A Simple Approach to Automated Spectral Clustering

Jicong Fan[1,2], **Yiheng Tu**[3,4], **Zhao Zhang**[5,*], **Mingbo Zhao**[6], **Haijun Zhang**[7]
[1]The Chinese University of Hong Kong, Shenzhen [2]Shenzhen Research Institute of Big Data
[3]Chinese Academy of Science, Beijing [4]University of Chinese Academy of Sciences, Beijing
[5]Hefei University of Technology, Hefei [6]Donghua University, Shanghai
[7]Harbin Institute of Technology, Shenzhen
fanjicong@cuhk.edu.cn yihengtu@gmail.com cszzhang@gmail.com
mzhao4@dhu.edu.cn hjzhang@hit.edu.cn

Abstract

The performance of spectral clustering heavily relies on the quality of affinity matrix. A variety of affinity-matrix-construction (AMC) methods have been proposed but they have hyperparameters to determine beforehand, which requires strong experience and leads to difficulty in real applications, especially when the inter-cluster similarity is high and/or the dataset is large. In addition, we often need to choose different AMC methods for different datasets, which still depends on experience. To solve these two challenging problems, in this paper, we present a simple yet effective method for automated spectral clustering. First, we propose to find the most reliable affinity matrix via grid search or Bayesian optimization among a set of candidates given by different AMC methods with different hyperparameters, where the reliability is quantified by the *relative-eigen-gap* of graph Laplacian introduced in this paper. Second, we propose a fast and accurate AMC method based on least squares representation and thresholding and prove its effectiveness theoretically. Finally, we provide a large-scale extension for the automated spectral clustering method, of which the time complexity is linear with the number of data points. Extensive experiments of natural image clustering show that our method is more versatile, accurate, and efficient than baseline methods.

1 Introduction

Clustering is an important approach to data mining and knowledge discovery. Particularly, spectral clustering [Weiss, 1999; Shi and Malik, 2000; Ng *et al.*, 2002; Von Luxburg, 2007] has superior performance than k-means clustering [Steinhaus and others, 1956], hierarchical clustering [Johnson, 1967], DBSCAN [Ester *et al.*, 1996], and mixtures of probabilistic principal component analyzers [Tipping and Bishop, 1999] in many applications. Roughly speaking, spectral clustering consists of two steps: 1) construct an affinity matrix in which each element denotes the similarity between two data points; 2) perform normalized cut [Shi and Malik, 2000] on the graph corresponding to the affinity matrix. K-nearest neighbors (K-NN) and Gaussian kernel $k(\boldsymbol{x},\boldsymbol{y}) = \exp(-\|\boldsymbol{x}-\boldsymbol{y}\|^2/(2\varsigma^2))$ are two popular methods to construct affinity matrices, where k and ς are hyperparameters.

As the performance of spectral clustering heavily relies on the quality of affinity matrix, in recent years, a variety of methods have been proposed to construct or learn affinity matrices for spectral clustering. Many of them are in the framework of self-expressive [Roweis and Saul, 2000; Elhamifar and Vidal, 2013] model, i.e., $\text{minimize}_{\boldsymbol{C}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 + \lambda\mathcal{R}(\boldsymbol{C})$. Here the columns of $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ are the data points drawn from a union of subspaces. $\boldsymbol{C} \in \mathbb{R}^{n\times n}$ is a coefficient matrix. $\mathcal{R}(\boldsymbol{C})$ denotes a regularization operator on $\boldsymbol{C}$. λ is a hyperparameter to be determined in advance.

*Corresponding author

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Elhamifar and Vidal [2013] proposed to use $\mathcal{R}(\boldsymbol{C}) = \|\boldsymbol{C}\|_1 := \sum_{i=1}^n \sum_{j=1}^n |c_{ij}|$ under a constraint $\text{diag}(\boldsymbol{C}) = \boldsymbol{0}$. In [Elhamifar and Vidal, 2013], the affinity matrix for spectral clustering is given by $\boldsymbol{A} = |\boldsymbol{C}| + |\boldsymbol{C}|^\top$. The method is called Sparse Subspace Clustering (SSC). Some theoretical results of SSC can be found in [Wang and Xu, 2013; Soltanolkotabi *et al.*, 2014].

Following the self-expressive framework, Liu *et al.* [2013] let $\mathcal{R}(\boldsymbol{C}) = \|\boldsymbol{C}\|_*$ (nuclear norm of $\boldsymbol{C}$) and proposed a Low-Rank Representation (LRR) method for subspace clustering. Lu *et al.* [2012] and Pan Ji *et al.* [2014] used the least squares representation (LSR) model for subspace clustering. A few variants of LRR and SSC can be found in [Patel and Vidal, 2014; Li and Vidal, 2015; Patel *et al.*, 2015; Shen and Li, 2016; Li and Vidal, 2016; Fan and Chow, 2017; Fan *et al.*, 2018; Lu *et al.*, 2018; Pan and Kang, 2021; Kang *et al.*, 2022]. Recently, deep learning methods were also used to learn affinity matrices for spectral clustering [Ji *et al.*, 2017; Zhang *et al.*, 2019b,a; Lv *et al.*, 2021] and have achieved state-of-the-art performance on many benchmark datasets.

One common limitation of these spectral or subspace clustering methods is that they have at least one hyperparameter to determine. In the codes of SSC[2] and its variants provided by their authors, there is usually one more thresholding parameter for affinity matrix, which affects the clustering accuracy a lot. In the deep learning clustering methods such as [Ji *et al.*, 2017] and [Zhang *et al.*, 2019b], we need to determine the network structures and regularization parameters, which is much more difficult. Since clustering is an unsupervised learning problem, the hyperparameters cannot be tuned by cross-validation widely used in supervised learning. Thus we have to tune the hyperparameters in spectral clustering by experience, which is difficult when the dataset is quite different from those in our experience and/or the inter-class similarity is high compared to the intra-cluster similarity. Note that SSC, LRR, and their kernel or deep learning extensions have quadratic or even cubic time complexity (per iteration), which further increases the difficulty of hyperparameter selection in clustering large datasets, though there have been a few works improving the computational efficiency [Peng *et al.*, 2013; Cai and Chen, 2014; Wang *et al.*, 2014; Peng *et al.*, 2015; You *et al.*, 2016a,b; Li and Zhao, 2017; You *et al.*, 2018; Matsushima and Brbic, 2019; Li *et al.*, 2020; Chen *et al.*, 2020; Kang *et al.*, 2020; Fan, 2021; Cai *et al.*, 2022]. On the other hand, different datasets often require different AMC methods, which is hard to tackle by experience.

This paper aims at model and hyperparameter selection for spectral clustering and wants to improve the convenience, accuracy, and efficiency of spectral clustering. Our contributions are as follows.

- We propose a *relative-eigen-gap* based automated spectral clustering (AutoSC) method. It finds the Laplacian matrix with largest *relative-eigen-gap* among a set of candidates constructed by different models with different hyperparameters.
- We also implement the AutoSC method via Bayesian optimization. The method can select the possibly best model and optimize the hyperparameters automatically. Note that any AMC methods (e.g. SSC) can be included in the framework of AutoSC.
- To improve the accuracy and efficiency of AutoSC, we propose a new AMC method based on least squares representation and thresholding and prove its effectiveness theoretically.
- We provide an extension for AutoSC to cluster large-scale datasets.

Experiments on seven benchmark image datasets demonstrate the effectiveness of our method. Particularly, our method outperforms state-of-the-art methods of large-scale clustering.

2 Related work

Exploiting eigenvalue information for clustering As the number of zero eigenvalues of a Laplacian matrix is equal to the number of connected components of the graph [Von Luxburg, 2007], a few researchers took advantage of eigenvalue information in spectral clustering [Meila *et al.*, 2005; Meila and Shortreed, 2006; Ji *et al.*, 2015; Hu *et al.*, 2017; Lu *et al.*, 2018]. For instance, Ji *et al.* [2015] utilized eigen-gap to determine the rank of the Shape Interaction Matrix. But the method requires determining another hyperparameter γ beforehand and needs to perform spectral clustering multiple times. The methods of [Meila *et al.*, 2005; Hu *et al.*, 2017; Lu *et al.*, 2018] are based on

[2] Wang and Xu [2013] and Soltanolkotabi *et al.* [2014] provided lower and upper bounds for the λ in SSC theoretically, which however depend on the unknown noise level.

iterative optimization (need to perform eigenvalue decomposition at every iteration) and hence are not effective in handling large-scale datasets. In addition, the BDR method of [Lu *et al.*, 2018] has two hyperparameters (λ, γ) to determine by experience, although it outperformed SSC and LRR on some datasets. A comparison is shown in Figure 1.

Automated machine learning Automated model and hyperparameter selection for supervised learning have been extensively studied [Hutter *et al.*, 2019]. In contrast, the study for unsupervised learning is very limited. The reason is that in unsupervised learning there is no ground truth or reliable metric to evaluate the performance of algorithms. Concurrently to our work, Poulakis [2020] also attempted to do automated clustering. Specifically, Poulakis [2020] proposed to use meta-learning to select clustering algorithm and use a heuristic combination of some clustering validity metrics such as Silhouette coefficient [Liu *et al.*, 2010] and S_Dbw [Halkidi and Vazirgiannis, 2001] as an objective to maximize via grid search or Bayesian optimization [Jones *et al.*, 1998]. One problem is that these metrics are mainly based on Euclidean distance or densities and hence may not be suitable to evaluate the clustering performance of non-distance or non-density based clustering algorithms. Another one is that there is no unified metric to compare different clustering algorithms.

3 Automated Spectral Clustering (AutoSC)

3.1 Preliminary Knowledge

Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ be an affinity matrix constructed from a given data matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$. The corresponding graph is denoted by $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ is the vertex set and $E = \{e_1, \ldots, e_l\}$ is the edge set. The degree matrix of a graph G is defined as $\boldsymbol{D} = \text{diag}(\boldsymbol{A}\mathbf{1})$, where $\mathbf{1} = [1, \ldots, 1]^\top$. Our goal is to partition the vertices into k disjoint nonempty subsets $C_1, \ldots, C_k$. Let $\mathcal{C} = \{C_1, \ldots, C_k\}$. It is expected to find a partition $\mathcal{C}$ that minimizes the following metric.

Definition 3.1 (MNCut)**.** The multiway normalized cut (MNCut) [Meila, 2001] is defined as

$$\text{MNCut}(\mathcal{C}) = \sum_{i=1}^{k} \sum_{j\neq i} \frac{\text{Cut}(C_i, C_j)}{\text{Vol}(C_i)}, \tag{1}$$

where $\text{Cut}(C_i, C_j) = \sum_{u\in C_i} \sum_{v\in C_j} A_{uv}$ and $\text{Vol}(C_i)$ denotes the sum of vertex degrees of C_i.

The normalized graph Laplacian matrix is defined as

$$\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}, \tag{2}$$

where $\boldsymbol{I}$ is an identity matrix. The normalized graph Laplacian is often more effective than the unnormalized one in spectral clustering (some theoretical justification was given by [Von Luxburg, 2007]). Let $\sigma_i(\boldsymbol{L})$ be the i-th smallest eigenvalue of $\boldsymbol{L}$. The following claim shows the connection between $\text{MNCut}(\mathcal{C})$ and $\boldsymbol{L}$.

Claim 3.2. *The sum of the k smallest singular values of $\boldsymbol{L}$ quantifies the potential connectivity among $C_1, \ldots, C_k$:* $\text{MNCut}(\mathcal{C}) \geq \sum_{i=1}^{k} \sigma_i(\boldsymbol{L})$.

The claim can be easily proved by using Lemma 4 of [Meila, 2001]. We defer all proof of this paper to the appendices. Because the multiplicity k of the eigenvalue 0 of $\boldsymbol{L}$ equals the number of connected components in G [Von Luxburg, 2007], we expect to construct an affinity matrix $\boldsymbol{A}$ from $\boldsymbol{X}$ such that $\boldsymbol{L}$ has k zero eigenvalues. Thus the optimal partition means $\text{MNCut}(\mathcal{C}) = \sum_{i=1}^{k} \sigma_i(\boldsymbol{L}) = 0$.

3.2 Relative Eigen-Gap Guided Search

In practice, we may construct an $\boldsymbol{A}$ such that $\sum_{i=1}^{k} \sigma_i(\boldsymbol{L})$ is as small as possible because guaranteeing zero eigenvalues is difficult. But this is not enough because $\boldsymbol{L}$ may have $k+1$ or more very small or even zero eigenvalues. The second smallest eigenvalue of the Laplacian matrix of a graph G is called the algebraic connectivity of G (denoted by $ac(G)$) [Fiedler, 1973]. We have $ac(G) = 0$ if and only if G is not connected. When G has k disjointed components, there are k algebraic connectivities, denoted by $ac(C_1), \ldots, ac(C_k)$. Based on this, we have

Claim 3.3. *The k+1th smallest eigenvalue of $\boldsymbol{L}$ quantifies the least potential connectivity of partitions $C_1, \ldots, C_k$ of $\mathcal{C}$:*

$$\min_{1 \le i \le k} \text{MNCut}(C_i) \ge \sigma_{k+1}(\boldsymbol{L}). \tag{3}$$

In other words, $\sigma_{k+1}(\boldsymbol{L})$ measures the difficulty in segmenting each of C_i into two subsets. Hence, when $\sigma_{k+1}(\boldsymbol{L})$ is large, the partitions $C_1, \ldots, C_k$ are stable. Based on Claim 3.2 and Claim 3.3, we may construct an $\boldsymbol{A}$ that has small $\sum_{i=1}^{k} \sigma_i(\boldsymbol{L})$ and large $\sigma_{k+1}(\boldsymbol{L})$ simultaneously, by solving

$$\begin{aligned} &\underset{\theta}{\text{maximize}} \;\; \sigma_{k+1}(\boldsymbol{L}) - \frac{1}{k}\sum_{i=1}^{k} \sigma_i(\boldsymbol{L}), \\ &\text{subject to} \;\; \boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}, \;\; \boldsymbol{A} = f_\theta(\boldsymbol{X}). \end{aligned} \tag{4}$$

where $f_\theta : \mathbb{R}^{m \times n} \to \mathbb{R}^{n \times n}$ is a function with parameter θ, e.g. $\boldsymbol{A} = [\exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/(2\varsigma^2))] - \boldsymbol{I}$. It is difficult to solve (4) because of the composition of f_θ, symmetric normalized Laplacian, and eigenvalue decomposition. On the other hand, in (4), we have to choose f in advance, which requires domain expertise or strong experience because different dataset usually needs different f.

Note that different f can result in very different distributions of eigenvalues and the small eigenvalues are sensitive to f, θ, and noise. Hence the objective in (4) is not effective to compared different f and θ. In this paper, we define a new metric *relative-eigen-gap* as follows

$$\text{reg}(\boldsymbol{L}) := \frac{\sigma_{k+1}(\boldsymbol{L}) - \frac{1}{k}\sum_{i=1}^{k} \sigma_i(\boldsymbol{L})}{\frac{1}{k}\sum_{i=1}^{k} \sigma_i(\boldsymbol{L}) + \varepsilon}, \tag{5}$$

where ε is a small constant (e.g. 10^{-6}) to avoid zero denominator. reg($\boldsymbol{L}$) is not sensitive to the scale of the small eigenvalues. Therefore, instead of (4), we propose to solve

$$\begin{aligned} &\underset{(f,\theta) \in \mathcal{F} \times \Theta}{\text{maximize}} \;\; \text{reg}(\boldsymbol{L}), \\ &\text{subject to} \;\; \boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}, \;\; \boldsymbol{A} = f_\theta(\boldsymbol{X}), \end{aligned} \tag{6}$$

where $\mathcal{F}$ is a set of pre-defined functions and Θ is a set of hyperparameters. In fact, (6) is equivalent to choosing one $\boldsymbol{A}$ (or $\boldsymbol{L}$) from a set of candidates constructed by different f with different θ, of which the *relative-eigen-gap* is largest. The best θ can be found using grid search (or even random search). For convenience, we call the method AutoSC-GD. Table 1 shows a few examples of f and its parameters. One may use a weighted sum of affinity matrices given by different f, like [Huang *et al.*, 2012], which however will introduce more hyperparameters.

Table 1: A few examples of f and its θ for AMC (AASC: [Huang *et al.*, 2012])

f	K-NN	ϵ-neighborhood	Gaussian kernel	SSC	LRR	LSR	KSSC	AASC
θ	K	ϵ	σ	λ	λ	λ	λ, σ	$\sigma_1, \sigma_2, \ldots$

The following theorem[3] shows the connection between reg($\boldsymbol{L}$) and the stability of the clustering $\mathcal{C}$.

Theorem 3.4. *Let $\mathcal{C}$ and $\mathcal{C}'$ be two partitions of the vertices of G, where $|\mathcal{C}| = |\mathcal{C}'| = k$. Define the distance between $\mathcal{C}$ and $\mathcal{C}'$ as* $\text{dist}(\mathcal{C}, \mathcal{C}') = 1 - \frac{1}{k}\sum_{C_i \in \mathcal{C}}\sum_{C'_j \in \mathcal{C}'} \frac{(\text{Vol}(C_i \cap C'_j))^2}{\text{Vol}(C_i)\text{Vol}(C'_j)}$. *Suppose $\eta k\varepsilon \ge \sum_{i=1}^{k} \sigma_i(\boldsymbol{L}) \ge k\varepsilon$ and* reg($\boldsymbol{L}$) $> (k-1)\eta/2$. *Let* $\delta = \max\big(\text{MNCut}(\mathcal{C}) - \sum_{i=1}^{k} \sigma_i(\boldsymbol{L}), \text{MNCut}(\mathcal{C}') - \sum_{i=1}^{k} \sigma_i(\boldsymbol{L})\big)$. *Then*

$$\text{dist}(\mathcal{C}, \mathcal{C}') < \frac{1.5\delta\varepsilon^{-1}}{\text{reg}(\boldsymbol{L}) + (1-k)\eta/2}. \tag{7}$$

It indicates that when reg($\boldsymbol{L}$) is large and δ is small, the partitions $\mathcal{C}$ and $\mathcal{C}'$ are close to each other. Thus, the clustering has high stability. When $\sum_{i=1}^{k} \sigma_i(\boldsymbol{L}) = k\varepsilon$, we have $\text{dist}(\mathcal{C}, \mathcal{C}') < \frac{6\delta}{\sigma_{k+1}(\boldsymbol{L}) - k\varepsilon}$, which means the larger $\sigma_{k+1}(\boldsymbol{L})$ the more stable clustering.

[3]This theorem is a modified version of Theorem 1 in [Meila *et al.*, 2005], which is for the eigen-gap $\sigma_{k+1} - \sigma_k$ of $\boldsymbol{L}$. Here we consider reg($\boldsymbol{L}$) instead.

Compare reg($\boldsymbol{L}$) with [Meila and Shortreed, 2006] It is worth noting that Meila and Shortreed [2006] proposed to minimize $\mathrm{J}(\boldsymbol{L}) := \frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L}) + \alpha\left(\sigma_k(\boldsymbol{L}) - \sigma_{k+1}(\boldsymbol{L})\right)^2$ to find a good affinity matrix for spectral clustering. Although reg($\boldsymbol{L}$) and J($\boldsymbol{L}$) seem similar, they are essentially different. As mentioned before, different AMC methods may lead to different scales for the small eigenvalues, which makes it difficult to compare different AMC methods using J($\boldsymbol{L}$). In addition, J($\boldsymbol{L}$) has a hyperparameter α to determine beforehand, which violates our goal of searching models and hyperparameters. A comparative study ($\alpha \leq 0$) is in Section 4.1 (Table 3).

3.3 AutoSC via Bayesian Optimization

Bayesian optimization (BO) [Jones *et al.*, 1998] has become a promising tool for hyperparameter optimization of supervised machine learning algorithms [Snoek *et al.*, 2012; Klein *et al.*, 2017]. Given a black-box function $g : \mathcal{X} \to \mathbb{R}$, BO aims to find an $\boldsymbol{x}^* \in \mathcal{X}$ that globally minimizes g and usually has three steps. The first step is finding the most promising point $\boldsymbol{x}_{t+1} \in \mathrm{argmax}_{\boldsymbol{x}}\, a_{p(g)}(\boldsymbol{x})$ by numerical optimization, where $a_{p(g)} : \mathcal{X} \to \mathbb{R}$ is an acquisition function (e.g. Expected Improvement) relying on an prior $p(g)$ (e.g. Gaussian processes [Williams and Rasmussen, 2006]). The second step is evaluating the expensive and possibly noisy function $y_{t+1} \sim g(\boldsymbol{x}) + \mathcal{N}(0, \sigma^2)$ and adding the new sample $(\boldsymbol{x}_{t+1}, y_{t+1})$ to the observation set $\mathcal{D}_t = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_t, y_t)\}$. The last step is updating $p(g)$ and $a_{p(g)}$ using $\mathcal{D}_{t+1}$.

As an alternative to the grid search for (6), we can maximize reg($\boldsymbol{L}$) via BO. Suppose we have a set of different AMC models, i.e., $\mathcal{F} = \{f_1, f_2, \ldots, f_M\}$. For $i = 1, 2, \ldots, M$, let

$$g_i(\theta^{(i)}) := -\mathrm{reg}(\boldsymbol{L}(f_i(\theta^{(i)}|\mathbb{X}))),$$

where $\theta^{(i)}$ denotes the hyperparameters in f_i and $\mathbb{X}$ denotes the dataset. Then we use BO to find

$$\theta_*^{(i)} = \underset{\theta^{(i)} \in S^{(i)}}{\mathrm{argmin}}\; g_i(\theta^{(i)}), \tag{8}$$

where $S^{(i)}$ denotes the set of constraints. Finally we get the best model with its best hyperparameters

$$f_\star(\theta_*^{(\star)}|\mathbb{X}), \quad \text{where } \star = \underset{1 \leq i \leq M}{\mathrm{argmin}}\, g_i(\theta_*^{(i)}). \tag{9}$$

For convenience, we denote the method by AutoSC-BO. Note that $\mathcal{F}$ can include any AMC methods such as those in Table 1 and even DSC [Ji *et al.*, 2017] (see Appendix D.5).

In AutoSC-BO, we use Expected Improvement (EI) acquisition function

$$a_{\mathrm{EI}}(\boldsymbol{s}|\mathcal{D}_t) = \mathbb{E}_p\left[\max(g_{\min} - g(\boldsymbol{s}), 0)\right], \tag{10}$$

where $g_{\min}$ is the best function value known. The closed-form formulation is

$$a_{\mathrm{EI}}(\boldsymbol{s}|\mathcal{D}_t) = (g_{\min} - \mu)\Phi\left(\frac{g_{\min} - \mu}{\sigma}\right) + \phi\left(\frac{g_{\min} - \mu}{\sigma}\right), \tag{11}$$

where $\mu = \mu(\boldsymbol{s}|\mathcal{D}_t, \theta_K)$ and $\sigma = \sigma(\boldsymbol{s}|\mathcal{D}_t, \theta_K)$ are the mean value and variance of the Gaussian process, ϕ and Φ are standard Gaussian cumulative density function and probability density function respectively, and θ_K denotes the hyperparameters of the Gaussian process. For the covariance function, we use the automatic relevance determination (ARD) Matérn $5/2$ kernel [Matérn, 2013]

$$k_{M52}(\boldsymbol{s}, \boldsymbol{s}') = \theta_0\left(1 + \sqrt{5r^2(\boldsymbol{s}, \boldsymbol{s}')} + \frac{5}{3}r^2(\boldsymbol{s}, \boldsymbol{s}')\right) \times \exp\left(-\sqrt{5r^2(\boldsymbol{s}, \boldsymbol{s}')}\right), \tag{12}$$

where $r^2(\boldsymbol{s}, \boldsymbol{s}') = \sum_{j=1}^{d}(s_j - s_j')^2/\theta_j^2$.

3.4 Discussion on AMC Methods for AutoSC and LSR with Thresholding

In AutoSC, the size of searching space is $|\mathcal{F}| \times \prod_j |\theta_j|$, which should be large enough to include effective models and their hyperparameters. A large number of works have shown that the self-expressive models [Elhamifar and Vidal, 2013; Liu *et al.*, 2013; Lu *et al.*, 2018; Ji *et al.*, 2017] often outperform other AMC models such as Gaussian kernel. However, the self-expressive model based

AMC methods often require iterative optimization and has at least quadratic time complexity per iteration, which leads to huge time cost in AutoSC. Although LSR [Lu *et al.*, 2012] has closed-form solution, the clustering accuracy is not satisfactory [Lu *et al.*, 2018]. In this work, we will show that LSR with a simple post-processing operation can be a good AMC method and can outperform SSC, LRR, and BDR [Lu *et al.*, 2018]. Specifically, the LSR model is given as

$$\underset{\boldsymbol{C}}{\text{minimize}} \quad \tfrac{1}{2}\|\boldsymbol{X}-\boldsymbol{X}\boldsymbol{C}\|_F^2+\tfrac{\lambda}{2}\|\boldsymbol{C}\|_F^2, \tag{13}$$

of which the closed-form solution is $\boldsymbol{C}=(\boldsymbol{X}^\top\boldsymbol{X}+\lambda\boldsymbol{I})^{-1}\boldsymbol{X}^\top\boldsymbol{X}$. Let $\text{diag}(\boldsymbol{C})=\boldsymbol{0}$ and $\boldsymbol{C}\leftarrow|\boldsymbol{C}|$, the affinity matrix can be constructed as $\boldsymbol{A}=(\boldsymbol{C}+\boldsymbol{C}^\top)/2$. One problem is that the off-diagonal elements of $\boldsymbol{A}$ are dense (leading to a connected graph), which can result in low clustering accuracy. Therefore, we propose to truncate $\boldsymbol{C}$ by keeping only the largest τ elements of each column of $\boldsymbol{C}$. Nevertheless, it is not easy to determine τ beforehand. When τ is too small, the corresponding graph will have $k+1$ or more connected components. When τ is too large, the corresponding graph will have $k-1$ or less connected components. However, τ can be automatically determined by our AutoSC-GD and AutoSC-BO.

In the case that the data have some low-dimensional nonlinear structures, the similarity between pair-wise columns of $\boldsymbol{X}$ cannot be well recognized by the linear regression (13). Therefore, we also consider the following nonlinear regression model

$$\underset{\boldsymbol{C}}{\text{minimize}} \quad \tfrac{1}{2}\|\phi(\boldsymbol{X})-\phi(\boldsymbol{X})\boldsymbol{C}\|_F^2+\tfrac{\lambda}{2}\|\boldsymbol{C}\|_F^2, \tag{14}$$

where ϕ denotes a nonlinear feature map performed on each column of the matrix, i.e. $\phi(\boldsymbol{X})=[\phi(\boldsymbol{x}_1),\ldots,\phi(\boldsymbol{x}_n)]$. In (14), letting ϕ be some feature map induced by a kernel function $k(\cdot,\cdot)$ (e.g. polynomial kernel $k(\boldsymbol{x}_i,\boldsymbol{x}_j)=(\boldsymbol{x}_i^\top\boldsymbol{x}_j+b)^q$ and Gaussian kernel), we get the kernel LSR (KLSR):

$$\underset{\boldsymbol{C}}{\text{minimize}} \quad \tfrac{1}{2}\text{Tr}\left(\boldsymbol{K}-2\boldsymbol{K}\boldsymbol{C}+\boldsymbol{C}^\top\boldsymbol{K}\boldsymbol{C}\right)+\tfrac{\lambda}{2}\|\boldsymbol{C}\|_F^2, \tag{15}$$

where $\boldsymbol{K}=\phi(\boldsymbol{X})^\top\phi(\boldsymbol{X})$ and $[\boldsymbol{K}]_{ij}=k(\boldsymbol{x}_i,\boldsymbol{x}_j)$. The closed-form solution is $\boldsymbol{C}=(\boldsymbol{K}+\lambda\boldsymbol{I})^{-1}\boldsymbol{K}$. The post-processing is the same as that for the solution of LSR.

We introduce the following property, a necessary condition of successful subspace clustering, which is similar to the one used in [Wang and Xu, 2013; Soltanolkotabi *et al.*, 2014].

Definition 3.5 (Subspace Detection Property)**.** A symmetric affinity matrix $\boldsymbol{A}$ obtained from $\boldsymbol{X}$ has subspace detection property if for all i, the nonzero elements of $\boldsymbol{a}_i$ correspond to the columns of $\boldsymbol{X}$ in the same subspace as $\boldsymbol{x}_i$.

For convenience, let $\pi(i)$ be the index of the subspace $\boldsymbol{x}_i$ belongs to and C_j be the index set of the columns of $\boldsymbol{X}$ in subspace j. We consider the following deterministic model.

Definition 3.6 (Deterministic Model)**.** The columns of $\boldsymbol{X}\in\mathbb{R}^{m\times n}$ are drawn from a union of k different subspaces and are further corrupted by noise, where $\dim(\mathcal{S}_1\cup\cdots\cup\mathcal{S}_k)=d<m\leq n$. Let $\boldsymbol{X}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the SVD of $\boldsymbol{X}$, where $\boldsymbol{\Sigma}=\text{diag}(\sigma_1,\ldots,\sigma_n)$ and $\sigma_1\geq\sigma_2\geq\cdots\sigma_n$. Let $\gamma=\sigma_{d+1}/\sigma_d$. Denote $\boldsymbol{v}_i=(v_{i1},\ldots,v_{in})$ the i-th row of $\boldsymbol{V}$ and let $\bar{\boldsymbol{v}}_i=(v_{i1},\ldots,v_{id})$. Suppose the following conditions hold[4]: 1) for every $i\in[n]$, the $\bar{\tau}$-th largest element of $\{|\bar{\boldsymbol{v}}_i^\top\bar{\boldsymbol{v}}_j|:j\in C_{\pi(i)}\}$ is greater than α; 2) $\max_{i\in[n]}\max_{j\in[n]\setminus C_{\pi(i)}}|\bar{\boldsymbol{v}}_i^\top\bar{\boldsymbol{v}}_j|\leq\beta$; 3) $\max_{i,j,l}|v_{il}v_{jl}|\leq\mu$.

Then the following theorem verifies the effectiveness of (13) followed by the truncation (thresholding) operation in subspace detection.

Theorem 3.7. *Suppose $\boldsymbol{X}$ is given by Definition 3.6 and $\boldsymbol{C}$ is given by* (13) *with*

$$\frac{\left(\rho-\sqrt{\rho^2-4(2\mu d-\Delta)(2\mu m-2\mu d-\Delta)}\right)\sigma_d^2}{4\mu d-2\Delta}<\lambda<\frac{\left(\rho+\sqrt{\rho^2-4(2\mu d-\Delta)(2\mu m-2\mu d-\Delta)}\right)\sigma_d^2}{4\mu d-2\Delta} \tag{16}$$

where $\rho=2\mu m\gamma^2-\Delta(1+\gamma^2)$ and $\Delta=\alpha-\beta$. Then the $\boldsymbol{C}$ truncated by $\tau\leq\bar{\tau}$ has the subspace detection property.

[4] γ measures the noise level, β is dominated by the difference between subspaces, and μ quantifies the incoherence in the singular vectors.

In Theorem 3.7, the width of the range of λ is $w = \frac{\sqrt{\rho^2 - 4(2\mu d - \Delta)(2\mu m - 2\mu d - \Delta)}\sigma_d^2}{2\mu d - \Delta}$. We see that a larger σ_d, Δ, or smaller γ, d leads to a wider range of λ, which corresponds to a simper clustering problem. When $\rho^2 \leq 4(2\mu d - \Delta)(2\mu m - 2\mu d - \Delta)$, λ does not exist. Theorem 3.7 can be extended to the kernel case (15) without the restriction of $d < m$ even when the columns of $\boldsymbol{X}$ are drawn from a union of nonlinear low-dimensional manifolds. See Definition C.1, Definition C.2, and Theorem C.3 in Appendix C. Based on Theorem 3.7 and Theorem C.3, the follow proposition indicates that AutoSC can cluster the data correctly.

Proposition 3.8. *Suppose the affinity matrix $\boldsymbol{A}$ given by AutoSC has the subspace or manifold detection property (defined in Appendix C) and* $\text{reg}(\boldsymbol{L}) = \frac{\sigma_{k+1}}{\epsilon} > 0$. *Then each component of G consists of all columns of $\boldsymbol{X}$ in the same subspace or manifold.*

Now we see that LSR and KLSR with thresholding can provide effective self-expressive affinity matrices for AutoSC without performing iterative optimization. On the other hand, the relative-eigen-gap is able to compare LSR with KLSR, compare different kernels, and evaluate λ, τ, and kernel parameters. Note that if we use SSC and KSSC instead of LSR and KLSR, AutoSC will be very time-consuming. If we use LSR and KLSR without thresholding, AutoSC may not provide high clustering accuracy. We hope that AutoSC is not only automatic but also accurate and efficient.

3.5 AutoSC+NSE for Large-Scale Data

Since the time and space complexity of AutoSC are quadratic with n, it cannot be directly applied to large-scale datasets. To solve the problem, we propose to perform Algorithm 1 on a set of s landmarks of the data (denoted by $\hat{\boldsymbol{X}}$) to get a $\hat{\boldsymbol{Z}}$. The landmarks can be generated by k-means or randomly. Then we regard $\hat{\boldsymbol{Z}}$ as a feature matrix and learn a map $g : \mathbb{R}^m \to \mathbb{R}^k$ from $\hat{\boldsymbol{X}}$ to $\hat{\boldsymbol{Z}}$. According to the universal approximation theorem [Sonoda and Murata, 2017] of neural networks, we approximate g by a two-layer neural network and solve

$$\underset{\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2}{\text{minimize}} \ \frac{1}{2s} \|\hat{\boldsymbol{Z}} - \boldsymbol{W}_2 \text{ReLU}(\boldsymbol{W}_1 \hat{\boldsymbol{X}} + \boldsymbol{b}_1 \mathbf{1}_s^\top) - \boldsymbol{b}_2 \mathbf{1}_s^\top\|_F^2 + \frac{\gamma}{2} \left(\|\boldsymbol{W}_1\|_F^2 + \|\boldsymbol{W}_2\|_F^2 \right), \quad (17)$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{d \times m}$, $\boldsymbol{W}_2 \in \mathbb{R}^{k \times d}$, $\boldsymbol{b}_1 \in \mathbb{R}^d$, and $\boldsymbol{b}_2 \in \mathbb{R}^k$. Since $\boldsymbol{A}$ is sparse, k is often less than m, and a neural network is used, we call (17) Neural Sparse Embedding (NSE). We use mini-batch Adam [Kingma and Ba, 2014] to solve NSE. It is worth noting that NSE is different the method proposed by [Li *et al.*, 2020]. In [Li *et al.*, 2020], the regression is for an affinity matrix, which leads to high computational cost. The network learned from (17) is applied to $\boldsymbol{X}$ to extract a k-dimensional feature matrix $\boldsymbol{Z}$:

$$\boldsymbol{Z} = \hat{g}(\boldsymbol{X}) = \boldsymbol{W}_2 \text{ReLU}(\boldsymbol{W}_1 \boldsymbol{X} + \boldsymbol{b}_1 \mathbf{1}_n^\top) + \boldsymbol{b}_2 \mathbf{1}_n^\top. \quad (18)$$

Finally, we perform k-means on $\boldsymbol{Z}$ to get the clusters. The procedures are summarized into Algorithm 2 (see Appendix B). Note that the time complexity of AutoSC+NSE is $O(dmn + \tilde{d}s^2)$, where $\tilde{d}$ depends on the specific AMC method. When $s \ll n$, the time complexity of AutoSC+NSE is linear with the number of data points n. Proposition C.4 in Appendix C.2 shows that a small number of hidden nodes in NSE are sufficient to make the clustering succeed.

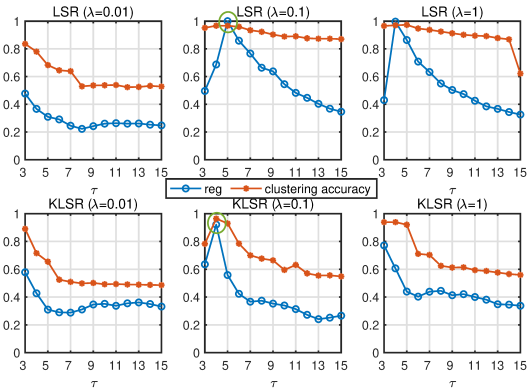
4 Experiments

We test our AutoSC on Extended Yale B Face [Kuang-Chih *et al.*, 2005], ORL Face [Samaria and Harter, 1994], COIL20 [Nene *et al.*, 1996], AR Face [Martínez and Kak, 2001], MNIST [LeCun *et al.*, 1998], Fashion-MNIST [Xiao *et al.*, 2017], GTSRB [Stallkamp *et al.*, 2012], subsets and extracted features of MNIST and Fashion-MNIST. The descriptions for the datasets are in Appendix D.1. Our MATLAB codes are available at `https://github.com/jicongfan/Automated-Spectral-Clustering`.

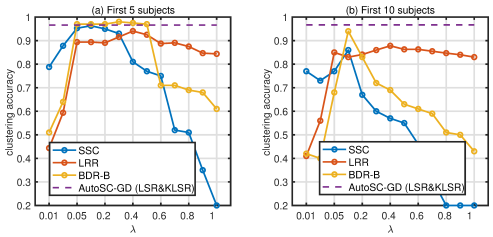
4.1 Intuitive validation of AutoSC

Performance of Relative-Eigen-Gap First, we use LSR and KLSR to show the effectiveness of the proposed $\text{reg}(\boldsymbol{L})$. Figure 1(i) shows an intuitive example of the performance of LSR and KLSR

in clustering a subset of the Extended Yale B database, where for (15) we use Gaussian kernel with $\varsigma = \frac{1}{n^2}\sum_{ij}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$. We see that: 1) in LSR and KLSR, for a fixed λ (or τ), the τ (or λ) with larger reg$(\boldsymbol{L})$ provides higher clustering accuracy; 2) for a fixed λ and a fixed τ, if LSR has a larger reg$(\boldsymbol{L})$, its clustering accuracy is higher than that of KLSR, and vice versa. We conclude that roughly a larger reg$(\boldsymbol{L})$ indeed leads to a higher clustering accuracy, which is consistent with our theoretical analysis in Section 3.2.



(i) Clustering accuracy and reg$(\boldsymbol{L})$ (rescaled between 0 and 1) and clustering accuracy of LSR and KLSR on the first 5 subjects of the Extended Yale Face B database.



(ii) Clustering accuracies of SSC, LRR, and BDR-B with different hyperparameter λ in comparison to AutoSC-GD with LSR and KLSR on the Extended Yale Face B database. The value of λ used in BDR-B has been divided by 10. The γ in BDR-B is chosen from $\{0.01, 0.1, 1\}$ and the best one is used for each λ. In Case (b), the time costs of SSC, LRR, BDR-B, and AutoSC-GD are 9.5s, 33.0s, 7.6s, and 1.6s respectively.

Figure 1: Examples about reg$(\boldsymbol{L})$, clustering accuracy, and hyperparameters on Extended Yale B.

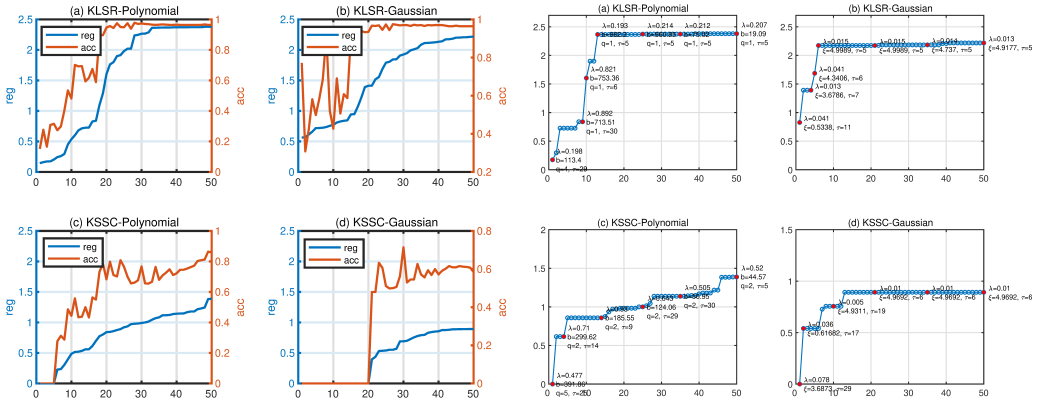
Now we show the superiority of LSR and KLSR compared to a few important AMC methods. Figure 1(ii) shows the clustering accuracy of SSC [Elhamifar and Vidal, 2013], LRR [Liu *et al.*, 2013], and BDR-B [Lu *et al.*, 2018] with different hyperparameters and our method Auto-GD with LSR and KLSR (detailed by Algorithm 1 in the supplementary material) on the Extended Yale Face B subset. SSC and BDR-B are sensitive to the value of λ, especially for the relatively difficult task, say Figure 1(ii-b). LRR is not sensitive to the value of λ but its accuracy is low. LSR and KLSR are more accurate and efficient than other methods.

The performance of AutoSC-BO We show the performance of AutoSC-BO with many AMC methods such as SSC [Elhamifar and Vidal, 2013] and LSR. Taking the KLSR model (15) with polynomial kernel as an example, the parameters are $\theta = (\lambda, b, q, \tau)^\top$ and the constraints are given by $S = \{\lambda \in \mathbb{R} : \lambda_{min} \leq \lambda \leq \lambda_{max};\ b \in \mathbb{R} : b_{min} \leq b \leq b_{max};\ q \in \mathbb{Z}^+ : q_{min} \leq q \leq q_{max};\ \tau \in \mathbb{Z}^+ : \tau_{min} \leq \tau \leq \tau_{max}\}$. More details are in Appendix D.5. Shown in Table 2, larger reg corresponds to higher clustering accuracy and KLSR with polynomial kernel (the optimal q is 1) performs best. Figure 2 shows the performance of KSSC [Patel and Vidal, 2014] and KLSR in each iteration of AutoSC-BO.

Table 2: Clustering accuracy of AutoSC-BO with many methods on Yale Face B dataset (first 10 subjects). All hyperparameters of the kernel functions were optimized via Bayesian optimization.

AMC	ϵ-neigh -borhood	Polynomial kernel	Gaussian kernel	KSSC (Gauss)	KSSC (Poly)	KLSR (Gauss)	KLSR (Poly)
reg_{max}	0.776	1.294	1.307	0.892	1.388	2.217	2.379
Accuracy	0.325	0.389	0.393	0.584	0.859	0.963	0.966

Relative-Eigen-Gap versus Eigen-Gap We compare the proposed relative-eigen-gap (reg$(\mathbf{L})$) with eigen-gap (denoted by eg$(\mathbf{L})$), and the regularizer J$(\mathbf{L}, \alpha)$ proposed by [Meila and Shortreed, 2006] with different α. Note that in AutoSC, we need to minimize J$(\mathbf{L}, \alpha)$ instead. The clustering results of AutoSC-GD are reported in Table 3 (details about the datasets are in Section 4). We see that our reg$(\mathbf{L})$ outperforms eg$(\mathbf{L})$ and J$(\mathbf{L}, \alpha)$ in all cases except AR.



(i) reg($\boldsymbol{L}$) and clustering accuracy of KLSR and KSSC in each iteration of BO.

(ii) reg($\boldsymbol{L}$) and hyperparameters of KLSR and KSSC in each iteration of BO.

Figure 2: AutoSC-BO with KSSC and KLSR on the first 10 subjects of YaleB Face dataset.

Table 3: The comparison of AutoSC-GD with reg($\mathbf{L}$), eg($\mathbf{L}$), and J($\mathbf{L}, \alpha$)

	YaleB	ORL	COIL20	AR	MNIST	F-MNIST
eg($\mathbf{L}$)	0.790	0.768	0.619	0.805	0.663	0.562
J($\mathbf{L}, 0$)	0.823	**0.795**	0.750	0.804	0.726	0.516
J($\mathbf{L}, -0.1$)	0.818	0.788	0.768	0.826	0.718	0.519
J($\mathbf{L}, -1$)	0.812	0.785	0.769	0.817	0.722	0.523
J($\mathbf{L}, -10$)	0.804	0.788	0.768	**0.832**	0.731	0.539
J($\mathbf{L}, -100$)	0.788	0.765	0.769	0.817	0.735	0.545
reg($\mathbf{L}$)	**0.897**	**0.795**	**0.782**	0.786	**0.755**	**0.595**

4.2 Comparative studies of AutoSC and baselines

First we compare AutoSC with SSC, LRR [Liu *et al.*, 2013], LSR [Lu *et al.*, 2012], EDSC [Pan Ji *et al.*, 2014], KSSC, SSC-OMP [You *et al.*, 2016b], BDR-Z [Lu *et al.*, 2018], and BDR-B [Lu *et al.*, 2018] on six smaller datasets. The clustering accuracy and time cost are reported in Table 4. AutoSC-GD and AutoSC-BO outperformed other methods significantly in almost all cases. SSC-OMP and AutoSC-GD are more efficient than SSC, LRR, EDSC, and KSSC. The time cost of AutoSC-BO is much higher than that of AutoSC-GD because the former optimizes all hyperparameters including λ, τ, and kernel parameters (e.g. b, q).

Table 4: Clustering performance on the six small datasets. For the MNIST-1k and Fahsion-MNIST-1k, we report the average results of 20 trials because the subset is formed randomly. AutoSC chose LSR for Yale B and AR and chose KLSR for others datasets. The NMI results are in Table 7 (See Appendix D.3).

		SSC	LRR	LSR	EDSC	KSSC	SSC-OMP	BDR-Z	BDR-B	AutoSC-GD	AutoSC-BO
Yale B	acc	0.723	0.643	0.592	0.806	0.649	0.768	0.596	0.719	**0.897**	**0.909**
	time	273.8	928.1	7.3	58.6	464.3	**8.9**	368.8	368.8	19.1	78.3
ORL	acc	0.711	0.762	0.680	0.712	0.707	0.665	0.739	0.735	**0.795**	**0.803**
	time	2.7	8.8	**0.5**	2.0	2.6	**0.4**	3.9	3.9	2.3	20.5
COIL20	acc	0.871	0.729	0.695	0.759	**0.912**	0.658	0.713	0.791	0.782	**0.878**
	time	61.8	221.2	**1.4**	15.4	100.6	**2.5**	86.8	86.8	7.6	39.2
AR	acc	0.718	0.769	0.665	0.673	0.726	0.669	0.745	0.751	**0.786**	**0.826**
	time	317.5	1220.6	**14.5**	69.1	627.4	57.6	578.7	578.7	**43.4**	130.6
MNIST -1k	acc	0.596 (0.054)	0.513 (0.037)	0.554 (0.041)	0.536 (0.035)	0.577 (0.053)	0.542 (0.038)	0.576 (0.037)	0.578 (0.043)	**0.615** (0.041)	**0.619** (0.038)
	time	24.9	69.1	**0.8**	5.2	32.8	**1.3**	26.9	26.9	2.5	21.7
Fashion-MNIST-1k	acc	0.553 (0.025)	0.515 (0.014)	0.563 (0.023)	0.544 (0.017)	0.548 (0.016)	0.566 (0.034)	0.574 (0.019)	0.563 (0.031)	**0.581** (0.025)	**0.584** (0.021)
	time	24.1	68.5	**0.7**	5.1	35.9	**1.2**	25.7	25.7	2.6	22.7

We compare our method with LSR, LSC-K [Chen and Cai, 2011], SSC-OMP [You *et al.*, 2016b], and S^5C [Matsushima and Brbic, 2019], and S^3COMP-C [Chen *et al.*, 2020] on the larger datasets. The parameter settings are in Appendix D.6. Table 5 shows the clustering accuracy and standard deviation of 10 repeated trials on the raw-pixel data of MNIST and Fashion-MNIST. Table 6 shows the results on MNIST and Fashion-MNIST features and GTSRB. Our methods have the highest clustering accuracy in every case. Note that if NSE is ablated, the accuracy of AutoSC-GD on MNIST(features)-10k, -20k, and -30k are 0.9772, 0.9783 and 0.9864 respectively, higher than those of AutoSC-GD+NSE, though the time costs increased.

It is worth mentioning that, to the best of our knowledge, the deep clustering method of [Zhang *et al.*, 2019a] has SOTA performance on Yale B (acc=0.98) and ORL (acc=0.89), the method proposed by [Zhang *et al.*, 2019b] has SOTA performance on Fashion-MNIST (acc=0.72), the method proposed by [Mahon and Lukasiewicz, 2021] has SOTA performance on MNIST (acc=0.99), and our method has SOTA performance on GTSRB. Nevertheless, we focus on automated spectral clustering.

Table 5: Clustering accuracy and time cost (second) on MNIST and Fashion MNIST. "—" means the computation is out of memory.

		LSR	LSC-K	SSC-OMP	S^5C	S^3COMP-C	AutoSC-GD+NSE	AutoSC-BO+NSE
MNIST-10k	acc	0.583(0.007)	0.652(0.037)	0.431(0.014)	0.646(0.045)	0.623(0.028)	**0.687**(0.035)	**0.679**(0.034)
	time	154.9	**18.9**	26.4	82.3	710.4/20	**16.2**	48.3
MNIST	acc	—	0.665(0.021)	0.453(0.017)	0.627(0.025)	—	**0.755**(0.022)	**0.750**(0.009)
	time	—	329.2	1178.3	961.5	—	**86.9**	**123.6**
Fashion-MNIST-10k	acc	0.561(0.008)	0.571(0.025)	0.509(0.038)	0.565(0.021)	0.569(0.024)	**0.576**(0.011)	**0.572**(0.019)
	time	153.6	**18.6**	26.8	107.3	707.2/20	**17.3**	50.9
Fashion-MNIST	acc	—	0.561(0.015)	0.359(0.017)	0.559(0.013)	—	**0.586**(0.008)	**0.578**(0.012)
	time	—	335.1	1156.6	932.6	—	**88.7**	**122.8**

Table 6: Clustering accuracy (mean value and standard deviation) and time cost (second) on MNIST and Fashion-MNIST with feature extraction. "/" means the algorithm was performed on a computational platform not comparable to ours. The underlined values are from [Chen *et al.*, 2020].

		LSC-K	SSC-OMP	S^5C	S^3COMP-C	AutoSC-GD+NSE	AutoSC-BO+NSE
MNIST	acc	0.8659(0.0215)	<u>0.8159</u>	0.7829(0.0283)	<u>0.9632</u>	**0.9775**(0.0034)	**0.9741**(0.0044)
	time	273.6	<u>280.6</u>	907.5	<u>416.8</u>	**59.2**	**115.3**
Fashion-MNIST	acc	0.6131(0.0298)	0.3796(0.0217)	0.6057(0.0227)	—	**0.6398**(0.0133)	**0.6461**(0.0104)
	time	251.8	1013.9	913.2	—	**61.9**	**112.6**
GTSRB	acc	0.8711(0.0510)	<u>0.8252</u>	0.9044(0.0267)	<u>0.9554</u>	**0.9873**(0.0126)	**0.9881**(0.0078)
	time	**31.2**	/	98.7	/	**16.8**	69.4

5 Conclusion

We have proposed an automated spectral clustering method. Extensive experiments showed the effectiveness and superiority of our methods over baseline methods. The efficiency improvement is from the closed-form solutions of the least squares regressions. The accuracy improvement is from the effectiveness of LSR and KLSR with thresholding and the automation of model and hyperparameter selection. One limitation of this work is that we only considered automated spectral clustering while there are many other clustering methods (e.g. [Fan, 2021]) not relying on affinity matrix.

Acknowledgments

The work of Jicong Fan was supported in part by the research funding T00120210002 of Shenzhen Research Institute of Big Data and the Youth program 62106211 of the National Natural Science Foundation of China. The work of Yiheng Tu was supported in part by the National Natural Science Foundation of China under Grant no. 32171078. The work of Zhao Zhang was supported in part by the National Natural Science Foundation of China under Grant no. 62072151 and Anhui Provincial Natural Science Fund for the Distinguished Young Scholars (2008085J30). The work of Mingbo Zhao was supported in part by the National Natural Science Foundation of China under Grant no. 61971121. The work of Haijun Zhang was supported in part by the National Natural Science Foundation of China under Grant no. 61972112 and no. 61832004, the Guangdong Basic and Applied Basic Research Foundation under Grant no. 2021B1515020088.

The authors appreciate the reviewers' comments and time.

References

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011.

Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

Deng Cai and Xinlei Chen. Large scale spectral clustering via landmark-based sparse representation. *IEEE transactions on cybernetics*, 45(8):1669–1680, 2014.

Jinyu Cai, Jicong Fan, Wenzhong Guo, Shiping Wang, Yunhe Zhang, and Zhao Zhang. Efficient deep embedded subspace clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–10, June 2022.

Xinlei Chen and Deng Cai. Large scale spectral clustering with landmark-based representation. In *Twenty-fifth AAAI conference on artificial intelligence*. Citeseer, 2011.

Ying Chen, Chun-Guang Li, and Chong You. Stochastic sparse subspace clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4155–4164, 2020.

E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

Jicong Fan and Tommy W.S. Chow. Sparse subspace clustering for data with missing entries and high-rank matrix completion. *Neural Networks*, 93:36–44, 2017.

Jicong Fan, Zhaoyang Tian, Mingbo Zhao, and Tommy W.S. Chow. Accelerated low-rank representation for subspace clustering and semi-supervised classification on large-scale data. *Neural Networks*, 100:39–48, 2018.

Jicong Fan, Yuqian Zhang, and Madeleine Udell. Polynomial matrix completion for missing data imputation and transductive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3842–3849, 2020.

Jicong Fan. Large-scale subspace clustering via k-factorization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '21, page 342–352, New York, NY, USA, 2021. Association for Computing Machinery.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.

M. Halkidi and M. Vazirgiannis. Clustering validity assessment: finding the optimal partitioning of a data set. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 187–194, 2001.

N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

Harold V Henderson and Shayle R Searle. On deriving the inverse of a sum of matrices. *Siam Review*, 23(1):53–60, 1981.

Juhua Hu, Qi Qian, Jian Pei, Rong Jin, and Shenghuo Zhu. Finding multiple stable clusterings. *Knowledge and Information Systems*, 51(3):991–1021, 2017.

Hsin-Chien Huang, Yung-Yu Chuang, and Chu-Song Chen. Affinity aggregation for spectral clustering. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 773–780. IEEE, 2012.

Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

Pan Ji, Mathieu Salzmann, and Hongdong Li. Shape interaction matrix revisited and robustified: Efficient subspace clustering with corrupted and incomplete data. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In *Advances in Neural Information Processing Systems*, pages 24–33, 2017.

Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

Zhao Kang, Wangtao Zhou, Zhitong Zhao, Junming Shao, Meng Han, and Zenglin Xu. Large-scale multi-view subspace clustering in linear time. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4412–4419, 2020.

Zhao Kang, Zhiping Lin, Xiaofeng Zhu, and Wenbo Xu. Structured graph learning for scalable subspace clustering: From single view to multiview. *IEEE Transactions on Cybernetics*, 52(9):8976 – 8986, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial Intelligence and Statistics*, pages 528–536. PMLR, 2017.

Lee Kuang-Chih, J. Ho, and D. J. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):684–698, 2005.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Chun-Guang Li and Rene Vidal. Structured sparse subspace clustering: A unified optimization framework. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 277–286, 2015.

Chun-Guang Li and Rene Vidal. A structured sparse plus structured low-rank framework for subspace clustering and completion. *IEEE Transactions on Signal Processing*, 64(24):6557–6570, 2016.

Jun Li and Handong Zhao. Large-scale subspace clustering by fast regression coding. In *IJCAI*, 2017.

Jun Li, Hongfu Liu, Zhiqiang Tao, Handong Zhao, and Yun Fu. Learnable subspace clustering. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2020.

Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu. Understanding of internal clustering validation measures. In *2010 IEEE international conference on data mining*, pages 911–916. IEEE, 2010.

G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013.

Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. Robust and efficient subspace segmentation via least squares regression. In *European conference on computer vision*, pages 347–360. Springer, 2012.

Canyi Lu, Jiashi Feng, Zhouchen Lin, Tao Mei, and Shuicheng Yan. Subspace clustering by block diagonal representation. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):487–501, 2018.

Juncheng Lv, Zhao Kang, Xiao Lu, and Zenglin Xu. Pseudo-supervised deep subspace clustering. *IEEE Transactions on Image Processing*, 30:5252–5263, 2021.

Louis Mahon and Thomas Lukasiewicz. Selective pseudo-label clustering. In *German Conference on Artificial Intelligence (Künstliche Intelligenz)*, pages 158–178. Springer, 2021.

Aleix M Martínez and Avinash C Kak. PCA versus LDA. *IEEE transactions on pattern analysis and machine intelligence*, 23(2):228–233, 2001.

Bertil Matérn. *Spatial variation*, volume 36. Springer Science & Business Media, 2013.

Shin Matsushima and Maria Brbic. Selective sampling-based scalable sparse subspace clustering. In *Advances in Neural Information Processing Systems*, pages 12416–12425, 2019.

Marian Meila and Susan Shortreed. Regularized spectral learning. *Journal of machine learning research*, 1(1):1–20, 2006.

Marian Meila, Susan Shortreed, and Liang Xu. Regularized spectral learning. In *AISTATS*. PMLR, 2005.

Marina Meila. The multicut lemma. *UW Statistics Technical Report*, 417, 2001.

S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-20). Report, Columbia University, 1996.

Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002.

Erlin Pan and Zhao Kang. Multi-view contrastive graph clustering. *Advances in Neural Information Processing Systems*, 34, 2021.

Pan Ji, M. Salzmann, and Hongdong Li. Efficient dense subspace clustering. In *IEEE Winter Conference on Applications of Computer Vision*, pages 461–468, 2014.

V. M. Patel and R. Vidal. Kernel sparse subspace clustering. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 2849–2853, Oct 2014.

V. M. Patel, Nguyen Hien Van, and R. Vidal. Latent space sparse and low-rank subspace clustering. *Selected Topics in Signal Processing, IEEE Journal of*, 9(4):691–701, 2015.

Xi Peng, Lei Zhang, and Zhang Yi. Scalable sparse subspace clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 430–437, 2013.

Xi Peng, Huajin Tang, Lei Zhang, Zhang Yi, and Shijie Xiao. A unified framework for representation-based subspace clustering of out-of-sample and large-scale data. *IEEE transactions on neural networks and learning systems*, 27(12):2499–2512, 2015.

Giannis Poulakis. Unsupervised automl: a study on automated machine learning in the context of clustering. Master's thesis, $\Pi\alpha\nu\varepsilon\pi\iota\sigma\tau\acute{\eta}\mu\iota o$ $\Pi\varepsilon\iota\rho\alpha\iota\acute{\omega}\varsigma$, 2020.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

F. S. Samaria and A. C. Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*, pages 138–142, Dec 1994.

Jie Shen and Ping Li. Learning structured low-rank representation via matrix factorization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 500–509, Cadiz, Spain, 09–11 May 2016. PMLR.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Mahdi Soltanolkotabi, Ehsan Elhamifar, and Emmanuel J Candes. Robust subspace clustering. *The Annals of Statistics*, 42(2):669–699, 2014.

Sho Sonoda and Noboru Murata. Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis*, 43(2):233 – 268, 2017.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.

Hugo Steinhaus et al. Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci*, 1(804):801, 1956.

Michael E Tipping and Christopher M Bishop. Mixtures of probabilistic principal component analyzers. *Neural computation*, 11(2):443–482, 1999.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Yu-Xiang Wang and Huan Xu. Noisy sparse subspace clustering. In *International Conference on Machine Learning*, pages 89–97. PMLR, 2013.

Shusen Wang, Bojun Tu, Congfu Xu, and Zhihua Zhang. Exact subspace clustering in linear time. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 2113–2120, 2014.

Y. Weiss. Segmentation using eigenvectors: a unifying view. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 975–982 vol.2, 1999.

Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Chong You, Chun-Guang Li, Daniel P Robinson, and René Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3928–3937, 2016.

Chong You, Daniel Robinson, and René Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3918–3927, 2016.

Chong You, Chi Li, Daniel P Robinson, and René Vidal. Scalable exemplar-based subspace clustering on class-imbalanced data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–83, 2018.

Junjian Zhang, Chun-Guang Li, Chong You, Xianbiao Qi, Honggang Zhang, Jun Guo, and Zhouchen Lin. Self-supervised convolutional subspace clustering network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5473–5482, 2019.

Tong Zhang, Pan Ji, Mehrtash Harandi, Wenbing Huang, and Hongdong Li. Neural collaborative subspace clustering. In *International Conference on Machine Learning*, pages 7384–7393. PMLR, 2019.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [N/A]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A More discussion about LSR and KLSR

Note that if $n \ll m$, using the *push-through identity* [Henderson and Searle, 1981], we reformulate $\boldsymbol{C} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{X}$ as $\boldsymbol{C} = \boldsymbol{X}^\top (\lambda \boldsymbol{I} + \boldsymbol{X}\boldsymbol{X}^\top)^{-1} \boldsymbol{X}$ to reduce the computational cost from $O(n^3)$ to $O(mn^2)$. In $\boldsymbol{C} = (\boldsymbol{K} + \lambda \boldsymbol{I})^{-1} \boldsymbol{K}$, when n is large (e.g.> 5000), we perform randomized SVD [Halko *et al.*, 2011] on $\boldsymbol{K}$: $\boldsymbol{K} \approx \boldsymbol{V}_r \boldsymbol{\Sigma}_r \boldsymbol{V}_r^\top$. Then $\boldsymbol{C} \approx \boldsymbol{V}_r \boldsymbol{\Sigma}_r^{1/2} (\lambda \boldsymbol{I} + \boldsymbol{\Sigma})^{-1} \boldsymbol{\Sigma}_r^{1/2} \boldsymbol{V}_r^\top$, where $r = 20k$ works well in practical applications. The time complexity of computing $\boldsymbol{C}$ is $O(r\tau n + rn^2)$. The computation of the smallest $k+1$ eigenvalues of $\boldsymbol{L}$ is equivalent to compute the largest $k+1$ eigenvalues and eigenvectors of $\boldsymbol{D}^{-1/2} \boldsymbol{A} \boldsymbol{D}^{-1/2}$, which is sparse. The time complexity is $O(k\tau n)$. We have the following result.

Proposition A.1. *Let $\hat{\boldsymbol{c}}$ be the optimal solution of* $\text{minimize}_{\boldsymbol{c}} \frac{1}{2}\|\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\boldsymbol{c}\|^2 + \frac{\lambda}{2}\|\boldsymbol{c}\|^2$, *where ϕ is induced by Gaussian kernel and $\boldsymbol{y}$ is arbitrary. Then* $\|\hat{c}_i - \hat{c}_j\| \leq \sqrt{2 - 2\exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/(2\varsigma^2)\right)}$.

It shows that when two data points in $\boldsymbol{X}$, e.g. $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, are close to each other, the corresponding two elements in $\hat{\boldsymbol{c}}$, e.g. $\hat{c}_i$ and $\hat{c}_j$, have small difference. Hence KLSR with Gaussian kernel utilizes local information to enhance $\boldsymbol{C}$.

In LSR and KLSR, let $\lambda \in \Lambda$, $\tau \in \mathcal{T}$, and $\Theta = \Lambda \times \mathcal{T}$. The algorithm of AutoSC-GD with only LSR and KLSR is shown in Algorithm 1. The total time complexity is

$$O\left(|\Lambda|(mn^2 + r\bar{\tau}n + rn^2) + 2|\Lambda||\mathcal{T}|k\bar{\tau}n\right),$$

where $\bar{\tau}$ denotes the mean value in $\mathcal{T}$. The time complexity is at most $O(|\Lambda|\left(mn^2 + |\mathcal{T}|kmn\right))$ when $\tau \leq r \leq m \leq n$. It is worth noting that Algorithm 1 can be easily implemented parallelly, which will reduce the time complexity to $O(\max(m, r)n^2 + kmn)$. On the contrary, SSC, LRR, and their variants require iterative optimization and hence their time complexity is about $O(tmn^2)$, where t denotes the iteration number and is often larger than 100.

Algorithm 1 AutoSC-GD with Only LSR and KLSR

Input: $\boldsymbol{X}$, k, $\mathcal{F}$, Λ, $\mathcal{T}$
Normalize the columns of $\boldsymbol{X}$ to have unit ℓ_2 norm.
for f_u in $\mathcal{F}$ **do**
  for λ_i in Λ **do**
    Construct $\boldsymbol{C}$ by (13) or (15).
    for τ_j in $\mathcal{T}$ **do**
      $\boldsymbol{C} \leftarrow |\boldsymbol{C} \odot (\boldsymbol{1} - \boldsymbol{I})|$.
      Truncate $\boldsymbol{C}$ with parameter τ_j.
      For $j = 1, \ldots, n$, let $\boldsymbol{c}_j \leftarrow \boldsymbol{c}_j / |\boldsymbol{c}_j|_1$.
      $\boldsymbol{A} = (\boldsymbol{C} + \boldsymbol{C}^\top)/2$.
      $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2} \boldsymbol{A} \boldsymbol{D}^{-1/2}$.
      Compute $\sigma_1, \ldots, \sigma_{k+1}$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$.
      $\Delta_{uij} = \text{REG}(\boldsymbol{L})$, $\mathcal{V}_{uij} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]$.
    end for
  end for
end for
$\boldsymbol{Z} = \mathcal{V}_{\bar{u}\bar{i}\bar{j}}^\top$, where $\{\bar{u}, \bar{i}, \bar{j}\} = \text{argmax}_{u,i,j} \Delta_{uij}$.
Normalize the columns of $\boldsymbol{Z}$ to have unit ℓ_2 norm.
Perform k-means on $\boldsymbol{Z}$.
Output: k clusters: $C_1, \ldots, C_k$.

B The algorithm of AutoSC+NSE

See Algorithm 2.

C More theoretical results

Algorithm 2 AutoSC+NSE

Input: $\boldsymbol{X}$, k, $\mathcal{F}$, Θ, $\hat{n}$.

1: Select $\hat{n}$ landmarks from $\boldsymbol{X}$ by k-means to form $\hat{\boldsymbol{X}}$.
2: Apply AutoSC-GD or AutoSC-BO to $\hat{\boldsymbol{X}}$ with $\mathcal{F}$ and Θ.
2: Get $\hat{\boldsymbol{Z}}$ from the best Laplacian matrix given by AutoSC-G or AutoSC-BO.
3: Use mini-batch Adam to solve (17).
4: Compute $\boldsymbol{Z}$ by (18).
5: Perform k-means on $\boldsymbol{Z}$.

Output: k clusters: $C_1, \ldots, C_k$.

C.1 Theoretical guarantee for KLSR

Definition C.1 (Polynomial Deterministic Model)**.** The columns of $\boldsymbol{X}_0 \in \mathbb{R}^{m\times n}$ are drawn from a union of k different polynomials $\{g_j : \mathbb{R}^r \to \mathbb{R}^m,\ r < m\}_{j=1}^k$ of order at most p and are further corrupted by noise, say $\boldsymbol{X} = \boldsymbol{X}_0 + \boldsymbol{E}$. Denote the eigenvalue decomposition of the kernel matrix $\boldsymbol{K}$ of $\boldsymbol{X}$ as $\boldsymbol{K} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_n$. Let $\gamma = \sigma_{d+1}/\sigma_d$. Denote $\boldsymbol{v}_i = (v_{i1}, \ldots, v_{in})$ the i-th row of $\boldsymbol{V}$ and let $\bar{\boldsymbol{v}}_i = (v_{i1}, \ldots, v_{id})$, where $d < n$. Suppose the following conditions hold: 1) for every $i \in [n]$, the $\bar{\tau}$-th largest element of $\{|\bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j| : j \in C_{\pi(i)}\}$ is greater than α; 2) $\max_{i\in[n]} \max_{j\in[n]\setminus C_{\pi(i)}} |\bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j| \leq \beta$; 3) $\max_{i,j,l} |v_{il}v_{jl}| \leq \mu$.

Here we consider polynomials because they are easy to analyze and can well approximate smooth functions provided that p is sufficiently large. Clustering the columns of $\boldsymbol{X}$ given by Definition C.1 according to the polynomials is actually a manifold clustering problem beyond the setting of subspace clustering. Similar to the subspace detection property, we define

Definition C.2 (Manifold Detection Property)**.** A symmetric affinity matrix $\boldsymbol{A}$ obtained from $\boldsymbol{X}$ has manifold detection property if for all i, the nonzero elements of $\boldsymbol{a}_i$ correspond to the columns of $\boldsymbol{X}$ lying on the same manifold as $\boldsymbol{x}_i$.

The following theorem verifies the effectiveness of (15) followed by the truncation operation in manifold detection.

Theorem C.3. *Suppose $\boldsymbol{X}$ and $\boldsymbol{K}$ are given by Definition C.1 and $\boldsymbol{C}$ is given by* (15)*, where the kernel function is a polynomial kernel of order q,* $\mathrm{rank}(\boldsymbol{K}_0) = d$ *($\boldsymbol{K}_0$ is from $\boldsymbol{X}_0$), and*

$$\frac{\left(\rho - \sqrt{\rho^2 - 4(2\mu d - \Delta)(2\mu n - 2\mu d - \Delta)}\right)\sigma_d^2}{4\mu d - 2\Delta} < \lambda < \frac{\left(\rho + \sqrt{\rho^2 - 4(2\mu d - \Delta)(2\mu n - 2\mu d - \Delta)}\right)\sigma_d^2}{4\mu d - 2\Delta} \tag{19}$$

where $\rho = 2\mu n\gamma^2 - \Delta(1+\gamma^2)$. Then $d \leq k\binom{r+pq}{pq}$ and the $\boldsymbol{C}$ truncated by $\tau \leq \bar{\tau}$ has the manifold detection property.

In the theorem, σ_d can be much larger than σ_{d+1} provided that the noise is small enough. Then we get a wide range for λ. Compared to Theorem 3.7, Theorem C.3 allows a much larger d, which means the kernel method is able to handle more difficult clustering problems than the linear method.

C.2 Theoretical analysis for NSE

The following proposition shows that a small number of hidden nodes in NSE are sufficient to make the clustering succeed.

Proposition C.4. *Suppose the columns (with unit ℓ_2 norm) of $\boldsymbol{X}$ are drawn from a union of k independent subspaces of dimension r: $\sum_{j=1}^k \dim(\mathcal{S}_j) = \dim(\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_k) = kr$. For $j = 1, \ldots, k$, let $\boldsymbol{U}^j$ be the bases of $\mathcal{S}_j$ and $\boldsymbol{x}_i = \boldsymbol{U}^j\boldsymbol{v}_i$, if $\boldsymbol{x}_i \in \mathcal{S}_j$. Suppose $\max\{\|\boldsymbol{U}_{:l}^{i\top}\boldsymbol{U}^j\| : 1 \leq l \leq r, 1 \leq i \neq j \leq k\} \leq \mu$. Suppose that for all $i = 1, \ldots n$, $\max\{v_{1i}, \ldots, v_{ri}\} > \mu$. Then there exist $\boldsymbol{W}_1 \in \mathbb{R}^{d\times m}$, $\boldsymbol{W}_2 \in \mathbb{R}^{k\times d}$, $\boldsymbol{b}_1 \in \mathbb{R}^d$, and $\boldsymbol{b}_2 \in \mathbb{R}^k$ such that performing k-means on $\boldsymbol{Z}$ given by* (18) *identifies the clusters correctly, where $d = kr$.*

D More about the experiments

D.1 Dataset description

The description for the benchmark image datasets considered in this paper are as follows.

- **Extended Yale B Face** [Kuang-Chih *et al.*, 2005] (Yale B for short): face images (192×168) of 38 subjects. Each subject has about 64 images under various illumination conditions. We resize the images into 32×32.
- **ORL Face** [Samaria and Harter, 1994]: face images (112×92) of 40 subjects. Each subject has 10 images with different poses and facial expressions. We resize the images into 32×32.
- **COIL20** [Nene *et al.*, 1996]: images (32×32) of 20 objects. Each object has 72 images of different poses.
- **AR Face** [Martínez and Kak, 2001]: face images (165×120) of 50 males and 50 females. Each subject has 26 images with different facial expressions, illumination conditions, and occlusions. We resize the images into 42×30.
- **MNIST** [LeCun *et al.*, 1998]: 70,000 grey images (28×28) of handwritten digits $0 - 9$.
- **MNIST-1k(10k)**: a subset of MNIST containing 1000(10000) samples, 100(1000) randomly selected samples per class.
- **Fashion-MNIST** [Xiao *et al.*, 2017]: 70,000 gray images (28×28) of 10 types of fashion product.
- **Fashion-MNIST-1k(10k)**: a subset of Fashion-MNIST containing 1000(10000) samples, 100(1000) randomly selected samples per class.
- **MNIST-feature**: following the same procedures of [Chen *et al.*, 2020], we compute a feature vector of dimension 3,472 using the scattering convolution network [Bruna and Mallat, 2013] and then reduce the dimension to 500 using PCA.
- **Fashion-MNIST-feature**: similar to MNIST-feature.
- **GTSRB** [Stallkamp *et al.*, 2012]: consisting of 12,390 images of street signs in 14 categories. Following [Chen *et al.*, 2020], we extract a 1568-dimensional HOG feature, and reduce the dimension to 500 by PCA.

All experiments are conducted in MATLAB on a MacBook Pro with 2.3 GHz Intel i5 Core and 8GB RAM.

D.2 Hyperparameter settings for the small datasets

We select λ from $\{0.01, 0.02, 0.05, 0.1, 0.2, \ldots, 0.5\}$ for SSC, LRR, and KSSC. The λ in BDR is chosen from $\{5, 10, 20, \ldots, 80\}$. The γ in BDR-B and BDR-Z is chosen from $\{0.01, 0.1, 1\}$. The parameter s in SSC-OMP is chosen from $\{3, 4, \ldots, 15\}$. We report the results of these methods with their best hyperparameters. In AutoSC, we set $\Lambda = \{0.01, 0.1, 1\}$ and $\mathcal{T} = \{5, 6, \ldots, 15\}$. In AutoSC-BO, we consider two models: 1) Gaussian kernel similarity; 2) KLSR with polynomial kernel; 3) KLSR with Gaussian kernel, in which the hyperparameters of kernels are optimized adaptively. Then we needn't to consider LSR explicitly because it is a special case of KLSR with polynomial kernel. See Appendix D.5.

D.3 Clustering results in terms of NMI

In addition to the clustering accuracy reported in Table 4, here we also compare the normalized mutual information (NMI) in Table 7. We see that the comparative performance of all methods are similar to the results in Table 4 and our methods AutoSC-GD and AutoSC-BO outperformed other methods in almost all cases.

Table 7: Normalized Mutual Information on the six small datasets

	SSC	LRR	EDSC	KSSC	SSC-OMP	BDR-Z	BDR-B	AutoSC-GD	AutoSC-BO
Yale B	0.817	0.703	0.835	0.730	0.841	0.666	0.743	**0.919**	**0.928**
ORL	0.849	0.872	0.856	0.872	0.815	0.875	0.865	**0.907**	**0.903**
COIL20	0.954	0.706	0.843	**0.983**	0.671	0.843	0.873	0.897	**0.963**
AR	0.818	0.872	0.825	0.809	0.691	0.865	0.861	**0.887**	**0.904**
MNIST-1k	0.612	0.538	0.631	0.626	0.546	0.634	0.580	**0.667**	**0.652**
Fashion-MNIST-1k	0.616	0.601	0.621	0.621	0.559	0.614	0.605	**0.633**	**0.629**

D.4 The stability of AutoSC

Though we have used a relatively compact search space in AutoSC to reduce the highly unnecessary computational cost, the search space can be arbitrarily large. Figure 3 shows the clustering accuracy and the corresponding relative-eigen-gap. We can see that the region with highest relative-eigen-gap is in accordance with the region with highest clustering accuracy.

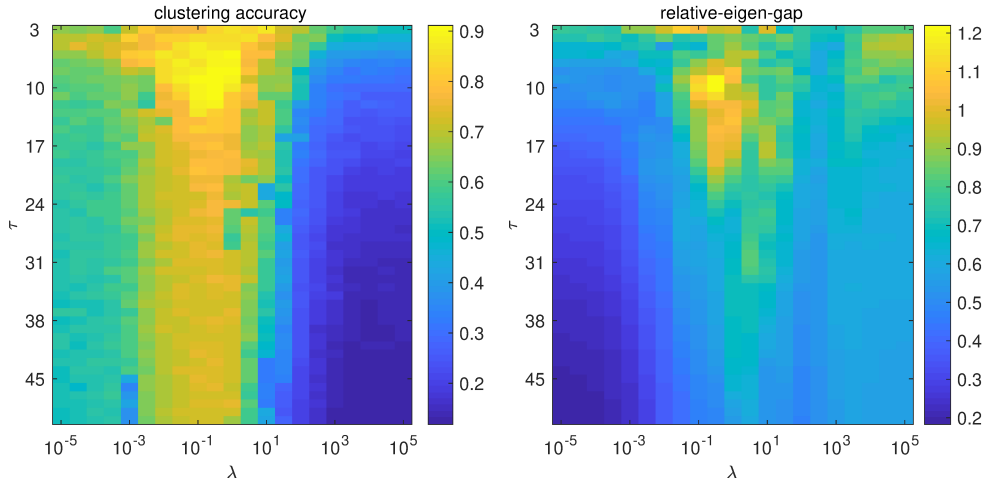


Figure 3: Visualization of the clustering accuracy and the corresponding relative-eigen-gap when a large search space is used.

D.5 More about AutoSC-BO in the experiments

For SSC, we consider the following problem

$$\underset{\boldsymbol{C}}{\text{minimize}} \ \tfrac{1}{2}\text{Tr}\left(\boldsymbol{K} - 2\boldsymbol{K}\boldsymbol{C} + \boldsymbol{C}^\top \boldsymbol{K}\boldsymbol{C}\right) + \lambda \|\boldsymbol{C}\|_1, \tag{20}$$

where $\boldsymbol{K}$ is an $n \times n$ kernel matrix with $[\boldsymbol{K}]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Note that when we use a linear kernel function, (20) reduces to the vanilla SSC. We solve the optimization via alternating direction method of multipliers (ADMM) [Boyd *et al.*, 2011], where the Lagrange parameter is 0.1 and the maximum number of iterations is 500. In this study, we consider polynomial kernel and Gaussian kernel, and optimize all hyperparameters including the order of the polynomial kernel. Particularly, for Gaussian kernel, we set $\varsigma = \frac{\xi}{n^2}\sum_{ij}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ and optimize ξ. The search space for the hyperparameters are as follows: $10^{-3} \leq \lambda \leq 1$, $5 \leq \tau \leq 50$, $0 \leq b \leq 10^3$, $1 \leq q \leq 5$, $0.5 \leq \xi \leq 5$.

In addition to Figure 2 of the main paper, here we report the best hyperparameters of the four models found by AutoSC-BO in Table 8. It can be found that the accuracy of KLSR with a linear kernel is higher than other models, which is consistent with its highest reg.

D.6 Hyperparameter settings of large-scale clustering

On MNIST-10k, MNIST, Fashion-MNIST-10k, and Fashion-MNIST, the parameter settings of [Chen and Cai, 2011], SSSC [Peng *et al.*, 2013], SSC-OMP [You *et al.*, 2016b], and S^{5}C [Mat-

Table 8: The best hyperparameters and the corresponding clustering accuracy given by AutoSC$_{BO}$ on the first 10 subjects of YaleB Face dataset.

method	hyperparameters	reg	accuracy
KLSR (Polynomial)	$\lambda = 0.207$, $b = 19.09$, $q = 1$, $\tau = 5$	2.379	0.966
KLSR (Gaussian)	$\lambda = 0.013$, $\xi = 4.92$, $\tau = 5$	2.217	0.963
KSSC (Polynomial)	$\lambda = 0.519$, $b = 44.57$, $q = 2$, $\tau = 5$	1.388	0.859
KSSC (Gaussian)	$\lambda = 0.0011$, $\xi = 4.97$, $\tau = 6$	0.892	0.584

sushima and Brbic, 2019], and S^{3}COMP-C [Chen *et al.*, 2020], and AutoSC+NSE are shown in Table 9. These hyper parameters have been determined via grid search and the best (as possible) values are used.

Table 9: Hyper-parameter settings of the compared methods on MNIST-10k, MNIST, Fashion-MNIST-10k, and Fashion-MNIST. s denotes the number of landmark data points. In the optimization (mini-batch Adam) of AutoSC+NSE, the epoch number, batch size, and step size are 200, 128, and 10^{-3} respectively.

LSC-K	$s = 1000$, $r = 3$
SSSC	$s = 1000$, $\lambda = 0.01$
SSC-OMP	$K = 10$ (sparsity)
S^{5}C	$s = 1000$, $\lambda = 0.1$ or 0.2
S^{3}COMP-C	$T = 20$, $\lambda = 0.4$, $\delta = 0.9$
AutoSC+NSE	$s = 1000$, $d = 200$, $\gamma = 10^{-5}$
AutoSC$_{BO}$+NSE	$s = 1000$, $d = 200$, $\gamma = 10^{-5}$

D.7 Influence of hyper-parameters in AutoSC+NSE

We investigate the effects of the type of activation function and the number (d) of nodes in the hidden layer of NSE. For convenience, we used a fixed random seed of MATLAB (rng(1)). Figure 4 shows the clustering accuracy on MNIST given by AutoSC+NSE with different activation function and different d. We see that ReLU outperformed tanh consistently. The reason is that the nonlinear mapping g from the data space to the eigenspace of the Laplacian matrix is nonsmooth and ReLU is more effective than tanh in approximating nonsmooth functions. In addition, when d increases, the clustering accuracy of AutoSC+NSE with ReLU often becomes higher because a wider network often has a higher ability of function approximation.

Figure 5 shows the clustering accuracy on MNIST given by AutoSC+NSE with different γ and α. When α is too small (say 10^{-4}, the clustering accuracy is low, because the training error is quite large in 200 epochs. In fact, by increasing the training epochs, the clustering accuracy can be improved, which however will increase the time cost. When α is relatively large, the clustering accuracy is often higher than 0.755. On the other hand, AutoSC+NSE is not sensitive to γ provided that it is not too large.

Figure 6 shows the mean value and standard deviation (10 repeated trials) of the clustering accuracy on MNIST given by AutoSC+NSE with different number (denoted by s) of landmark points. It can be found that when the s increases, the clustering accuracy increases and its standard deviation becomes smaller. When s is large enough, the improvement is not significant.

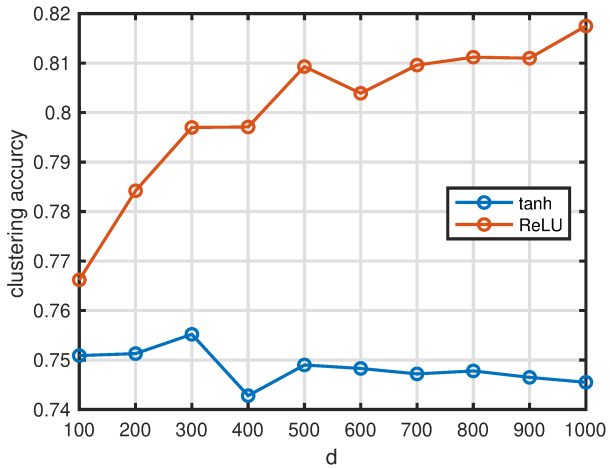


Figure 4: ReLU v.s. tanh (hyperbolic tangent) in the hidden layer of AutoSC+NSE on MNIST. When using ReLU, we set $\gamma = 10^{-5}$ and $\alpha = 10^{-3}$ (the step size in Adam). When using tanh, we set $\gamma = 10^{-3}$ and $\alpha = 10^{-2}$, which perform best in this case. Notice that the clustering accuracy when using ReLU is higher than 0.78 in almost all cases, which is higher than the value (say 0.755) we reported in the main paper. The reason is that in the main paper, we reported the mean value of 10 repeated trials but here we report the value of a single trial.

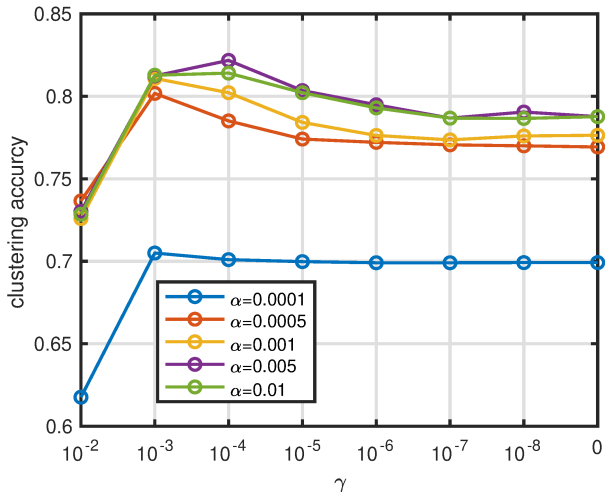


Figure 5: Influence of γ and α in AutoSC+NSE on MNIST. We set $d = 200$ and use ReLU.

E Proof for the theoretical results

E.1 Proof for Claim 3.2

Proof. The stochastic transition matrix of G is defined as

$$\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}. \tag{21}$$

In [Meila, 2001], it was showed that

$$\text{MNCut}(\mathcal{C}) \geq k - \sum_{i=1}^{k} \varrho_i(\boldsymbol{P}), \tag{22}$$

where $\varrho_i(\boldsymbol{P})$ denotes the i-th largest eigenvalue of $\boldsymbol{P}$ and $1 = \varrho_1(\boldsymbol{P}) \geq \varrho_2(\boldsymbol{P}) \geq \cdots \varrho_k(\boldsymbol{P})$. According to Lemma 3 of [Meila, 2001], we have

$$\sigma_i(\boldsymbol{L}) = 1 - \varrho_i(\boldsymbol{P}), \quad \forall i = 1, \ldots, n. \tag{23}$$

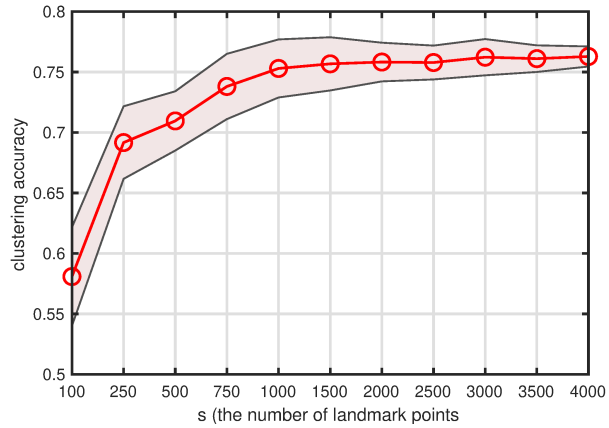


Figure 6: Influence of the number of landmark points in AutoSC+NSE on MNIST. We set $d = 200$, $\gamma = 10^{-5}$, and $\alpha = 10^{-3}$. The shadow denotes the standard deviation of 10 trials.

Substituting (23) into (22), we have

$$\text{MNCut}(\mathcal{C}) \geq \sum_{i=1}^{k} \sigma_i(\boldsymbol{L}). \tag{24}$$

□

Remark E.1. $\mathcal{C}$ can be any partition of the nodes of G. Let $\mathcal{C}^*$ be the optimal partition. Then $\text{MNCut}(\mathcal{C}^*) = \sum_{i=1}^{k} \sigma_i(\boldsymbol{L})$. If $\sum_{i=1}^{k} \sigma_i(\boldsymbol{L}) = 0$, there are no connections (edges) among $C_1^*, \ldots, C_k^*$.

E.2 Proof for Claim 3.3

Proof. For $i = 1, \ldots, k$, we aim to partition C_i into two subsets, denoted by C_i^1 and C_i^2. Then we define

$$\text{MNCut}(C_i) = \frac{Cut(C_i^1, C_i^2)}{Vol(C_i^1)} + \frac{Cut(C_i^2, C_i^1)}{Vol(C_i^2)}. \tag{25}$$

It follows that

$$\text{MNCut}(C_i) \geq \sum_{j=1}^{2} \sigma_j(\boldsymbol{L}_{C_i}) \geq \sigma_2(\boldsymbol{L}_{C_i}) = ac(C_i), \tag{26}$$

where $\boldsymbol{L}_{C_i}$ denotes the Laplacian matrix of C_i an $i = 1, \ldots, k$. Since $\sigma_{k+1}(\boldsymbol{L}) = \min\{ac(C_1), \ldots, ac(C_k)\}$, we have

$$\min_{1 \leq i \leq k} \text{MNCut}(C_i) \geq \sigma_{k+1}(\boldsymbol{L}). \tag{27}$$

Therefore, $\sigma_{k+1}(\boldsymbol{L})$ measures the least connectivity of $C_1, \ldots, C_k$. This finished the proof. □

Remark E.2. When $\sigma_{k+1}(\boldsymbol{L})$ is large, the connectivity in each of $C_1, \ldots, C_k$ is strong. Otherwise, the connectivity in each of $C_1, \ldots, C_k$ is weak. When $\sigma_{k+1}(\boldsymbol{L}) = 0$, at least one of $C_1, \ldots, C_k$ contains at least two components, which means the nodes of G can be partitioned into $k+1$ or more clusters.

E.3 Proof for Theorem 3.4

Proof. According to Theorem 1 of [Meila *et al.*, 2005], we have

$$\text{dist}(\mathcal{C}, \mathcal{C}') < \frac{3\delta}{\sigma_{k+1}(\boldsymbol{L}) - \sigma_k(\boldsymbol{L})}. \tag{28}$$

Since $\text{reg}(\boldsymbol{L}) = \dfrac{\sigma_{k+1}(\boldsymbol{L}) - \frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L})}{\frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L}) + \epsilon}$, we have

$$\sigma_{k+1}(\boldsymbol{L}) - \sigma_k(\boldsymbol{L}) = \text{reg}(\boldsymbol{L})(\bar{\sigma} + \epsilon) + \bar{\sigma} - \sigma_k(\boldsymbol{L}), \tag{29}$$

where $\bar{\sigma} = \frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L}) \geq \epsilon$. Invoking (29) into (28), we arrive at

$$\begin{aligned}
\text{dist}(\mathcal{C}, \mathcal{C}') &< \frac{3\delta}{\text{reg}(\boldsymbol{L})(\bar{\sigma} + \epsilon) + \bar{\sigma} - \sigma_k(\boldsymbol{L})} \\
&\leq \frac{3\delta}{2\epsilon\text{reg}(\boldsymbol{L}) + \bar{\sigma} - k\bar{\sigma}} \\
&\leq \frac{3\delta}{2\epsilon\text{reg}(\boldsymbol{L}) + (1-k)\eta\epsilon} \\
&\leq \frac{1.5\delta\epsilon^{-1}}{\text{reg}(\boldsymbol{L}) + (1-k)\eta/2}.
\end{aligned}$$

This finished the proof. □

E.4 Proof for Proposition A.1

Proof. Since $\hat{\boldsymbol{c}}$ is the optimal solution, we have

$$\begin{aligned}
\phi(\boldsymbol{x}_i)^\top \left(\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\right) + \lambda\hat{c}_i = 0, \\
\phi(\boldsymbol{x}_j)^\top \left(\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\right) + \lambda\hat{c}_j = 0.
\end{aligned}$$

It follows that

$$\begin{aligned}
\|\hat{c}_i - \hat{c}_j\| &= \| \left(\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)\right)^\top \left(\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\right) \| \\
&\leq \|\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)\| \|\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\| \\
&= \sqrt{k(\boldsymbol{x}_i, \boldsymbol{x}_i) - 2k(\boldsymbol{x}_i, \boldsymbol{x}_j) + k(\boldsymbol{x}_j, \boldsymbol{x}_j)} \\
&\quad \times \|\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\| \\
&= \sqrt{2 - 2k(\boldsymbol{x}_i, \boldsymbol{x}_j)} \|\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\| \\
&\leq \sqrt{2 - 2k(\boldsymbol{x}_i, \boldsymbol{x}_j)} \|\phi(\boldsymbol{y})\| \\
&= \sqrt{2 - 2\exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\varsigma^2}\right)}.
\end{aligned} \tag{30}$$

In the second and last equalities, we used the fact that $\|\phi(\boldsymbol{y})\| = \|\phi(\boldsymbol{x})\| = 1$. In the second inequality, we used the fact that $\frac{1}{2}\|\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\hat{\boldsymbol{c}}\|^2 + \frac{\lambda}{2}\|\hat{\boldsymbol{c}}\|^2 \leq \frac{1}{2}\|\phi(\boldsymbol{y}) - \phi(\boldsymbol{X})\boldsymbol{0}\|^2 + \frac{\lambda}{2}\|\boldsymbol{0}\|^2 = \frac{1}{2}\|\phi(\boldsymbol{y})\|^2$ because $\hat{\boldsymbol{c}}$ is the optimal solution. □

E.5 Proof for Theorem 3.7

Proof. Invoking the SVD of $\boldsymbol{X}$ into the closed-form solution of LSR, we get

$$\boldsymbol{C} = \boldsymbol{V}\text{diag}\left(\frac{\sigma_1^2}{\sigma_1^2 + \lambda}, \ldots, \frac{\sigma_n^2}{\sigma_n^2 + \lambda}\right)\boldsymbol{V}^\top. \tag{31}$$

It means

$$\begin{aligned}
c_{it} =& \sum_{l=1}^{n} \frac{v_{il}v_{jl}\sigma_l^2}{\sigma_l^2 + \lambda} \\
=& \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_t - \sum_{l=1}^{d} \frac{v_{il}v_{tl}\lambda}{\sigma_l^2 + \lambda} + \sum_{l=d+1}^{n} \frac{v_{il}v_{tl}\sigma_l^2}{\sigma_l^2 + \lambda}.
\end{aligned} \tag{32}$$

Suppose $j \in C_{\pi(i)}$ and $k \in [n] \setminus C_{\pi(i)}$. We have

$$
\begin{aligned}
&|c_{ij}| - |c_{ik}| \\
&= \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j - \sum_{l=1}^{d} \frac{v_{il} v_{jl} \lambda}{\sigma_l^2 + \lambda} + \sum_{l=d+1}^{n} \frac{v_{il} v_{jl} \sigma_l^2}{\sigma_l^2 + \lambda} \right| \\
&\quad - \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_k - \sum_{l=1}^{d} \frac{v_{il} v_{kl} \lambda}{\sigma_l^2 + \lambda} + \sum_{l=d+1}^{n} \frac{v_{il} v_{kl} \sigma_l^2}{\sigma_l^2 + \lambda} \right| \\
&\geq \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j \right| - \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_k \right| - \left| \sum_{l=1}^{d} \frac{v_{il} v_{jl} \lambda}{\sigma_l^2 + \lambda} \right| - \left| \sum_{l=1}^{d} \frac{v_{il} v_{kl} \lambda}{\sigma_l^2 + \lambda} \right| \\
&\quad - \left| \sum_{l=d+1}^{n} \frac{v_{il} v_{jl} \sigma_l^2}{\sigma_l^2 + \lambda} \right| - \left| \sum_{l=d+1}^{n} \frac{v_{il} v_{kl} \sigma_l^2}{\sigma_l^2 + \lambda} \right| \\
&\geq \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j \right| - \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_k \right| - 2\mu \sum_{l=1}^{d} \frac{\lambda}{\sigma_l^2 + \lambda} - 2\mu \sum_{l=d+1}^{n} \frac{\sigma_l^2}{\sigma_l^2 + \lambda} \\
&\geq \left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j \right| - \beta - \frac{2\mu d \lambda}{\sigma_d^2 + \lambda} - \frac{2\mu a \sigma_{d+1}^2}{\sigma_{d+1}^2 + \lambda},
\end{aligned}
\tag{33}
$$

where $a = \min(m, n) - d = m - d$.

To ensure that there exist at least $\bar{\tau}$ elements of $\{|c_{ij}| : j \in C_{\pi(i)}\}$ greater than $|c_{ik}|$ for all $k \in [n] \backslash C_{\pi(i)}$, we need

$$
\left| \bar{\boldsymbol{v}}_i^\top \bar{\boldsymbol{v}}_j \right| - \beta - \frac{2\mu d \lambda}{\sigma_d^2 + \lambda} - \frac{2\mu a \sigma_{d+1}^2}{\sigma_{d+1}^2 + \lambda} > 0 \tag{34}
$$

holds at least for $\bar{\tau}$ different j, where $j \in C_{\pi(i)}$. It is equivalent to ensure that

$$
\alpha - \beta - \frac{2\mu d \lambda}{\sigma_d^2 + \lambda} - \frac{2\mu a \sigma_{d+1}^2}{\sigma_{d+1}^2 + \lambda} > 0. \tag{35}
$$

We rewrite (35) as

$$
u_1 \lambda^2 + u_2 \lambda + u_3 > 0, \tag{36}
$$

where $u_1 = \alpha - \beta - 2\mu d$, $u_2 = (\alpha - \beta)(\sigma_d^2 + \sigma_{d+1}^2) - 2\mu(d + a)\sigma_{d+1}^2$, and $u_3 = (\alpha - \beta - 2\mu a)\sigma_d^2 \sigma_{d+1}^2$. The definition of μ, α, and β imply $u_1 < 0$. Then we solve (36) and obtain

$$
\begin{cases}
\lambda > \frac{2\mu m \sigma_{d+1}^2 - (\alpha - \beta)(\sigma_d^2 + \sigma_{d+1}^2) - \sqrt{w}}{2(2\mu d - (\alpha - \beta))} \\
\lambda < \frac{2\mu m \sigma_{d+1}^2 - (\alpha - \beta)(\sigma_d^2 + \sigma_{d+1}^2) + \sqrt{w}}{2(2\mu d - (\alpha - \beta))}
\end{cases}
\tag{37}
$$

where $w = u_2^2 - 4u_1 u_3$. To simplify the notations, we let $\Delta = \alpha - \beta$, $\sigma_{d+1} = \gamma \sigma_d$ and get

$$
\begin{cases}
\lambda > \frac{\left(2\mu m \gamma^2 - \Delta(1 + \gamma^2) - \sqrt{(\Delta(1 + \gamma^2) - 2\mu m \gamma^2)^2 - 4(\Delta - 2\mu d)(\Delta - 2\mu m + 2\mu d)}\right)\sigma_d^2}{4\mu d - 2\Delta} \\
\lambda < \frac{\left(2\mu m \gamma^2 - \Delta(1 + \gamma^2) + \sqrt{(\Delta(1 + \gamma^2) - 2\mu m \gamma^2)^2 - 4(\Delta - 2\mu d)(\Delta - 2\mu m + 2\mu d)}\right)\sigma_d^2}{4\mu d - 2\Delta}
\end{cases}
\tag{38}
$$

Further, let $\rho = 2\mu m \gamma^2 - \Delta(1 + \gamma^2)$, we arrive at

$$
\begin{cases}
\lambda > \frac{\left(\rho - \sqrt{\rho^2 - 4(\Delta - 2\mu d)(\Delta - 2\mu m + 2\mu d)}\right)\sigma_d^2}{4\mu d - 2\Delta} \\
\lambda < \frac{\left(\rho + \sqrt{\rho^2 - 4(\Delta - 2\mu d)(\Delta - 2\mu m + 2\mu d)}\right)\sigma_d^2}{4\mu d - 2\Delta}
\end{cases}
\tag{39}
$$

That means, if (39) holds, for every i, the indices of the largest $\bar{\tau}$ absolute elements in the i-th column of C are in $C_{\pi(i)}$. Therefore, the truncation operation with parameter $\tau \leq \bar{\tau}$ ensures the subspace detection property. This finished the proof.

□

E.6 Proof for Proposition 3.8

Proof. The condition of reg means

$$\frac{\sigma_{k+1}(\boldsymbol{L}) - \frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L})}{\frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L}) + \epsilon} = \frac{\sigma_{k+1}(\boldsymbol{L})}{\epsilon} > 0.$$

For convenience, denote $\vartheta = \frac{1}{k}\sum_{i=1}^{k}\sigma_i(\boldsymbol{L})$. We have

$$-\vartheta\epsilon = \vartheta\sigma_{k+1}.$$

It indicates $\vartheta = 0$ and $\sigma_{k+1} \neq 0$. Therefore the graph has exactly k connected components. Since the subspace or manifold detection property hold for $\boldsymbol{A}$, each component of G is composed of the columns of $\boldsymbol{X}$ in the same subspace or manifold. Thus, all the columns of $\boldsymbol{X}$ in the same subspace or manifold must be in the same component. Otherwise, the number of connected components is larger than k. □

E.7 Proof for Theorem C.3

The proof is nearly the same as that for Theorem 3.7, except that $d < n$ and $\text{rank}(\boldsymbol{K}_0) \leq k\binom{r+pq}{pq}$, where $\boldsymbol{K}_0 = \phi(\boldsymbol{X}_0)^\top\phi(\boldsymbol{X}_0)$. In this case, $\boldsymbol{K}$ can be well approximately by a low-rank matrix of rank at most $k\binom{r+pq}{pq}$ provided that the noise is small enough. More details about $\boldsymbol{K}_0$ can be found in [Fan *et al.*, 2020].

E.8 Proof for Proposition C.4

Proof. We only need to provide an example of $\boldsymbol{W}_1 \in \mathbb{R}^{d\times m}$, $\boldsymbol{W}_2 \in \mathbb{R}^{k\times d}$, $\boldsymbol{b}_1 \in \mathbb{R}^d$, and $\boldsymbol{b}_2 \in \mathbb{R}^k$, where $d = kr$, such that the clusters can be recognized by k-means.

We organize the rows of $\boldsymbol{W}_1$ into k groups: $\boldsymbol{W}_1^j \in \mathbb{R}^{r\times m}$, $j = 1, \ldots, k$. Let $\boldsymbol{W}_1^j = \boldsymbol{U}^{j\top}$, $j = 1, \ldots, k$. Let $\boldsymbol{W}_1\boldsymbol{x}_i = \boldsymbol{\alpha}_i = (\boldsymbol{\alpha}_i^1, \ldots, \boldsymbol{\alpha}_i^r)$. When $\boldsymbol{x}_i \in \mathcal{S}_j$, we have

$$\boldsymbol{\alpha}_i^j = \boldsymbol{U}^{j\top}\boldsymbol{x}_i = \boldsymbol{U}^{j\top}\boldsymbol{U}^j\boldsymbol{v}_i = \boldsymbol{v}_i. \tag{40}$$

It follows from the assumption that

$$\max_p \alpha_{pi}^j > \mu. \tag{41}$$

Let $\boldsymbol{b}_1 = [\boldsymbol{b}_1^1; \ldots; \boldsymbol{b}_1^k] = -\mu\boldsymbol{1}$. Then $\boldsymbol{h}_i^j = \text{ReLU}(\boldsymbol{\alpha}_i^j + \boldsymbol{b}_1^j)$ has at least one positive element. On the other hand, since

$$\boldsymbol{\alpha}_i^l = \boldsymbol{U}^{l\top}\boldsymbol{x}_i = \boldsymbol{U}^{l\top}\boldsymbol{U}^j\boldsymbol{v}_i \quad l \neq j, \tag{42}$$

using the assumption of μ, we have

$$|\alpha_{pi}^l| = |\boldsymbol{U}_{:p}^{l\top}\boldsymbol{U}^j\boldsymbol{v}_i| \leq \|\boldsymbol{U}_{:p}^{l\top}\boldsymbol{U}^j\|\|\boldsymbol{v}_i\| \leq \mu, \tag{43}$$

where we have used the fact $\|\boldsymbol{v}_i\| = 1$ because $\|\boldsymbol{x}_i\| = 1$. It follows that

$$\boldsymbol{h}_i^l = \text{ReLU}(\boldsymbol{\alpha}_i^l + \boldsymbol{b}_1^l) = \boldsymbol{0}, \quad l \neq j.$$

Now we formulate $\boldsymbol{W}_2$ as

$$\boldsymbol{W}_2 = \begin{bmatrix} \boldsymbol{q}_{11} & \boldsymbol{q}_{12} & \cdots & \boldsymbol{q}_{1k} \\ \boldsymbol{q}_{21} & \boldsymbol{q}_{22} & \cdots & \boldsymbol{q}_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{q}_{k1} & \boldsymbol{q}_{k2} & \cdots & \boldsymbol{q}_{kk} \end{bmatrix}, \tag{44}$$

where $\boldsymbol{q}_{lj} \in \mathbb{R}^{1\times r}$, $l, j = 1, \ldots, k$. We have

$$z_{ji} = \boldsymbol{q}_{j1}\boldsymbol{h}_i^1 + \boldsymbol{q}_{j2}\boldsymbol{h}_i^2 \cdots + \boldsymbol{q}_{jk}\boldsymbol{h}_i^k = \boldsymbol{q}_{jj}\boldsymbol{h}_i^j.$$

and

$$z_{li} = \boldsymbol{q}_{l1}\boldsymbol{h}_i^1 + \boldsymbol{q}_{l2}\boldsymbol{h}_i^2 \cdots + \boldsymbol{q}_{lk}\boldsymbol{h}_i^k = \boldsymbol{q}_{lj}\boldsymbol{h}_i^j.$$

Here we have let $\boldsymbol{b}_2 = \boldsymbol{0}$. Let $\boldsymbol{q}_{jj} \geq \boldsymbol{0}$ and $\boldsymbol{q}_{lj} = \boldsymbol{0}$, we have

$$z_{ji} > z_{li} = 0.$$

Therefore, if $\boldsymbol{x}_i \in \mathcal{S}_j$, we have $z_{ji} > 0$ and $z_{li} = 0\ \forall 1 \leq j \neq l \leq k$. Now performing k-means on $\boldsymbol{Z} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n]$ can identify the clusters trivially.

□