Component Preserving Laplacian Eigenmaps for Data Reconstruction and Dimensionality Reduction

Hua Meng[1], Hanlin Zhang[1], Yu Ding[1], Shuxia Ma[1] and Zhiguo Long[2*]

[1]School of Mathematics, Southwest Jiaotong University, Chengdu, 611756, Sichuan, China.
[2]School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, 611756, Sichuan, China.

*Corresponding author(s). E-mail(s): zhiguolong@swjtu.edu.cn;
Contributing authors: menghua@swjtu.edu.cn; zhl85@my.swjtu.edu.cn;
dingy@my.swjtu.edu.cn; masuxia@swjtu.edu.cn;

Abstract

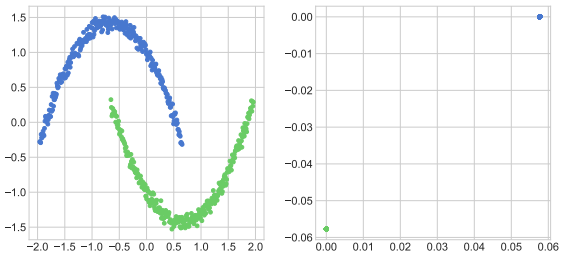
Laplacian Eigenmaps (LE) is a widely used dimensionality reduction and data reconstruction method. When the data has multiple connected components, the LE method has two obvious deficiencies. First, it might reconstruct each component as a single point, resulting in loss of information within the component. Second, it only focuses on local features but ignores the location information between components, which might cause the reconstructed components to overlap or to completely change their relative positions. To solve these two problems, this article first modifies the optimization objective of the LE method, by characterizing the relative positions between components of data with the similarity between high-density core points, and then solves the optimization problem by using a gradient descent method to avoid the over-compression of data points in the same connected component. A series of experiments on synthetic data and real-world data verify the effectiveness of the proposed method.

Keywords: Dimensionality reduction, cluster analysis, Laplacian Eigenmaps, spectral methods

1 Introduction

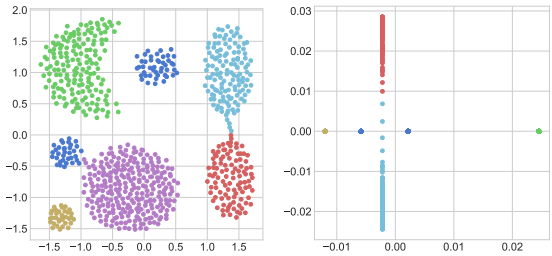
Dimensionality reduction and reconstruction of data are important to research in data mining [1]. This kind of methods can avoid the curse of dimensionality and improve the accuracy of classification and clustering algorithms. It can also reconstruct data with manifold or graph structure in low-dimensional Euclidean space for data visualization [2]. This technique has been widely used in various fields such as face recognition [3] and gene detection [4]. Standard linear dimensionality reduction methods, such as Principal Component Analysis (PCA) [5], Linear Discriminant Analysis (LDA) [6], and SLDA [7], look for a linear projection that projects the data into a subspace. For relatively complex structured data such as biological data and financial data, linear dimensionality reduction methods might not be suitable anymore, and one solution is to introduce kernel functions to improve linear dimensionality reduction methods such as KPCA [8] and KLDA [9].

Data reconstruction methods constitute another direction of dimensionality reduction, including Multidimensional Scaling (MDS) [10], t-SNE [11], Laplacian Eigenmaps (LE) [12], UMAP [13], Locally Linear Embedding (LLE) [14], Isometric Mapping (Isomap) [15], and



(a) cresent (b) LE

Fig. 1 Dataset with two connected components (left) and LE reconstruction results (with zero eigenvalues).



(a) Aggregation (b) LE

Fig. 2 Dataset with multiple connected components (left) and LE reconstruction results (without zero eigenvalues).

their variants are also widely used in manifold learning [16–18]. LE is an outstanding representative of the data reconstruction methods, which has been widely used [19–21] and has many variants, e.g., linear ones like LPP [22], ILPP-L1 [23], LAPP [24], ULGE [25] and LPP_SG [26], and non-linear ones like KSLE-ML [27] and COLES [28]. LE models data as a weighted undirected graph, where edges exist between similar sample points and the given weight on an edge corresponds to the degree of similarity between two points. It reconstructs data by performing a spectral decomposition of the Laplacian matrix of the graph. The method attempts to maintain the local similarity between sample points during the reconstruction process.

LE has several shortcomings for different types of data. For example, for data with uneven distribution, if k nearest neighbours are used to construct local similarity matrices or similarity maps, the results are often not as desired, and thus the Hessian LE (HLE) [29] method is proposed to solve this problem. By noticing that LE can break a single manifold structure into multiple local areas, SALE [30] proposes to use a soft adaptive loss to better maintain the manifold structure. In [28], the authors extend LE with contrastive learning, to resolve the common problem of loss functions in variants of LE that unrelated points cannot be guaranteed to be separated. Although various improvements have been made in the literature to address different shortcomings of LE and its variants, few have considered the performance of LE on multi-component data.

As shown in Fig. 1, the left side is the original multi-component data and the right side is the data after LE dimensionality reduction. In this figure, the reconstructed multi-component data are often compressed into a single point and the relative position between components will be destroyed. The main reason is that for multi-component data, the Laplacian matrix of LE method will have multiple zero eigenvalues, and zero eigenvalues will make the projection result of a whole component become a point in the new subspace. However, if zero eigenvalues are simply ignored, information about the differences between the different components might be lost.

Moreover, LE only cares about local similarities of nearest neighbours, which might not include information about components, because components are relatively far away from each other. This could make the projection results of components overlap and/or have random placements. For example, in Fig. 2, in addition to the problem of compressing some components into single points, the relative positions of components are completely destroyed, e.g., the green, brown, and dark blue components are on the same line after projection while they have completely different relative positions before projection. The latter problem exists not only in the original LE but also in many of its variants. These issues will be analysed in more detail in Section 2.

In this article, we focus on the above two problems of LE and model the overall structure by finding core points in data, in order to maintain the structure within the connected components of data and the relative positions between the components.

Contribution. The main contributions of this work are as follows:

- We propose a novel data reconstruction and dimensionality reduction approach that can reconstruct points within the same component

to be close together and also maintain the relative positions of different components.
- We introduce similarities for core points within a component and between different components to capture the global structural information of data.
- We devise a gradient descent method to accomplish data reconstruction and dimensionality reduction, which can help to avoid the over-compression problem of LE and similar approaches.

This work is a substantial extension of the conference paper [31], where the main differences are:

- We give more detailed analysis in the introduction section for the problems of LE, and added a section to discuss the research motivation.
- We give more detailed descriptions and analysis of the proposed method in Section 3.
- In experimental analysis, we give more detailed discussion on how hyperparameters are set, evaluate the proposed model on more synthetic datasets and against more baseline methods on real-world datasets, and demonstrate more detailed results.
- We add a section to analyse the effects of hyperparameters and the contributions of each component in the proposed model.
- In the conclusion section, we include more discussions on the advantages and limitations of the proposed model and provided more insights into future work.

The rest of the article is organized as follows: Section 2 briefly introduces the original LE and the related spectral clustering algorithm, analyses the algorithm limitations, and then presents our motivation for the study; Section 3 proposes the Component Preserving Laplacian Eigenmaps model (CPLE); Section 4 evaluates the performance of the new method on synthetic and real-world datasets; Section 5 discusses the effects of hyperparameters and components of the proposed model; Section 6 concludes the article.

2 Preliminaries and problem analysis

Suppose the data of interest contain n samples and each sample has m features, which can be represented as a matrix $X_{m\times n} = (x_1, x_2, \ldots, x_n)$, where each $x_i \in \mathbb{R}^m$ is an m-dimensional column vector.

2.1 Laplacian Eigenmaps

Laplacian Eigenmaps (LE) constructs the similarity relationship between points and their neighbours from a local perspective and then reconstructs this relationship in a new space with the following steps.

Step 1. Construct the adjacency graph. Denote by $N_k(x_i)$ the set of the k nearest neighbours of x_i under the Euclidean distance between points. Here, we require that $x_i \in N_k(x_i)$. If $x_j \in N_k(x_i)$ or $x_i \in N_k(x_j)$, then x_i and x_j are connected by an edge in the adjacency graph.

Step 2. Calculate the similarity matrix W. If x_i and x_j are connected by an edge, then the similarity between them is defined as

$$W_{ij} = \begin{cases} \exp(-\dfrac{\|x_i - x_j\|_2^2}{\sigma^2}) & x_j \in N_k(x_i) \text{ or } x_i \in N_k(x_j) \\ 0 & \text{otherwise} \end{cases}$$

where σ is hyperparameter.

Step 3. Reconstruct the data into d-dimensional space by solving the following generalized eigenvector problem. $Lf = \lambda Df$, where D is a diagonal matrix with $D_{ii} = \sum_{j=1}^{n} W_{ij}$, and $L = D - W$ is the corresponding Laplacian matrix. The calculated eigenvalues are sorted in ascending order, and the eigenvectors corresponding to the d smallest eigenvalues are selected to form the result matrix $Y_{n\times d}$. The transpose of each row vectors of Y, say $y_i \in \mathbb{R}^d$, is the result of dimensionality reduction of the original x_i.

2.2 Research motivation

The core steps of LE and its variants are to model data as a weighted graph, and then to reconstruct data by performing spectral decomposition of the Laplacian matrix L of the graph. It has been shown that L has the following properties [32].

Property 1. Let $G = (V, E)$ be an undirected *connected* graph with non-negative connection weights, and $W = (W_{ij})_{n\times n}$ be the weight matrix,

then the corresponding Laplacian matrix L is a semi-positive definite real symmetric matrix, 0 is its eigenvalue, and $(c, c, \ldots, c)^T$ is an eigenvector corresponding to the eigenvalue 0, where $c \neq 0$ is a constant real number.

Property 2. Let $G = (V, E)$ be an undirected graph with non-negative connection weights and $W = (W_{ij})_{n\times n}$ be the weight matrix. If G has s connected components: $G_1, G_2, \ldots, G_s$, then

- The multiplicity of the eigenvalue 0 of the corresponding Laplacian matrix L is s.
- For any $1 \leq i \leq s$, let $f_i = (f_{i1}, f_{i2}, \ldots, f_{in})^T$, where
$$f_{ij} = \begin{cases} c & v_j \in G_i, \\ 0 & \text{otherwise}, \end{cases}$$
then f_i is an eigenvector corresponding to the eigenvalue 0 of L, and when $i \neq j$, f_i is orthogonal to f_j.

From Property 1, we can actually conclude that if there is only one connected component in G, then the eigenvalue 0 can be ignored. This is because, all the element values in the eigenvector of the eigenvalue 0 are the same e.g., they are all ones, and thus there is no difference for data when reconstructed to this dimension. That is, if the eigenvector $(c, c, \ldots, c)^T$ of the eigenvalue 0 was used for reconstruction, then obtained Y will be

$$\begin{pmatrix} c & y_{12} & \cdots & y_{1d} \\ c & y_{22} & \cdots & y_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ c & y_{n2} & \cdots & y_{nd} \end{pmatrix} \tag{1}$$

The reconstructed points will be in the form of $(c, y_{i2}, \ldots, y_{id})^T$, where the first dimension is redundant.

From Property 2, if there are multiple components in G, the orthogonal eigenvectors of the eigenvalue 0 contain important information of the difference between components. When only these eigenvectors are used for reconstruction, different components will become different single points, which will be in the form of $(0, \ldots, 0, c_i, 0, \ldots, 0)^T$.

When these eigenvectors are not used, then different components might be overlapping after reconstruction, because the difference information reflected in these eigenvectors are lost. These can be seen from Fig. 1 and Fig. 2. Note that in Fig. 2, in addition to the over-compression and overlapping problems, the relative positions are drastically changed after reconstruction. The reason is that the original LE model did not consider this kind of information.

This article will then consider to design a model to keep the data as close as appropriate within the same component (instead of compressing them into a point), while maintaining the relative position relationship between components (avoiding overlaps and displacement). The main idea is to

- replace the spectral decomposition Laplacian matrix method with the gradient descent method to prevent over-compression; and
- introduce core points to better describe the structure of the data, maintain the relationship between core points, and balance the relative positions of components.

3 The CPLE algorithm

To characterize the component structure of data, inspired by the Density Peaks Clustering (DPC) [33, 34] method, this article introduces the concept of core points, which, roughly speaking, are the *local* density peak points.

3.1 Core points

Definition 1 For any $x_i \in X$, and a given number of nearest neighbours k, the density of x_i is defined as:

$$\rho_i = \sum_{j\in N_k(x_i)} \exp(-\mathsf{dist}_{ij}^2), \tag{2}$$

where $\mathsf{dist}_{ij} = \begin{cases} \|x_i - x_j\|_2 & x_j \in N_k(x_i) \\ 0 & \text{otherwise} \end{cases}$.

Note that here we did not adopt the density defined in DPC [33], as the absolute amount of density is less important than relative ones, and adopting this simpler way to calculate density can be more efficient.

Definition 2 For any $x_i \in X$, its *leader point* $\mathsf{Lead}(x_i)$ is defined as follows.

- If $N_k(x_i)$ contains points with higher density than x_i, then the point $x_j \in N_k(x_i)$ with higher density than x_i and closest to x_i is selected as the leader point of x_i, i.e., $\mathsf{Lead}(x_i) = x_j$.

- If x_i has higher density than that of any other point in $N_k(x_i)$, then the leader point of x_i is itself, i.e., $\mathsf{Lead}(x_i) = x_i$. In this case, x_i is said to be a *core point*.

We denote by $\mathsf{core}(X)$ all the core points in X. By the definition, one can identify a core point simply by checking if it has higher density than its neighbours. Also, it is easy to find that Lead is an operator from X to X. If x_i is a core point then $\mathsf{Lead}(x_i)$ will still be x_i. If x_i is not a core point, then $\mathsf{Lead}(x_i)$ will map x_i to a higher density point in the nearest neighbours. For any x_i, by applying the Lead operator repeatedly, x_i will be mapped to a unique core point, which is called the *core leader* of x_i and is denoted as $\mathsf{core}(x_i)$.

Fig. 3 shows the distribution of core points in the "Aggregation" dataset ($k = 10$), where the points with red circles are the core points. The edges reflect leader point relationship. Intuitively, the core points of the data represent the overall structure of the data in a coarse granularity.

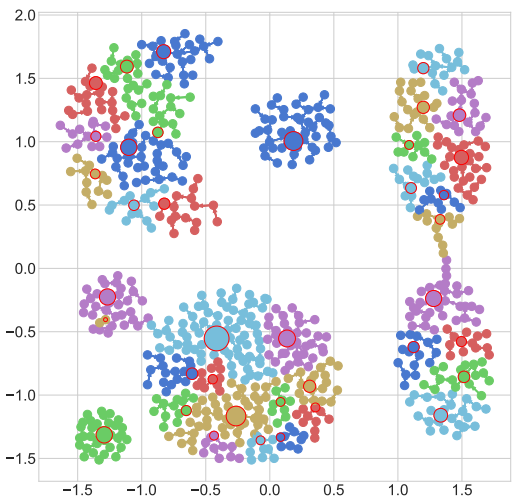


Fig. 3 Distribution of core points in the "Aggregation" dataset ($k = 10$).

3.2 Construct the similarity matrix

To solve the problems of over-compression to single points and changes of relative positions of components in the reconstruction process, this article considers building multiple similarity matrices to reveal the structure of the data from different scales. In particular, we propose two similarity measures (W^{comp} and W^{core}) to characterize the structural information within and between components, respectively. By characterizing the similarities within a component, the points in the same component will be reconstructed to be close together, and each point will follow its core leader after reconstruction. By characterizing the similarities between components, the relative positions of components will be maintained and overlaps or changes of relative positions will be avoided. The problem of over-compression will be resolved by replacing the direct spectral decomposition solution of the optimization problem with an iterative approximation, so that points in the same component will not be reconstructed into a single point. The specific steps are as follows.

First, we construct the similarity matrix W^{comp} within a component, which is composed of two parts W^{TT} and W^{TC}.

(1) Similar to the LE method, the similarity matrix W^{TT} of points and neighbouring points is defined as:

$$W_{ij}^{\text{TT}} = \begin{cases} \exp(-\frac{\|x_i - x_j\|_2^2}{\sigma_1^2}) & x_j \in N_k(x_i) \text{ or } \\ & x_i \in N_k(x_j) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $\sigma_1 \in \mathbb{R}$. W^{TT} characterizes the similarity between each point and its neighbours, so that after reconstruction each point is still close to its neighbours. However, a point and some of its neighbours might not be necessarily in the same component, so we will include another similarity measure W^{TC} to help points follow their core leaders which are usually from the same component as the points themselves.

(2) The similarity matrix W^{TC} between x_i and its core leader $\mathsf{core}(x_i)$ is to bring the point closer to its core point after reconstruction, which is defined as follows:

$$W_{ij}^{\text{TC}} = \begin{cases} \exp(-\frac{\|x_i - x_j\|_2^2}{\sigma_1^2}) & \mathsf{core}(x_i) = x_j \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

As mentioned before, a point is usually in the same component as its core leader, so by characterizing the similarities between them, the structural information within a component can be better

maintained, and points within a component will be reconstructed as close together.

The local and intra-component similarity matrices are obtained by weighted merging of W^{TT} and W^{TC}.

$$W^{\mathrm{comp}} = W^{\mathrm{TT}} + \alpha * W^{\mathrm{TC}} \tag{5}$$

where $\alpha \in \mathbb{R}$ is a hyperparameter.

Second, in order to maintain relative positions between components, we need to characterize similarities between components. The idea here is to make use of core points to represent components. Therefore, we construct the similarity matrix W^{core} between core points, which is also composed of two parts, i.e., W^{CC_1} and W^{CC_2}.

(1) Core points should maintain relative positions in data reconstruction, so we define the similarity matrix W^{CC_1} between core points:

$$W_{ij}^{\mathrm{CC}_1} = \begin{cases} \exp(-\frac{\|x_i - x_j\|_2^2}{\sigma_2^2}) & x_i, x_j \in \mathsf{core}(X) \\ & \text{and } x_i \neq x_j \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $\sigma_2 \in \mathbb{R}$ is a hyperparameter.

(2) In order to distinguish core points in the same component and in different components, we introduce another similarity measure between core points, which is given by the shortest path distance in a graph, denoted as $\mathsf{dist\_}G_{ij}$, so that core points in the same component are more similar than those in different components. The graph G used here is induced by the adjacency matrix dist in Definition 1, which is actually a k-nearest neighbour graph.

$$W_{ij}^{\mathrm{CC}_2} = \begin{cases} \exp(-\mathsf{dist\_}{G_{ij}}^2) & x_i, x_j \in \mathsf{core}(X) \\ & \text{and } x_i \neq x_j \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Combining W^{CC_1} and W^{CC_2}, we get the matrix of position relations between core points:

$$W^{\mathrm{core}} = W^{\mathrm{CC}_1} + \beta * W^{\mathrm{CC}_2} \tag{8}$$

where $\beta \in \mathbb{R}$ is a hyperparameter.

3.3 Optimization goals

First, we hope that after reconstruction, similar points are still close, and the distance between points and their core points is as close as possible. Based on this, the following objective are obtained:

$$\arg\min_{Y} \ \mathrm{tr}(Y^T L_b Y) \tag{9}$$

where $Y = (Y_{ij})_{n \times d}$ is the result of data reconstruction, $L_b = D^b - W^{\mathrm{comp}}$ and D^b is the diagonal matrix with the diagonal element values: $D_{ii}^b = \sum_{j=1}^n W_{ij}^{\mathrm{comp}}$.

Second, the reconstructed data should maintain the relative positions of the core points, i.e., the core points that are originally close should also be relatively close after reconstruction. Thus, we need to optimize the following objective:

$$\arg\min_{Y} \ \mathrm{tr}(Y^T L_c Y) \tag{10}$$

where $L_c = D^c - W^{\mathrm{core}}$, and D^c is the diagonal matrix with diagonal element values: $D_{ii}^c = \sum_{j=1}^n W_{ij}^{\mathrm{core}}$.

Combining Eqs. 9 and 10, the objective function is:

$$\arg\min_{Y} \ \mathrm{tr}(Y^T (L_b + L_c) Y) \tag{11}$$

where $Y = (Y_{ij})_{n \times d}$ is the result after data reconstruction; I is an identity matrix. However, if there is no constraint on Y, then a trivial solution to the above objective function is $Y = \mathbf{0}$, which is not desired.

Therefore, to remove the effect of scaling on the model, borrowing the idea from the LE method, we add the orthogonal constraint on Y:

$$Y^T D^{\mathrm{core}} Y = I \tag{12}$$

where D^{core} is the diagonal matrix with diagonal element values: $D_{ii}^{\mathrm{core}} = \sum_{j=1}^n W_{ij}^{\mathrm{CC}_1}$; I is the identity matrix.

Hence, the optimization problem becomes:

$$\begin{aligned} &\arg\min_{Y} \ \mathrm{tr}(Y^T (L_b + L_c) Y) \\ &\text{s.t.} \quad Y^T D^{\mathrm{core}} Y = I. \end{aligned} \tag{13}$$

This is an optimization problem with manifold constraints, which can be solved iteratively with

Augmented Lagrangian Method [35], by minimizing:

$$\mathrm{tr}(Y^T(L_b + L_c)Y) - \frac{1}{2}\langle \Lambda, Y^T D^{\mathrm{core}} Y - I\rangle + \frac{\theta}{4}\|Y^T D^{\mathrm{core}} Y - I\|_F^2 \quad (14)$$

where Λ is the Lagrangian multiplier matrix, $\langle\cdot,\cdot\rangle$ is the inner product of matrix, and θ is a balancing coefficient.

Note that for the data reconstruction problem here, the orthogonal constraint is used to remove the correlations between attributes and to normalize the length of vectors to 1, so as to prevent points from being reconstructed as a single point. Therefore, to simplify the optimization process, we remove the Lagrangian term and take the orthogonal constraint as an orthogonal regularization term in the objective function:

$$\arg\min_{Y} \ \mathrm{tr}(Y^T(L_b + L_c)Y) + \frac{\theta}{4}\|Y^T D^{\mathrm{core}} Y - I\|_F^2 \quad (15)$$

By replacing the constraint to a regularization term, first the length requirement is slightly relaxed, and the length of vectors is scaled around 1. Fortunately, the relative distance of reconstructed points will not be drastically different from the ones given by the original model, as scaling is a linear transformation and is controlled to a small amount by the regularization term. Moreover, the orthogonal requirement is also slightly relaxed so that there are weak correlations of attributes, which however, is acceptable as only a small amount of redundancy is introduced, and the performance will not be seriously affected.

In order to solve Eq. 15 and to prevent the data from being over-compressed, the gradient descent method is used in this article for solving the model. In particular, we iteratively update Y as

$$Y^{(k+1)} = Y^{(k)} - \eta(2(L_b + L_c)Y^{(k)} + \theta D^{\mathrm{core}} Y^{(k)} ({Y^{(k)}}^T D^{\mathrm{core}} Y^{(k)} - I)) \quad (16)$$

By introducing the stopping criterion $|\mathsf{loss} - \mathsf{loss}_{\mathrm{new}}| < \epsilon$ to compare the loss function values before and after each iteration, the over-compression of reconstructed data can be avoided, as the iteration can be stopped before the reconstructed data become too compact. Algorithm 1 lists the detailed steps of the proposed Component Preserving Laplacian Eigenmaps model (CPLE).

Algorithm 1 CPLE

Require: Initial data $X_{m\times n}$; Nearest neighbour parameter k; Target dimension d; α, β, θ; Iteration threshold ϵ

Ensure: $Y_{n\times d}$

1: $X \leftarrow$ StandardScaler(X)
2: **for** each $x_i \in X$ **do**
3: Calculate $N_k(x_i)$ and the matrix dist
4: Calculate $\rho_i \leftarrow$ Definition 1
5: **end for**
6: **for** each $x_i \in X$ **do**
7: Calculate $\mathsf{core}(x_i) \leftarrow$ Definition 2
8: get the $\mathsf{core}(X)$
9: **end for**
10: Calculate $W^{\mathrm{comp}} \leftarrow$ Eqs. 3-5
11: Calculate $W^{\mathrm{CC}_1} \leftarrow$ Eq. 6
12: Calculate $W^{\mathrm{core}} \leftarrow$ Eqs. 6-8
13: Randomly generate one $Y_{n\times d}$
14: $\mathsf{loss}_{\mathrm{new}} \leftarrow 0$
15: **while** *True* **do**
16: Update $Y_{n\times d}$ by Eq. 16
17: Calculate loss according to Eq. 15
18: **if** $|\mathsf{loss} - \mathsf{loss}_{\mathrm{new}}| < \epsilon$ **then**
19: break
20: **else**
21: $\mathsf{loss}_{\mathrm{new}} = \mathsf{loss}$
22: **end if**
23: **end while**
24: return Y

3.4 Time complexity analysis

Suppose the number of sample points in X is n, the dimensionality of each point is m, the number of nearest neighbours is k, the number of core points in X is K, and the target dimensionality is d. For line 1, the StandardScaler transforms each $x \in X$ to be $\frac{x-\mu}{\sigma}$, so the time complexity for this step is $\mathcal{O}(nm)$. The for-loop on lines 2-5 is to get the k nearest neighbours and density of each point, which can be obtained by using a KD-tree, and the time complexity is $\mathcal{O}((m+k)n\log n)$. The for-loop on lines 6-9 is to obtain the core leader of each x_i and all the core points in X, where the core

Table 1 Time complexity comparison.

Algorithm	Time Complexity
CPLE (ours)	$\mathcal{O}((m+K)n\log n + t(n^2d + d^2n))$
LE [12]	$\mathcal{O}(mn^2)$
LPP [22]	$\mathcal{O}(mn^2 + nm^2 + m^3)$
Isomap [15]	$\mathcal{O}(kn^2\log n + n^3)$
ILPP-L1 [23]	$\mathcal{O}(kmdnt + mdn + md^2)$
LAPP [24]	$\mathcal{O}(t(mn^2 + nm^2 + m^3))$
MDS [10]	$\mathcal{O}(n^3)$
t-SNE [11]	$\mathcal{O}(mn^2)$

leaders can be determined by using a depth-first search with time complexity $\mathcal{O}(kn)$ as each node can have at most k edges, and all the core points can be determined by going through the k-nearest neighbours of each x_i with time complexity $\mathcal{O}(kn)$. Lines 10-12 calculate W^{comp}, W^{CC_1}, and W^{core}, which takes $\mathcal{O}(kn)$, $\mathcal{O}(K^2)$, and $\mathcal{O}(Kn\log n + knK)$ time, respectively. Line 13 takes $\mathcal{O}(nd)$ to generate $Y_{n\times d}$. For each iteration of the while-loop on lines 15-23, the main step on lines 16-17 updates with gradient descent and calculate loss, which takes $\mathcal{O}(n^2d + nd^2)$ time. So the total time complexity for the while-loop is $\mathcal{O}(t(n^2d + nd^2))$, where t is the number of iterations of the loop. In total, the time complexity of the proposed algorithm CPLE is $\mathcal{O}(nm + (m+k)n\log n + kn + K^2 + Kn\log n + knK + nd + t(n^2d + d^2n)) = \mathcal{O}((m+k+K)n\log n + knK + t(n^2d + d^2n))$. Note that k is usually much smaller than $m + K$ and $\log n$, so the time complexity can be simplified to $\mathcal{O}((m+K)n\log n + t(n^2d + d^2n))$.

Table 1 shows the comparison of the time complexity of CPLE and other baseline algorithms. From the table, we can see that CPLE has an order of time complexity that is approximately comparable to the baseline algorithms, e.g., LE is $\mathcal{O}(mn^2)$ which is similar to that of CPLE.

4 Experimental analysis

To illustrate the effectiveness of the CPLE method, experiments are conducted on synthetic datasets with multi-connected component structures and real-world datasets, respectively. The baseline algorithms involved are: LE [12], LPP [22], Isomap [15], ILPP-L1 [23], LAPP [24], MDS [10], and t-SNE [11]. Here, the implemention of LPP is from its authors. LE, ILPP-L1 and LAPP are implemented according to the descriptions in the corresponding papers. The implementations of Isomap, MDS, and t-SNE are from scikit-learn library. The default parameter values of these algorithms are used.

For the hyperparameters of CPLE, note that both a too small and a too large value of k for CPLE will result in worsened performance, which will be illustrated in Section 5.1.1. Based on these observations and for the sake of fairness, the parameter k involved in the experimental part of this article is fixed as a moderate value, i.e., $k = 10$. α and β are to balance different aspects of similarity, and according to the discussion in Section 5.1.2, the values of α and β are set as 5 for real-world datasets. For synthetic datasets, in order to better visualize the results, the values of α and β are set separately for each dataset. The balancing coefficient θ in Eq. 15 should not be too large, otherwise the quadratic term will become dominant and the solution will be unstable [35]. In this article, we set $\theta = \frac{0.5}{\|Y^{(0)T} D^{\text{core}} Y^{(0)}\|_F}$ to normalize the quadratic term, where $Y^{(0)}$ is the initial random value of Y. The value of the threshold ϵ for the decrement in loss should be set as a small value, so that the iteration stops when the decrement in loss is very small and the reconstructed results are almost not improving. Here, we set $\epsilon = 10^{-7}$, which is a reasonable small value. For σ_1 and σ_2, they control the similarity values based on distance between points, and when the distance is larger than the parameter value, the similarity value will become relatively small. In order to properly characterize similarities, the values of σ_1 and σ_2 should be set according to the distance values in a dataset, e.g., one can use mean or median of the values of the distance between points. Here, we set $\sigma_1 = \sigma_2 = \max(\mathsf{dist}) \times 20\%$ based on experience and observations in experiments.

Note that CPLE will actually reconstruct data in a $d-1$ dimensional subspace for a target dimensionality d. This is because, the Laplacian matrix $(L_b + L_c)$ in the objective function (Eq. 13) has at least one zero eigenvalue, and $(1, 1, \ldots, 1)^T$ will be an eigenvector of it. The reconstructed data will thus have the same value in one of directions, which results in the reconstructed data to be in a $d-1$ dimensional subspace. Traditional methods like LE will ignore the eigenvector corresponding to the zero eigenvalue, while this is

Table 2 Details of datasets

	Name	Instances	Features	Class
synthetic datasets	cresent	600	2	2
	Aggregation	788	2	7
	R15	600	2	15
	D31	3100	2	31
real-world datasets	abta	2282	8	4
	alphabet	814	892	3
	breastEW	569	30	2
	isolet2	1560	617	2
	mfeat-mor	2000	6	10
	psis	32	4	3
	segmentation	2100	19	7
	spliceEW	3175	60	3
	ver2	310	6	3
	wdbc	569	30	2

not possible for CPLE as it obtains reconstruction in an iterative manner. Therefore, for visualization, in order to better illustrate the distribution of the reconstructed data in a d dimensional space, we first apply CPLE to reconstruct the data in $d+1$ dimensional space and then apply PCA on the reconstructed data to visualize them in d dimensional space.

The specific arrangement of this part is as follows: Section 4.1 introduces datasets; Section 4.2 studies the effectiveness of gradient descent; Sections 4.3 and 4.4 are the effectiveness analysis of the algorithm on synthetic datasets and real-world datasets, respectively.

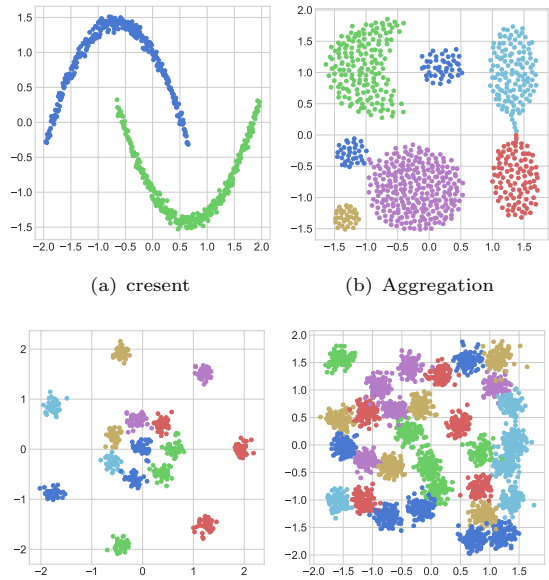
4.1 Datasets

The information of the synthetic and real-world datasets selected for this article is shown in Table 2.

The synthetic datasets all have multiple connected components, and they are visually illustrated in Fig. 4. The "cresent" dataset (a) is shaped like a pair of crescents with two connected components; the "Aggregation" dataset (b) consists of irregular non-spherical components; "R15" (c) resembles a firework shape with 15 components; and the "D31" dataset(d) has a multi-component combination with a more complex structure and 31 components.

4.2 Visualization of synthetic data with the number of iterations

In order to test the effectiveness of replacing the spectral decomposition method with the gradient descent method, we test the reconstruction



(a) cresent (b) Aggregation

(c) R15 (d) D31

Fig. 4 Synthetic datasets.

performance of CPLE by varying the number of iterations of gradient descent.

Fig. 5 and Fig. 6 show the visualization results of the "cresent" and "Aggregation" datasets after different numbers of iterations (8,000 to 40,000 iterations, with a step size of 8,000), where the points with red circles in the figure are the core points. We can find that, as the number of iterations increases, the same-component data become more and more compact and the relative positions between different core points can be well maintained. Moreover, after a certain number of iterations, the data can maintain as much intra-cluster similarity structure as possible, while keeping the relative position relationship between clusters.

4.3 Visualization of synthetic datasets

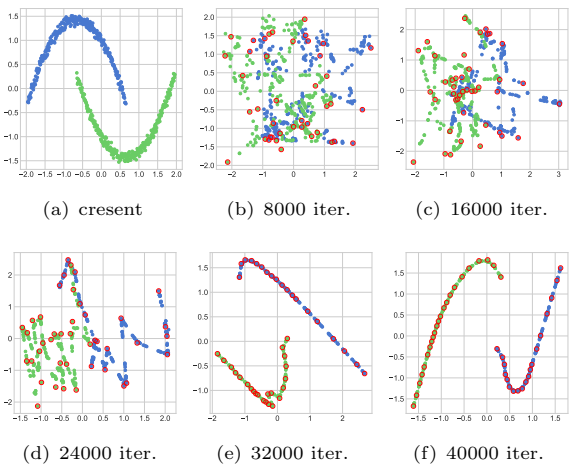
To verify the effectiveness of the CPLE algorithm on datasets with multiple connected components, we compare the visualization of LE, LPP, Isomap, ILPP-L1, LAPP, MDS, t-SNE and CPLE algorithms on three synthetic datasets, respectively. The specific experimental procedure is that each of these algorithms reconstructs the synthetic data in two dimensions and then visualizes them, which can visualize whether our algorithm solves the problems existing in traditional LE. For our

Table 3 Comparison of NMI on the ten real-world datasets

	LE	LPP	Isomap	ILPP-L1	LAPP	MDS	t-SNE	CPLE
abta	7.66±0.03	8.17±1.15	9.54±0	**12.09±0.45**	11.42±0.04	11.99±0.13	9.86±1.32	11.01±1.03
alphabet	5.75±0.06	0.56±0	5.34±0.06	4.57±1.95	0.49±0	**7.75±0.55**	5.23±1.17	6.79±1.26
breastEW	63.47±0	4.85±0	61.99±0	12.01±4.76	45.14±0.22	58.50±1.12	52.93±6.59	**64.73±0.75**
isolet2	2.06±0	0.13±0	1.85±0	0.12±0.11	0.60±0	0.15±0.08	1.50±0.27	**2.44±0.14**
mfeat-mor	62.67±1.19	40.83±0.16	68.40±0.10	67.22±1.18	66.51±0.38	65.95±0.20	65.33±0.50	**68.85±0.02**
psis	28.03±0	20.17±0	52.71±0	50.09±10.66	3.03±0.59	41.64±0	7.92±3.74	**56.27±8.10**
segmentation	53.46±0	52.87±0.13	54.03±0.04	36.38±10.08	53.74±0.17	57.16±0.46	52.88±1.43	**60.79±2.31**
spliceEW	1.46±0	0.91±0.05	5.08±0.58	3.88±0.96	2.24±0.65	3.15±0.71	3.33±1.55	**6.02±1.06**
ver2	31.71±0	32.86±0.91	32.26±0	24.40±5.04	7.60±0	31.17±1.11	14.15±2.99	**36.05±0.90**
wdbc	**66.27±0**	38.95±0.45	63.86±0	34.86±17.11	40.29±0	58.40±0.73	59.59±5.27	64.00±0.11

Table 4 Comparison of ACC on the ten real-world datasets

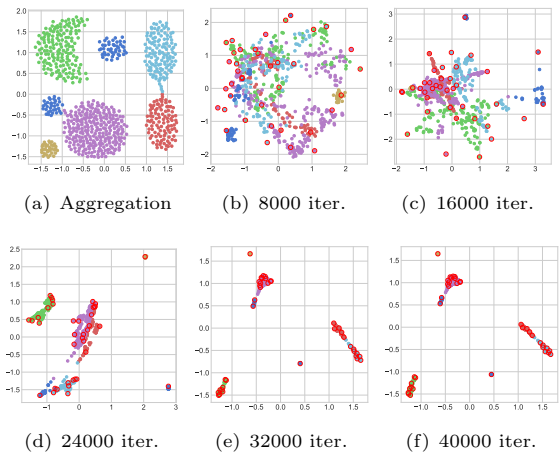
	LE	LPP	Isomap	ILPP-L1	LAPP	MDS	t-SNE	CPLE
abta	35.78±0.05	37.48±0.65	35.93±0	42.48±1.29	38.05±0.04	**42.78±0.21**	38.62±0.72	40.92±2.01
alphabet	**49.30±0.06**	47.17±0	41.78±0.27	41.93±3.15	43.37±0	43.91±1.35	43.50±2.70	47.29±0.80
breastEW	91.92±0	66.43±0	**92.27±0**	70.40±2.53	88.01±0.07	91.92±0.27	90.12±2.11	91.97±0.27
isolet2	58.14±0	50.06±0	57.76±0	51.39±0.83	54.49±0	52.21±0.66	56.94±0.61	**58.87±0.29**
mfeat-mor	58.32±1.45	45.25±0.29	64.13±0.11	**65.03±1.65**	64.47±0.13	64.27±0.19	63.36±0.34	64.56±0.02
psis	53.13±0	50.00±0	**71.88±0**	69.06±6.00	41.25±1.88	68.75±0	45.00±6.12	**71.88±4.84**
segmentation	46.67±0	52.29±0.10	49.90±0.02	43.65±7.17	58.30±0.28	54.14±0.32	59.94±0.40	**62.97±3.18**
spliceEW	52.19±0	39.84±0.27	48.43±1.52	43.71±1.21	41.31±0.61	41.77±1.81	43.92±2.85	**55.65±1.02**
ver2	48.06±0	54.97±0.30	48.71±0	49.74±3.39	43.87±0	51.65±1.49	47.03±2.46	**55.00±4.47**
wdbc	**93.67±0**	82.00±0.21	92.97±0	81.83±10.35	84.01±0	91.97±0.21	92.13±1.38	93.32±0



(a) cresent (b) 8000 iter. (c) 16000 iter.

(d) 24000 iter. (e) 32000 iter. (f) 40000 iter.

Fig. 5 The results of the CPLE algorithm on "cresent" with different numbers of iterations (α=5, β=1).



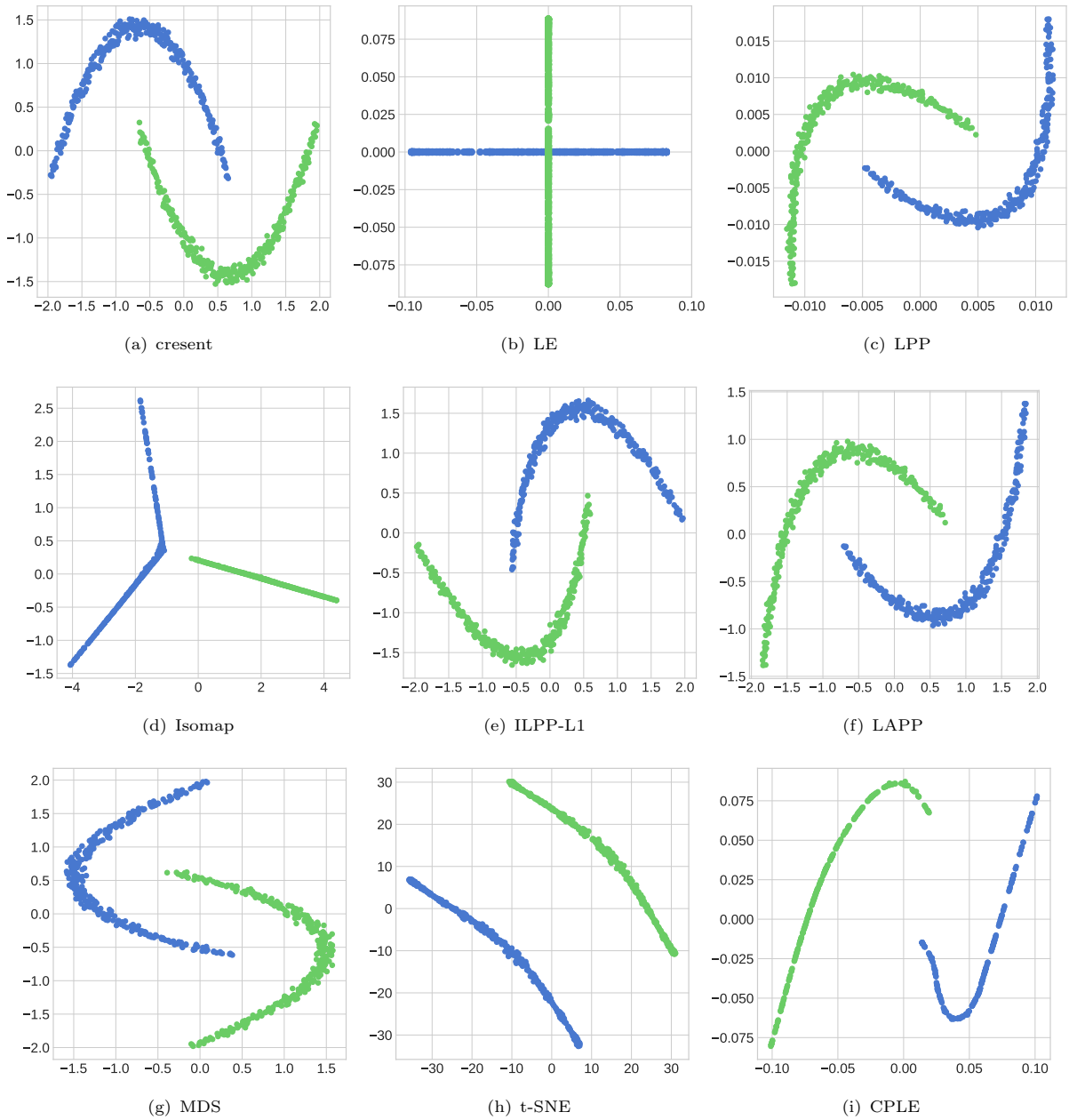
(a) Aggregation (b) 8000 iter. (c) 16000 iter.

(d) 24000 iter. (e) 32000 iter. (f) 40000 iter.

Fig. 6 The results of the CPLE algorithm on "Aggregation" with different numbers of iterations (α=5, β=5).

method, the parameters are set as follows: "cresent": $\alpha = 5$, $\beta = 1$; "Aggregation": $\alpha = 5$, $\beta = 5$; "R15": $\alpha = 2$, $\beta = 1$; "D31": $\alpha = 5$, $\beta = 1$.

Fig. 7-10 show the two-dimensional reconstruction visualization of the synthetic datasets "cresent", "Aggregation", "R15" and "D31" based on LE, LPP, Isomap, ILPP-L1, LAPP, MDS, t-SNE, and CPLE methods (ours), respectively. The traditional LE method ((b) in Fig. 7-10) also performs poorly on these datasets, for example, on "Aggregation" and "R15", the original structure of the data is not maintained, and overlap occurs between connected components, which destroys



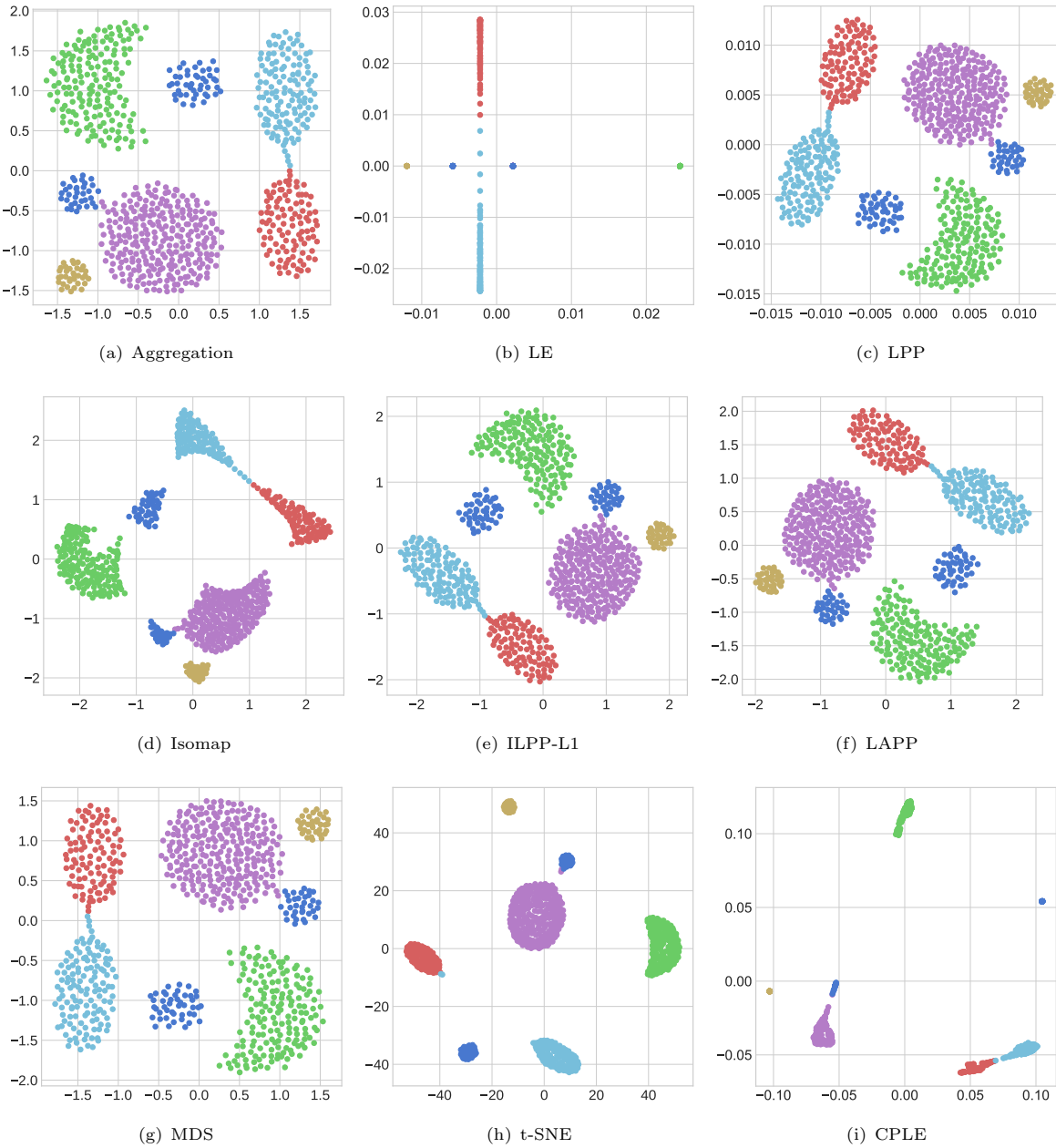
(a) cresent (b) LE (c) LPP

(d) Isomap (e) ILPP-L1 (f) LAPP

(g) MDS (h) t-SNE (i) CPLE

Fig. 7 Data reconstruction results for cresent.

the key information of the original data structure. The LPP method ((c) in Fig. 7-10) is a linear approximation of the LE method, so there is little difference between the two-dimensional reconstruction of the two-dimensional data. This also applies to ILPP-L1 and LAPP ((e) and (f) in Fig. 7-10), which are variants of LPP. Isomap ((d) in Fig. 7-10) tries to maintain the geodesic distance between neighbours, but it still deviates from the original inter-component structure after reconstructing the "R15" and "D31" datasets. From the results of t-SNE ((h) in Fig. 7-10), it tends to separate different components, and sometimes destroys neighbouring information and relative positions of components. For example, for R15, originally there are two green components



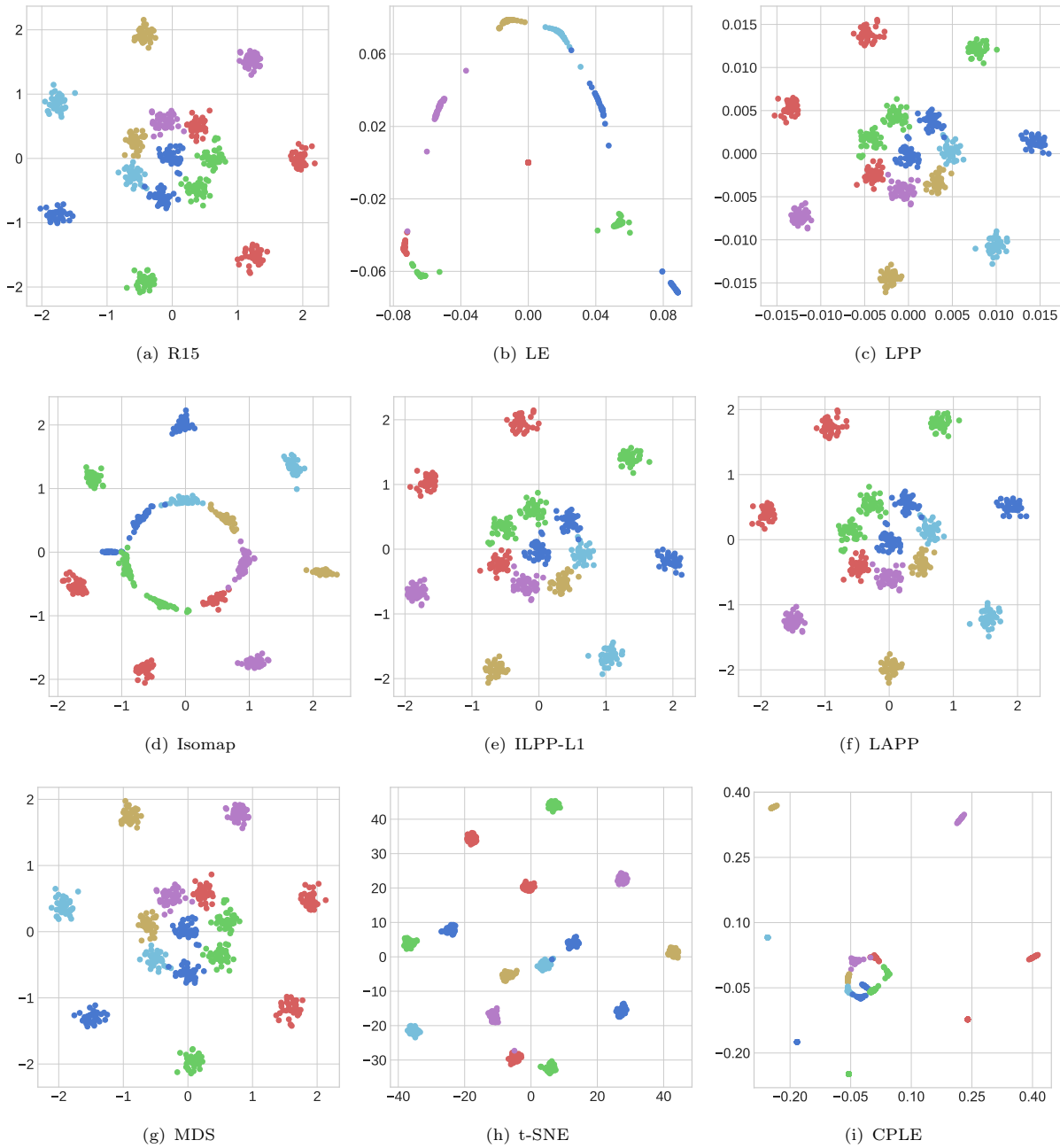
(a) Aggregation (b) LE (c) LPP

(d) Isomap (e) ILPP-L1 (f) LAPP

(g) MDS (h) t-SNE (i) CPLE

Fig. 8 Data reconstruction results for Aggregation.

that are close with each other, but after reconstruction, they become far apart. Unlike other methods, the CPLE method ((f) in Fig. 7-10) performs well on all four datasets. For example, after reconstructing the "Aggregation" and "R15" datasets, the structure of different components of the original data and the relative positions of the components remain unchanged and are well maintained; in addition, for the more complex "D31" dataset, CPLE still maintains a similar structure of the original data, respecting the relative positions of components.



(a) R15 (b) LE (c) LPP

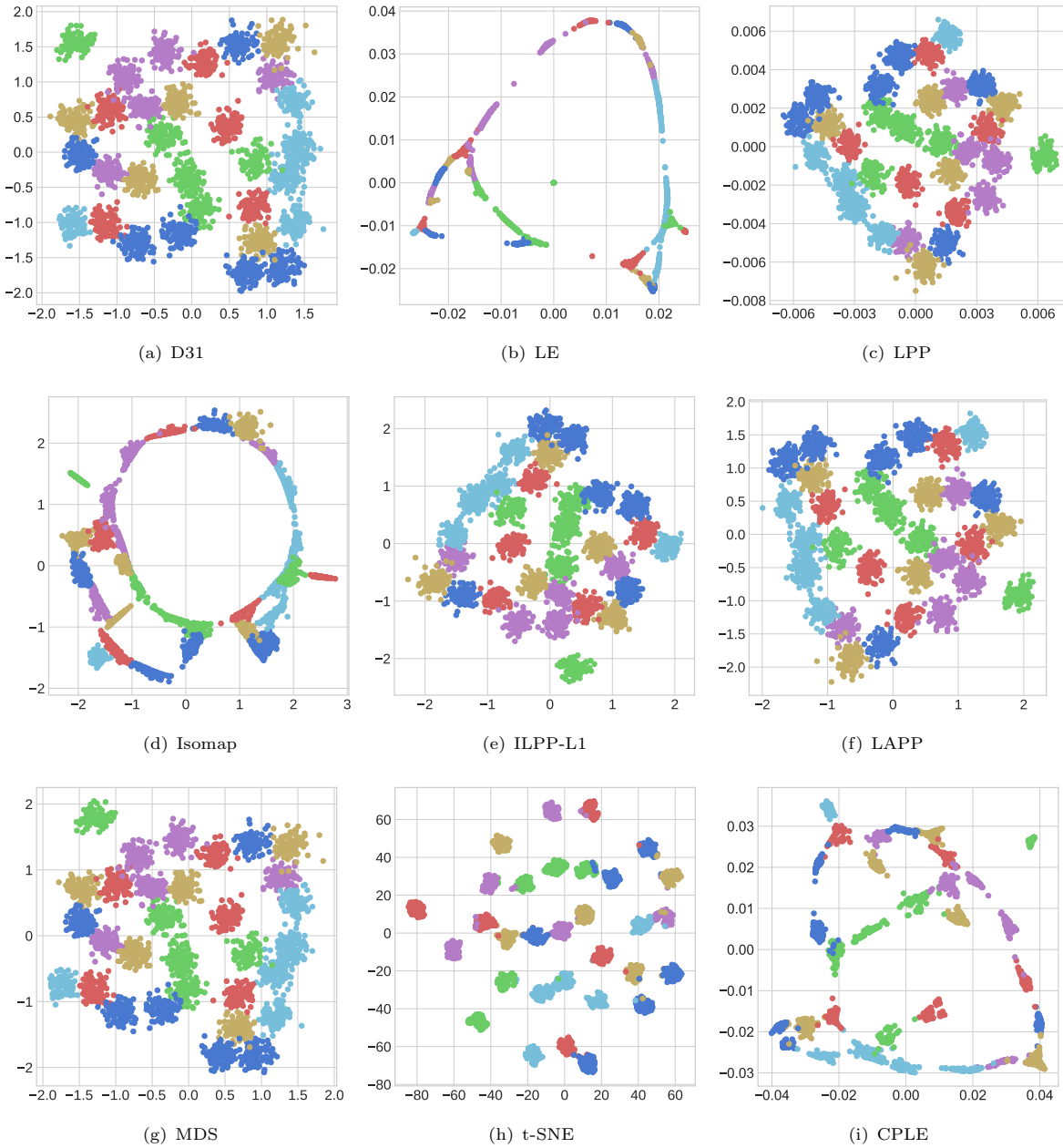
(d) Isomap (e) ILPP-L1 (f) LAPP

(g) MDS (h) t-SNE (i) CPLE

Fig. 9 Data reconstruction results for R15.

4.4 Results on real-world datasets

Clustering analysis of reconstructed data can reflect some intrinsic properties and patterns among the data, so to demonstrate the reliability of CPLE, we also conducted experiments on real-world datasets, and the final results $Y_{n\times d}$ obtained from CPLE are passed on to Kmeans algorithm to obtain the corresponding clustering metrics. To better evaluate the clustering results, we used three external metrics: NMI (Normalized Mutual Info), ACC (Accuracy) and ARI (Adjusted Rand Index). In addition, we combined the previous baseline algorithms with Kmeans to calculate the corresponding clustering metrics, respectively.



(a) D31 (b) LE (c) LPP

(d) Isomap (e) ILPP-L1 (f) LAPP

(g) MDS (h) t-SNE (i) CPLE

Fig. 10 Data reconstruction results for D31.

Because both the gradient descent method and Kmeans method have some randomness, clusterings with the considered algorithms are performed 10 times for each dataset, and we take the average value as the final result, and additionally calculate the standard deviation of the 10 experiments. The target dimensionality of reconstruction is set to n_class, which is the number of classes of the dataset; the number of classes of Kmeans clustering is also set to n_class; however, as discussed at the beginning of Section 4, CPLE actually reconsrtucts data in a $d-1$ dimensional subspace for the target dimensionality d, and thus the dimensionality of reconstruction by CPLE is set to $n\_class+1$ to have the data be reconstructed in a n_class dimensional subspace.

Table 5 Comparison of ARI on the ten real-world datasets

	LE	LPP	Isomap	ILPP-L1	LAPP	MDS	t-SNE	CPLE
abta	6.44±0.01	5.01±1.26	6.85±0	8.57±0.47	**8.62±0.04**	8.24±0.05	7.04±0.93	8.40±1.21
alphabet	6.53±0.06	0.13±0	2.31±0.16	1.71±2.15	-1.34±0	3.63±0.62	2.91±1.24	**7.51±1.20**
breastEW	69.91±0	6.52±0	**71.16±0**	13.48±4.80	57.46±0.21	70.03±0.92	64.41±6.77	70.08±0.93
isolet2	2.59±0	0±0	2.35±0	0.06±0.10	0.74±0	0.15±0.12	1.88±0.33	**3.09±0.21**
mfeat-mor	46.41±1.79	27.88±0.19	54.86±0.10	54.31±1.26	52.06±0.74	53.10±0.15	50.77±0.44	**54.98±0.02**
psis	21.67±0	3.15±0	45.70±0	42.38±9.88	-2.69±0.78	36.65±0	1.07±4.62	**47.09±9.42**
segmentation	31.32±0	40.10±0.15	37.43±0.06	25.92±8.36	41.77±0.17	44.89±0.40	43.46±0.81	**50.69±3.53**
spliceEW	0.55±0	1.16±0.10	7.39±0.92	3.41±1.14	2.15±0.05	2.93±0.73	3.41±1.62	**8.99±2.82**
ver2	23.08±0	**27.08±0.36**	24.78±0	16.42±5.29	1.38±0	22.53±1.31	14.11±3.00	23.86±4.45
wdbc	**76.08±0**	39.46±0.58	73.61±0	44.17±20.36	45.16±0	70.22±0.71	70.91±4.72	74.87±0

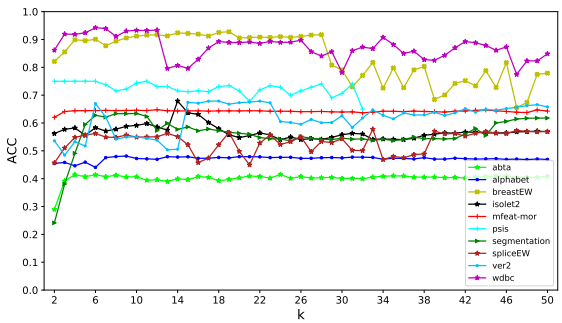


Fig. 11 ACC performance of CPLE w.r.t. k.

Table 3-5 show the clustering metrics (NMI[36], ACC[37], ARI[38]) obtained from the experiments of clustering with the considered algorithms on 10 real-world datasets. In the tables, bold text means the best average metric value for a data set among the compared methods. In total, there are 18 cases where CPLE has the best average result compared to other baseline methods. Based on t-test at 5% significance level, the improvement is also significant in 97% of the cases where CPLE outperforms a baseline method. For those cases where the difference between CPLE and a baseline method is insignificant, the main reason is that the variation of CPLE is relatively large, e.g., on the ver2 dataset, LPP has an average ACC of 54.97 and CPLE 55.00, but the variance of CPLE is 4.47 which is larger than 0.30 of LPP. From the NMI metrics in Table 3, we can see that the CPLE performs better on most of the datasets, and CPLE can generally improve by more than four or five percentage points compared to the traditional LE method. Similarly, in Table 4 ACC metrics and Table 5 ARI metrics, CPLE still performs well on most of the datasets, indicating that our improved method is effective.

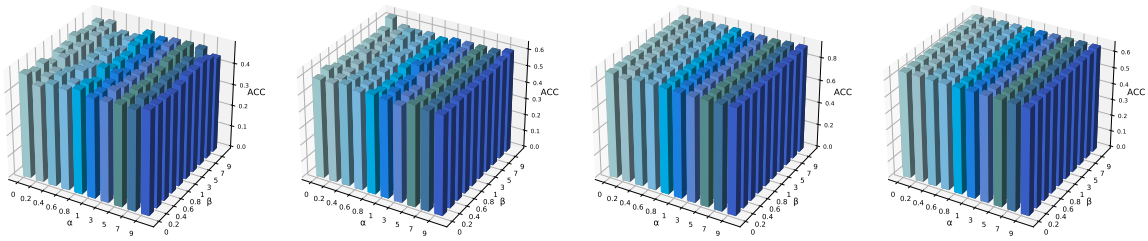
5 Detailed analysis of CPLE

5.1 Effects of hyperparameters

There are three major hyperparameters in the proposed CPLE algorithm, i.e., the number of nearest neighbours k, and the two weighting parameters α and β in Eq. 5 and Eq. 8, respectively. In the following, we will demonstrate the effects of these three hyperparameters on the performance of CPLE. All the results here are averaged over 10 runs of CPLE on each dataset.

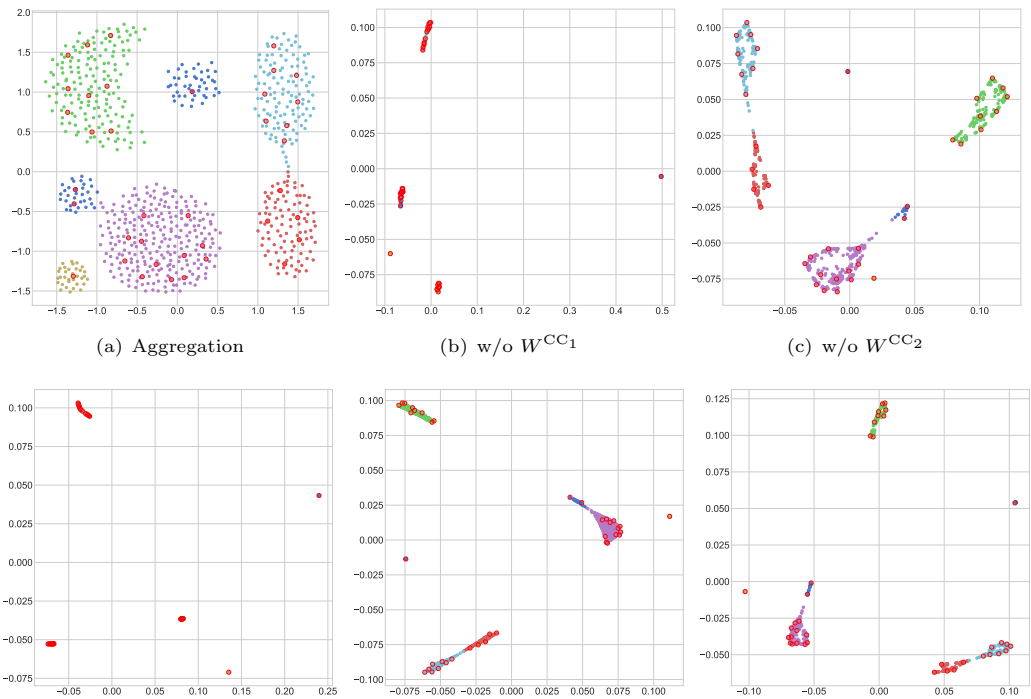
5.1.1 Effects of k

In practice, a common range of k is $[2, 50]$. For each of the values in this range, the results of the ACC metric on the real-world datasets are shown in Fig. 11, where the other two hyperparameters are fixed as $\alpha = 5$ and $\beta = 5$ like in previous experiments on real-world datasets. Note that different k can reflect different levels of structures: a too small k will result in a too detailed structure, where almost every point is separated with others; a too large k will result in a too coarse structure, where almost every point is connected with others. Therefore, we expect that the performance of CPLE will change when different k was used. As seen in Fig. 11, when k is too small, the performance of CPLE on several datasets (e.g., segmentation, breastEW, and spliceEW) is relatively lower than that when k is moderately larger. When k becomes too large, on most of the datasets, the performance will start to fluctuate largely, because components might get combined in an undesirable way. This indicates that it is



(a) segmentation (b) isolet2 (c) breastEW (d) mfeat-mor

Fig. 12 ACC performance of CPLE w.r.t. α and β.



(a) Aggregation (b) w/o W^{CC_1} (c) w/o W^{CC_2}

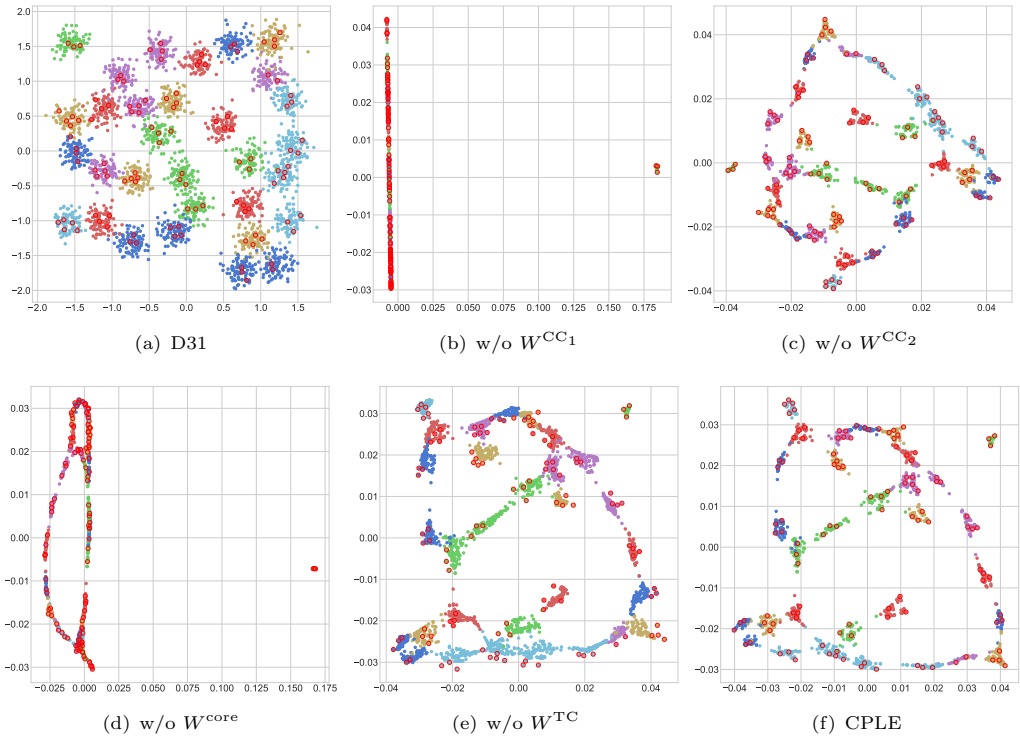
(d) w/o W^{core} (e) w/o W^{TC} (f) CPLE

Fig. 13 Ablation study results on Aggregation.

important to choose a reasonable value of k, which usually lies between 5 and 15.

5.1.2 Effects of α and β

In Eq. 5, α controls the contribution of W^{TC}, which is the similarity measure between a point and its core point. This similarity measure sometimes has an overlap with the similarity measure (W^{TT}) between a point and its neighbours, depending on the specific structure of the corresponding dataset. Therefore, the value of α will have effects on the performance, but sometimes the effect is not so significant. In Eq. 8, β controls the contribution of W^{CC_2}, which is the similarity measure between two core points that are in the same component. In other words, β will affect how much of the structural information within a component will be maintained. Generally speaking, this structural information will only have a subtle effect on the clustering performance of the reconstructed data. Fig. 12 shows the results of the ACC metric of the CPLE algorithm by varying α and β on four datasets. As can be seen from the figure, for the first two datasets, the effects of



(a) D31 (b) w/o W^{CC_1} (c) w/o W^{CC_2}

(d) w/o W^{core} (e) w/o W^{TC} (f) CPLE

Fig. 14 Ablation study results on D31.

α and β are more significant than that for the last two datasets. In practice, one can choose a moderate value of α or β (e.g., 5) if the structure of the dataset is not clear.

5.2 Ablation study

By characterizing the structure of data from different aspects, with different similarities, CPLE solves the problems of over-compression of data points and random changes of the relative positions between connected components for data reconstruction. To evaluate the effectiveness of different similarities in CPLE, we compare the performance of the full model of CPLE against the four variants by removing W^{TC}, W^{CC_1}, W^{CC_2}, and W^{core}, respectively.

5.2.1 Ablation study on synthetic datasets

First, we compare the results on synthetic datasets, to get intuitions about how different similarities help. Fig. 13 and Fig. 14 are the visualization results. We can draw the following conclusions:

- W^{CC_1} is the most important factor to obtain good reconstruction performance. As shown in (b) of Fig. 13 and Fig. 14, when W^{CC_1} is absent, the algorithm reconstructs almost every component into a single point (over-compression), and the relative positions of components change randomly in reconstruction results. The reason is that W^{CC_1} characterizes the relative positions between core points and each connected component contain several core points (red circles in the figure).
- W^{CC_2} plays a certain role in maintaining the structure within the component. This is expected, because this similarity measure considers the similarities of core points within the same component, by the use of the shortest path distance on the connection graph. As shown in (c) of Fig. 13 and Fig. 14, when W^{CC_2} is missing, the sub-clusters and the core points which are previously close to each other become separated after reconstruction, because the similarities between core points within the same component are not correctly characterized.
- W^{core} is a weighted sum of W^{CC_1} and W^{CC_2}, so removing it means removing the similarities

Table 6 Ablation study on real-world datasets

Datasets	indexs	Methods				
		w/o W^{TC}	w/o W^{CC_1}	w/o W^{CC_2}	w/o W^{core}	CPLE
abta	ARI	**0.092**	0.058	0.081	0.064	0.084
	NMI	**0.119**	0.082	0.108	0.084	0.110
	ACC	0.407	0.376	0.401	0.382	**0.409**
alphabet	ARI	0.055	0.067	0.061	0.067	**0.075**
	NMI	0.045	0.067	0.057	0.065	**0.068**
	ACC	0.462	**0.473**	0.466	0.470	**0.473**
breastEW	ARI	0.651	0.699	0.673	**0.705**	0.701
	NMI	0.605	0.634	0.625	0.639	**0.647**
	ACC	0.904	0.919	0.912	**0.921**	0.920
isolet2	ARI	0.044	0.034	0.029	**0.053**	0.031
	NMI	0.034	0.027	0.023	**0.040**	0.024
	ACC	0.601	0.591	0.586	**0.612**	0.589
mfeat-mor	ARI	**0.550**	0.549	**0.550**	0.544	**0.550**
	NMI	0.684	0.688	0.688	0.681	**0.689**
	ACC	0.645	**0.646**	**0.646**	0.645	**0.646**
psis	ARI	0.402	0.469	**0.529**	0.506	0.471
	NMI	0.493	0.547	**0.614**	**0.614**	0.563
	ACC	0.688	0.722	**0.744**	0.716	0.719
segmentation	ARI	0.483	0.373	0.493	0.343	**0.507**
	NMI	0.590	0.531	0.597	0.512	**0.608**
	ACC	0.614	0.512	**0.631**	0.494	0.630
spliceEW	ARI	**0.092**	0.075	0.083	0.069	0.090
	NMI	0.051	0.051	0.053	0.053	**0.060**
	ACC	0.534	0.554	0.548	0.547	**0.557**
ver2	ARI	**0.251**	0.215	0.226	0.226	0.239
	NMI	0.298	0.343	0.360	0.360	**0.361**
	ACC	0.527	0.525	0.536	0.535	**0.550**
wdbc	ARI	0.705	0.748	0.744	0.732	**0.749**
	NMI	0.593	0.638	0.634	0.622	**0.640**
	ACC	0.920	**0.933**	0.932	0.928	**0.933**

between all core points. As a result, in (d) of Fig. 13 and Fig. 14, the sub-clusters and core points are over-compressed after reconstruction.

- W^{TC} has only a slight impact on the overall reconstruction result, but has a strong impact on the relative position of core points to other points. As shown in (e) of these two figures, especially Fig. 14, we find that after reconstruction, a large portion of core points are located at the boundary of or far away from the core part of sub-clusters. This reflects that the reconstructed data have different density structures from the original data, which can be undesirable.

In summary, the multiple similarity matrices proposed in CPLE algorithm play a certain role in the whole algorithm.

5.2.2 Ablation study on real-world datasets

We also carried out comparisons on the real-world datasets, as shown in Table 6, where the metrics are averaged over 10 runs of the methods for each dataset. We have the following observations for Table 6:

- The full CPLE model can achieve the best results in most of the cases, and when it is not the best, it is usually close to the best. This indicates that the improvements made by CPLE algorithm are effective.
- W^{CC_1} is the most important element in the model, because without it the performance of the algorithm drops drastically.
- The absence of W^{CC_2} and W^{TC} have only subtle effects on the final result, but they are still important to maintain specific structures, e.g., the relative distances of sub-clusters within the same component and certain density structures of each sub-cluster, as analyzed in the experiments on synthetic datasets.

6 Conclusion

In this article, we propose the LE-based dimensionality reduction and reconstruction method for multi-component data based on data core points (CPLE), which solves the problem that the traditional LE method will overly "compress" the dataset with multiple connected components. In order to keep the similarity structure within the components as much as possible while maintaining the relative position relationship between the components, we modified the objective function of the LE method based on the classical LE method. First, we introduce core points to describe the structure of data and optimize the objective to increase the similarity between common nodes and their core nodes. Secondly, we present a global similarity matrix between core nodes to describe the relative positions between components. Finally, to prevent the data from being over-compressed, we use the gradient descent method instead of the traditional spectral decomposition method to update the information of the similarity within the class and the similarity between the classes. At last, we have achieved good results in both synthetic data visualization and real-world data clustering analysis. The advantages of the proposed method over existing LE variants mainly lie in two aspects: the relative positions between components are properly maintained, and the reconstructed data are no longer over-compressed, which helps to retain important structural information of data after reconstruction. Still, there are several limitations of the proposed method. It is an iterative method, so it may need to iterate many times before a satisfactory result is obtained, and the result obtained is just a local approximation of the actual solution of the corresponding optimization problem. There are also several hyperparameters (e.g., $k, \alpha, \beta, \sigma_1, \sigma_2$, and ϵ) that can affect the performance significantly and be hard to tune in practice, although we recommended default values for them based on experimental results and experience.

As for future work, note that here we considered maintaining relative positions between components, whereas there are many other important information within or between components or even other types of structures, e.g., holes or loops in components, distribution of points in a component, and ordering of the neighbours of a point. Also, the model here did not give an explicit function to reconstruct data, so it has to re-calculate the reconstruction for new data. To alleviate this problem, one can combine a neural network structure with the objective function to approximate the nonlinear mapping between original and reconstructed data, so the mapping can be reused for new data. Explorations on how to accomplish this and the performance of this idea will be interesting for future work.

Acknowledgements. This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/10.1007/S10489-023-05012-6.

This work was supported by the National Natural Science Foundation of China [grant numbers 11871353, 12231007, 61806170], the National Key Research and Development Program of

China [grant number 2019YFB1706104], and the Fundamental Research Funds for the Central Universities [grant numbers 2682022ZTPY082, 2682023ZTPY027]. We would also like to thank the anonymous reviewers for their helpful comments and suggestions to improve the article.

Declarations

Competing interest. The authors declare that they have no competing interest to this work.

Author contribution statement. Hua Meng: Conceptualization, Methodology, Supervision, Writing - Reviewing and Editing. **Hanlin Zhang**: Software, Visualization, Data curation, Investigation, Validation, Writing-Reviewing and Editing. **Yu Ding**: Visualization, Data curation. **Shuxia Ma**: Supervision, Project administration, Writing-Reviewing and Editing. **Zhiguo Long**: Conceptualization, Formal analysis, Supervision, Writing-Original Draft, Funding acquisition.

Ethical and informed consent for data used. The data used in this article are either from public datasets or generated by the authors. The research in this article does not involve humans or animals.

Data availability and access. The datasets generated and analysed during the current study are available from the corresponding author on reasonable request.

References

[1] Nie, F., Wang, Z., Wang, R., Li, X.: Adaptive local embedding learning for semi-supervised dimensionality reduction. IEEE Transactions on Knowledge and Data Engineering **34**(10), 4609–4621 (2022)

[2] Das, S., Pal, N.R.: Nonlinear dimensionality reduction for data visualization: An unsupervised fuzzy rule-based approach. IEEE Transactions on Fuzzy Systems **30**(7), 2157–2169 (2022)

[3] Wang, R., Nie, F., Hong, R., Chang, X., Yang, X., Yu, W.: Fast and orthogonal locality preserving projections for dimensionality reduction. IEEE Transactions on Image Processing **26**(10), 5019–5030 (2017)

[4] Leem, S., Park, T.: An empirical fuzzy multifactor dimensionality reduction method for detecting gene-gene interactions. BMC Genomics **18**(2), 1–12 (2017)

[5] Abdi, H., Williams, L.J.: Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics **2**(4), 433–459 (2010)

[6] Balakrishnama, S., Ganapathiraju, A.: Linear discriminant analysis-a brief tutorial. Institute for Signal and Information Processing **18**(1998), 1–8 (1998)

[7] Wang, S., Lu, J., Gu, X., Du, H., Yang, J.: Semi-supervised linear discriminant analysis for dimension reduction and classification. Pattern Recognition **57**, 179–189 (2016)

[8] Schölkopf, B., Smola, A., Müller, K.-R.: Kernel principal component analysis. In: International Conference on Artificial Neural Networks, pp. 583–588 (1997)

[9] Mika, S., Ratsch, G., Weston, J., Scholkopf, B., Mullers, K.-R.: Fisher discriminant analysis with kernels. In: IEEE Workshop on Neural Networks for Signal Processing, pp. 41–48 (1999)

[10] Cox, M.A., Cox, T.F.: Multidimensional Scaling vol. 1. Springer, New York, USA (2008)

[11] Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research **9**(11) (2008)

[12] Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation **15**(6), 1373–1396 (2003)

[13] McInnes, L., Healy, J., Melville, J.: Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426 (2018)

[14] Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)

[15] Tenenbaum, J.B., Silva, V.d., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)

[16] Tan, C., Chen, S., Geng, X., Ji, G.: A novel label enhancement algorithm based on manifold learning. Pattern Recognition **135**, 109189 (2023)

[17] Valem, L.P., Pedronette, D.C.G., Latecki, L.J.: Rank flow embedding for unsupervised and semi-supervised manifold learning. IEEE Transactions on Image Processing **32**, 2811–2826 (2023)

[18] Gao, Z., Wu, Y., Fan, X., Harandi, M., Jia, Y.: Learning to optimize on riemannian manifolds. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(5), 5935–5952 (2023)

[19] Liu, C., JaJa, J., Pessoa, L.: LEICA: Laplacian eigenmaps for group ICA decomposition of fMRI data. NeuroImage **169**, 363–373 (2018)

[20] Ye, X., Li, H., Imakura, A., Sakurai, T.: An oversampling framework for imbalanced classification based on Laplacian eigenmaps. Neurocomputing **399**, 107–116 (2020)

[21] Li, B., Li, Y.-R., Zhang, X.-L.: A survey on Laplacian eigenmaps based manifold learning methods. Neurocomputing **335**, 336–351 (2019)

[22] He, X., Niyogi, P.: Locality preserving projections. Advances in Neural Information Processing Systems **16** (2003)

[23] Yu, W., Wang, R., Nie, F., Wang, F., Yu, Q., Yang, X.: An improved locality preserving projection with l_1-norm minimization for dimensionality reduction. Neurocomputing **316**, 322–331 (2018)

[24] Wang, A., Zhao, S., Liu, J., Yang, J., Liu, L., Chen, G.: Locality adaptive preserving projections for linear dimensionality reduction. Expert Systems with Applications **151**, 113352 (2020)

[25] Nie, F., Zhu, W., Li, X.: Unsupervised large graph embedding based on balanced and hierarchical k-means. IEEE Transactions on Knowledge and Data Engineering **34**(4), 2008–2019 (2022)

[26] Lu, X., Long, J., Wen, J., Fei, L., Zhang, B., Xu, Y.: Locality preserving projection with symmetric graph embedding for unsupervised dimensionality reduction. Pattern Recognition **131**, 108844 (2022)

[27] Tai, M., Kudo, M., Tanaka, A., Imai, H., Kimura, K.: Kernelized supervised Laplacian Eigenmap for visualization and classification of multi-label data. Pattern Recognition **123**, 108399 (2022)

[28] Zhu, H., Sun, K., Koniusz, P.: Contrastive Laplacian Eigenmaps. In: Advances in Neural Information Processing Systems, pp. 5682–5695 (2021)

[29] Donoho, D.L., Grimes, C.: Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. Proceedings of the National Academy of Sciences **100**(10), 5591–5596 (2003)

[30] Chen, B., Gao, Y., Wu, S., Pan, J., Liu, J., Fan, Y.: Soft adaptive loss based laplacian eigenmaps. Applied Intelligence **52**(1), 321–338 (2022)

[31] Zhang, H., Ding, Y., Meng, H., Ma, S., Long, Z.: Component preserving and adaptive laplacian eigenmaps for data reconstruction and dimensionality reduction. In: International Conference on Intelligent Systems and Knowledge Engineering, pp. 642–649 (2022)

[32] Von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing **17**(4), 395–416 (2007)

[33] Rodriguez, A., Laio, A.: Clustering by fast search and find of density peaks. Science **344**(6191), 1492–1496 (2014)

[34] Long, Z., Gao, Y., Meng, H., Yao, Y., Li, T.: Clustering based on local density peaks and

graph cut. Information Sciences **600**, 263–286 (2022)

[35] Gao, B., Liu, X., Yuan, Y.: Parallelizable algorithms for optimization problems with orthogonality constraints. SIAM Journal on Scientific Computing **41**(3), 1949–1983 (2019)

[36] Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 267–273 (2003)

[37] Yang, Y., Xu, D., Nie, F., Yan, S., Zhuang, Y.: Image clustering using local discriminant models and global integration. IEEE Transactions on Image Processing **19**(10), 2761–2773 (2010)

[38] Steinley, D.: Properties of the Hubert-Arable Adjusted Rand Index. Psychological Methods **9**(3), 386–396 (2004)