At present, DPC algorithm has huge potential application value in image segmentation, medical image, community discovery, etc. However, the DPC algorithm also has a few insufficient: 1) manual selection of the cutoff distance and subjective selection of the local density formula have relatively large impact on the selection of the initial cluster centers; 2) assignment strategy of remaining points is easy to form the consecutive assignment error. A point assignment error may lead to a series of subsequent point assignment errors, which will affect the effect of clustering.

Due to the above shortcomings, many scholars have made improvements to the DPC algorithm. Aiming at selecting cutoff distance and the definition of local density: the DPC-KNN-PCA algorithm (Du et al. 2016) was proposed to replace the selection of cutoff distance with K-nearest neighbors. Yang et al. (2018) achieved the adaptive cutoff distance to propose a cutoff distance adaptive algorithm based on information entropy. Liu et al. (2020) use the reverse natural nearest neighbor to optimize the local density formula to find the initial cluster centers better and get the clustering result better. Wu et al. (2020) redefine the method of selecting initial cluster centers by considering the weight of local density and relative distance. Fang et al. (2020) proposed the CFDP algorithm by establishing the connection between the cutoff distance and the data, which calculates local density and clustering adaptively. Lv et al. (2021) proposed using geodetic distance and dynamic neighborhood to measure the similarity between points. It has a nice clustering effect in manifold datasets. Tang et al. (2021) proposed the NNN-DPC algorithm. In the NNN-DPC algorithm, Local density is obtained through natural nearest neighbor, and the process is parameterless.

For the consecutive assignment error in the assignment: Xie et al. (2016) proposed the FKNN-DPC algorithm, which divides the remaining points into core points, ordinary points, and outliers, and gives them different sequences and weights, respectively, and then assigns them. The DPCSA algorithm (Yu et al. 2019) was proposed. It introduces a weighted local density sequence and a two-stage assignment strategy, improves efficiency by designing the nearest neighbor dynamic table. This algorithm breaks the local density decreasing assignment sequence and reduces the propagation of assignment errors. Wu et al. (2019) proposed the DPC-SNR algorithm to calculate K-nearest neighbors and reverse K-nearest neighbors for each point, thereby drawing asymmetric neighborhood graph and using the symmetric neighborhood graph to assign the remaining points, thereby reducing assignment errors. Chen et al. (2020) defined the similarity between clusters to mergers clusters, thereby reducing the possibility of assignment errors.

For these shortcomings, we also propose a density peaks clustering algorithm based on natural nearest neighbor and multi-cluster mergers. In this algorithm, the local density is redefined applying the idea of natural nearest neighbor and weighting and considering the relationship between the whole and the part. Furthermore, the probability of misassignment of the DPC algorithm is reduced by a two-stage optimization algorithm. The experimental results indicated that the effect of this algorithm is effective.
and robust.

The structure of this paper: In Sect.2, we introduce the DPC algorithm and its shortcomings. In Sect.3, the DPC++ algorithm is proposed. In Sect.4, we conduct an experimental comparison of the DPC++ algorithm with other related algorithms. Finally, we conclude in Sect. 5.

2 DPC Algorithm

In the part, we introduce the DPC algorithm and its shortcomings.

2.1 Algorithmic principle

The basic assumptions of the DPC algorithm are: 1) each cluster center is surrounded by low local density neighbors; 2) there is a relatively large distance between the cluster centers. For any point \( x_i \) in the dataset \( D \), firstly, the DPC algorithm calculate the local density and relative distance of point \( x_i \); then select the density peaks (i.e., cluster centers) by decision graph is formed by the local density and relative distance of each point; finally complete clustering according to the assignment strategy.

Local density \( \rho_i \) indicates the density around the point \( x_i \). There are two kinds of calculation equations: Cut-off kernel Eq. (1) and Gaussian kernel Eq. (2).

\[
\rho_i = \sum_{j \neq i} \chi(d_{ij} - d_i)
\]

\[
\rho_i = \sum_{j \neq i} \exp \left(-\left(\frac{d_{ij}}{d_i^c}\right)^\gamma\right)
\]

where \( d_{ij} \) is the Euclidean distance between point \( x_i \) and point \( x_j \); \( d_i^c \) is cutoff distance; the selection criteria of \( d_i^c \) is that the average number of neighbors per point is about 1% to 2% of the total number of points in the dataset; if \( d < d_i^c \), \( \chi(d) = 1 \), otherwise \( \chi(d) = 0 \).

Relative distance \( \delta_i \) refers to the minimum the distance between point \( x_i \) and point that have a higher local density than point \( x_i \) and is the closest distance to point \( x_i \), \( \delta_i \) be set as Eq. (3).

\[
\delta_i = \begin{cases} 
\min_{j \neq i} \{d_{ij} : \exists j, \rho_j > \rho_i \} & \text{if } \exists j, \rho_j > \rho_i \\
\max_{j \neq i} \{d_{ij} \} & \text{otherwise}
\end{cases}
\]

After obtaining \( \rho \) and \( \delta \) of all points, the DPC algorithm takes \( \rho_i \) and \( \delta_i \) of each point \( x_i \) as horizontal and vertical coordinates in the two-dimensional decision graph, to select the points with higher \( \rho \) and \( \delta \) to as the cluster centers. After the cluster centers have been found, each remaining point is assigned to the same cluster as its nearest neighbor of higher density than it.

2.2 Existing problems

The DPC algorithm has some shortcomings.

1) Choosing different cutoff distance and density formula, the clustering results are quite different. In the DPC algorithm, cutoff distance is randomly selected in a range, make the result of the density
formula is affected. At the same time, the DPC algorithm has two formulas to calculating local density. For larger datasets, the Cut-off kernel clustering is better, and for smaller datasets, the Gaussian kernel clustering is better. However, there are no objective criteria for measuring the size of a dataset.

In Fig. 1(a) and (b), the clustering effect of \( d_c = 3 \) is better than that of \( d_c = 2 \) when both using Gaussian kernel. In Fig. 1(b) and (c), clustering effect of the Gaussian kernel is better than that of the Cut-off kernel when using the same \( d_c \).

2) The assignment strategy of the DPC algorithm is easy to produce consecutive assignment errors. After the density peaks is found, the assignment strategy of DPC algorithm is easy to produce a “Domino effect”, that is, one point error may lead to a series of point errors. In Fig. 2, point \( a \) is a point of set \( s_{1,2,...,26} \), and has the Maximum local density. Both point \( b \) and point \( c \) have a greater local density than point \( a \). Point \( a \) should be in the same cluster with point \( c \), not point \( b \). However, the distance from point \( a \) to point \( b \) is smaller than the distance from point \( a \) to point \( c \). Therefore, point \( a \) is assigned to the same cluster as point \( b \), causing the point \( a \) to be misallocated. The point \( a \) is misallocated, all point in set \( s \) are misassigned. This assignment error is determined by both the assignment strategy and the definition of the local density of DPC algorithm.

(a) Gaussian kernel and \( d_c = 2 \)  
(b) Gaussian kernel and \( d_c = 3 \)  
(c) Cut-off kernel and \( d_c = 3 \)

Fig. 1 The results of clustering Flame dataset using the DPC algorithm

Fig. 2 Clustering of Spiral dataset using Gaussian kernel as the local density formula
3 DPC++ Algorithm

In this part, we propose DPC++ algorithm. In order to solve these questions, we proposed a density peaks clustering algorithm based on natural nearest neighbor and multi-cluster mergers (DPC++). This algorithm will improve the DPC algorithm in two aspects: 1) the local density is defined uniformly; 2) changing the assignment strategy of the remaining points after finding the cluster centers. The prime idea of this algorithm is to calculate the local density of each point based on natural nearest neighbors. After finding the cluster centers and the potential cluster centers, the remaining points are assigned to correct clusters using shared nearest neighbor and finite iterations (Zhu et al. 2014; Zhu et al. 2016; Levent et al. 2003).

3.1 Nature Nearest Neighbor

In the study of data structure, the concept of the nearest neighbors is proposed, the most commonly used \( k \)-neighborhood and \( \delta \)-neighborhood. \( k \)-neighborhood represents the \( k \) points that are closest to the target point in all the points. \( \delta \)-neighborhood represents all points within the \( \delta \) distance range of the target point. The natural nearest neighbor is an adaptive new neighbor relationship, in which the natural neighbors of each point is automatically generated by the algorithm. It makes points distributed in serried areas have more natural neighbors, and points distributed in discrete areas have fewer natural neighbors.

**Definition 1 (Nearest Neighbors)** \( rNN(i) \) represents the \( r \) nearest neighbors of the point \( x_i \), \( r \) is automatically generated by the algorithm.

**Definition 2 (Natural Nearest Neighbor)** \( NNN(i) \) represents the natural nearest neighbor of point \( x_i \). It is expressed as Eq. (4).

\[
NNN(i) = \{ x_j \in D | x_j \in N_j(i) \& x_j \in N_j(j), i \neq j \}
\]

It can be seen from the above definition that the natural nearest neighbor is asymmetric neighbor relationship, in which the value of the parameter \( r \) is only an intermediate variable in the natural nearest neighbor search algorithm. The termination condition of the search is: during two consecutive iterations, the number of outlier points (that is, points without natural neighbors) no longer changes.

**Algorithm 1** The natural nearest neighbor search algorithm

| Input: | Dataset \( D \) |
| Output: | \( NNN \) (set of the natural nearest neighbor of each point in the \( D \)) |
| \( r \) isStable \((r, r + 1)\): \( x \) indicates whether the number of outliers is stable. If the number of outliers has not changed from the previous time, returns True; otherwise returns False |
| 1: | \( r = 1 \) |
| 2: | for each \( x_i \) of \( D \): |
| 3: | calculate \( N_j(i) \) and \( NNN(i) \) |
| 4: | end for |
| 5: | while(True): |
| 6: | \( r = r + 1 \) |
| 7: | for each \( x_i \) of \( D \): |
| 8: | calculate \( N_j(i) \) and \( NNN(i) \) |
| 9: | end for |
| 10: | if (isStable\((r, r + 1)\)) or ( \( \forall x_i \in D, NNN(i) \neq \emptyset \)): |
| 11: | return \( NNN \) |
| 12: | end if |
| 13: | end while |

The natural nearest neighbor search process starts with \( r=1 \), gets the \( r \) nearest neighbors of each point
in turn, and then records the number of times each point appears in the other point's $r$ nearest neighbors. Let $r=r+1$, repeat the previous operation. When there are no outliers in the dataset or the number of outliers is no longer reduced, the algorithm stops. The natural nearest neighbor search process is described in algorithm 1.

The natural nearest neighbor has great advantages: 1) compared with $k$-neighborhood and $\delta$-neighborhood, it can be generated without any parameters; 2) natural nearest neighbor can better and more accurately represent the neighbor relationship between points. Compared with $k$-neighborhood, natural nearest neighbor can avoid outliers with no real neighbors or less than $k$ neighbors forcing distant points to become neighbors, thus better reflect the true neighbor relationship. Compared $\delta$-neighborhood, natural nearest neighbor can avoid the circumstance which makes the neighbor relationship are not inelastic because of setting $\delta$, thus to better reflect the true neighbor relationship.

For example, $d_{ij} = d_{i\bar{j}}$, but $d_{i\bar{j}} < d_i$ and $d_{i\bar{j}} > d_j$, but $x_i$ belongs to the $\delta$-neighborhood of $x_j$, and $x_{\bar{j}}$ does not belong to the $\delta$-neighborhood of $x_i$.

### 3.2 Local density

We aim to solve the question of inconsistent rules of measurement for the local density of the DPC algorithm.

We redefined the local density based on the idea of weighting and natural nearest neighbor. Moreover, the definition of local density considers both local and global information.

First, in this paper, the idea of a natural nearest neighbor is incorporated join the design of local density. Because it can better express the proximity relationship between points, thus more accurately and naturally evaluate the local density of each point. Compared with the local density calculated by cutoff distance or the K-nearest neighbor method, this approach makes the local density of each point can be calculated adaptively without manual parameters, so it is very robust.

**Definition 3** (dependent neighbor) $BNN(i)$ is the dependent neighbor of point $x_i$. It represents that set of points that belong to nearest neighbors of point $x_i$, but do not belong to natural neighbors of point $x_i$. It is expressed as Eq. (5).

$$BNN(i) = \{x_j \in D \mid x_j \in NN(i) \& x_j \notin NNN(i)\}$$

(5)

**Definition 4** (local density $\rho_i$) A new definition of local density based on natural nearest neighbors. It is expressed as Eq. (6).

$$\rho_i = \sum_{j \in BNN(i)} \exp(-d_{ij}) + \sum_{j \in BNN(i)} \exp\left(-\frac{d_{ij}}{j - \kappa_i}\right) + \sum_{j \in BNN(i)} \exp\left(-\frac{d_{ij}}{s(j - \kappa_i)}\right)$$

(6)

$k_i$ is the number of the $NN(i)$; $j$ is the order that the distance from each point to point $x_i$, from near to far; $s$ is the empirical value, usually $s = 2$.

Second, the density formula in this paper also shows the idea of weighting, giving different weights according to the different importance of each neighbor. This paper classifies the neighbors of each point into two classes: the natural nearest neighbor and the dependent neighbor and gives them different
Finally, the third part of the formula calculates the density contribution of the point except for the nearest neighbors. This definition of local density takes into account both local and global information of the points. This way can avoid the situation that sometimes the selection of the cluster centers focuses on the local area, which makes the cluster centers deviate from the central area, leading to incorrect clustering results.

In Fig.3(b)and(c), the red circles express the cluster centers, the yellow circles express the potential cluster centers, and the black circles represents outliers. It is obvious from the red circles in this graph that the cluster centers of the DPC++ algorithm is easier to find. The potential cluster centers are obvious in the graph of the DPC++ algorithm, but the DPC algorithm is not easy to find. Easier to find potential cluster centers are very helpful for the assignment strategy of this algorithm. For the black circles, we can see clearly that the outlier is obvious in the DPC++ algorithm. However, the outliers are too dense to pick out the true outliers in the DPC algorithm.

![Fig.3 Flame Dataset and its clustering decision graph](image)

3.3 Two-stage assignment Strategy

Aiming at the consecutive assignment error of the DPC algorithm, a new two-phase assignment strategy is proposed. It can largely avoid this problem.

**Phase I:** Selecting potential cluster centers and forming multiple clusters.

After the cluster centers $c_c$ are selected, there are still some points in the decision graph, which have higher local density and relative distance than many common points. These points far from other peaks and are surrounded by points with low local density. These points are defined as the potential cluster centers $p_{cc}$. For any point $x_i$. It is expressed as Eq. (7).

$$
\text{s.t.} \sum_{j \in D} \delta_j > \frac{1}{2n} \sum_{j \in D} \delta_j \land \rho_i > w \sum_{j \in D} \rho_j, x_i \not\in c_c
$$

$w$ is a density weight, $w \in [0.2, 2]$, generally $w=0.5$. $w$ is highly robust when the scale of the dataset is huge.

Taking $p_{cc}$ and $c_c$ as density peaks, each remaining point is assigned to the same cluster as its nearest neighbor of higher density than it, and some clusters are formed by taking $p_{cc}$ and $c_c$ as peaks.

**Phase II:** Mergers clusters that $p_{cc}$ as the peaks into cluster $c_c$ as the peaks.

**Definition 5** (Shared Nearest Neighbors) $SNN_ij(i, j)$ is the shared nearest neighbor of point $x_i$ and
point $x_j$. It is expressed as Eq. (8).

$$SNN_j(i,j) = \{x_k \in D \mid x_k \in KNN_j(i) \& x_k \in KNN_j(j)\}$$

(8)

$KNN_i(i)$ represents $k$ points that are closest to point $x_i$.

**Definition 6** $c_{oi}$ is the set of points. It consists of $cc$ and points whose local density is bigger than $pcc_i$ in $pcc$. $pcc$ is any point in $pcc$. It is expressed as Eq. (9).

$$c_{oi} = \{x_j \in D \mid (x_j \in cc) \lor (x_j \in pcc \& \rho_j > \rho_i)\}$$

(9)

The second stage is a brief process: let $k = r$ (r is r of $NN_i(i)$), calculating the shared nearest neighbor between each $pcc_i$ and corresponding $c_{oi}$. Then record point $x_i$ in $c_{oi}$, make the number of $(i, j)_{SNN \ pcc}$ is the most and not equal to zero. If $x_i$ exists, the cluster of $pcc_i$ is merged into the cluster of $x_i$, then delete $pcc_i$ from $pcc$. After that, let $k = k + 0.02n (n$ is the length of a dataset), and repeat the previous step until $pcc$ is empty, second stage ends. The detailed process of second stage is described by algorithm 2.

| Algorithm 2 Phase II assignment strategy |
|-----------------|----------------|----------------|----------------|
| **Input**: potential cluster centers $pcc$, and cluster centers $cc$ |
| **Output**: clustering result |
| $^\mathbb{f}$getSet($pcc_i, c_{oi}$)is used to get some points in $c_{oi}$, which have the largest number of shared nearest neighbor with $pcc_i$, and if this number is equal to 0, an empty set is returned $^\mathbb{f}$/ |
| 1. $k = r$ |
| 2. while ($pcc \neq \emptyset$) |
| 3. for each $pcc_i$ of $pcc$ |
| 4. for each $x_{oi}$ of $c_{oi}$ |
| 5. Calculate $SNN_i(pcc_i,x_{oi})$ |
| 6. for each $pcc$ of $pcc$ |
| 7. set$_i = getSet(pcc_i, c_{oi})$ |
| 8. if $x_i$ is only one point of set$_i$ |
| 9. cluster of $pcc_i$ is merged into cluster of $x_i$ |
| 10. $pcc_i$ was removed from the $pcc$ |
| 11. end if |
| 12. if multiple points in the set$_i$ |
| 13. $x_i$ is the closest point to $pcc_i$ in the set$_i$ |
| 14. cluster of $pcc_i$ is merged into cluster of $x_i$ |
| 15. $pcc_i$ was removed from the $pcc$ |
| 16. end if |
| 17. end for |
| 18. $k = k + 0.02n$ |
| 19.end while |

**3.4 Time complexity analysis**

The time complexity of the DPC algorithm from calculation of distance matrix between points, local density, relative distance, and the assignment strategy. Their time complexity respectively is $O(n^2), O(n^2), O(n^2)$ and $O(n)$. Therefore, the total time complexity of DPC algorithm is $O(n^2)$.

The time complexity of the DPC++ algorithm in this paper consists of the following parts: calculation of the distance matrix $O(n^2)$; Sort each row of the distance matrix $O(n^2 \log n)$; calculate local density $O(n^2)$; calculate relative distance $O(n^2)$; Phase I is $O(n)$; Phase II is $O(mq^2)$. Among them, $m$ is the number of iterations. $m$ is a constant with a maximum value of 50 because $k$ increases the total dataset by 2% each time. $q$ is the number of potential cluster centers $pcc$, and $q \ll n$. Therefore, the total time complexity of the DPC++ algorithm is $O(n^2 \log n)$. Compared with DPC, the time complexity of the DPC++ algorithm is slightly higher. The increase of time complexity comes from the sorting of each row.
of the distance matrix. Although it costs lots of time, gets more precise local density and more good
Clustering result.

4 Experimental results and analysis

In the part, we conduct an experimental comparison of the DPC++ algorithm with other related
algorithms.

4.1 Datasets and Data pre-processing

For the purpose of verifying the performance of the DPC++ algorithm, artificial datasets and real-world
datasets are used for testing and evaluation. These datasets are described in detail in Table 1 and 2. In the
datasets of Table 1 and 2, the size, attributes, and classes of each dataset are different. Artificial datasets
have arbitrary shapes such as Spirals, Circles, Flames, Moon, and so on. Real-world datasets are derived
from UCI and are used to reflect real-world situations. The two datasets complement each other two
verify the effectiveness of the algorithm. In order to avoid the influence of the attribute value range, the
real-world datasets are normalized as Eq. (10), but the artificial datasets do not need to consider this
problem.

\[ x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \]  

(10)

| Table 1. Artificial datasets |
|-----------------------------|
| Dataset          | Size | Attributes | Classes |
| Aggregation       | 788  | 2          | 7       |
| Flame            | 240  | 2          | 3       |
| D15              | 600  | 2          | 15      |
| D31              | 3100 | 2          | 31      |
| Spiral           | 312  | 2          | 3       |
| Spiral, unbalance| 567  | 2          | 2       |
| Moon             | 500  | 2          | 2       |
| Circle           | 500  | 2          | 2       |

| Table 2. Real-world datasets |
|-----------------------------|
| Dataset    | Size | Attributes | Classes |
| dermatology| 366  | 34         | 6       |
| wdbc       | 569  | 30         | 2       |
| iris       | 150  | 4          | 3       |
| libras     | 360  | 91         | 15      |
| parkinsons | 195  | 23         | 2       |
| wine       | 178  | 13         | 3       |
| seeds      | 210  | 7          | 3       |
| segmentation| 2310 | 19        | 7       |

4.2 Parameter optimization of each algorithm

We introduce the source and parameter optimization of each algorithm.

In this paper, the DPC++ algorithms are compared with the DPC (Rodríguez et al. 2014), DPCSA
(Yu et al. 2019), CFDP (Fang et al. 2020), and DBSCAN (Ester et al. 1996) algorithms. The CFDP
and the DBSCAN algorithms are programmed concerning the original literature. The original code of the
DPC and the DPCSA algorithms is provided by the original author.

For guarantee the best clustering result of each algorithm, it is necessary to optimize the parameters
of each algorithm. DPC++ algorithm needs density weight $w$ to select the potential cluster centers, and the range of $w$ is [0.2,2], and $w$ is robust when the dataset size is huge. The DPC algorithm needs to select cutoff distance $d_c$ and needs to choose between two density formulas. The DPCS and CFDPC algorithms do not require parameters. The parameters required by DBSCAN algorithm are neighborhood radius and threshold, and the change of the parameters has a great influence on the result. After many experiments, all parameters are the best.

### 4.3 Clustering Evaluation

To compare different algorithms, a suitable and uniform evaluation index is needed. In this paper, three popular evaluation indicators are chosen as accuracy (ACC), Adjusted Mutual Information (AMI), and Adjusted Rand Index (ARI) (Cai et al. 2005; Vinh et al. 2010). The range of ACC is [0,1], the scope of AMI is [-1,1], and the range of ARI is [-1,1]. The closer each index is to 1, the better the clustering performance is. Their expression is Eq. (11), Eq. (12), and Eq. (13).

\[
\text{ACC} = \frac{\sum_{i=1}^{n} \delta(s, \text{map}(r_i))}{n}
\]

\[
\text{AMI} = \frac{I(U,V) - E[I(M)|a,b]}{\sqrt{H(U)H(V) - E[I(M)|a,b]}}
\]

\[
\text{ARI} = \frac{RI - E[RI]}{\max\{RI\} - E[RI]}
\]

### 4.4 Result analysis of artificial datasets

From Table 3, we can find that: the clustering result of the DPC++ algorithm on the Aggregation dataset is almost equal to that of the best DPC algorithm and better than other algorithms. In the Flame dataset, the clustering effect of the DPC++ algorithm is only slightly lower than that of the best algorithm. In the D15 dataset, the clustering performance of the DPC++ algorithm is slightly lower than that of the DPC algorithm and CFDPC algorithm. However, the difference of DPC++ algorithm is very small and better than other algorithms. Except for the above datasets, in D31, Spiral, Spiral_unbalance, Moon, and Circle 5 datasets, the clustering effect of the DPC++ algorithm is better than or equal to that of other algorithms.
Table 3. Performance of five algorithms on eight artificial datasets

| Algorithm | ACC  | AMI  | ARI  | ACC  | AMI  | ARI  |
|-----------|------|------|------|------|------|------|
| Aggregation |      |      |      | Flame |      |      |
| DPC++     | 0.9975 | 0.9923 | 0.9956 | 0.9917 | 0.9353 | 0.9666 |
| DPC       | **0.9987** | **0.9956** | **0.9978** | **1.0000** | **1.0000** | **1.0000** |
| DPCSA     | 0.9734 | 0.9570 | 0.9581 | **1.0000** | **1.0000** | **1.0000** |
| CFDPC     | 0.9975 | 0.9923 | 0.9956 | **1.0000** | **1.0000** | **1.0000** |
| DBSCAN    | 0.9937 | 0.9826 | 0.9881 | 0.9791 | 0.8894 | 0.8894 |
| D15       |      |      |      | D31  |      |      |
| DPC++     | 0.9950 | 0.9907 | 0.9893 | **0.9697** | **0.9574** | **0.9388** |
| DPC       | **0.9967** | **0.9938** | **0.9927** | 0.9687 | 0.9567 | 0.9371 |
| DPCSA     | 0.9933 | 0.9885 | 0.9857 | 0.9677 | 0.9551 | 0.9353 |
| CFDPC     | **0.9967** | **0.9938** | **0.9927** | 0.9674 | 0.9547 | 0.9345 |
| DBSCAN    | 0.9250 | 0.9356 | 0.9164 | 0.9084 | 0.8973 | 0.8157 |
| Spiral    |      |      |      | Spiral_unbalance |      |      |
| DPC++     | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| DPC       | **1.0000** | **1.0000** | **1.0000** | 0.6067 | 0.0949 | -0.1018 |
| DPCSA     | **1.0000** | **1.0000** | **1.0000** | 0.6190 | 0.0902 | -0.1045 |
| CFDPC     | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| DBSCAN    | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| Moon      |      |      |      | Circle |      |      |
| DPC++     | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| DPC       | 0.6820 | 0.2498 | 0.1315 | 0.5360 | 0.0024 | 0.0032 |
| DPCSA     | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** |
| CFDPC     | 0.3320 | 0.4748 | 0.2240 | 0.6420 | 0.6666 | 0.6258 |
| DBSCAN    | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** | **1.0000** |

![Fig.4 The effect of the five algorithms on the Aggregation dataset](image)

(a) DPC++  (b) DPC  (c) DPCSA  (d) CFDPC  (e) DBSCAN
In Fig. 4, we can observe that the Aggregation dataset has composed of 7 heap-like clusters, whose characteristics are obvious, but there is cross-linking between clusters. The clustering performance of the DPC++ algorithm is slightly weaker than that of the DPC algorithm, ranking second. The DPC++ algorithm can find cluster centers exactly right. The main assignment error is the point on the right side where the two cluster boundaries are connected. The label of the cluster boundaries may be disputed because the dataset is artificial, so the performance of the DPC++ algorithm is good.

![Images of Fig. 4](image1)

(a) DPC++  (b) DPC

In Fig. 5, we can observe that the DPC++ algorithm can correctly find the peak density, but five boundary points are misassigned. The DPC, DPCSA, and CFDPC algorithms can find the density peaks accurately and assign the points correctly. DBSCAN algorithm is weaker than the DPC++ algorithm.

![Images of Fig. 5](image2)

(a) DPC++  (b) DPC

(c) DPCSA  (d) CFDPC  (e) DBSCAN

Fig. 5 The effect of five algorithms on Flame dataset

In Fig. 6, we can observe that the DPC++ algorithm can correctly find the peak density, but five boundary points are misassigned. The DPC, DPCSA, and CFDPC algorithms can find the density peaks accurately and assign the points correctly. DBSCAN algorithm is weaker than the DPC++ algorithm.

![Images of Fig. 6](image3)

(a) DPC++  (b) DPC

(c) DPCSA  (d) CFDPC  (e) DBSCAN

Fig. 6 The effect of five algorithms on D15 dataset
In Fig.6, we can observe that: the D15 data set has fifteen clusters. The outermost seven clusters are far away from each other, all clustering algorithms can cluster these seven clusters correctly, and the innermost eight clusters are adjacent to each other and intertwined, so the samples of these eight clusters are easy to be misassigned. Except the DBSCAN algorithm, the clustering accuracy of other algorithms for the D15 data set is above 99%.

In Fig.7, we can observe that: the D31 dataset contains thirty-one class clusters. There are many kinds of clusters, and they are connected. These clustering algorithms can perform clustering well, but the DPC++ algorithm is the best.

In Fig.8, we can observe that: the Spiral dataset is manifold dataset composed of three Spiral clusters. These clustering algorithms can assign the points to correct clustering.
In Fig. 9, we can observe that: the Spiral_unbalance dataset is a manifold data set consisting of two clusters whose Spiral are unbalanced. The DPC++, DPC, and DBSCAN algorithms are completely clustered correctly, while the DPCSA and the CFDPC algorithms have poor clustering effects.

In Fig. 10, we can observe that: the Moon dataset is manifold dataset composed of two kinds of moonlike clusters, and a certain proportion of noise is added. DPC++ and DPCSA algorithms clustering algorithms are successful, but the DPC, CFDPC, and DBSCAN algorithms have more or less clustering errors.
In Fig. 11, we can observe that: the Circles dataset is a manifold dataset consisting of two circle-like clusters, one is big, and the other is small. Among them, DPC++, DPCSA, and DBSCAN clustering algorithms cluster successfully, and DPC and CFDPC clustering algorithms have larger errors.

In Fig. 12, the mean of all indexes of the DPC++ algorithm is the first among all clustering algorithms. The results show that the comprehensive clustering performance of the DPC++ algorithm is the best among all algorithms.

4.5 Analysis of experimental results of real-world datasets

For verify the clustering performance of the DPC++ algorithm in real-world datasets, we compare the DPC++ algorithm with four other kinds of clustering algorithms on eight real-world datasets. The clustering results of each algorithm are shown in Table 4. The experimental results show that the ACC and ARI indices of the DPC++ algorithms are slightly lower than that of the DPCSA algorithm when dealing with the iris dataset. When dealing with the parkinsons dataset, the clustering result AMI index of the DPC++ algorithm is slightly lower than that of the DPC algorithm. In the rest datasets of
dermatology, the wdbc, libras, wine, seeds and segmentation, the clustering effect of the DPC++ algorithm is better than the compared algorithms.

### Table 4. Performance of five clustering algorithms on eight real-world datasets

|               | ACC   | AMI   | ARI   |               | ACC   | AMI   | ARI   |
|---------------|-------|-------|-------|---------------|-------|-------|-------|
| **dermatology** |       |       |       | **wdbc**      |       |       |       |
| DPC++         | 0.8142| 0.8032| 0.7891| DPC           | 0.8700| 0.5148| 0.5389|
| DPC           | 0.7431| 0.7628| 0.6775|               | 0.6309| 0.0035| 0.0048|
| DPCSA         | 0.7123| 0.7451| 0.6061|               | 0.8137| 0.3361| 0.3771|
| CFDP          | 0.5246| 0.6427| 0.4048|               | 0.8084| 0.4098| 0.4484|
| DBSCAN        | 0.5956| 0.5752| 0.3782|               | 0.7135| 0.1027| 0.1640|
| **iris**      |       |       |       |               |       |       |       |
| DPC++         | 0.9600| 0.8689| 0.8858| DPC           | 0.5194| 0.6332| 0.4137|
| DPC           | 0.9600| 0.8625| 0.8857|               | 0.4222| 0.5339| 0.2850|
| DPCSA         | **0.9667**| 0.8331| **0.9038**|               | 0.4667| 0.5709| 0.3553|
| CFDP          | 0.6267| 0.6746| 0.5992|               | 0.3194| 0.5289| 0.2706|
| DBSCAN        | 0.8600| 0.6590| 0.6246|               | 0.3417| 0.4463| 0.1357|
| **parkinsons**|       |       |       |               |       |       |       |
| DPC++         | **0.8205**| 0.2467| **0.2686**|               | 0.9101| 0.7501| 0.7414|
| DPC           | 0.6564| **0.2517**| 0.0869|               | 0.8820| 0.7060| 0.6720|
| DPCSA         | **0.8205**| 0.1772| **0.2686**|               | 0.9101| 0.7480| 0.7414|
| CFDP          | 0.1179| 0.1705| 0.0271|               | 0.3258| 0.3853| 0.1805|
| DBSCAN        | 0.7846| 0.1792| 0.2047|               | 0.8146| 0.5858| 0.5292|
| **seeds**     |       |       |       |               |       |       |       |
| DPC++         | **0.9190**| 0.7309| **0.7789**|               | 0.6619| 0.6505| 0.5401|
| DPC           | 0.9000| 0.7212| 0.7341|               | 0.6381| 0.6226| 0.5039|
| DPCSA         | 0.8810| 0.6873| 0.6609|               | 0.5571| 0.5130| 0.3668|
| CFDP          | **0.9190**| 0.7309| **0.7789**|               | 0.3714| 0.5861| 0.3515|
| DBSCAN        | 0.6048| 0.4761| 0.4116|               | 0.5143| 0.3974| 0.2009|
| **segmentation** |       |       |       |               |       |       |       |
| DPC++         |       |       |       |               |       |       |       |
| DPC           |       |       |       |               |       |       |       |
| DPCSA         |       |       |       |               |       |       |       |
| CFDP          |       |       |       |               |       |       |       |
| DBSCAN        |       |       |       |               |       |       |       |

**Fig.13** The mean of three evaluation indexes on real-world datasets

In Fig.13, the mean of all indexes of the DPC++ algorithm is the first among all clustering algorithms. The results show that the DPC++ algorithm is the best among all algorithms.
5 Conclusion

In the DPC algorithm, selection of cutoff distance is difficult, and the definition of local density is not uniform. In addition, it is easy to make consecutive assignment errors. Based on the above shortcomings, we propose a density peaks clustering algorithm based on natural nearest neighbor and multi-cluster mergers. The DPC++ algorithm combines the idea of weighting and natural nearest neighbors, redefines the local density, measures the local density of each point more accurately, and solves the problem that the selection of cutoff distance and the measurement of local density definition are not uniform. At the same time, the DPC++ algorithm is combined with a shared nearest neighbor and fast and finite iteration to mergers multi-clusters, which reduces the possibility of consecutive assignment errors. Experimental results on various datasets show that the DPC++ algorithm can find density peaks and process datasets with different shapes more precisely, and the performance of clustering is very good.

The DPC++ algorithm does not involve the extraction and processing of noise, so that the clustering of boundary points in some datasets may be wrong. The next step is to further identify and process the noise based on this method to improve the accuracy of clustering, which is the focus of the next stage of research.

Acknowledgments This work was supported by the Natural Science Foundation of Zhejiang Province (LY20F020006) and the Ministry of Education of the People's Republic of China(19YJA870013).

Compliance with ethical standards

Conflicts of interest The authors declared that they have no conflicts with this work.

Reference

Ankerst M, Breunig MM, Kriegel HP, Sander J (1999) OPTICS: Ordering Points To Identify the Clustering Structure. Dissertation. University of Munich
Cai D, He X, Han J (2005) Document clustering using locality preserving indexing. IEEE Transactions on Knowledge and Data Engineering 17(12):1624-1637
Chen L, Wu RX, Li PW (2020) Weighted K-nearest neighbors and Multi-clusters Mergers Density Peaks Clustering Algorithm. Journal of Frontiers of Computer Science and Technology https://kns.cnki.net/kcms/detail/11.5602.tp.20210416.1428.002.html
Dasgupta S, Long PM (2005) Performance guarantees for hierarchical clustering. Journal of Computer and System Sciences. 70(4):555-569 https://doi.org/10.1016/j.jcss.2004.10.006
Deng C, Song JW, Sun Z et al. (2018) GRIDEN: An effective grid-based and density-based spatial clustering algorithm to support parallel computing. Pattern Recognition Letters. 109:81-88 https://doi.org/10.1016/j.patrec.2017.11.011
Du MJ, Ding SF, Jia HJ (2016) Study on density peaks clustering based on k-nearest neighbors and principal component analysis. Knowledge-Based Systems 99:135-145 https://doi.org/10.1016/j.knosys.2016.02.001
Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. Knowledge Discovery and Data Mining 226–231

Fahad A, Alshatri N, Tari Z et al. (2014) A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis. IEEE Transactions on Emerging Topics in Computing 2(3):267-279

Fang F, Qiu L, Yuan SF (2020) Adaptive core fusion-based density peak clustering for complex data with arbitrary shapes and densities. Pattern Recognition https://doi.org/10.1016/j.patcog.2020.107452

Frey BJ, Dueck D (2007) Clustering by Passing Messages Between Data Points. Science 315(5814):972-976

Guo LJ, Ai CY, Wang XM, Cai ZP (2009) Real time clustering of sensory data in wireless sensor networks. 2009 IEEE 28th International Performance Computing and Communications Conference 33-40

Guyon I, Luxburg UV, Williamson RC (2012) Clustering: Science or Art? Journal of Machine Learning Research 27:65-80

Han J, Kamber M, Pei J (2011) Data Mining: Concepts and Techniques, Third Edition. Morgan Kaufmann, San Francisco

Jain AK (2009) Data clustering: 50 years beyond K-means. Pattern Recognition Letters 31(8):651-666 https://doi.org/10.1016/j.patrec.2009.09.011

Kaufman LJP (2008) Rousseeuw: finding groups in data: an introduction to cluster analysis. Wiley, London

Kriegel HP, Kröger P, Sander J, Zimek A (2011) Density-based clustering. Wiley interdisciplinary reviews: data mining and knowledge discovery 1(3):231-240 https://doi.org/10.1002/widm.30

Leong SH, Ong SH (2017) Similarity measure and domain adaptation in multiple mixture model clustering: an application to image processing. Plos One https://doi.org/10.1371/journal.pone.0180307

Levent E, Steinbach M, Kumar V (2003) Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data. Proceedings of the Third SIAM International Conference on Data Mining https://doi.org/10.1137/1.9781611972733.5

Liu J, Wan J (2020) Optimized Density Peak Clustering Algorithm by Natural Reverse Nearest Neighbor. J Front Comput Sci Technol. https://kns.cnki.net/kcms/detail/11.5602.TP.20200917.1251.002.html

Lv L, Wang JY Wu RX, Wang H (2021) Density peaks clustering based on geodetic distance and dynamic neighbourhood. International Journal of Bio-Inspired Computation 17(1)24-33 https://doi.org/10.1504/IJBIC.2021.113363

Pappas NT (1992) An adaptive clustering algorithm for image segmentation. IEEE Transactions on Signal Processing 40(4):901-914

Rodríguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science 344(6191):1492-1496

Rui X, Wunsch DI (2005) Survey of clustering algorithms. IEEE Transactions on Neural Networks 16(3):645-678

Tang XY, Zhang ZJ, Chu J, Yan T (2021) Density Peaks Clustering Algorithm based on Natural Nearest Neighbor. Computer Science 48(03):151-157
Vinh NX, Epps J, Bailey J (2010) Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. Journal of Machine Learning Research 11(10):2837-2854

Wu B, Lu HL, Jiang HJ (2020) Adaptive density peaks clustering algorithm. J. of Computer Applications 40(6):1654-1661 https://doi.org/10.11772/j.issn.1001-9081.2019111881

Wu C, Lee J, Isokawa T, Yao J, Xia Y (2019) Efficient Clustering Method Based on Density Peaks with Symmetric Neighborhood Relationship. IEEE Access 7:60684 -60696 https://doi.org/10.1109/ACCESS.2019.2912332

Xie JY, Gao HC, Xie WX, Liu XH, Grant PW (2016) Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors. Information Sciences 354:19-40 https://doi.org/10.1016/j.ins.2016.03.011

Yang Z, Wang HJ, Zhou X (2018) A Clustering Algorithm with Adaptive Cut-off Distance and Cluster Centers. Data Analysis and Knowledge Discovery 2(03):39-48

Yu D, Liu G, Guo M, Liu X, Yao S (2019) Density Peaks Clustering Based on Weighted Local Density Sequence and Nearest Neighbor Assignment. IEEE Access 7:34301-34317 https://doi.org/10.1109/ACCESS.2019.2904254

Zhu Q, Feng J, Huang J (2016) Natural neighbor: A self-adaptive neighborhood method without parameter K. Pattern Recognition Letters 80:30-36 https://doi.org/10.1016/j.patrec.2016.05.007

Zhu QS, Tang H, Feng Y (2014) Outlier Detection Algorithm Based on Nearest Neighbor. Computer Science 41(3):276-305