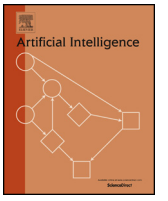
Artificial Intelligence 320 (2023) 103924

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint



Spectral clustering with robust self-learning constraints

Liang Bai, Minxue Qi, Jiye Liang *

Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Institute of Intelligent Information Processing, Shanxi University, Taiyuan, 030006, Shanxi, China

ARTICLE INFO

Article history:
Received 10 May 2022
Received in revised form 13 April 2023
Accepted 15 April 2023
Available online 19 April 2023

Keywords:
Cluster analysis
Spectral clustering
Self-learning constraints
Robustness

ABSTRACT

Spectral clustering is a leading unsupervised classification algorithm widely used to capture complex clusters in unlabeled data. Additional prior information can further enhance the quality of spectral clustering results to satisfy users' expectations. However, it is challenging for users to find the prior information under unsupervised scenes. To get rid of the deficiency, we propose a spectral clustering model with robust self-learning constraints. In this model, we first extend the optimization problem of spectral clustering by seeing label constraints as variables to learn the constraints and the clustering result simultaneously. Furthermore, we add a robust term to the proposed model so that we can learn multiple groups of label constraints to guide the clustering process and find a robust self-constrained spectral clustering result. The robust term can reduce the impact of uncertainty in the quality of a single set of label constraints on the performance of the proposed model. An iterative strategy with update formulas for variables is proposed to solve the self-constrained spectral clustering problem. We provide the theoretical analysis to explain the importance of the learned constraints in spectral clustering. Furthermore, we analyze the convergence of our optimization scheme. Finally, we have done many experiments on benchmark data sets to illustrate the effectiveness of the proposed algorithm.

1. Introduction

Clustering is an important field in machine learning and artificial intelligence [1]. The goal of clustering is to identify objects that look similar into a common cluster and discover patterns from huge data like humans. To solve this problem, various types of clustering algorithms have been developed in the literature (e.g., [2] and references therein).

Spectral clustering (SC) [3,4] is a representative of graph clustering. It has shown greater promise than other traditional clustering algorithms in learning hidden nonlinear structures from data. It transforms a clustering problem into a graph-partitioning problem and then uses the spectrum (eigenvalues) of the graph to learn the label features of data. Since it exploits nonlinear pairwise similarity between data, it can recognize different shapes of clusters. Currently, many improved spectral clustering methods have been developed to enhance the performance of spectral clustering, which can be found in Section 2. However, since spectral clustering works without supervision information, its clustering result may differ from the users' expectations. Many studies [5] have demonstrated that even a small amount of supervision information can lead to significant improvements in the performance of spectral clustering. In the field of machine learning, label and pairwise

* Corresponding author.
E-mail addresses: bailiang@sxu.edu.cn (L. Bai), qiminxuert@qq.com (M. Qi), ljy@sxu.edu.cn (J. Liang).

https://doi.org/10.1016/j.artint.2023.103924
0004-3702/

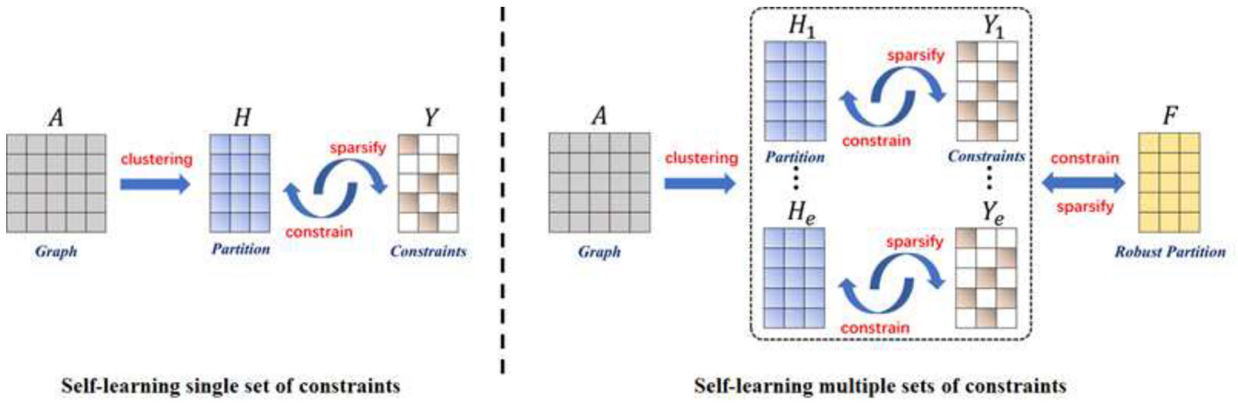


Fig. 1. Self-constrained spectral clustering.

constraints are two widely used types of supervision information. The relationship between these two types of constraints has been explored by [6], where it was shown that label constraints may be converted into pairwise constraints. Compared to label constraints, pairwise constraints are weak supervision signals. Currently, many semi-supervised spectral clustering algorithms with different types of supervision information have been developed, which are detailed in Section 2. The supervision information can provide additional discriminative information to improve the clustering accuracy in semi-supervised clustering algorithms. Unfortunately, the semi-supervised clustering results are sensitive to the quality of prior supervision information. Inexact supervision information often can not improve the clustering result but reduce its effectiveness. Besides, it is very challenging for users to obtain some prior supervision information from a data set under an unsupervised scene.

To solve the problem, we try to learn label constraints automatically from unlabeled data and then convert a spectral clustering problem into a self-supervised clustering problem. Based on the idea, we develop a spectral clustering model with self-learning constraints, where label constraints are seen as variables and learned by using a sparse regularization term. After label constraints are extracted automatically, this model can make use of semi-supervised learning techniques to improve the clustering results. However, the performance of this model is very sensitive to the quality of the learned constraints. Since there is certain uncertainty in the learning process of label constraints, we can not guarantee that the learned constraints are of high quality. To overcome the shortcoming, we further build a robust self-constrained spectral clustering model which can learn multiple sets of label constraints to guide the clustering process. Compared to a single set of self-learning constraints, multiple sets can help us to get a more robust clustering result. The diagram of self-constrained spectral clustering with single and multiple sets of self-learning constraints is shown in Fig. 1. According to this figure, we can see their difference. Furthermore, we provide the theoretical and experimental analysis to illustrate the effectiveness of the proposed model.

The main contributions of this paper are described as follows:

- We build a robust self-constrained spectral clustering model which learns multiple sets of label constraints to guide the spectral clustering process and obtain the robust clustering result.
- We derive the update formulas for different variables and propose an iterative method to solve the optimization problem of spectral clustering with robust self-learning constraints.
- We provide the theoretical analysis to investigate the importance of the learned constraints for spectral clustering. Furthermore, we analyze the convergence of the proposed algorithm.
- By the experimental analysis, we illustrate the effectiveness of the proposed algorithm on the benchmark data sets.

The outline of the rest of this paper is as follows. Section 2 reviews the related work of spectral clustering. Section 3 introduces the preliminaries of spectral clustering and label propagation. Sections 4 and 5 present spectral clustering with single and multiple sets of self-learning constraints, respectively. Section 6 shows the description of the proposed algorithm. Sections 7 and 8 provide the theoretical and convergence analysis for the proposed algorithm, respectively. Section 9 demonstrates the performance of the proposed algorithm. Section 10 concludes the paper with some remarks.

2. Related work

According to the supervision scenarios, we introduce the related works from three parts, i.e., unsupervised spectral clustering, semi-supervised spectral clustering, and self-supervised clustering, which are reviewed as follows.

(1) *Unsupervised spectral clustering:* There are two key factors influencing the performance of the spectral clustering algorithm, namely, the quality of the pairwise similarity matrix and the expensive computational cost. Different definitions of the similarity matrix often result in spectral clustering outputs of varying qualities. To address this issue, many studies

have focused on learning an appropriate pairwise similarity matrix from data for spectral clustering. For instance, sparse subspace clustering [7–10] utilizes a self-representation optimization model to learn a sparse similarity matrix. In [11], a low-rank similarity matrix was learned for subspace clustering. Zhang et al. [12] further extended this method to learn the affinity matrix from the low-dimensional space of the original data. In [13–15], the authors proposed to learn a Laplacian-rank similarity matrix with precisely connected components. In addition to improving the similarity matrix, many studies focus on how to reduce the calculation cost for spectral clustering. Dhillon et al. [16] demonstrated the equivalence between spectral clustering and kernel k-means, and used the iterative optimization method of kernel k-means instead of Eigen decomposition to solve the spectral clustering problem. Liu et al. [17] applied this technique to the cluster ensemble problem. Moreover, various methods have been proposed to compress the original graph into a sparse sub or bipartite graph, in order to reduce the time cost of spectral clustering, such as [18–25].

(2) *Semi-supervised spectral clustering:* Currently, different types of semi-supervised spectral clustering algorithms have been developed, which can make use of additional prior information to improve the spectral clustering results. For example, Zhou et al. [5] proposed a label propagation algorithm that can be seen as spectral clustering with Positive-Label constraints. Zoidi et al. further extended the label propagation algorithm to propose a version with Negative-Label constraints [26]. Bai et al. proposed a label propagation with pairwise constraints [24]. Furthermore, they developed a spectral clustering algorithm with the integration of different types of constraints [6]. Besides, other types of constrained spectral clustering algorithms have also been developed in [27–29]. Although these methods can improve the spectral clustering results, their performance depends on the quality of prior information. Poor supervision information often brings bad clustering results. However, finding high-quality prior information requires high costs. Besides, it is difficult for users to discover good prior information in many scenarios, especially unsupervised scenarios.

(3) *Self-supervised clustering:* Different self-learning paradigms, such as label learning and pairwise learning, have been developed to tackle the insufficiency of discriminative information [30]. Each paradigm has its own application scenarios. Pairwise learning, for example, is suitable when the number of clusters is unknown on a dataset. When learning pairwise relations between objects, we do not need to know the number of clusters and only consider whether they belong to the same clusters. However, if the number of clusters is given and is much smaller than the number of objects in a dataset, the computational cost of learning labels is lower than that of pairwise learning. Based on these paradigms, several deep clustering algorithms have been proposed that self-learn label or pairwise constraints from unlabeled data to train deep neural networks for clustering tasks [31–34]. Deep embedded clustering (DEC) [34] and joint unsupervised learning (JULE) [35] are the early representatives of deep clustering models based on label learning. Lv et al. [36] learned pseudo-labels to train deep subspace clustering network [37]. Li et al. [38] adopted a confidence-based criterion to select pseudo-labels for boosting contrastive clustering. In [39], a deep spectral clustering network was proposed to learn pairwise similarity between data points to enhance the performance of spectral clustering. Chang et al. proposed a deep self-evolutionary clustering (DSEC) [40] which uses the similarity between points as supervision information. Compared to traditional clustering algorithms, these deep methods can enhance the clustering results' effectiveness by self-supervision. However, their performance strongly depends on the capability of deep neural networks for data representation, which requires expensive training costs, such as parameter tuning, sufficient data, large storage space, and time costs.

3. Preliminaries

In this section, we first give some notations used in this paper. For any matrix M, its element of the ith row and the jth column is represented by $[M]_{ij}$, its ith row is represented by $[M]_{i.}$, its jth column is represented by $[M]_{.j}$. The trace of M is denoted as $Tr(M)$, and the transpose of M is denoted as M^T. $diag(M)$ is the diagonal matrix of M. $||M||_F$ is Frobenius norm of M. Next, we briefly introduce some base concepts of spectral clustering and label propagation.

3.1. Spectral clustering

Let X be a $n \times d$ data matrix with n objects and d features, x_i be the ith row of X which is used to represent the ith object. Given X, people can use a similarity measure to get its affinity matrix A. In general, Gaussian kernel is used to define A as follows.

$$A_{ij} = \exp\left(-\frac{||[X]_{i.} - [X]_{j.}||^2}{\delta}\right), \tag{1}$$

where δ is a kernel parameter. Spectral clustering is to see A as a graph and find its partition such that the sum of weights of edges between the two sets is minimized. Its objective function is described as

$$\min_H Tr(H^T L H), s.t., H^T H = I, \tag{2}$$

where $L = I - \hat{A}$ is a normalized Laplacian matrix and H is a $n \times k$ membership matrix. $\hat{A}$ is the normalized similarity matrix $D^{-1/2} A D^{-1/2}$ or $D^{-1} A$, D is a diagonal matrix whose entries are row sums of A. The spectral clustering problem is the standard trace minimization problem which is solved by the matrix H by containing the first k eigenvectors of L as columns.

3.2. Semi-supervised spectral clustering

Label propagation [5] is the representative of semi-supervised spectral clustering methods, which uses additional prior information, i.e., label constraints, to enhance the performance of spectral clustering. Its optimization function Ω is described as

$$\min_{H} Tr(H^T LH) + \alpha||H - Y||_F^2, \tag{3}$$

where Y is a $n \times k$ pre-given label constraint matrix, which reflects the relations between objects and clusters. If the ith object belongs to the lth cluster, Y_{il} is set to 1; otherwise, 0. α is a parameter that is used to balance the importance of each term in the objective function. It is set to 0.01 by default. If $\alpha = 0$, the function is equivalent to spectral clustering. In [5], the authors provided the optimal solution of Eq. (3), which is described as follows. Differentiating Ω with respect to H, we have

$$\frac{\partial \Omega}{\partial H} = 2(H - \hat{A}H) + 2\alpha(H - Y) = 0. \tag{4}$$

Its closed-form solution $\tilde{H}$ is

$$\tilde{H} = \frac{\alpha}{1+\alpha}(I - \frac{1}{1+\alpha}\hat{A})^{-1}Y. \tag{5}$$

Based on Eq. (5), we can get a clustering result H with pre-given label constraints Y. In general, people do not directly compute $(I - \frac{1}{1+\alpha}\hat{A})^{-1}$ but get $\tilde{H}$ by iterative updating formula

$$H = \frac{1}{1+\alpha}\hat{A}H + \frac{\alpha}{1+\alpha}Y. \tag{6}$$

According to Eq. (6), we can see H is a non-negative matrix if Y is required to be non-negative. The non-negative property corresponds to the meanings represented by H and Y. Because each element in H or Y reflects the membership of an object to a cluster, which is assumed to be non-negative in many clustering algorithms.

From Eq. (6), it is evident that H is non-negative when we require Y to be non-negative. The non-negative property corresponds to the meanings represented by H and Y. This is because each element in either H or Y represents an object's membership in a cluster, which is assumed to be non-negative in many clustering algorithms.

4. Spectral clustering with self-learning constraints

We can extend the objective function of spectral clustering by seeing label constraints as variables to simultaneously learn the label constraints and the clustering result. The new objective function is defined as

$$\min_{H,Y} Tr(H^T LH) + \alpha||H - Y||_F^2 + \eta||Y||_{2,1}, \tag{7}$$

where $||Y||_{2,1}$ is a regularization norm of $L_{2,1}$ to make Y sparse and η is a parameter. $L_{2,1}$-norm regularization was proposed for multi-task feature selection [41,42]. It is a combination between L_2-norm and L_1-norm to control the sparsity of columns and rows of a feature matrix, respectively. For variable Y, each of its columns represents a class label. Since there is some overlap between class labels, we need a smooth regular term, i.e., L_2-norm, to sparse the column values in each row. Each row of Y represents an object. We only require some objects to get high-credible label constraints, and the label constraints of other objects are deleted. Then we choose a strongly sparse regularization, i.e., the norm of L_1, to constrain each row of Y.

This optimization problem forms a class of nonconvex optimization problems. To minimize it, we can randomly initialize Y and iteratively update H and Y to get its approximate solution. Next, we introduce the specific update formulas for H and Y, respectively.

Updating H: When Y is fixed, the optimization problem (7) becomes a label propagation problem [5], i.e., Eq. (3). Therefore, the update formula of H equals to Eq. (5).

Updating Y: When H is fixed, the optimization problem (7) is reduced to a problem

$$\min_{Y} \Delta = \alpha||H - Y||_F^2 + \eta||Y||_{2,1}. \tag{8}$$

Thus, minimizing it becomes a classical problem of $L_{2,1}$-norm regularization whose solving method is described as follows [11]. Differentiating Eq. (8) with respect to Y, we have

$$\frac{\partial \Delta}{\partial Y} = 2\alpha(Y - H) + 2\eta UY, \tag{9}$$

where $U = diag(1/||[Y]_{i.}||_2)$. Therefore, for $1 \le l \le e$, Y is updated by the following formula

$$[Y]_{i.} = \begin{cases} \left(1 - \dfrac{\eta}{\alpha||[H]_{i.}||_2}\right)[H]_{i.}, & \alpha||[H]_{i.}||_2 > \eta, \\ 0, & otherwise. \end{cases} \tag{10}$$

As seen from Eq. (10), the updating formula of Y is equal to the sparsification of H. This operation enables us to extract reliable label constraints from H, which we use to update Y. Moreover, we note that if the initial value of Y is non-negative, then both the updated Y and H are non-negative.

5. Spectral clustering with robust self-learning constraints

The solution of self-constrained spectral clustering is sensitive to the initialization of Y. Before the proposed model runs, we need to randomly initialize Y which will provide the first guidance for the clustering task. However, a poor initialization may result in incorrect guidance and subsequently low-quality learned label constraints. If we only learn a group of label constraints, the clustering result of the proposed model is sensitive to the quality of the learned label constraints. Therefore, to mitigate this issue, we incorporate a robust function into Eq. (7). Learning multiple sets (groups) of label constraints can correct erroneous initialization information to some extent that can guide the spectral clustering process in a more accurate and robust manner.

The robust function is defined as

$$\Phi = \min_{F,G} Tr(F^T LF) + \beta \sum_{l=1}^{e} ||Y_l - FG_l||_F^2, \tag{11}$$

where β is a parameter, e is the number of learned label constraints, Y_l is the lth matrix of label constraints, F is the final clustering result, $G = [G_l]_{l=1}^e$ where G_l is a relation matrix between F and Y_l. $\sum_{l=1}^e ||Y_l - FG_l||_F^2$ is a consensus measure to evaluate the difference between the final clustering result F and each matrix of the learned label constraints Y_l. We wish to minimize the robust function to find the most consensus clustering result with all the label constraints.

Based on Eqs. (7) and (11), the objective function of robust self-constrained spectral clustering is defined as

$$\begin{aligned} \min_{H,Y,F,G} \Omega = &\sum_{l=1}^{e} \left[Tr(H_l^T LH_l) + \alpha||H_l - Y_l||_F^2 \right] + \eta||Y||_{2,1} \\ &+ Tr(F^T LF) + \beta \sum_{l=1}^{e} ||Y_l - FG_l||_F^2, \\ s.t., F \ge 0, H_l \ge 0, Y_l \ge 0, G_l \ge 0, \end{aligned} \tag{12}$$

where $H = [H_l]_{l=1}^e$ and H_l is the eth clustering result based on Y_l.

We need to iteratively update these variables to minimize Eq. (12). Next, we introduce the specific optimization process of their update formulas, respectively.

Updating H: When updating H, other variables are fixed. In this case, each Θ_l for H_l is independent and nonnegative, where

$$\begin{aligned} \Theta_l = &Tr(H_l^T LH_l) + \alpha||H_l - Y_l||_F^2 \\ &+ \beta||Y_l - FG_l||_F^2. \end{aligned} \tag{13}$$

Thus, the optimization problem is equal to minimizing each Θ_l. Differentiating Θ_l with respect to H_l, we can obtain

$$\frac{\partial \Theta_l}{\partial H_l} = 2LH_l + 2\alpha(H_l - Y_l) = 0. \tag{14}$$

Thus, for $1 \le l \le e$, H_l is computed by the same equation as Eq. (5), i.e.,

$$H_l = \frac{\alpha}{1+\alpha}(I - \frac{1}{1+\alpha}\hat{A})^{-1} Y_l. \tag{15}$$

Updating Y: Given other variables, the optimization problem becomes minimizing

$$\Delta = \alpha \sum_{l=1}^{e} ||H_l - Y_l||_F^2 + \beta \sum_{l=1}^{e} ||FG_l - Y_l||_F^2 + \eta||Y||_{2,1}. \tag{16}$$

Since

$$\sum_{l=1}^{e} ||H_l - Y_l||_F^2 = ||H - Y||_F^2 \tag{17}$$

and

$$\sum_{l=1}^{e} ||FG_l - Y_l||_F^2 = ||FG - Y||_F^2, \tag{18}$$

we have

$$\Delta = \alpha||H - Y||_F^2 + \beta||FG - Y||_F^2 + \eta||Y||_{2,1}. \tag{19}$$

Thus, minimizing Δ becomes a classical problem of $L_{2,1}$-norm regularization whose solving method is described as [11]. Differentiating Δ with respect to Y, we have

$$\frac{\partial \Delta}{\partial Y} = 2\alpha(Y - H) + 2\beta(Y - FG) + 2\eta UY, \tag{20}$$

where $U = diag(1/||[Y]_{i.}||_2)$. Therefore, for $1 \le l \le e$, Y_l is updated by the following formula

$$[Y_l]_{i.} = \begin{cases} \left(1 - \dfrac{\eta}{||[W]_{i.}||_2}\right)[W_l]_{i.}, & ||[W]_{i.}||_2 > \eta, \\ 0, & otherwise, \end{cases} \tag{21}$$

where $W_l = \alpha H_l + \beta FG_l$ and $W = [W_l]_{l=1}^{e}$.

Updating F and G: According to Eq. (18), we have

$$tr(F^T LF) + \beta \sum_{l=1}^{e} ||Y_l - FG_l||_F^2 = tr(F^T LF) + \beta||Y - FG||_F^2. \tag{22}$$

When H and Y are fixed, the optimization problem becomes minimizing

$$\min_{F,G} tr(F^T LF) + \beta||Y - FG||_F^2, \ s.t.\ F \ge 0, G \ge 0. \tag{23}$$

This problem can be seen as a non-negative matrix factorization with graph regularization [43]. Therefore, we can get the update formulas for F and G as follows.

$$[F]_{ij} \leftarrow [F]_{ij} \frac{[YG^T + \frac{1}{\beta}\hat{A}F]_{ij}}{[FGG^T + \frac{1}{\beta}F]_{ij}} \tag{24}$$

and

$$[G_l]_{ij} \leftarrow [G_l]_{ij} \frac{[F^T Y_l]_{ij}}{[F^T FG_l]_{ij}}. \tag{25}$$

Based on the above updating formulas of H, Y, F, and G, we can iteratively solve the optimization problem in Eq. (12).

6. Algorithm description

A spectral clustering with robust self-learning constraints (RSLC) algorithm is summarized in Algorithm 1. In this algorithm, we consider two cases, i.e., $e > 1$ and $e = 1$. If $e = 1$, we only need to learn a set of label constraints. Thus, in this case, we only update H and Y to return H as the clustering results. If $e > 1$, we need to update H, Y, F, and G to learn multiple sets of constraints and return F as a robust clustering result. For the initialization of Y_l, we only randomly select k objects from a data set and assign different labels for them.

The time complexity of the proposed algorithm is made up of three parts, computing similarity matrix $O(n^2m)$, self-label propagation $O(n^2ket)$, non-negative matrix factorization $O(nket)$, where t is the number of iterations. Therefore, its overall time complexity is $O(n^2ket + nket + n^2m)$. We know that the time complexity of classical spectral clustering with fast eigenvalue decomposition is $O(n^2m + n^2k)$. We can see that the proposed algorithm needs more computational costs than traditional spectral clustering. These additional costs are used to iteratively compute update formulas, which can help us to improve the performance of spectral clustering. However, since the time complexity is quadratic with the number of objects on a data set, it can not efficiently deal with large-scale data sets. Therefore, we need to study the acceleration mechanism of the proposed algorithm in future work to make it suitable for large-scale data.

Algorithm 1: The RSLC algorithm.

Input: A, k, α, β, η, e and t
Output: F
Randomly initialize Y_l, for $1 \le l \le e$;
Repeat
  If $e > 1$
    Update H and Y by Eqs. (15) and (21);
    Update F and G by Eqs. (24) and (25);
  Else
    Update H and Y by Eqs. (5) and (10);
    $F = H$;
  End
Until the desired number of iterations is reached;
Return F;

7. Theoretical analysis

In this section, we try to answer two questions: (1) *Why do the learned constraints can improve the performance of spectral clustering?* (2) *Is spectral clustering with multiple sets of constraints better than that with a single set of constraints?* To address these questions, we provide the generalization and robustness analysis to show the importance of the learned constraints.

7.1. Importance of single set of the learned constraints

We first use the relations between stability and generalization of a learning algorithm (which can be found in [44]) to analyze the role of a single set of the learned constraints in spectral clustering. We first give the notations for the analysis.

Let $X = \{x_1, x_2, \cdots, x_n\}$ and $X^{(i)} = \{x_1, \cdots, x_{i-1}, x', x_{i+1}, \cdots, x_n\}$ which differ at just the ith data point from X. We assume all the data points in $\{x_1, \cdots, x_n, x'\}$ are independent and identically distributed and subject to the same data distribution Z. For a data point x_i, its prediction loss of spectral clustering is described as

$$\begin{aligned} f(H, x_i) =& \mathfrak{L}(H, x_i) + \alpha \mathfrak{R}(H, x_i), \\ where\ \ \mathfrak{L}(H, x_i) =& \sum_{j=1}^{n} [A]_{ij} || \frac{H(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H]_{j.}}{\sqrt{[D]_{jj}}} ||^2, \\ \mathfrak{R}(H, x_i) =& ||H(x_i) - [Y]_{i.}||^2, \end{aligned} \tag{26}$$

where $H(x_i)$ is the representation of data x_i in gained Hilbert space formed by H, and we have $H(x_i) = \sum_{j=1}^{n} [\tilde{A}]_{ij} [H]_{j.}$.

The overall prediction loss is the mean value of $f(H, x_i)$ for all the data points, which is described as

$$\begin{aligned} f(H, X) =& \frac{1}{n} \sum_{x_i \in X} f(H, x_i) = \mathfrak{L}(H, X) + \alpha \mathfrak{R}(H, X), \\ where\ \ \mathfrak{L}(H, X) =& \frac{1}{n} \sum_{x_i \in X} \mathfrak{L}(H, x_i),\ \ and\ \ \mathfrak{R}(H, X) = \frac{1}{n} \sum_{x_i \in X} \mathfrak{R}(H, x_i). \end{aligned} \tag{27}$$

It is easy to verify that this loss function $f(H, X)$ is strongly convex, so we have

$$f(H, X) - f(H', X) \ge \alpha ||H - H'||_F^2 \tag{28}$$

where H and H' are two different clustering results for spectral clustering. For H and H', we have

$$\begin{aligned} f(H, X) - f(H', X) =& [\mathfrak{L}(H, X) + \alpha \mathfrak{R}(H, X)] - [\mathfrak{L}(H', X) + \alpha \mathfrak{R}(H', X)] \\ =& [\mathfrak{L}(H, X^{(i)}) + \alpha \mathfrak{R}(H, X^{(i)}) + \frac{f(H, x_i) - f(H, x')}{n}] \\ & - [\mathfrak{L}(H', X^{(i)}) + \alpha \mathfrak{R}(H', X^{(i)}) + \frac{f(H', x_i) - f(H', x')}{n}] \\ =& [\mathfrak{L}(H, X^{(i)}) + \alpha \mathfrak{R}(H, X^{(i)})] - [\mathfrak{L}(H', X^{(i)}) + \alpha \mathfrak{R}(H', X^{(i)})] \\ & + \frac{f(H, x_i) - f(H', x_i)}{n} + \frac{f(H', x') - f(H, x')}{n} \\ =& [f(H, X^{(i)}) - f(H', X^{(i)})] + \frac{\mathfrak{L}(H, x_i) - \mathfrak{L}(H', x_i)}{n} + \frac{\mathfrak{L}(H', x') - \mathfrak{L}(H, x')}{n} \end{aligned} \tag{29}$$

Furthermore, we suppose H_X is a matrix that minimizes $f(H, X)$, then $H_{X^{(i)}}$ stands for the optimal result for minimizing $f(H, X^{(i)})$. Let $H = H_{X^{(i)}}$ and $H' = H_X$ in (29). In this case, it is obvious that $f_{X^{(i)}}(H) \leq f_{X^{(i)}}(H')$.

Then, we have

$$f(H_{X^{(i)}}, X) - f(H_X, X) \leq \frac{\mathfrak{L}(H_{X^{(i)}}, x_i) - \mathfrak{L}(H_X, x_i)}{n} + \frac{\mathfrak{L}(H_X, x') - \mathfrak{L}(H_{X^{(i)}}, x')}{n} \tag{30}$$

Comparing this equation with (28), we can get

$$\alpha||H_{X^{(i)}} - H_X||_F^2 \leq \frac{\mathfrak{L}(H_{X^{(i)}}, x_i) - \mathfrak{L}(H_X, x_i)}{n} + \frac{\mathfrak{L}(H_X, x') - \mathfrak{L}(H_{X^{(i)}}, x')}{n} \tag{31}$$

The loss term $\mathfrak{L}(H, x_i)$ is bounded for given data point x_i and clustering matrix H, which is easy to verify, and we assume the upper bound is M. Next, we analyze the upper bound for the right two terms in (31). The notations H and H' are reused to clarify the following derivation.

$$\begin{aligned}
\mathfrak{L}(H, x_i) - \mathfrak{L}(H', x_i) &= \sum_j^n [A]_{ij} \left(||\frac{H(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H]_{j.}}{\sqrt{[D]_{jj}}}||^2 - ||\frac{H'(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H']_{j.}}{\sqrt{[D]_{jj}}}||^2 \right) \\
&\leq \sum_j^n [A]_{ij} [(\frac{H(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H]_{j.}}{\sqrt{[D]_{jj}}}) + (\frac{H'(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H']_{j.}}{\sqrt{[D]_{jj}}})]^T \\
&\quad [(\frac{H'(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H']_{j.}}{\sqrt{[D]_{jj}}}) - (\frac{H'(x_i)}{\sqrt{[D]_{ii}}} - \frac{[H']_{j.}}{\sqrt{[D]_{jj}}})] \\
&\leq 2\sqrt{M} \sum_j^n \sqrt{[A]_{ij}} ||(\sum_{r=1}^n \gamma_r [H]_{r.} - \beta_j [H]_{j.}) - (\sum_{r=1}^n \gamma_k [H']_{r.} - \beta_j [H']_{j.})|| \\
&\leq 2\sqrt{M} \sum_{r=1}^n \hat{\gamma}_r ||[H]_{r.} - [H']_{r.}|| \\
&\leq 2\sqrt{M} \sigma_i ||H - H'||_F
\end{aligned} \tag{32}$$

where $\gamma_r = \sqrt{[A]_{ij}}/\sqrt{[D]_{ii}}$, $\beta_j = 1/\sqrt{[D]_{jj}}$, $\hat{\gamma}_r = \sum_{j=1}^n \sqrt{[A]_{ij}}\gamma_r - \sqrt{[A]_{ir}}\beta_r$, $\sigma_i = \max(\hat{\gamma}_r)$, and $i, j, r \in \{1, 2, \cdots, n\}$.

Then for $H = H_{X^{(i)}}$ and $H' = H_X$, we have

$$\mathfrak{L}(H_{X^{(i)}}, x_i) - \mathfrak{L}(H_X, x_i) \leq 2\sqrt{M}\sigma_i ||H_{X^{(i)}} - H_X||_F, \tag{33}$$

$$\mathfrak{L}(H_X, x') - \mathfrak{L}(H_{X^{(i)}}, x') \leq 2\sqrt{M}\sigma' ||H_{X^{(i)}} - H_X||_F, \tag{34}$$

where σ_i and σ' are concerned with the data set, and we represent the biggest one as $\sigma_* = \max\{\sigma_1, \cdots, \sigma_n, \sigma'\}$.

Utilizing the above two equations, we can reformulate Eq. (31) as

$$\alpha||H_{X^{(i)}} - H_X||_F^2 \leq \frac{4\sqrt{M}\sigma_*}{n} ||H_{X^{(i)}} - H_X||_F \tag{35}$$

which yields

$$||H_{X^{(i)}} - H_X||_F \leq \frac{4\sqrt{M}\sigma_*}{n\alpha} \tag{36}$$

Taking this back into Eq. (32), we have

$$\mathfrak{L}(H_{X^{(i)}}, x_i) - \mathfrak{L}(H_X, x_i) \leq \frac{8M\sigma_*^2}{n\alpha} \tag{37}$$

Since this equation holds for all data points X, using the theorem in [44], we have the following equation

$$\mathop{\mathbb{E}}_{X \sim Z^n} [\mathfrak{L}[(H_X, Z) - \mathfrak{L}(H_X, X)] = \mathop{\mathbb{E}}_{(X, x') \sim Z^{n+1}, i \sim U(n)} [\mathfrak{L}(H_{X^{(i)}}, x_i) - \mathfrak{L}(H_X, x_i)] \tag{38}$$

In our derivation, each data point x_i is assumed to be chosen randomly and subject to a uniform distribution. Thus, based on Eq. (38), we can calculate the expectations on the Eq. (37) to conclude

$$\mathop{\mathbb{E}}_{X \sim Z^n} [\mathfrak{L}(H_X, Z) - \mathfrak{L}(H_X, X)] \leq \frac{8M\sigma_*^2}{n\alpha}. \tag{39}$$

The above conclusion shows that the constraint term influences the upper generalization bounds of the spectral clustering loss. As α increases, the upper bounds decreases, which means that the learned constraints can promote the generalization ability of the spectral clustering model.

7.2. Importance of multiple sets of the learned constraints

Next, we analyze the role of multiple sets of the learned constraints in spectral clustering. The proposed algorithm uses the robust function Φ to integrate multiple self-learning constraints to obtain the final clustering result F. If we assume

$$[F]_{ij} = \begin{cases} \frac{1}{\sqrt{|C_l|}}, & if\,[X]_i \in C_l \\ 0, & otherwise, \end{cases} \tag{40}$$

where C_l is a set of objects belonging to the lth cluster, for $1 \le l \le k$.

By minimizing Φ, we can obtain $G_l = F^T Y_l$. In this case, we have

$$\begin{aligned} ||Y_l - FG_l||^2 &= Tr(Y_l Y_l^T) - Tr(F^T Y_l Y_l^T F) \\ &\le \frac{1}{2}Tr(I) + Tr(Y_l Y_l^T) - Tr(F^T Y_l Y_l^T F) \\ &= \frac{1}{2}Tr(Y_l Y_l^T) + \frac{1}{2}||Y_l Y_l^T - FF^T||_F^2. \end{aligned} \tag{41}$$

According to Eqs. (40) and (41), we have

$$\begin{aligned} \Phi &= tr(F^T LF) + \beta \sum_{l=1}^{e} ||Y_l - FG_l||_F^2 \\ &= Tr(I) + \beta \sum_{l=1}^{e} Tr(Y_l Y_l^T) - Tr(F^T(\hat{A} + 2\beta \sum_{l=1}^{e} Y_l Y_l^T)F) \\ &\le Tr(I) + Tr(\hat{A}) + \beta \sum_{l=1}^{e} Tr(Y_l Y_l^T) - Tr(F^T(\hat{A} + 2\beta \sum_{l=1}^{e} Y_l Y_l^T)F) \\ &= \frac{1}{2}Tr(I + \hat{A} + \beta \sum_{l=1}^{e} Y_l Y_l^T) + \frac{1}{2}||\sum_{l=1}^{e}(\frac{1}{e}\hat{A} + 2\beta Y_l Y_l^T) - FF^T||_F^2. \end{aligned} \tag{42}$$

When each Y_l is given, minimizing Φ is equivalent to minimizing

$$\Phi' = ||\frac{1}{e}\sum_{l=1}^{e}(\hat{A} + \gamma Y_l Y_l^T) - FF^T||_F^2. \tag{43}$$

If we replace β with a parameter γ and assume $\beta = \frac{1}{2e}\gamma$, Φ' can be seen as

$$\Phi' = ||\frac{1}{e}\sum_{l=1}^{e}(\hat{A} + \gamma Y_l Y_l^T) - FF^T||_F^2. \tag{44}$$

If $E(B) = FF^T$ and $B_l = \hat{A} + \gamma Y_l Y_l^T$ are seen as the expectation and estimation of the final clustering result, respectively, we have

$$\Phi' = ||\frac{1}{e}\sum_{l=1}^{e} B_l - \mathbb{E}(B)||_F^2. \tag{45}$$

According to the above equation, we can see that the larger e, the closer the mean of B_l is to the expectation of B, and the lower the value of Φ'. Thus, we can conclude that the proposed algorithm with multiple sets of self-learning constraints can better learn the final clustering result compared to that with a single set of self-learning constraints.

8. Convergence analysis

Minimization of the proposed objective function Ω forms a class of constrained nonlinear optimization problems whose solutions are unknown. Therefore, we provide an iterative method to solve this optimization problem. In this section, we apply Zangwill's theorem [45] to discuss the convergence of the proposed algorithm. The theorem and its generalizations can be used to obtain convergence proofs for almost all of the classical iterative optimization algorithms, e.g., steepest descent, Newton's method, etc. [46], by using this approach as an alternative to more conventional arguments. The theorem is described as follows.

Theorem 1. *[45] Let $f : D_f \subset \mathbb{R}^m \to \mathbb{R}$, $S = \{x^* \in D_F : f(x^*) < f(y) \quad \forall y \in B^0(x^*, r)\}$, where $B^0(x^*, r) = \{y \in \mathbb{R}^m : \| x^* - y \| < r\}$, $\| \cdot \|$ any norm on $\mathbb{R}^m$, $\mathcal{A} : D_f \to D_f$ be an iterative algorithm, $x_{k+1} = \mathcal{A}(x_k)$, and g be attached to sequences of iterations generated by $\mathcal{A}$ to monitor the progress of $\mathcal{A}$ in seeking a solution $x^* \in S$. If the following conditions hold, g is a descent function for $\{\mathcal{A}, S\}$, $\mathcal{A}$ is continuous on $D_f \setminus S$, and the iterative sequence $\{\mathcal{A}(x_k) : k = 1, 2, \cdots; x_1 \in D_f\} \subset K$ are contained in a compact set $K \subseteq D_f$ for arbitrary $x_1 \in D_f$, then for each iterative sequence $\{x_k\}$ generated by $\mathcal{A}$, we have either $\{x_k\}$ terminates at a solution $x^* \in S$ or $\exists$ a sequence $\{x_{k_j}\} \subseteq \{x_k\}$ so that $\{x_{k_j}\} \to x^* \in S$.*

According to the theorem above, the optimization algorithm needs to satisfy three conditions, i.e., non-increasing, continuous, and compact set properties, to ensure that the algorithm can converge. In order to prove that the proposed algorithm satisfies the three conditions, we give some symbolic definitions for clarity of proof. We assume that $\mathcal{A}_\Omega$ represents the proposed algorithm in the paper. Since the algorithm is updated by iteration with four matrices $(H^{(t)}, Y^{(t)}, F^{(t)}, G^{(t)})$, and the iteration of variables produces corresponding sequences $\{(H^{(t)}, Y^{(t)}, F^{(t)}, G^{(t)}) : t = 1, 2, \cdots\}$. We define the domains of H, Y, F, G as M_h, M_y, M_f, M_g. As the updating formula of each variable is based on fixing all the other three variables, we define N_h as the ranges of the fixed vectors Y, F, G. Similarly, we have N_y, N_f, N_g. Then we define the updating functions for H as $\Phi_h : N_h \to M_h$, i.e., $\Phi_h(Y, F, G) = H$, and H in calculated by the Eq. (15). Likewise, we can define the updating functions Φ_y, Φ_f, Φ_g for Y, F, G. Now we have the updating algorithm $\mathcal{A}_\Omega$ be defined as $\mathcal{A}_\Omega : (M_h \times M_y \times M_f \times M_g) \to (M_h \times M_y \times M_f \times M_g)$, $\mathcal{A}_\Omega = \Phi_h \circ \Phi_y \circ \Phi_f \circ \Phi_g$.

Based on the Theorem 1, we will provide some theorems to prove the convergence of the proposed algorithm, i.e., it can converge in the limited number of iterations. In the following, Theorems 2-5 show the descending property of the objective function. Theorem 6 proves the iterative algorithm $\mathcal{A}_\Omega$ is continuous on the $(M_h \times M_y \times M_f \times M_g)$, and Theorem 7 asserts the iterative sequences calculated by the proposed algorithm are in the compact set.

8.1. Non-increasing property

To prove that the objective function is non-increasing under the updating rules, we essentially follow the idea in the proof of NMF [47] and GNMF [43]. Our proof will make use of their auxiliary function and the corresponding lemma which are described as follows.

Definition 1. [47] $Q(v, v')$ is an auxiliary function for $\mathcal{F}(v)$ if the conditions

$$Q(v, v') \geq \mathcal{F}(v), \quad Q(v, v) = \mathcal{F}(v) \tag{46}$$

are satisfied.

Lemma 1. *[47] If Q is an auxiliary function of $\mathcal{F}$, then $\mathcal{F}$ is non-increasing under the update formula:*

$$v^{(t+1)} = \arg\min_v Q(v, v^{(t)}), \tag{47}$$

where t denotes the tth iteration.

Proof.

$$\mathcal{F}(v^{(t+1)}) \leq Q(v^{(t+1)}, v^{(t)}) \leq Q(v^{(t)}, v^{(t)}) = \mathcal{F}(v^{(t)}) \quad \square \tag{48}$$

Next, based on the above auxiliary function and lemma, we provide four theorems to prove that the objective function is non-increasing under each updating rule.

Theorem 2. *The optimization function in Eq. (22) is non-increasing under the update formula of F in Eq. (24).*

Proof. The proof is mainly about designing a suitable auxiliary function that satisfies the inequality in Eq. (47). First, we represent the objective function in Eq. (22) as Θ_F and the element-wise objective function as $\Theta_{[F]_{ij}}$. We rewrite the update formula for F as

$$[F]_{ij} \leftarrow [F]_{ij} \frac{[\beta Y G^T + \hat{A} F]_{ij}}{[\beta F G G^T + F]_{ij}} \tag{49}$$

It is easy to get the derivatives of $\Theta_{[F]_{ij}}$ as

$$\Theta'_{[F]_{ij}} = \frac{\partial \Theta_F}{\partial [F]_{ij}} = [2LF + 2\beta F G G^T - 2\beta Y G^T]_{ij} \tag{50}$$

$$\Theta''_{[F]_{ij}} = \frac{\partial^2 \Theta_F}{\partial [F]_{ij}^2} = 2\beta [G G^T]_{jj} + 2[L]_{ii} \tag{51}$$

We set the auxiliary function for Θ_F as

$$Q_F(f,[F]_{ij}^{(t)}) = \Theta_{[F]_{ij}^{(t)}} + \Theta'_{[F]_{ij}^{(t)}}(f-[F]_{ij}^{(t)}) + \frac{[\beta FGG^T+F]_{ij}^{(t)}}{[F]_{ij}^{(t)}}(f-[F]_{ij}^{(t)})^2 \tag{52}$$

It is obvious that $Q_F([F]_{ij}^{(t)},[F]_{ij}^{(t)}) = \Theta_{[F]_{ij}^{(t)}}$. Then we need to prove that $Q_F(f,[F]_{ij}^{(t)}) \geq \Theta_f$. It can be derived using Taylor's expansion of Θ_f as follows.

$$\begin{aligned} Q_F(f,[F]_{ij}^{(t)}) \geq \Theta_f &= \Theta_{[F]_{ij}^{(t)}} + \Theta'_{[F]_{ij}^{(t)}}(f-[F]_{ij}^{(t)}) + \frac{1}{2}\Theta''_{f*}(f-[F]_{ij}^{(t)})^2 \\ &= \Theta_{[F]_{ij}^{(t)}} + \Theta'_{[F]_{ij}^{(t)}}(f-[F]_{ij}^{(t)}) + (\beta[GG^T]_{jj}+[L]_{ii})(f-[F]_{ij}^{(t)})^2. \end{aligned} \tag{53}$$

Based on the derivation, $Q_F(f,[F]_{ij}^{(t)}) \geq \Theta_f$ can be transformed as

$$\frac{[\beta FGG^T+F]_{ij}^{(t)}}{[F]_{ij}^{(t)}} \geq (\beta[GG^T]_{jj}+[L]_{ii}). \tag{54}$$

It is easy to obtain that

$$[\beta FGG^T]_{ij}^{(t)} = \beta\sum_{l=1}^{n}[F]_{il}^{(t)}[GG^T]_{lj}^{(t)} \geq \beta[F]_{ij}^{(t)}[GG^T]_{jj}^{(t)} \tag{55}$$

and

$$[F]_{ij}^{(t)} \geq [L]_{ii}[F]_{ij}^{(t)} \tag{56}$$

Therefore, Eq. (54) is clearly true, i.e., $Q_F(f,[F]_{ij}) \geq \Theta_f$. We conclude that the optimization function is non-increasing under the update formula for F. □

Theorem 3. *The optimization function Θ_l in Eq. (13) is non-increasing under the update formula in Eq. (15).*

Proof. We set Θ_H as the objective function for H and $\Theta_{[H]_{ij}}$ as the objective function with respect to the matrix element $[H]_{ij}$ in H. Then the first and the second derivatives can be calculated as

$$\Theta'_{[H]_{ij}} = \frac{\partial\Theta_H}{\partial[H]_{ij}} = [2LH+2\alpha H-2\alpha Y]_{ij} \tag{57}$$

$$\Theta''_{[H]_{ij}} = \frac{\partial^2\Theta_H}{\partial[H]_{ij}^2} = [2L+2\alpha I]_{ii} \tag{58}$$

We design the auxiliary function as follows.

$$Q_H(h,[H]_{ij}^{(t)}) = \Theta_{[H]_{ij}^{(t)}} + \Theta'_{[H]_{ij}^{(t)}}(h-[H]_{ij}^{(t)}) + \frac{[\alpha H+H]_{ij}^{(t)}}{[H]_{ij}^{(t)}}(h-[H]_{ij}^{(t)})^2 \tag{59}$$

Similar to the analysis in Θ_F, we can conclude that the optimization function is non-increasing under the update formula of H. □

Theorem 4. *The optimization function Δ in Eq. (16) is non-increasing under the update formula in Eq. (21).*

Proof. For the convenience of proof, we use the symbol Θ_Y for Y's loss function and $\Theta_{[Y]_{ij}}$ for $[Y]_{ij}$'s loss function. We can rewrite the update formula of Y in the element-wise form as

$$[Y]_{ij} \leftarrow [Y]_{ij}\frac{[\alpha H+\beta FG]_{ij}}{[(\alpha+\beta)I+\eta UY]_{ij}} \tag{60}$$

The derivative of Θ_Y for each variable Y_{ij} in Y can be calculated as

$$\Theta'_{[Y]_{ij}} = \frac{\partial\Theta_Y}{\partial[Y]_{ij}} = [2(\alpha+\beta)Y+2\eta UY-2\alpha H-2\beta FG]_{ij} \tag{61}$$

$$\Theta''_{[Y]_{ij}} = \frac{\partial^2 \Theta_Y}{\partial [Y]_{ij}^2} = [2(\alpha+\beta)I + 2\eta U]_{ii} \tag{62}$$

Then we assume the corresponding auxiliary function for Y is

$$Q_Y(y, [Y]_{ij}^{(t)}) = \Theta_{[Y]_{ij}^{(t)}} + \Theta'_{[Y]_{ij}^{(t)}}(y - [Y]_{ij}^{(t)}) + \frac{[(\alpha+\beta)Y + \eta UY]_{ij}^{(t)}}{[Y]_{ij}^{(t)}}(y - [Y]_{ij}^{(t)})^2 \tag{63}$$

Like the proof in Q_F, we can conclude that the given formula of Y in Eq. (16) can guarantee the non-increasing property of the optimization function. □

Theorem 5. *The optimization function in Eq. (64) is non-increasing under the update formula of G in Eq. (25).*

Proof. We have the following optimization function Θ_G for G

$$\Theta_G = \beta ||Y - FG||_F^2 \tag{64}$$

that is similar to the NMF model in [47]. Then we can utilize the convergence analysis in [47] to prove the above theorem. □

Given these theorems, we have

$$\begin{aligned}\Omega^{(t+1)} =& Tr([H^T]^{(t+1)} L H^{(t+1)}) + \alpha ||H^{(t+1)} - Y^{(t+1)}||_F^2 + \eta ||Y^{(t+1)}||_{2,1} \\ &+ Tr([F^T]^{(t+1)} L F^{(t+1)}) + \beta ||Y^{(t+1)} - F^{(t+1)} G^{(t+1)}||_F^2 \\ \leq& Tr([H^T]^{(t)} L H^{(t)}) + \alpha ||H^{(t)} - Y^{(t)}||_F^2 + \eta ||Y^{(t)}||_{2,1} \\ &+ Tr([F^T]^{(t)} L F^{(t)}) + \beta ||Y^{(t)} - F^{(t)} G^{(t)}||_F^2.\end{aligned} \tag{65}$$

According to Eq. (65), we conclude that the objective function in Eq. (12) is non-increasing with the updating formulas of the H, Y, F, and G.

8.2. Continuous property

The second requirement of the Theorem 1 is to make sure that the algorithm $\mathcal{A}_\Omega$ is continuous on the domain (M_{fh}). We give the following theorem to prove the continuous property.

Theorem 6. *The algorithm $\mathcal{A}_\Omega$ is continuous on $(M_h \times M_y \times M_f \times M_g)$.*

Proof. Since $\mathcal{A}_\Omega = \Phi_h \circ \Phi_y \circ \Phi_f \circ \Phi_g$, and the composition of the continuous functions is also continuous, it suffices to show that $\Phi_h, \Phi_y, \Phi_f, \Phi_g$ are each continuous. Then we prove that Φ_h is continuous in the (kn) variables $\{[H]_{lj}\}$. Note that Φ_h is a vector field, with the resolution by (kn) scalar field like the domain of H. Thus, it can be described as

$$\Phi_h = [\Phi_h^{(11)}, \Phi_h^{(12)}, \cdots, \Phi_h^{(lj)}, \cdots, \Phi_h^{(kn)}] : \mathbb{R}^{kn} \to \mathbb{R}^{kn}, \tag{66}$$

where $\Phi_h^{(lj)} : \mathbb{R}^{kn} \to \mathbb{R}$ defined in Eq. (15) can be calculated as

$$\Phi_h^{(lj)} \leftarrow [H]_{lj} \frac{[\alpha Y^T + \hat{A}H]_{lj}}{[\alpha H + H]_{lj}}. \tag{67}$$

It is evident that $[H]_{lj}$, $[\alpha Y^T + \hat{A}H]_{lj}$, $[\alpha H + H]_{lj}$ are element-wise functions and are continuous, and the denominator of Φ_h never vanishes under the given constraints of H. Therefore, the Φ_h is a continuous function. Next, we prove that Φ_y is also a continuous function of the $(2kn + k^2)$ variables $\{[Y]_{lj}\}$. Φ_y is a vector field with the resolution by (kn) variables

$$\Phi_y = [\Phi_y^{(11)}, \Phi_y^{(12)}, \cdots, \Phi_y^{(lj)}, \cdots, \Phi_y^{(kn)}] : \mathbb{R}^{k(2n+k)} \to \mathbb{R}^{kn} \tag{68}$$

where $\Phi_y^{(lj)} : \mathbb{R}^{k(2n+k)} \to \mathbb{R}$ is given in Eq. (60). Likewise, we know that each part of such element-wise updating function is continuous, so the Φ_y is continuous on their entire domains. Next we show that Φ_f is also a continuous function of the $(kn + kk)$ variables $\{[F]_{lj}\}$, and Φ_f is a vector field with the resolution by (kn) variables

$$\Phi_f = [\Phi_f^{(11)}, \Phi_f^{(12)}, \cdots, \Phi_f^{(lj)}, \cdots, \Phi_f^{(kn)}] : \mathbb{R}^{k(n+k)} \to \mathbb{R}^{kn} \tag{69}$$

Table 1
Description of benchmark data sets.

Data sets	Objects	Dimensions	Clusters
ORL	400	1024	40
Umist	575	1024	20
COIL20	1440	1024	20
Yale-B	2424	5120	38
OpticDigits	5620	10	63
Statlog	6435	36	6
COIL100	7200	1024	100
MNIST	10000	728	10
PenDigits	10992	16	10
USPS	11000	256	10

where $\Phi_f^{(lj)} : \mathbb{R}^{k(n+k)} \to \mathbb{R}$ is given in Eq. (24). Likewise, since each part of such element-wise updating function is continuous, Φ_f is continuous on their entire domains. Next we prove that Φ_g is also a continuous function of the $(2kn)$ variables $\{[G]_{lj}\}$, and Φ_g is a vector field with the resolution by (k^2) variables

$$\Phi_g = [\Phi_g^{(11)}, \Phi_g^{(12)}, \cdots, \Phi_g^{(lj)}, \cdots, \Phi_g^{(kn)}] : \mathbb{R}^{(2kn)} \to \mathbb{R}^{k^2} \tag{70}$$

where $\Phi_g^{(lj)} : \mathbb{R}^{(2kn)} \to \mathbb{R}$ is given in Eq. (25). Likewise, since each part of such element-wise updating function is continuous, Φ_g is continuous on their entire domains. Hence, $\mathcal{A}_\Omega = \Phi_h \circ \Phi_y \circ \Phi_f \circ \Phi_g$ is continuous on $(M_h \times M_y \times M_f \times M_g)$. □

8.3. Compact set property

The third condition in the Theorem 1 is to judge the compactness of $(M_h \times M_y \times M_f \times M_g)$, which contains all of the possible iterate sequences generated by $\mathcal{A}_\Omega$. Based on the idea of [48], we give the following theorem to prove the compact set property.

Theorem 7. *$(M_h \times M_y \times M_f \times M_g)$ is a compact set.*

Proof. Given the initial values of $H^{(0)} \in M_h, Y^{(0)} \in M_y, F^{(0)} \in M_f, G^{(0)} \in M_g$. Then we can iteratively calculate the values of the vector $F^{(t+1)}$ as

$$[F]_{ij}^{(t+1)} = [F]_{ij}^{(t)} \frac{[\beta Y G^T + \hat{A} F]_{ij}^{(t)}}{[\beta F G G^T + F]_{ij}^{(t)}}. \tag{71}$$

According to Eq. (12), we know $M_f = \{F : [F]_{ij} \geq 0\}$ and $F^{(t)} \in M_f$. We also can see that each element in $Y^{(t)}$, $G^{(t)}$ and $\hat{A}$ is non-negative, according to their definitions. Therefore, from Eq. (71), we have $[F^{(t+1)}]_{ij} \geq 0$ and $F^{(t+1)} \in M_f$. Based on the same way, we can infer that $H^{(t+1)} \in M_h, Y^{(t+1)} \in M_y, G^{(t+1)} \in M_g$. Therefore, we can conclude that M_h, M_y, M_f, M_g are all compact sets. Then according to the Heine-Borel theorem [48], we can conclude that $(M_h \times M_y \times M_f \times M_g)$ is a compact set. □

9. Experiment analysis

9.1. Experiment settings

To examine the performance of the proposed algorithm, we compare it with other eight versions of spectral clustering algorithms, including standard spectral clustering (SC) [3], bipartite graph clustering (ESCG) [18], spectral clustering using k-means-based landmark selection (LSC-K) [20], spectral clustering with approximate eigenvectors (FastESC) [22], ultra-scalable spectral clustering (U-SPEC) [25], constrained Laplacian rank clustering (CLR) [14], low-rank representation clustering (LRR) [11], graph regularized nonnegative matrix factorization (GNMF) [43]. Besides, we also compare the proposed algorithm with label propagation with different sizes of label information [5].

The comparisons are carried out on ten widely used benchmark data sets [49] whose detailed information is described in Table 1. We employ three clustering indices [2]: accuracy measure (ACC), the adjusted rand index (ARI) and the normalized mutual information (NMI) to evaluate the effectiveness of clustering results on each data set. If the clustering result is close to the true partition, its ACC, NMI, and ARI values are high. The experiment equipment is a personal computer with Intel i9-10900K, 64G RAM, Matlab R2018a, and windows 10.

Before the comparisons, we need to set some parameters as follows. For each algorithm, we set the number of clusters k to its true number of classes on a data set, and use the Gaussian kernel function to produce the similarity matrix and test each of these algorithms with different γ value of the kernel parameter, i.e., $\delta = \varepsilon_X / g$, $g \in [10, 100]$ with a step length of

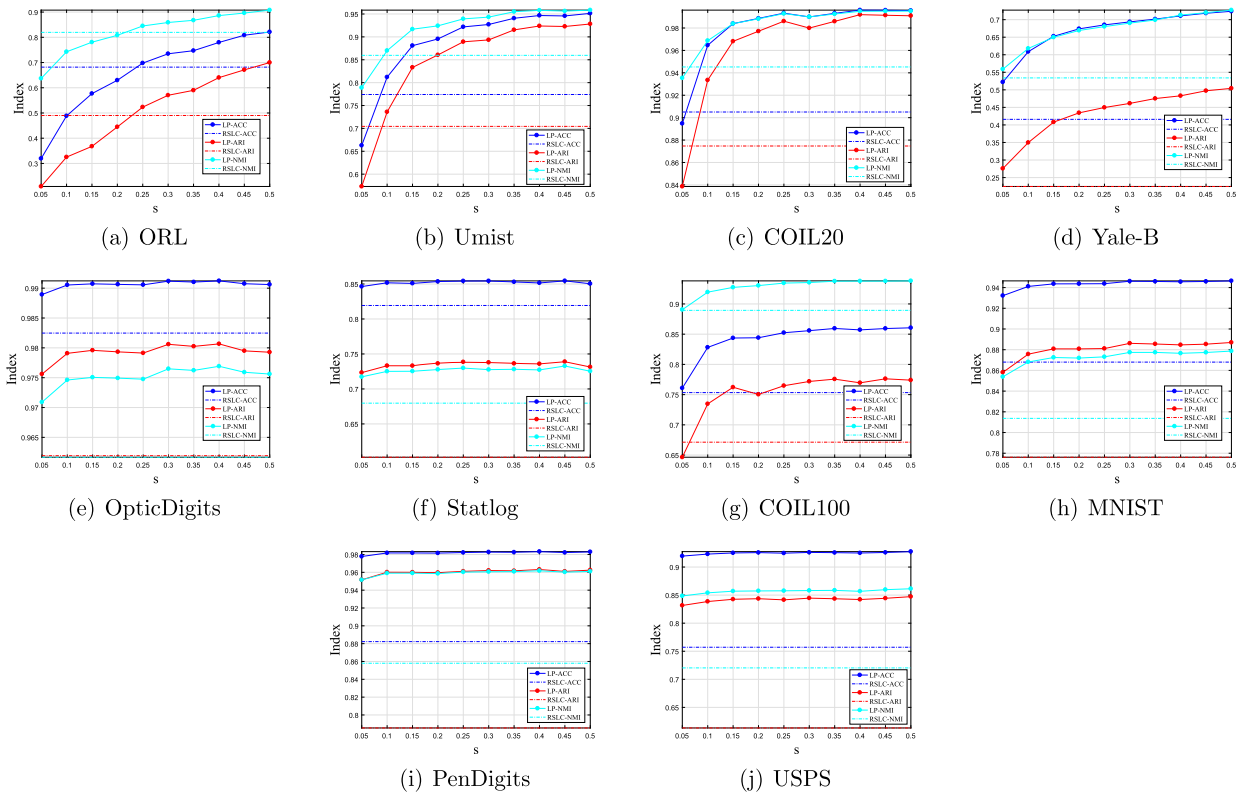


Fig. 2. Comparison of the proposed algorithm with semi-supervised spectral clustering. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 2
ACC values of different algorithms on benchmark data sets.

Dataset	SC	ESCG	LSC-K	Fast-ESC	U-SPEC	CLR	LRR	GNMF	RSLC
ORL	61.36 ± 1.26	60.51 ± 1.41	58.13 ± 1.67	59.64 ± 3.24	57.78 ± 1.61	64.75 ± 0.00	61.14 ± 1.53	59.22 ± 1.52	**67.88** ± 1.65
Umist	71.04 ± 0.44	60.09 ± 1.35	60.21 ± 3.31	52.03 ± 2.69	50.17 ± 1.08	75.10 ± 1.15	38.26 ± 0.76	71.09 ± 2.41	**77.37** ± 2.40
COIL-20	73.70 ± 1.24	74.23 ± 0.98	72.25 ± 2.46	61.12 ± 3.88	61.95 ± 3.04	78.35 ± 2.23	55.04 ± 1.19	84.30 ± 1.33	**88.86** ± 0.61
Yale-B	36.13 ± 0.75	29.33 ± 1.42	16.24 ± 0.88	23.48 ± 1.08	10.56 ± 0.38	32.27 ± 0.56	40.62 ± 1.30	27.06 ± 1.25	**42.12** ± 1.38
OpticDigits	95.25 ± 0.00	98.21 ± 0.01	92.08 ± 1.23	77.98 ± 3.47	82.88 ± 5.07	96.53 ± 0.00	80.53 ± 0.02	93.76 ± 3.20	**98.27** ± 0.19
Stalog	66.16 ± 0.01	76.00 ± 0.03	74.69 ± 0.82	68.80 ± 3.60	74.96 ± 0.72	66.08 ± 0.00	67.79 ± 0.00	60.78 ± 4.55	**82.17** ± 0.72
COIL-100	49.86 ± 1.01	59.61 ± 0.58	60.82 ± 1.26	42.62 ± 2.31	53.10 ± 1.26	66.64 ± 0.94	47.22 ± 0.99	67.30 ± 1.32	**75.13** ± 0.81
MNIST	47.90 ± 0.01	70.73 ± 0.81	68.66 ± 1.98	56.50 ± 3.22	59.69 ± 2.58	72.62 ± 0.20	48.67 ± 0.03	63.67 ± 3.14	**85.52** ± 3.16
PenDigits	71.51 ± 0.00	73.41 ± 0.01	80.18 ± 3.39	68.66 ± 1.70	74.36 ± 2.47	67.79 ± 0.00	72.59 ± 0.01	76.39 ± 4.10	**86.90** ± 2.33
USPS	56.13 ± 0.02	61.81 ± 2.36	54.38 ± 2.02	49.04 ± 2.20	48.60 ± 1.90	64.60 ± 1.17	54.40 ± 0.00	54.35 ± 4.24	**74.87** ± 2.94

10, where ε_X is the variance of data set X, to select the highest ACC, ARI and NMI values for comparisons. In the proposed algorithm, we set $\alpha = 0.25$, $\beta = 0.5$, $\eta = 0.1$, $e = 10$, and $t = 100$. In the following experiments, we explain the parameter settings. For other parameters of the compared algorithms, we set them according to the suggestions of their references.

9.2. Comparison with other versions of spectral clustering algorithms

We first analyze the difference in the clustering effectiveness between the proposed algorithm and the other eight spectral clustering algorithms. Tables 2, 3, and 4 show the mean and standard deviation of ACC, ARI, and NMI for the clustering results produced by each algorithm running 20 times on the tested data sets. According to the evaluation results, we can observe that the mean values of the ACC, ARI, and NMI for the proposed algorithm are obviously superior to other algorithms on the tested data sets. The experimental results tell us that the self-label learning operation of the proposed algorithm can very effectively improve the performance of spectral clustering. Besides, we can see that the standard deviation of the proposed algorithm on each data set is less than 0.04. Therefore, we can conclude that the proposed algorithm is robust to deal with these data sets.

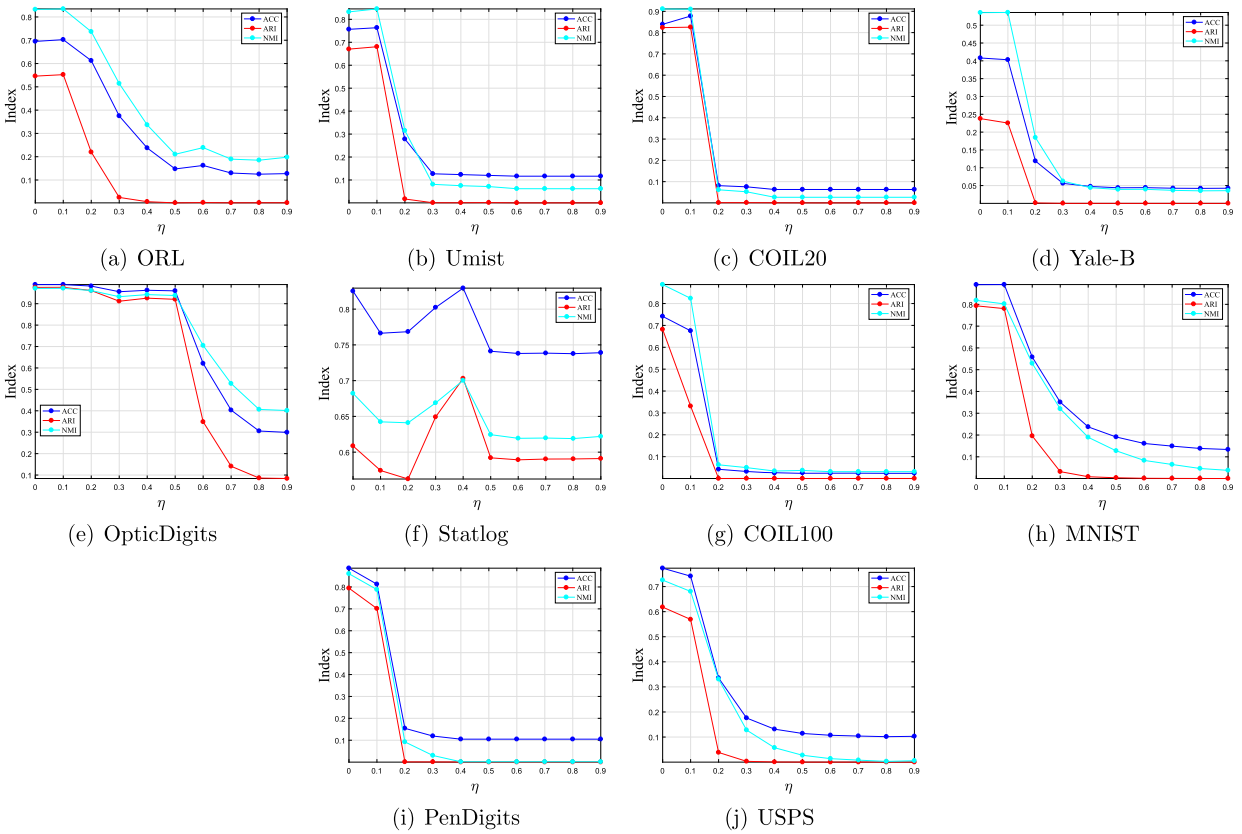


Fig. 3. Three clustering indices against parameter η.

Table 3
ARI values of different algorithms on benchmark data sets.

Dataset	SC	ESCG	LSC-K	Fast-ESC	U-SPEC	CLR	LRR	GNMF	RSLC
ORL	40.14 ± 1.70	42.35 ± 1.99	39.26 ± 2.23	35.04 ± 5.08	40.51 ± 1.61	37.57 ± 0.00	40.42 ± 2.30	36.89 ± 2.49	**48.39** ± 2.42
Umist	59.03 ± 0.76	45.48 ± 1.03	44.89 ± 3.59	35.52 ± 3.79	33.66 ± 0.97	62.15 ± 0.97	13.42 ± 0.88	58.60 ± 5.18	**70.05** ± 3.81
COIL-20	65.67 ± 1.41	64.09 ± 1.16	63.32 ± 2.68	49.06 ± 4.65	54.12 ± 2.62	73.34 ± 3.12	34.33 ± 2.14	76.50 ± 1.60	**86.04** ± 0.93
Yale-B	15.05 ± 0.51	13.74 ± 0.26	5.85 ± 0.42	11.34 ± 0.53	1.66 ± 0.23	7.78 ± 0.21	13.50 ± 0.57	08.26 ± 0.78	**23.72** ± 1.24
OpticDigits	90.43 ± 0.00	96.11 ± 0.02	84.01 ± 2.13	63.32 ± 3.77	70.90 ± 6.11	93.15 ± 0.00	67.91 ± 0.04	88.73 ± 5.59	**96.23** ± 0.40
Stalog	42.60 ± 0.01	53.16 ± 0.05	52.90 ± 1.60	43.44 ± 6.46	53.85 ± 1.47	42.67 ± 0.01	45.62 ± 0.00	32.81 ± 3.93	**61.26** ± 2.52
COIL-100	36.03 ± 0.87	38.47 ± 1.19	50.43 ± 1.49	24.09 ± 3.55	43.23 ± 1.10	56.95 ± 1.15	30.90 ± 1.53	46.43 ± 1.37	**67.10** ± 0.81
MNIST	25.74 ± 0.00	55.05 ± 0.98	50.29 ± 2.93	33.16 ± 2.88	38.61 ± 3.34	59.74 ± 0.09	25.22 ± 0.02	49.42 ± 6.55	**76.24** ± 3.61
PenDigits	56.40 ± 0.01	58.29 ± 0.95	65.36 ± 4.62	50.66 ± 2.10	57.75 ± 2.36	55.97 ± 0.00	54.86 ± 0.01	61.54 ± 6.07	**77.06** ± 2.30
USPS	29.49 ± 0.04	47.07 ± 2.77	37.86 ± 1.80	27.77 ± 1.95	30.01 ± 1.40	47.01 ± 1.69	33.94 ± 0.00	37.29 ± 2.86	**59.44** ± 4.67

Table 4
NMI values of different algorithms on benchmark data sets.

Dataset	SC	ESCG	LSC-K	Fast-ESC	U-SPEC	CLR	LRR	GNMF	RSLC
ORL	79.32 ± 0.56	75.92 ± 0.91	74.70 ± 1.02	75.89 ± 2.59	75.20 ± 0.66	80.15 ± 0.00	79.39 ± 0.73	75.67 ± 1.24	**81.87** ± 0.70
Umist	80.00 ± 0.28	73.36 ± 0.71	73.38 ± 2.08	66.15 ± 2.50	65.24 ± 0.85	86.17 ± 0.55	40.59 ± 0.91	83.69 ± 2.02	**86.07** ± 1.47
COIL-20	84.87 ± 0.60	83.53 ± 0.76	80.24 ± 1.34	73.16 ± 2.44	74.51 ± 1.68	88.14 ± 1.42	62.83 ± 1.33	89.36 ± 0.91	**93.03** ± 0.49
Yale-B	47.40 ± 0.79	38.64 ± 0.37	23.46 ± 0.61	39.30 ± 0.29	39.30 ± 0.29	37.07 ± 0.99	13.32 ± 0.54	36.37 ± 2.26	**53.34** ± 0.49
OpticDigits	92.79 ± 0.00	95.77 ± 0.02	85.82 ± 1.08	71.16 ± 2.26	77.20 ± 3.10	94.95 ± 0.00	74.34 ± 0.03	92.23 ± 2.52	**96.12** ± 0.25
Stalog	52.91 ± 0.02	64.94 ± 0.09	62.47 ± 0.86	56.05 ± 3.08	62.08 ± 0.57	53.00 ± 0.00	49.16 ± 0.00	44.53 ± 3.55	**68.29** ± 0.76
COIL-100	79.11 ± 0.38	79.47 ± 0.35	79.28 ± 0.52	72.00 ± 1.40	74.73 ± 0.44	84.69 ± 0.36	67.80 ± 0.77	86.50 ± 0.73	**88.91** ± 0.38
MNIST	42.01 ± 0.00	69.16 ± 0.43	60.57 ± 1.65	44.92 ± 2.10	50.29 ± 1.89	72.01 ± 0.23	38.02 ± 0.02	67.24 ± 3.35	**80.96** ± 1.40
PenDigist	72.89 ± 0.01	77.04 ± 0.48	76.81 ± 1.95	65.41 ± 1.49	68.45 ± 1.15	73.73 ± 0.00	63.95 ± 0.01	75.22 ± 3.13	**84.81** ± 1.21
USPS	48.88 ± 0.02	64.27 ± 1.48	52.90 ± 1.35	41.37 ± 1.86	44.89 ± 1.35	65.08 ± 0.43	46.20 ± 0.01	56.71 ± 1.99	**70.45** ± 2.74

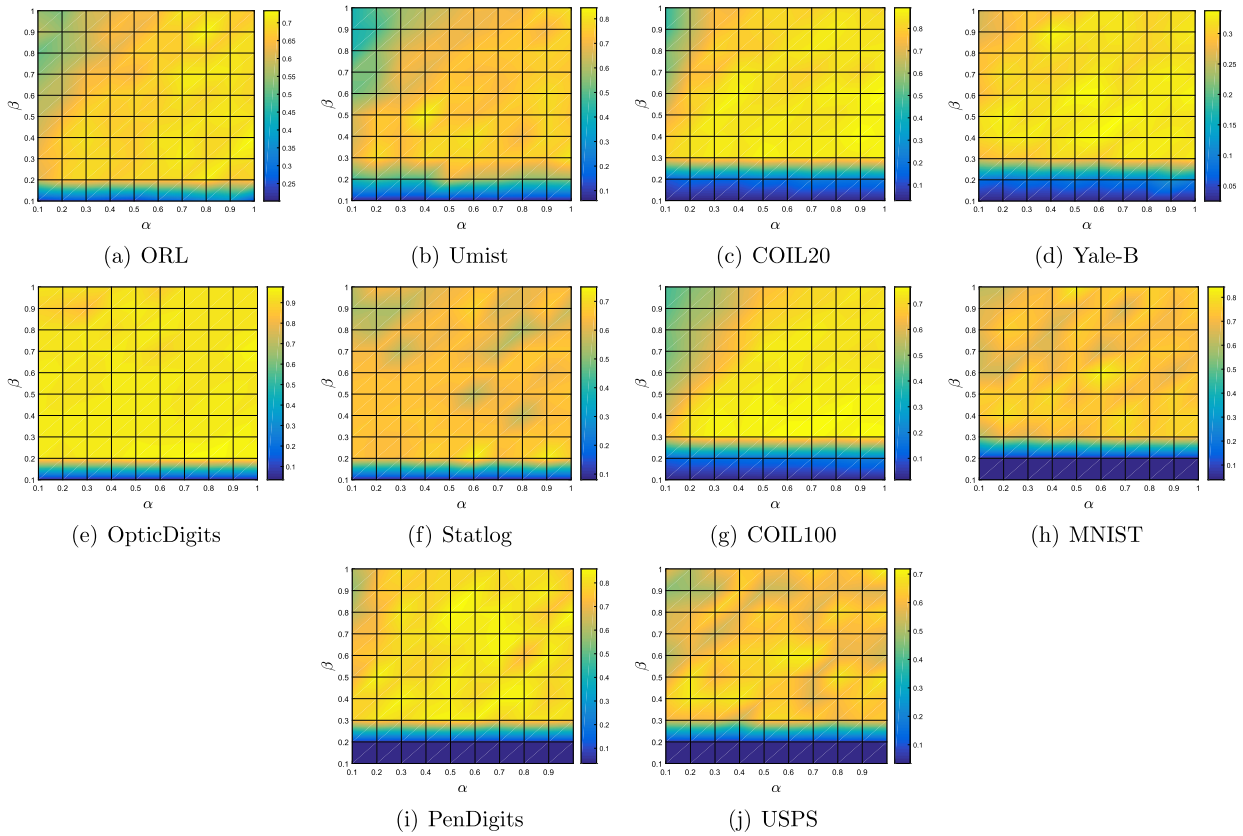


Fig. 4. Three clustering indices against parameters α and β.

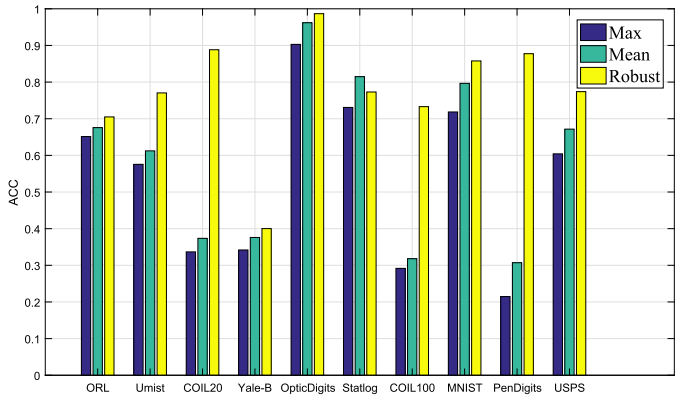
9.3. Comparison with semi-supervised spectral clustering

We analyze the difference in clustering effectiveness between the proposed algorithm and the semi-supervised spectral clustering algorithm, as shown in Fig. 2. We test spectral clustering with different sizes of label information. Let s be the proportion of labels to a whole data set. We select s from 5% to 50% with a step length of 5%. In these figures, the curves show the relations between the clustering indices of label propagation and the sizes of labels, and the lines reflect the clustering indices of the proposed algorithm. According to these figures, we can see that the clustering results of the proposed algorithm can reach or approach the semi-supervised results on four out of ten data sets. On other data sets, there are no small gaps between the clustering results of the proposed algorithm and label propagation with some labels. Therefore, we need to study further how to improve the proposed algorithm.

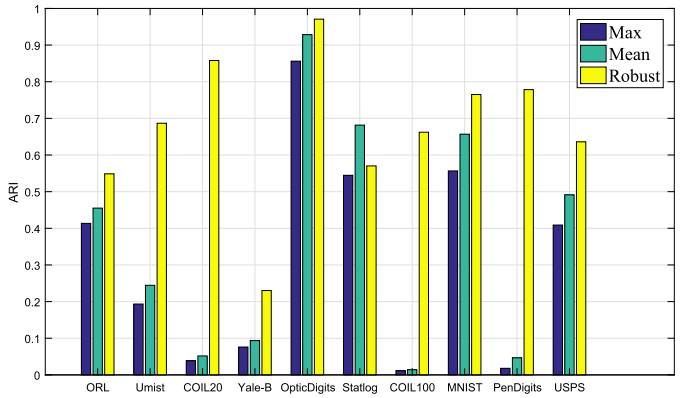
9.4. Parameter analysis

According to the description of the proposed algorithm, we can observe that there are three important parameters η, α, and β, which are used to balance the roles of main terms in the objective function. We first analyze the effect of η on the performance of the proposed algorithm. In the experiment, we test η in the interval $[0, 1]$ with the step length of 0.1 while α and β were fixed. Fig. 3 shows the relation between clustering indices and the parameter η on each data set. Based on these figures, we can see that the clustering quality decreases on all data sets except Statlog as the value of η increases. We know η is used to control the importance of the regularization term. The more η is, the more sparse Y is. We can also observe that if $\eta = 0.1$, we can get good clustering results on most of the data sets.

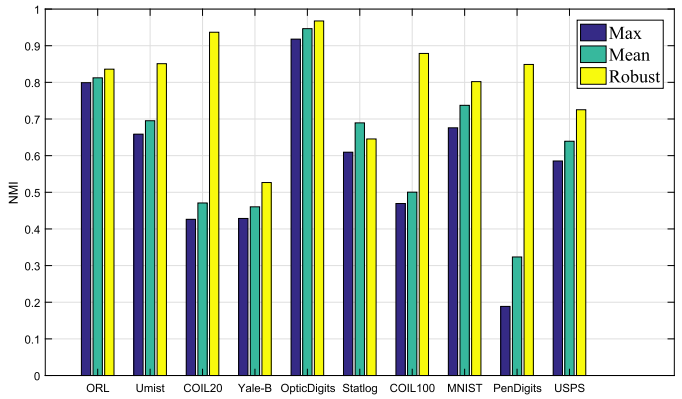
Furthermore, we analyze the co-influence of α and β on the performance of the proposed algorithm. We test the two parameters in the interval $[0, 1]$ with the step length of 0.1, while η is fixed. The tested results are shown in Fig. 4. According to these figures, we can observe that if α and β are close to 0, the performance of the proposed algorithm is very bad. The conclusion tells us that the self-learned constraint and consensus terms play important roles in the proposed algorithm. Besides, we can see that setting α and β to more than 0.2 can bring good clustering results on most of the data sets.



(a) ACC



(b) ARI

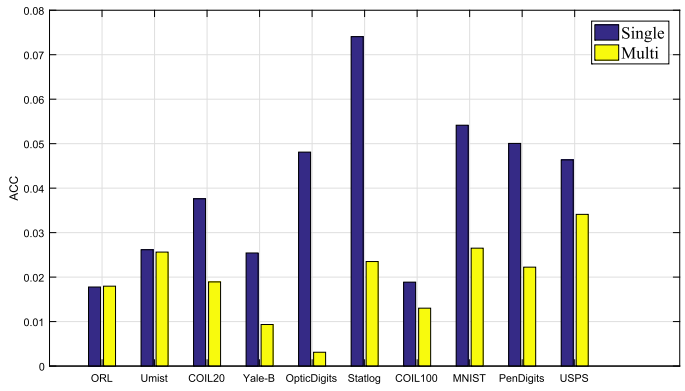


(c) NMI

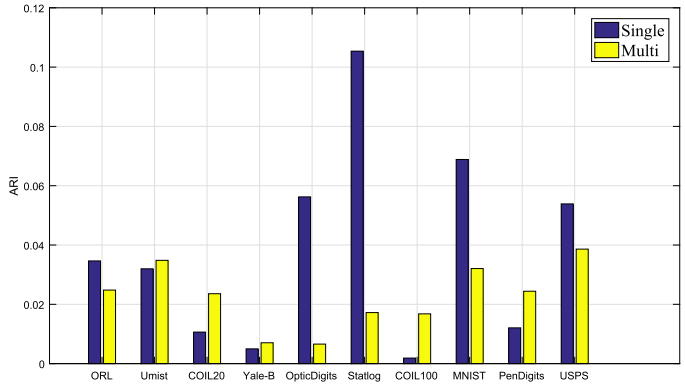
Fig. 5. Comparison of clustering accuracy between single and multiple sets of constraints.

9.5. Comparison of different numbers of constraints

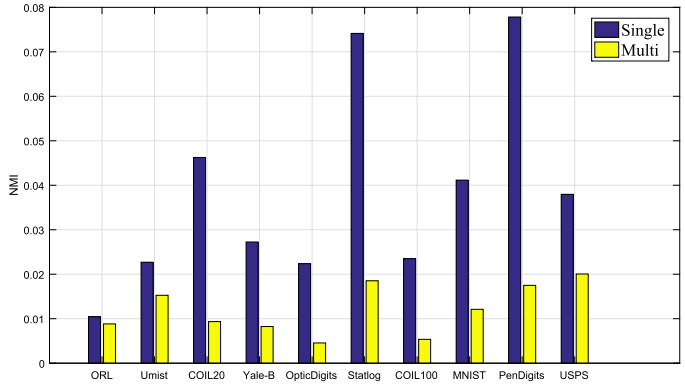
We first randomly produce ten different initialized Y_l. Then we test the proposed algorithm with every single set of Y_l and multiple sets including all the Y_l. In Fig. 5, 'Max' and 'Mean' denote the maximum and average values of clustering indices for the proposed algorithm with different Y_l, and 'Robust' represents the clustering indices for the proposed algorithm with all the Y_l. According to the figures, we can see that the proposed algorithm with multiple sets of constraints is obviously better than that with a single set of constraints. Besides, we also compare their robustness. In Fig. 6, we show the standard deviation of the clustering indices for the proposed algorithm with single and multiple sets of constraints. We can see that using multiple sets of constraints can enhance the robustness of the proposed algorithm.



(a) std of ACC



(b) std of ARI



(c) std of NMI

Fig. 6. Comparison of robustness between single and multiple constraints.

Additionally, we analyze the effects of the number of sets of self-learned constraints on the performance of the proposed algorithm. Fig. 7 shows the ACC, ARI, and NMI indices against different e on ten data sets, respectively. According to these figures, we can observe that the performance of the proposed algorithm basically increases as the value of e grows. However, the values of clustering indices increase slowly after the e value grows to a certain extent. The experimental results tell us on most tested data sets that (1) learning multiple sets of constraints is better than a single set; (2) learning a few sets of label constraints can effectively enhance the clustering results.

9.6. The effect of Y on the performance of the proposed algorithm

The performance of the proposed algorithm depends on the initialization of Y. In this paper, we need to select several objects and then use the relations between them and other objects to initialize Y. In this experiment, we employ

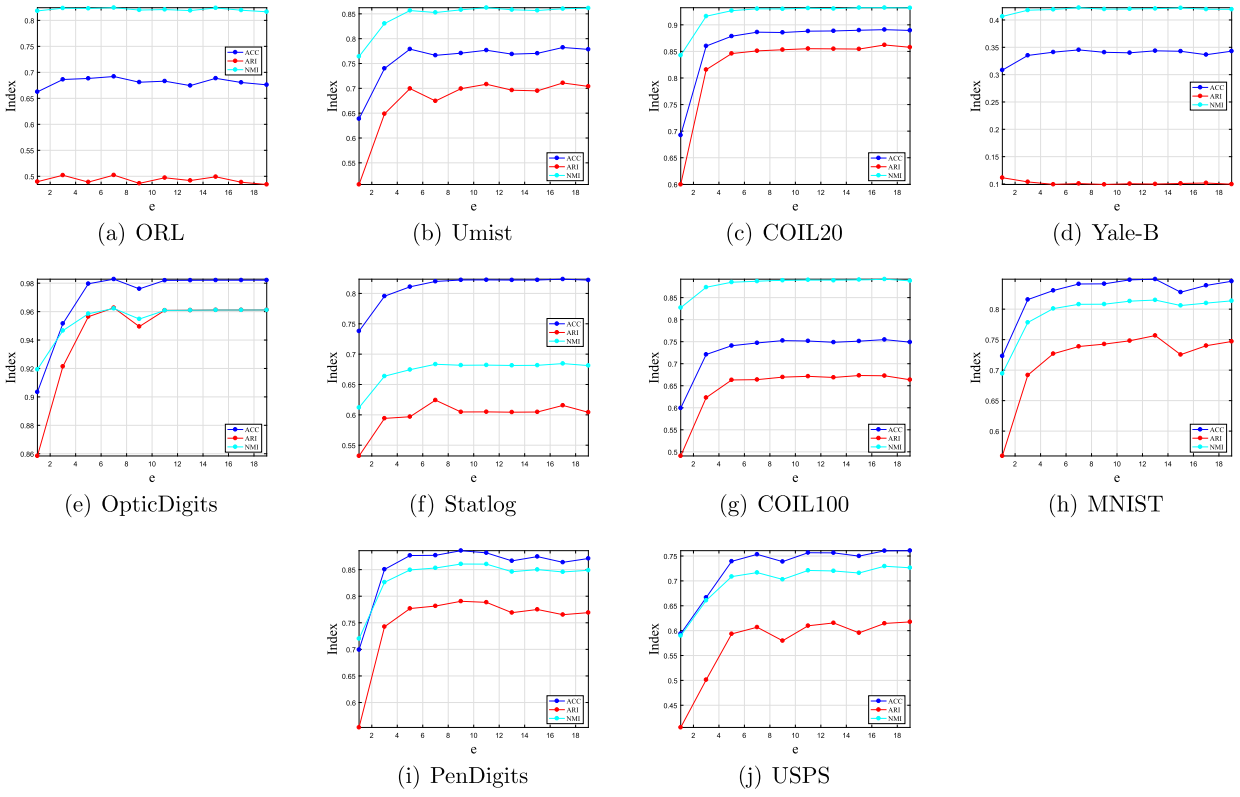
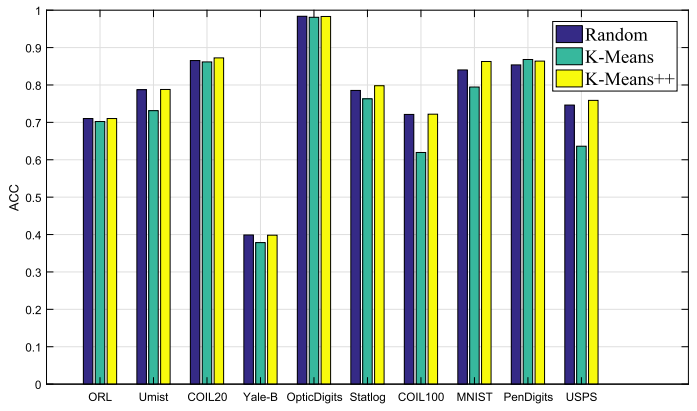


Fig. 7. Three clustering indices against the number of constraints.

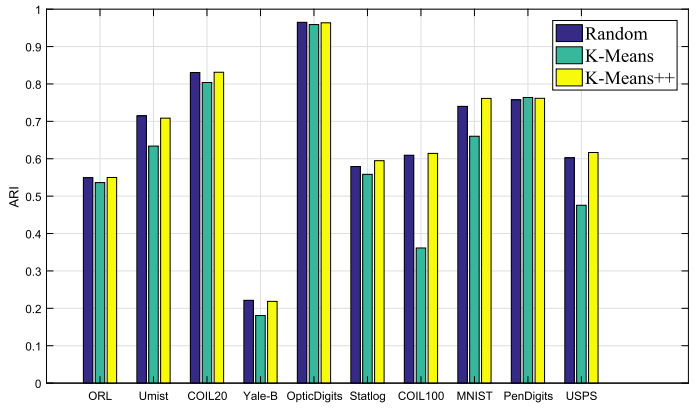
three selection strategies to initialize Y, i.e., random initialization (where we randomly select k objects from X), K-means initialization (where we implement k-means on X to produce k cluster centers and select the nearest objects for each of them from X) and K-means++ initialization (where we implement k-means++ on X to select k objects from X). We compare the proposed algorithm with the above three initialization methods on these data sets. The experimental results are shown in Fig. 8. We can see that the performance of the proposed algorithm with k-means++ initialization is better than the other two initialization methods. We can also see that random initialization is a better choice for simplicity and effectiveness.

Furthermore, we discuss the effect of the diversity of these constraints on the proposed algorithm. To detect the effect, we use overlapping ratios of different initial Y_l to reflect the consensus of the constraints. We assume the higher the overlapping ratio is, the smaller the difference among these constraints is. In Fig. 9, we show the clustering indices for the clustering results of the proposed algorithm with different overlapping ratios of the initial constraints on each data set. According to these figures, we can see that the overlap rate has less influence on the algorithm when it is less than a certain value on a data set. However, as the overlapping ratio continues increasing, the clustering quality is obviously decreasing on most of the data sets. The experimental result indicates that excessive overlap can lead to poor performance of the proposed algorithm. The main reasons are as follows. (1) The more consistent the different groups of the label constraints are, the closer the clustering performance with multiple groups of label constraints is that with the single group. Due the fact that we can not ensure the high quality of each group of the learned label constraints, the diversity of multiple groups can help us to reduce the effect of the quality of label constraints. (2) If a cluster is very complex, it is very difficult to learn a label column to represent it. We may need to learn multiple labels to describe it. Therefore, reducing overlapping between label matrices can help us to use multiple label columns to constrain the assignment of data objects to a cluster. Besides, the conclusion about the effect of the overlapping ratio of label constraints is dependent on the number of clusters. According to our experiments, we can see that the clustering quality is decreasing, as the overlapping ratio increases on the tested data sets with different numbers of clusters.

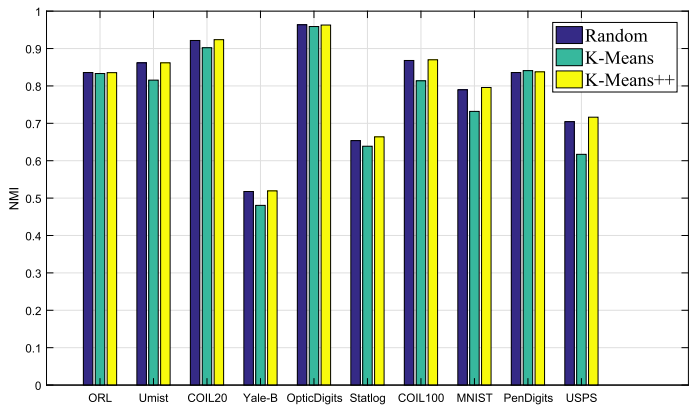
Finally, we analyze the quality of the self-label constraints in the proposed algorithm. In Figs. 10, we show the visualization of YY^T on each data set. According to these figures, we can see that the cluster structure can be easily found from the diagonal of the matrix YY^T on each of the tested data set. The experimental result indicates that the self-learned constraints have good quality, which can effectively guide the clustering process.



(a) ACC



(b) ARI



(c) NMI

Fig. 8. Comparison of different initialization for Y.

9.7. Convergence study of the proposed algorithm

In this paper, we have proved that the proposed algorithm is convergent. In this subsection, we further provide the experiment analysis to show the changing trend of the overall loss function Ω against the number of iterations, as seen in Fig. 11. According to the figures, we observe that as the number of iterations increases, the loss value decreases. We also see that the proposed algorithm can usually converge within 100 iterations.

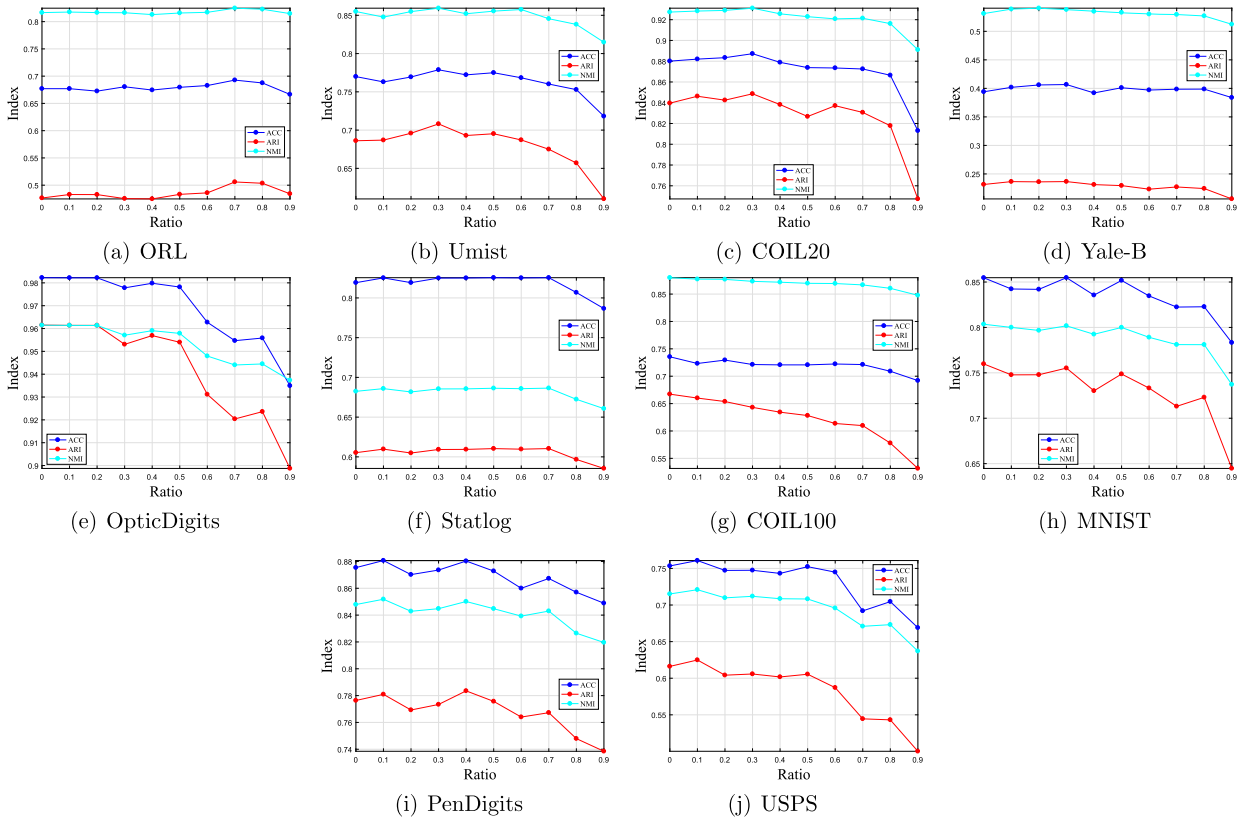


Fig. 9. Three clustering indices against the overlapping ratio.

Table 5
Comparison of running times (seconds).

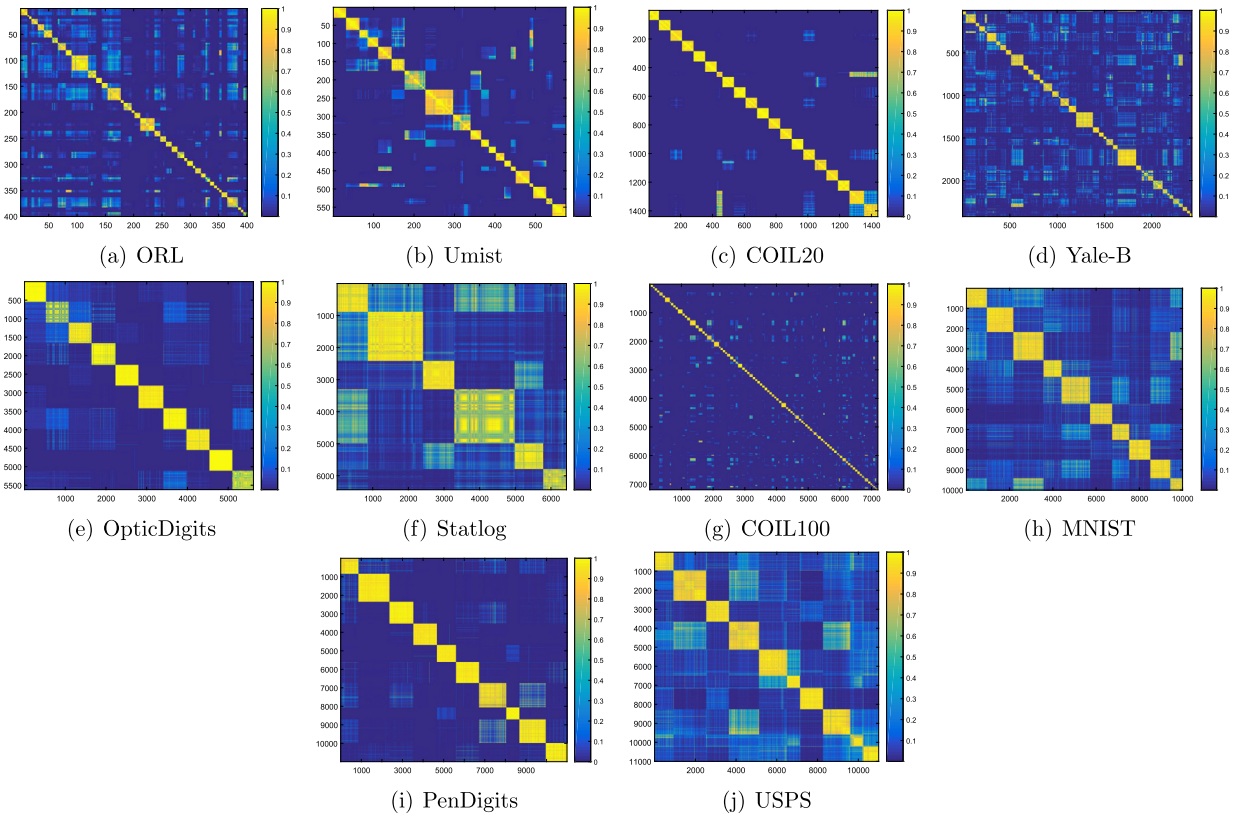
	ORL	Umist	COIL20	Yale-B	OpticDigits	Statlog	COIL100	MNIST	PenDigits	USPS
SC	0.2085	0.0895	0.1358	0.3414	0.1406	0.1412	3.5807	0.3260	0.4491	0.4712
RSLC-Single	0.2315	0.1023	0.1876	0.6551	0.5036	0.4473	6.1504	1.1658	1.1648	1.1683
RSLC-Multi	3.3347	1.6462	5.9965	21.9037	15.2059	10.6775	186.6335	32.7420	36.9136	32.2853

9.8. Comparison of computational costs

We compare the computational costs of the classical spectral clustering (SC) algorithm with fast Eigen decomposition and the proposed algorithm with single and multiple sets of self-learning constraints on each tested data set. The comparison result is shown in Table 5. According to the table, we can observe that the proposed algorithm needs additional time costs to iteratively update variables compared to the classical spectral clustering algorithm. Besides, we also can see that if we need to get a robust clustering result, we take more time costs to learn multiple sets of constraints compared to the proposed algorithm with a single set of constraints.

10. Conclusions

We proposed spectral clustering with robust self-learning constraints (RSLC) in this paper. In the new algorithm, we extend the objective function of the classical semi-supervised spectral clustering model by seeing label constraints as variables and adding a robust function. We wish to minimize the new objective function to learn multiple sets of label constraints and guide the spectral clustering process. We propose an iterative method to solve the optimization problem with update formulas for variables. The new algorithm can get robust self-constrained clustering results under unsupervised scenes. Furthermore, we provide the theoretical analysis to show the importance of the learned constraints in spectral clustering and then prove the convergence of the proposed algorithm. Finally, by experimental study, we show that the proposed algorithm performs well on benchmark data sets.

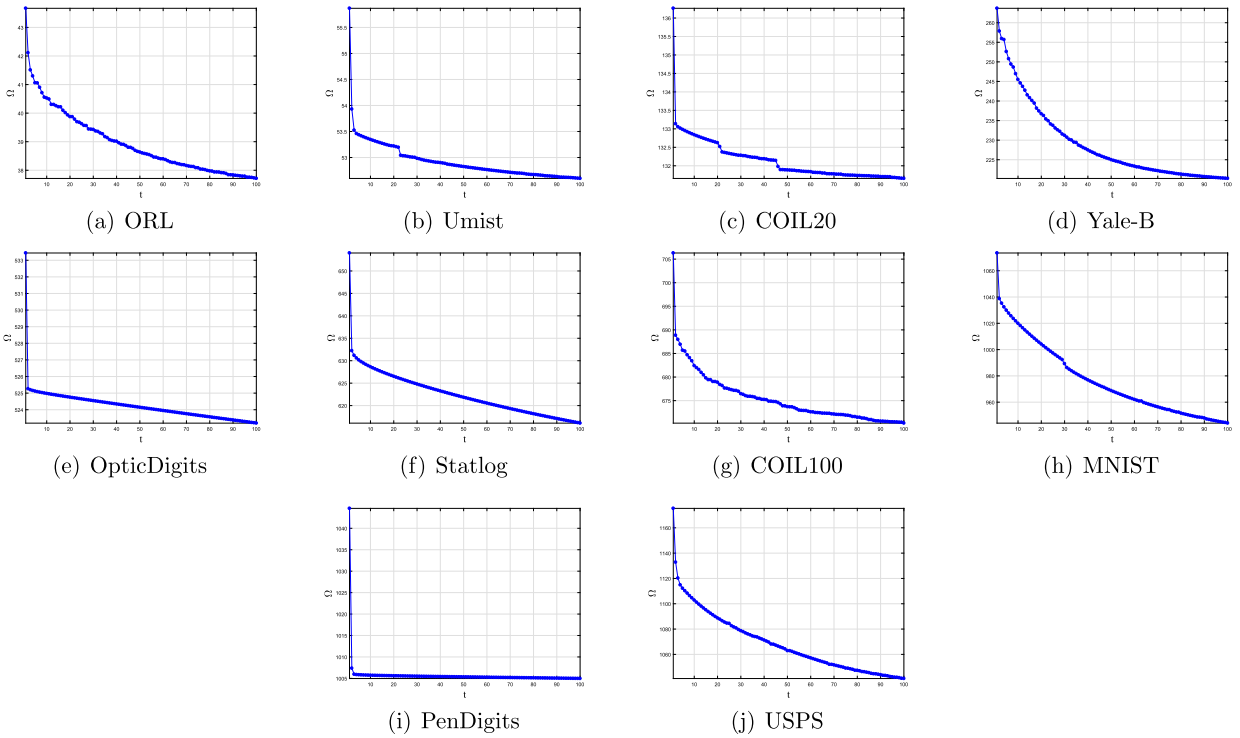


(a) ORL (b) Umist (c) COIL20 (d) Yale-B

(e) OpticDigits (f) Statlog (g) COIL100 (h) MNIST

(i) PenDigits (j) USPS

Fig. 10. Evaluation of the learned constraints.



(a) ORL (b) Umist (c) COIL20 (d) Yale-B

(e) OpticDigits (f) Statlog (g) COIL100 (h) MNIST

(i) PenDigits (j) USPS

Fig. 11. The loss values against the number of iterations.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

The authors are very grateful to the editors and reviewers for their valuable comments and suggestions. This work is supported by the National Key Research and Development Program of China (No. 2021ZD0113303), and the National Natural Science Foundation of China (Nos. 62022052, 62276159).

References

[1] A.K. Jain, Data clustering: 50 years beyond k-means, in: Machine Learning and Knowledge Discovery in Databases, Springer, Berlin, Heidelberg, 2008.
[2] C.C. Aggarwal, C.K. Reddy (Eds.), Data Clustering: Algorithms and Applications, CRC Press, 2014.
[3] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 888–905.
[4] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Advances in Neural Information Processing Systems, MIT Press, 2001, pp. 849–856.
[5] D. Zhou, O. Bousquet, T.N. Lal, J. Weston, B. Schölkopf, Learning with local and global consistency, in: Advances in Neural Information Processing Systems, vol. 16, 2003, pp. 321–328.
[6] L. Bai, J. Liang, F. Cao, Semi-supervised clustering with constraints of different types from multiple information sources, IEEE Trans. Pattern Anal. Mach. Intell. 43 (2021) 3247–3258.
[7] E. Elhamifar, R. Vidal, Sparse subspace clustering, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2009, pp. 2790–2797.
[8] M. Soltanolkotabi, E. Elhamifar, E. Candès, Robust subspace clustering, Ann. Stat. 42 (2013) 669–699.
[9] X. Peng, J. Feng, J.T. Zhou, Y. Lei, S. Yan, Deep subspace clustering, IEEE Trans. Neural Netw. Learn. Syst. 31 (2020) 5509–5521.
[10] Y. Chen, C.-G. Li, C. You, Stochastic sparse subspace clustering, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4154–4163.
[11] G. Liu, Z. Lin, Y. Yong, Robust subspace segmentation by low-rank representation, in: Proceedings of the 27th International Conference on Machine Learning (ICML-2010), 2010, pp. 663–670.
[12] X. Zhu, S. Zhang, Y. Li, J. Zhang, L. Yang, Y. Fang, Low-rank sparse subspace for spectral clustering, IEEE Trans. Knowl. Data Eng. 31 (2019) 1532–1543.
[13] F. Nie, X. Wang, H. Huang, Clustering and projected clustering with adaptive neighbors, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014.
[14] F. Nie, X. Wang, M.I. Jordan, H. Huang, The constrained laplacian rank algorithm for graph-based clustering, in: AAAI Conference on Artificial Intelligence, 2016.
[15] F. Nie, W. Chang, Z. Hu, X. Li, Robust subspace clustering with low-rank structure constraint, IEEE Trans. Knowl. Data Eng. 34 (2022) 1404–1415.
[16] I.S. Dhillon, Y. Guan, B. Kulis, Weighted graph cuts without eigenvectors: a multilevel approach, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 1944–1957.
[17] H. Liu, J. Wu, T. Liu, D. Tao, Y. Fu, Spectral ensemble clustering via weighted k-means: theoretical and practical evidence, IEEE Trans. Knowl. Data Eng. 29 (2017) 1129–1143.
[18] J. Liu, C. Wang, M. Danilevsky, J. Han, Large-scale spectral clustering on graphs, in: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, 2013, pp. 1486–1492.
[19] C.C. Fowlkes, S.J. Belongie, F.R.K. Chung, J. Malik, Spectral grouping using the Nyström method, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 214–225.
[20] D. Cai, X. Chen, Large scale spectral clustering via landmark-based sparse representation, IEEE Trans. Cybern. 45 (2015) 1669–1680.
[21] W.Y. Chen, H. Bai, H. Bai, E.Y. Chang, E.Y. Chang, Parallel spectral clustering in distributed systems, IEEE Trans. Pattern Anal. Mach. Intell. 33 (2011) 568–586.
[22] H. Li, N. Ray, Y. Guan, Z. Hong, Fast large-scale spectral clustering via explicit feature mapping, IEEE Trans. Cybern. 49 (2019) 1058–1071.
[23] M. Mohan, C. Monteleoni, Beyond the Nystrom approximation: speeding up spectral clustering using uniform sampling and weighted kernel k-means, in: Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017, pp. 2494–2500.
[24] L. Bai, J. Liang, A three-level optimization model for nonlinearly separable clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 3211–3218.
[25] D. Huang, C. Wang, J. Wu, J. Lai, C. Kwoh, Ultra-scalable spectral clustering and ensemble clustering, IEEE Trans. Knowl. Data Eng. 32 (2020) 1212–1226.
[26] O. Zoidi, A. Tefas, N. Nikolaidis, I. Pitas, Positive and negative label propagations, IEEE Trans. Circuits Syst. Video Technol. 28 (2018) 342–355.
[27] Z. Li, J. Liu, X. Tang, Constrained clustering via spectral regularization, in: Proceedings of IEEE Computer Vision and Pattern Recognition, 2009, pp. 421–428.
[28] Z. Lu, Y. Peng, Exhaustive and efficient constraint propagation: a graph-based learning approach and its applications, Int. J. Comput. Vis. 103 (2013) 306–325.
[29] H. Liu, Z. Tao, Y. Fu, Partition level constrained clustering, IEEE Trans. Pattern Anal. Mach. Intell. 40 (2018) 2469–2483.
[30] I. Jang, G. Danley, K. Chang, J. Kalpathy-Cramer, Decreasing Annotation Burden of Pairwise Comparisons with Human-in-the-Loop Sorting: Application in Medical Image Artifact Rating, in: NeurIPS Data-Centric AI Workshop, 2021.
[31] J. Zhang, C.G. Li, C. You, X. Qi, Z. Lin, Self-supervised convolutional subspace clustering network, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.
[32] X. Ji, A. Vedaldi, J. Henriques, Invariant information clustering for unsupervised image classification and segmentation, in: IEEE/CVF International Conference on Computer Vision, 2019, pp. 9864–9873.
[33] Y.M. Asano, C. Rupprecht, A. Vedaldi, Self-labelling via simultaneous clustering and representation learning, in: International Conference on Learning Representation, 2020.

[34] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, in: International Conference on Machine Learning, vol. 48, 2016, pp. 478–487.
[35] J. Yang, D. Parikh, D. Batra, Joint unsupervised learning of deep representations and image clusters, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5147–5156.
[36] J. Lv, Z. Kang, X. Lu, Z. Xu, Pseudo-supervised deep subspace clustering, IEEE Trans. Image Process. 30 (2021) 5252–5263.
[37] P. Ji, T. Zhang, H. Li, M. Salzmann, I. Reid, Deep subspace clustering networks, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., 2017.
[38] Yunfan Li, Mouxing Yang, Dezhong Peng, Taihao Li, Jiantao Huang, Xi Peng, Twin Contrastive Learning for Online Clustering, Int. J. Comput. Vis. 130 (2022) 2205–2221.
[39] U. Shaham, K. Stanton, H. Li, B. Nadler, R. Basri, Y. Kluger, Spectralnet: spectral clustering using deep neural networks, in: International Conference on Learning Representation, 2018.
[40] J. Chang, G. Meng, L. Wang, S. Xiang, C. Pan, Deep self-evolution clustering, IEEE Trans. Pattern Anal. Mach. Intell. 42 (2020) 809–823.
[41] A. Argyriou, T. Evgeniou, M. Pontil, Multi-task feature learning, in: Advances in Neural Information Processing Systems, 2006.
[42] F. Nie, H. Huang, X. Cai, C. Ding, Efficient and robust feature selection via joint 2, 1-norms minimization, in: Advances in Neural Information Processing Systems, 2010.
[43] D. Cai, X. He, J. Han, T.S. Huang, Graph regularized nonnegative matrix factorization for data representation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (2011).
[44] S. Shalev-Shwartz, S. Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, 2014.
[45] W.I. Zangwill, Nonlinear Programming: A Unified Approach, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1969.
[46] L. Bai, J. Liang, C. Dang, F. Cao, The impact of cluster representatives on the convergence of the k-modes type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 1509–1522.
[47] D.D. Lee, H.S. Seung, Algorithms for non-negative matrix factorization, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, vol. 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA, MIT Press, 2000, pp. 556–562.
[48] J.C. Bezdek, A convergence theorem for the fuzzy isodata clustering algorithms, IEEE Trans. Pattern Anal. Mach. Intell. 2 (1980) 1–8.
[49] C. Deng, Codes and dataset for feature learning, http://www.cad.zju.edu.cn/home/dengcai/Data/data.html, 2019.