# EXPLORING A PRINCIPLED FRAMEWORK FOR DEEP SUBSPACE CLUSTERING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Subspace clustering is a classical unsupervised learning task, built on a basic assumption that high-dimensional data can be approximated by a union of subspaces (UoS). Nevertheless, the real-world data are often deviating from the UoS assumption. To address this challenge, state-of-the-art deep subspace clustering algorithms attempt to jointly learn UoS representations and self-expressive coefficients. However, the general framework of the existing algorithms suffers from a catastrophic feature collapse and lacks a theoretical guarantee to learn desired UoS representation. In this paper, we present a Principled fRamewOrk for Deep Subspace Clustering (PRO-DSC), which is designed to learn structured representations and self-expressive coefficients in a unified manner. Specifically, in PRO-DSC, we incorporate an effective regularization on the learned representations into the self-expressive model, and prove that the regularized self-expressive model is able to prevent feature space collapse and the learned optimal representations under certain condition lie on a union of orthogonal subspaces. Moreover, we provide a scalable and efficient approach to implement our PRO-DSC and conduct extensive experiments to verify our theoretical findings and demonstrate the superior performance of our proposed deep subspace clustering approach.

## 1 INTRODUCTION

Subspace clustering is a classical unsupervised learning task, aiming to partition high dimensional data that are approximately lying on a union of subspaces (UoS), and finds wide-ranging applications, such as motion segmentation (Costeira & Kanade, 1998; Vidal et al., 2008; Rao et al., 2010), hybrid system identification (Vidal, 2004; Bako & Vidal, 2008), image representation and clustering (Hong et al., 2006; Lu et al., 2012), genes expression clustering (McWilliams & Montana, 2014) and so on.

Existing subspace clustering algorithms can be roughly divided into four categories: iterative methods (Tseng, 2000; Ho et al., 2003; Zhang et al., 2009), algebraic geometry based methods (Vidal et al., 2005; Tsakiris & Vidal, 2017), statistical methods (Fischler & Bolles, 1981), and spectral clustering-based methods (Chen & Lerman, 2009; Elhamifar & Vidal, 2009; Liu et al., 2010; Lu et al., 2012; You et al., 2016a; Zhang et al., 2021). Among them, spectral clustering based methods gain the most popularity due to the broad theoretical guarantee and superior performance.

The vital component in spectral clustering based methods is a so-called *self-expressive* model (Elhamifar & Vidal, 2009; 2013). Formally, given a dataset $\mathcal{X} := \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$ where $\boldsymbol{x}_j \in \mathbb{R}^D$, self-expressive model expresses each data point $\boldsymbol{x}_j$ by a linear combination of other points, i.e.,

$$\boldsymbol{x}_j = \sum_{i \neq j} c_{ij} \boldsymbol{x}_i, \tag{1}$$

where $c_{ij}$ is the corresponding self-expressive coefficient. The most intriguing merit of the self-expressive model is that the solution of the self-expressive model under proper regularizer on the coefficients $c_{ij}$ is guaranteed to satisfy a subspace-preserving property, namely, $c_{ij} \neq 0$ *only if* $\boldsymbol{x}_i$ *and* $\boldsymbol{x}_j$ *are in the same subspace* (Elhamifar & Vidal, 2013; Soltanolkotabi & Candes, 2012). Having had the optimal self-expressive coefficients $\{c_{ij}\}_{i,j=1}^N$, the data affinity can be induced by $|c_{ij}| + |c_{ji}|$ for which spectral clustering is applied to yield the partition of the data.

Despite the broad theoretical guarantee, the vanilla self-expressive model still faces great challenges when applied to the complex real-world data that may not well align with the UoS assumption. Earlier works devote to address this deficiency by learning a linear transform of the data (Patel et al., 2013; 2015) or introducing a nonlinear kernel mapping (Patel & Vidal, 2014) under which the representations of the data are supposed to be aligned with the UoS assumption. However, there is a lack of principled mechanism to guide the learning of the linear transforms or the design of the nonlinear kernels to guarantee the representations of the data to form a UoS structure.

To handle complex real-world data, in the past few years, there is a surge of interests in designing deep subspace clustering frameworks, e.g., (Peng et al., 2016; 2018; Ji et al., 2017; Zhou et al., 2018; Zhang et al., 2019a; Dang et al., 2020; Peng et al., 2020; Lv et al., 2021; Wang et al., 2023b). In these works, usually a deep neural network-based representation learning module is integrated to the self-expressive model, to learn the representations $\boldsymbol{Z} \in \mathbb{R}^{d \times N}$ and the self-expressive coefficients $\boldsymbol{C} = \{c_{ij}\}_{i,j=1}^{N}$ in a joint optimization framework. However, as analyzed in (Haeffele et al., 2021) that, the optimal representations $\boldsymbol{Z}$ of these methods tend to catastrophically collapse into subspaces with dimensions much lower than the ambient space, which is detrimental to subspace clustering and there is no evidence that the learned representations form a UoS structure.

In this paper, we attempt to propose a Principled fRamewOrk for Deep Subspace Clustering (PRO-DSC), which is able to simultaneously learn structured representations and self-expressive coefficients. Specifically, in PRO-DSC, we incorporate an effective regularization on the learned representations into the self-expressive model and prove that our PRO-DSC can effectively prevent feature collapse. Moreover, we demonstrate that our PRO-DSC under certain condition can yield structured representations forming a UoS structure and provide a scalable and efficient approach to implement PRO-DSC. We conduct extensive experiments on the synthetic data and six benchmark datasets to verify our theoretical results and evaluate the performance of the proposed approach.

**Contributions.** The contributions of the paper are highlighted as follows.

1. We propose a Principled fRamewOrk for Deep Subspace Clustering (PRO-DSC) that learns both structured representations and self-expressive coefficients simultaneously, in which an effective regularization on the learned representations is incorporated to prevent feature space collapse.
2. We provide a rigorous analysis for the optimal solution of our PRO-DSC, derive a sufficient condition that guarantees the learned representations to escape from feature collapse, and further demonstrate that our PRO-DSC under certain condition can yield structured representations of a UoS structure.
3. We conduct extensive experiments to verify our theoretical findings and to demonstrate the superior performance of the proposed approach.

To the best of our knowledge, this is the first principled framework for deep subspace clustering that is guaranteed to yield the desired UoS representations.

## 2 Deep Subspace Clustering: A Principled Framework, Justification, and Implementation

In this section, we review the deficiency that was suffering in the existing Self-Expressive Deep Subspace Clustering (SEDSC) frameworks at first, then present our principled framework for deep subspace clustering and provide a rigorous characterization of the optimal solution and the property of the learned structured representations. Finally we describe a scalable implementation based on differential programming for the proposed framework. Please refer to Appendix A for the detailed proofs of our theoretical results.

### 2.1 Prerequisite

To apply subspace clustering to complex real-world data that may not well align with the UoS assumption, there has been a surge of interests in exploiting deep neural networks to learn representations and then apply self-expressive model to the learned representations, e.g., (Peng et al., 2016; 2018; Ji et al., 2017; Zhou et al., 2018; Zhang et al., 2019a; Dang et al., 2020; Peng et al., 2020; Lv et al., 2021; Wang et al., 2023b).

Formally, the optimization problem of these SEDSC models can be formulated as:[1]

$$\begin{aligned} \min_{\boldsymbol{Z},\boldsymbol{C}} \quad & \frac{1}{2}\|\boldsymbol{Z}-\boldsymbol{Z}\boldsymbol{C}\|_F^2+\beta\cdot r(\boldsymbol{C}) \\ \text{s.t.} \quad & \|\boldsymbol{Z}\|_F^2=N, \end{aligned} \tag{2}$$

where $\boldsymbol{Z}\in\mathbb{R}^{d\times N}$ denotes the learned representation, $\boldsymbol{C}\in\mathbb{R}^{N\times N}$ denotes the self-expressive coefficient matrix, and $\beta>0$ is a hyper-parameter. The following lemma characterizes the property of the optimal solution $\boldsymbol{Z}$ for problem (2).

**Lemma 1** (Haeffele et al., 2021)**.** *The rows of the optimal solution $\boldsymbol{Z}$ for problem (2) are the eigenvectors that associate with the smallest eigenvalues of $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$.*

In other words, the optimal representation $\boldsymbol{Z}$ in SEDSC is restricted to an extremely "narrow" subspace whose dimension is much smaller than $d$, leading to an undesirable collapsed solution. [2]

### 2.2 Our Principled Framework for Deep Subspace Clustering

In this paper, we attempt to propose a principled framework for deep subspace clustering that provably learns structured representations with maximal intrinsic dimensions.

To be specific, we try to optimize the self-expressive model (2) while preserving the intrinsic dimension of the representation space. Other than using the rank, which is a common measure of the dimension, inspired by (Fazel et al., 2003; Ma et al., 2007; Yu et al., 2020; Liu et al., 2022), we propose to prevent the space collapse by incorporating the $\log\det(\cdot)$-based concave smooth surrogate which is defined as follows:

$$R(\boldsymbol{Z};\alpha)\coloneqq\log\det(\boldsymbol{I}+\alpha\boldsymbol{Z}^\top\boldsymbol{Z}),\quad \text{s.t. } \boldsymbol{Z}\in\mathbb{S}^{d-1}, \tag{3}$$

where $\alpha>0$ is a hyper-parameter. Unlike the commonly used nuclear norm, which is a convex surrogate of the rank, the $\log\det(\cdot)$-based function is concave but offers a closer approximation and encourages learning subspaces with maximal intrinsic dimensions.

By incorporating the maximization of $R(\boldsymbol{Z};\alpha)$ as a regularizer into the formulation of SEDSC in (2), we have a Principled fRamewOrk for Deep Subspace Clustering (PRO-DSC):

$$\begin{aligned} \min_{\boldsymbol{Z},\boldsymbol{C}} \quad & -\frac{1}{2}\log\det\left(\boldsymbol{I}+\alpha\boldsymbol{Z}^\top\boldsymbol{Z}\right)+\frac{\gamma}{2}\|\boldsymbol{Z}-\boldsymbol{Z}\boldsymbol{C}\|_F^2+\beta\cdot r(\boldsymbol{C}) \\ \text{s.t.} \quad & \|\boldsymbol{Z}\|_F^2=N, \end{aligned} \tag{4}$$

where $\gamma>0$ is a balancing hyper-parameter.

Now, we will give our theoretical findings about the optimal solution for problem (4). Note that the Lagrangian of problem (4) is:

$$\mathcal{L}(\boldsymbol{Z},\boldsymbol{C},\nu)\coloneqq-\frac{1}{2}\log\det\left(\boldsymbol{I}+\alpha\boldsymbol{Z}^\top\boldsymbol{Z}\right)+\frac{\gamma}{2}\|\boldsymbol{Z}-\boldsymbol{Z}\boldsymbol{C}\|_F^2+\beta r(\boldsymbol{C})+\nu(\|\boldsymbol{Z}\|_F^2-N), \tag{5}$$

where $\nu\in\mathbb{R}$ is the Lagrange multiplier. By analyzing the gradient flow dynamics of problem (5), we show that the eigenspaces of $\boldsymbol{Z}^\top\boldsymbol{Z}$ and $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$ align progressively, which is crucial for the subsequent analysis. Precisely, we have a theorem as follows.

**Theorem 1** (Informal Statement)**.** *Denote $\boldsymbol{G}\coloneqq\boldsymbol{Z}^\top\boldsymbol{Z}$, $\boldsymbol{M}\coloneqq(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$. Under mild condition, then $\boldsymbol{G}$ and $\boldsymbol{M}$ will converge to have eigenspaces aligned, i.e., $\boldsymbol{G}$ and $\boldsymbol{M}$ can be diagonalized simultaneously by $\boldsymbol{U}\in\mathcal{O}^{N\times N}$ where $\mathcal{O}^{N\times N}$ is an $N\times N$ orthogonal matrix group.*

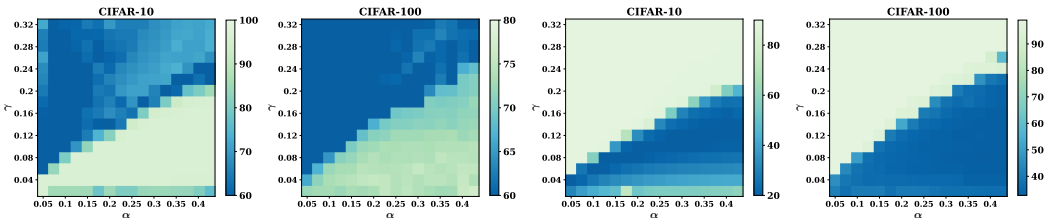
Next, we will analyze problem (4) from the perspective of alternating optimization. When $\boldsymbol{Z}$ is fixed, the optimization problem with respect to (w.r.t.) $\boldsymbol{C}$ reduces to a standard self-expressive model, which has been extensively studied in (Soltanolkotabi & Candes, 2012; Pimentel-Alarcon & Nowak, 2016; Wang & Xu, 2016; Tsakiris & Vidal, 2018). On the other hand, when $\boldsymbol{C}$ is fixed, the optimization problem w.r.t. $\boldsymbol{Z}$ becomes:

$$\begin{aligned} \min_{\boldsymbol{Z}} \quad & -\frac{1}{2}\log\det\left(\boldsymbol{I}+\alpha\boldsymbol{Z}^\top\boldsymbol{Z}\right)+\frac{\gamma}{2}\|\boldsymbol{Z}-\boldsymbol{Z}\boldsymbol{C}\|_F^2 \\ \text{s.t.} \quad & \|\boldsymbol{Z}\|_F^2=N, \end{aligned} \tag{6}$$

which is a *non-convex* optimization problem, whose optimal solution remains under-explored.

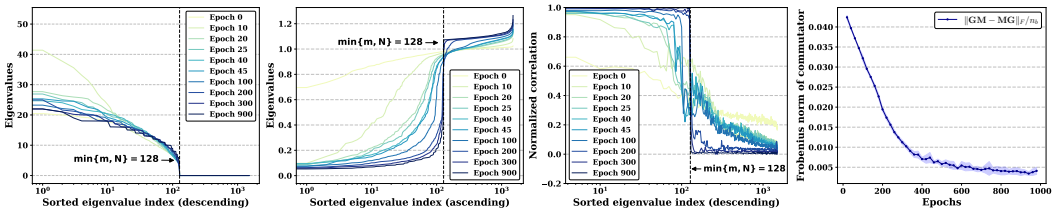
[1] Without loss of generality, we omit the constraint $\mathrm{Diag}(\boldsymbol{C})=\boldsymbol{0}$ throughout the analysis.

[2] The dimension equals to the multiplicity of the smallest eigenvalues of $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$.




(a) ACC (%) (b) SRE (%)

Figure 1: **Empirical validation to Theorem 2.** We train PRO-DSC on CIFAR-10 and CIFAR-100, and report the clustering accuracy (ACC%) and subspace-preserving representation error (SRE%) with varying $\alpha$ and $\gamma$. A clear linear phase transition phenomenon can be observed, which is consistent with the condition to avoid collapse: $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$.




(a) $\lambda(\boldsymbol{G})$ (b) $\lambda(\boldsymbol{M})$ (c) $\langle \boldsymbol{u}_j, \boldsymbol{G}\boldsymbol{u}_j/\|\boldsymbol{G}\boldsymbol{u}_j\|_2 \rangle$ (d) $\|\boldsymbol{G}\boldsymbol{M} - \boldsymbol{M}\boldsymbol{G}\|_F$

Figure 2: **Empirical validations to Theorem 1 and 2.** We train PRO-DSC on CIFAR-100. (a) Eigenvalues of $\boldsymbol{G}$. (b) Eigenvalues of $\boldsymbol{M}$. (c) For each eigenvector $\boldsymbol{u}_j$ of $\boldsymbol{M}$, we compute $\langle \boldsymbol{u}_j, \frac{\boldsymbol{G}\boldsymbol{u}_j}{\|\boldsymbol{G}\boldsymbol{u}_j\|_2} \rangle$ to measure the eigenspace alignment of $\boldsymbol{G}$ and $\boldsymbol{M}$. (d) We compute $\|\boldsymbol{L}\|_F$ to measure the eigenspace alignment of $\boldsymbol{G}$ and $\boldsymbol{M}$.

In light of the fact that $\boldsymbol{G}$ and $\boldsymbol{M}$ converge to share eigenspaces, we decompose $\boldsymbol{G}$ and $\boldsymbol{M}$ to $\boldsymbol{U}\,\mathrm{Diag}(\lambda_{\boldsymbol{G}}^{(1)}, \cdots, \lambda_{\boldsymbol{G}}^{(N)})\boldsymbol{U}^\top$ and $\boldsymbol{U}\,\mathrm{Diag}(\lambda_{\boldsymbol{M}}^{(1)}, \cdots, \lambda_{\boldsymbol{M}}^{(N)})\boldsymbol{U}^\top$, respectively. Recall that $\boldsymbol{G} \coloneqq \boldsymbol{Z}^\top\boldsymbol{Z}$, $\boldsymbol{M} \coloneqq (\boldsymbol{I} - \boldsymbol{C})(\boldsymbol{I} - \boldsymbol{C})^\top$, by using the eigenvalue decomposition, we have that problem (6) is transformed to a *convex* problem (detailed in Appendix A):

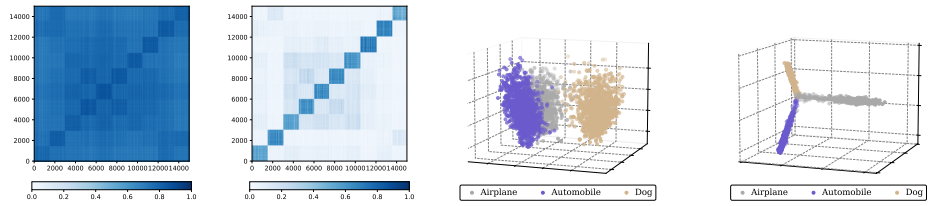
$$\begin{aligned} \min_{\{\lambda_{\boldsymbol{G}}^{(i)}\}_{i=1}^{\min\{d,N\}}} \quad & -\frac{1}{2}\sum_{i=1}^{\min\{d,N\}} \log(1 + \alpha\lambda_{\boldsymbol{G}}^{(i)}) + \frac{\gamma}{2}\lambda_{\boldsymbol{M}}^{(i)}\lambda_{\boldsymbol{G}}^{(i)} \\ \text{s.t.} \quad & \sum_{i=1}^{\min\{d,N\}} \lambda_{\boldsymbol{G}}^{(i)} = N, \lambda_{\boldsymbol{G}}^{(i)} \geq 0, \quad \text{for all } i = 1, \ldots, \min\{d, N\}, \end{aligned} \tag{7}$$

which is a classical reverse water-filling problem (Yu et al., 2004). By solving problem (7), we have the following theorem.

**Theorem 2.** *Suppose that $\boldsymbol{G}$ and $\boldsymbol{M}$ have eigenspaces aligned and $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$, then we have that* $\mathrm{rank}(\boldsymbol{Z}_\star) = \min\{d, N\}$ *and the singular values* $\sigma_{\boldsymbol{Z}_\star}^{(i)} = \sqrt{\frac{1}{\gamma\lambda_{\boldsymbol{M}}^{(i)} + \nu_\star} - \frac{1}{\alpha}}$, *for all* $i = 1, \ldots, \min\{d, N\}$, *where $\boldsymbol{Z}_\star$ and $\nu_\star$ are the primal optimal solution and dual optimal solution.*

The above theorem characterizes the optimal solution for problem (6). As shown, the rank of the minimizers for PRO-DSC satisfies that $\mathrm{rank}(\boldsymbol{Z}_\star) = \min\{d, N\}$. On the contrary, SEDSC yields a collapsed solution, where $\mathrm{rank}(\boldsymbol{Z}_\star) \ll \min\{d, N\}$. In Figure 1, we show the subspace clustering accuracy (ACC) and subspace-representation error[3] (SRE) as a function of the parameter $\alpha$ and $\gamma$. The phase transition phenomenon around $\gamma = (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$ well illustrates the sufficient condition in Theorem 2 to avoid collapse. In Figure 2, we illustrate the curves of the eigenvalues and the alignment of eigenspaces of $\boldsymbol{G}$ and $\boldsymbol{M}$, which demonstrate that the learned representation does no longer collapse and the two eigenspaces are approximately to be aligned.

[3]For each row $\boldsymbol{c}^{(j)}$ in $\boldsymbol{C}$, SRE is computed by $\frac{100}{N}\sum_j(1 - \sum_i w_{ij} \cdot |c_{ij}|)/\|\boldsymbol{c}^{(j)}\|_1$, where $w_{ij} \in \{0, 1\}$ is the ground-truth affinity.




(a) $|\boldsymbol{X}^\top \boldsymbol{X}|$ (b) $|\boldsymbol{Z}^\top \boldsymbol{Z}|$ (c) $\boldsymbol{X}^{(3)}$ via PCA (d) $\boldsymbol{Z}^{(3)}$ via PCA

Figure 3: **Empirical validation to Theorem 3.** We visualize the Gram matrices and the dimension reduction results via PCA for CLIP features $\boldsymbol{X}$ and the learned representations $\boldsymbol{Z}$ on CIFAR-10. For the clarity of visualization, we apply PCA to the samples from three categories $\boldsymbol{X}^{(3)}$ and $\boldsymbol{Z}^{(3)}$.

Furthermore, from the perspective of joint optimizing $\boldsymbol{Z}$ and $\boldsymbol{C}$, the following theorem demonstrates that PRO-DSC yields union-of-orthogonal-subspaces representations $\boldsymbol{Z}$ and block-diagonal self-expressive matrix $\boldsymbol{C}$ under certain condition.

**Theorem 3.** *Suppose that the sufficient conditions to prevent catastrophic feature collapse are satisfied. Without loss of generality, we further assume that the columns in matrix $\boldsymbol{Z}$ are arranged into $k$ blocks according to a certain $N \times N$ permutation matrix $\boldsymbol{\Gamma}$, i.e., $\boldsymbol{Z} = [\boldsymbol{Z}_1, \boldsymbol{Z}_2, \cdots, \boldsymbol{Z}_k]$. Then the condition for that PRO-DSC promotes the optimal solution $(\boldsymbol{Z}_\star, \boldsymbol{C}_\star)$ to have desired structure, i.e., $\boldsymbol{Z}_\star^\top \boldsymbol{Z}_\star$ and $\boldsymbol{C}_\star$ are both block-diagonal, is that $\langle (\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top, \boldsymbol{G}-\boldsymbol{G}^* \rangle \to 0$, where $\boldsymbol{G}^* \coloneqq \operatorname{Diag}\left(\boldsymbol{G}_{11}, \boldsymbol{G}_{22}, \cdots, \boldsymbol{G}_{kk}\right)$ and $\boldsymbol{G}_{jj}$ is the block Gram matrix corresponding to $\boldsymbol{Z}_j$.*

**Remark 1.** Theorem 3 suggests that our PRO-DSC is able to learn representations and self-expressive matrix with desired structures, i.e., the representations form a union of orthogonal subspaces and accordingly the self-expressive matrix is block-diagonal, when the condition $\langle (\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top, \boldsymbol{G}-\boldsymbol{G}^* \rangle \to 0$ is met. We call this condition *a compatibly structured coherence* (CSC), which relates to the properties of the distribution of the representations in $\boldsymbol{Z}$ and the self-coefficients in $\boldsymbol{C}$. While it is not possible for us to give a theoretical justification when the CSC condition will be satisfied in general, we do have the empirical evidence that our implementation for PRO-DSC with careful designs does approximately satisfy such a condition and thus yields representations and self-expressive matrix with desired structure (See Figures 3 and 4).[4]

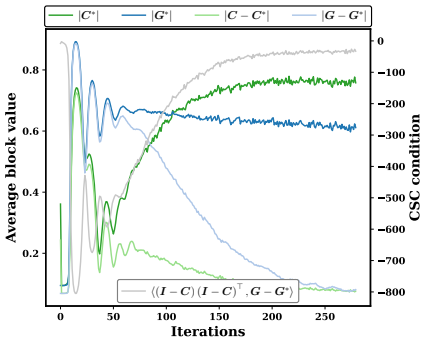



Figure 4: **Empirical validation to Theorem 3 on CIFAR-10.**

## 2.3 Scalable Implementation

Existing SEDSC models typically use autoencoders to learn the representations, and learn the self-expressive matrix $\boldsymbol{C}$ through an $N \times N$ fully-connected layer (Peng et al., 2016; 2018; Ji et al., 2017; Zhou et al., 2018; Zhang et al., 2019a). While such implementation is straightforward, there is two major drawbacks: a) since that the number of self-expressive coefficients is quadratic to the number of data points, solving these coefficients suffers from expensive computation burden; b) the learning process is transductive, i.e., the network parameters cannot be generalized to unseen data.

To address these issues, similar to (Zhang et al., 2021), we reparameterize the self-expressive coefficients $c_{ij}$ by a neural network. Specifically, the input data $\boldsymbol{x}_i$ is fed into a neural network $\boldsymbol{h}(\cdot; \boldsymbol{\Psi}) : \mathbb{R}^D \to \mathbb{R}^d$ to yield normalized representations, i.e.,

$$\boldsymbol{y}_i \coloneqq \boldsymbol{h}(\boldsymbol{x}_i; \boldsymbol{\Psi}) / \|\boldsymbol{h}(\boldsymbol{x}_i; \boldsymbol{\Psi})\|_2, \tag{8}$$

[4]Please refer to Appendix B.2 for more details about Figures 1–4.

where $\boldsymbol{\Psi}$ denotes all the parameters in $\boldsymbol{h}(\cdot)$. Then, the parameterized self-expressive matrix $\boldsymbol{C}_{\boldsymbol{\Psi}}$ is generated by:

$$\boldsymbol{C}_{\boldsymbol{\Psi}} \coloneqq \mathcal{P}(\boldsymbol{Y}^\top \boldsymbol{Y}), \tag{9}$$

where $\boldsymbol{Y} \coloneqq [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N] \in \mathbb{R}^{d \times N}$ and $\mathcal{P}(\cdot)$ is the sinkhorn projection (Cuturi, 2013), which has been widely applied in the deep clustering field (Caron et al., 2020; Ding et al., 2023).[5] To enable efficient representation learning, we introduce another learnable mapping $\boldsymbol{f}(\cdot; \boldsymbol{\Theta}) : \mathbb{R}^D \to \mathbb{R}^d$, for which

$$\boldsymbol{z}_j \coloneqq \boldsymbol{f}(\boldsymbol{x}_j; \boldsymbol{\Theta}) / \|\boldsymbol{f}(\boldsymbol{x}_j; \boldsymbol{\Theta})\|_2 \tag{10}$$

is the learned representation for the input $\boldsymbol{x}_j$, where $\boldsymbol{\Theta}$ denotes the parameters in $\boldsymbol{f}(\cdot)$ to learn the structured representation $\boldsymbol{Z}_{\boldsymbol{\Theta}} \coloneqq [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N] \in \mathbb{R}^{d \times N}$.

Therefore, our principled framework for deep subspace clustering (PRO-DSC) in (4) can be reparameterized and reformulated as follows:

$$\min_{\boldsymbol{\Theta}, \boldsymbol{\Psi}} \quad \mathcal{L}(\boldsymbol{\Theta}, \boldsymbol{\Psi}) \coloneqq -\frac{1}{2} \log\det\left(\boldsymbol{I} + \alpha \boldsymbol{Z}_{\boldsymbol{\Theta}}^\top \boldsymbol{Z}_{\boldsymbol{\Theta}}\right) + \frac{\gamma}{2} \|\boldsymbol{Z}_{\boldsymbol{\Theta}} - \boldsymbol{Z}_{\boldsymbol{\Theta}} \boldsymbol{C}_{\boldsymbol{\Psi}}\|_F^2 + \beta \cdot r\left(\boldsymbol{C}_{\boldsymbol{\Psi}}\right). \tag{11}$$

To strengthen the block-diagonal structure of self-expressive matrix, we choose the block-diagonal regularizer (Lu et al., 2018) for $r(\boldsymbol{C}_{\boldsymbol{\Psi}})$. To be specific, given the data affinity $\boldsymbol{A}_{\boldsymbol{\Psi}}$, which is induced by default as $\boldsymbol{A}_{\boldsymbol{\Psi}} \coloneqq \frac{1}{2}\left(|\boldsymbol{C}_{\boldsymbol{\Psi}}| + |\boldsymbol{C}_{\boldsymbol{\Psi}}^\top|\right)$, the block diagonal regularizer is defined as:

$$r\left(\boldsymbol{C}_{\boldsymbol{\Psi}}\right) \coloneqq \|\boldsymbol{A}_{\boldsymbol{\Psi}}\|_{\boxed{\kappa}}, \tag{12}$$

where $\|\boldsymbol{A}_{\boldsymbol{\Psi}}\|_{\boxed{\kappa}}$ is the sum of the $k$ smallest eigenvalues of the Laplacian matrix of the affinity $\boldsymbol{A}_{\boldsymbol{\Psi}}$.[6]

Consequently, the parameters in $\boldsymbol{\Theta}$ and $\boldsymbol{\Psi}$ of reparameterized PRO-DSC can be trained by Stochastic Gradient Descent (SGD) with the loss function $\mathcal{L}(\boldsymbol{\Theta}, \boldsymbol{\Psi})$ defined in (11). For clarity, we summarize the procedure for training and testing of our PRO-DSC in Algorithm 1.

**Remark 2.** We note that all the commonly used regularizers with extended block-diagonal property for self-expressive model as discussed in (Lu et al., 2018) can be used to improve the block-diagonal structure of self-expressive matrix. More interestingly, the specific type of the regularizers is not essential owning to the learned structured representation (Please refer to Table 4 for details), and using a specific regularizer or not is also not essential since that the SGD-based optimization also induces some implicit regularization, e.g., low-rank (Gunasekar et al., 2017; Arora et al., 2019).

**Algorithm 1** Scalable & Efficient Implementation of PRO-DSC via Differential Programming

**Input:** Dataset $\mathcal{X} = \mathcal{X}_{\text{train}} \cup \mathcal{X}_{\text{test}}$, batch size $n_b$, hyper-parameters $\alpha, \beta, \gamma$, number of iterations $T$, learning rate $\eta$

**Initialization:** Random initialize the parameters $\boldsymbol{\Psi}, \boldsymbol{\Theta}$ in the networks $\boldsymbol{h}(\cdot; \boldsymbol{\Psi})$ and $\boldsymbol{f}(\cdot; \boldsymbol{\Theta})$

**Training:**

1: **for** $t = 1, \ldots, T$ **do**
2: Sample a batch $\boldsymbol{X}_b \in \mathbb{R}^{D \times n_b}$ from $\mathcal{X}_{\text{train}}$
*# Forward propagation*
3: Compute self-expressive matrix $\boldsymbol{C}_b \in \mathbb{R}^{n_b \times n_b}$ by Eqs. (8–9)
4: Compute representations $\boldsymbol{Z}_b \in \mathbb{R}^{d \times n_b}$ by Eq. (10)
*# Backward propagation*
5: Compute $\nabla_{\boldsymbol{\Psi}} \coloneqq \frac{\partial \mathcal{L}}{\partial \boldsymbol{\Psi}}$, $\nabla_{\boldsymbol{\Theta}} \coloneqq \frac{\partial \mathcal{L}}{\partial \boldsymbol{\Theta}}$
6: Set $\boldsymbol{\Psi} \leftarrow \boldsymbol{\Psi} - \eta \cdot \nabla_{\boldsymbol{\Psi}}$, $\boldsymbol{\Theta} \leftarrow \boldsymbol{\Theta} - \eta \cdot \nabla_{\boldsymbol{\Theta}}$
7: **end for**

**Testing:**

8: Compute self-expressive matrix $\boldsymbol{C}_{\text{test}}$ by Eqs. (8–9) for $\mathcal{X}_{\text{test}}$
9: Apply spectral clustering on the affinity $\boldsymbol{A}_{\text{test}}$

[5] In practice, we set $\operatorname{diag}(\boldsymbol{C}_{\boldsymbol{\Psi}}) = \boldsymbol{0}$ to prevent trivial solution $\boldsymbol{C}_{\boldsymbol{\Psi}} = \boldsymbol{I}$.

[6] Recall that the number of zero eigenvalues of the Laplacian matrix equals to the number of connected components in the graph (von Luxburg, 2007).

# 3 EXPERIMENTS

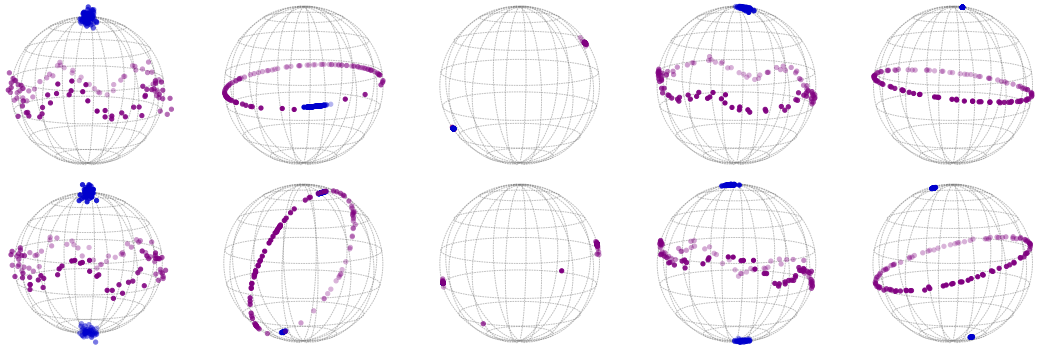
To validate the effectiveness of theoretical findings and to demonstrate the superior performance of our proposed framework, we conduct extensive experiments on synthetic data (Sec. 3.1) and real-world data (Sec. 3.2). Implementation details and more results are provided in Appendices B.1 and B.3, respectively.

## 3.1 EXPERIMENTS ON SYNTHETIC DATA

To validate that PRO-DSC addresses the collapse issue in SEDSC and learns representations with a UoS structure, we conduct experiments on synthetic data and visualize the results.

We first follow the procedure in (Ding et al., 2023) to generate synthetic data, as shown in the first row of Figure 5a. To conduct experiments on more complicated scenarios, we randomly select half of the points from the "Arctic" and place them near the "Antarctic" (Figure 5a, second row).

As shown in Figure 5b, the SEDSC models overly compress all the representations to a closed curve on the hypersphere. With increased weights (i.e., $\gamma \uparrow$) of the self-expressive term, the representations collapse to a few points (Figure 5c). The manifold linearizing and clustering method, MLC (Ding et al., 2023) approximately compresses the representations to the orthogonal subspaces. In contrast, our PRO-DSC learns linearized representations lying on orthogonal subspaces in both scenarios, confirming the validity of our theoretical analysis.



(a) Input Data (b) SEDSC (c) SEDSC ($\gamma \uparrow$) (d) MLC (e) PRO-DSC

Figure 5: Visualization of the input data and learned representations with different algorithms.

## 3.2 EXPERIMENTS ON REAL-WORLD DATA

To evaluate the performance of our proposed approach, we conduct experiments on six real-world image datasets: CIFAR-10, CIFAR-20, CIFAR-100, ImageNet-Dogs-15, Tiny-ImageNet-200, and ImageNet-1k. We measure clustering performance using clustering accuracy (ACC) and normalized mutual information (NMI). The results of our PRO-DSC in Table 1 are averaged over 10 trials (with $\pm$std) and other results of PRO-DSC are averaged over 3 trials.

**Main results.** Table 1 compares the clustering performance of our PRO-DSC on CLIP features (Radford et al., 2021) with various baseline methods, including classical clustering algorithms, e.g., $k$-means (MacQueen, 1967), spectral clustering (Shi & Malik, 2000), subspace clustering algorithm, e.g., EnSC (You et al., 2016a), SENet (Zhang et al., 2021), deep clustering algorithms, e.g., SCAN (Van Gansbeke et al., 2020), TEMI (Adaloglou et al., 2023), CPP (Chu et al., 2024), and deep subspace clustering algorithms, e.g., EDESC (Cai et al., 2022), DSCNet (Ji et al., 2017).[7] Since that the clustering performance with the CLIP feature is not reported for most baselines, we conduct experiments using the implementations provided by the authors. As shown in Table 1, PRO-DSC significantly outperforms the vanilla subspace clustering algorithms, achieving a 10% improvement on both datasets CIFAR-20 and CIFAR-100. In contrast, the previous deep subspace clustering algorithms perform less competitively, primarily because they do not learn structured representations forming a UoS. Compared to state-of-the-art deep clustering algorithms, our PRO-DSC improves

[7]Please refer to Appendix B.3 for the results on other pre-trained models.

Table 1: **Clustering results of our PRO-DSC on the CLIP features.** The best results are in bold font and the second best results are underlined. "OOM" means out of GPU memory.

| Method | CIFAR-10 | | CIFAR-20 | | CIFAR-100 | | TinyImgNet-200 | | ImgNetDogs-15 | | ImageNet-1k | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ACC | NMI | ACC | NMI | ACC | NMI | ACC | NMI | ACC | NMI | ACC | NMI |
| $k$-means | 83.5 | 84.1 | 46.9 | 49.4 | 52.8 | 66.8 | 54.1 | 73.4 | 52.7 | 53.6 | 53.9 | 79.8 |
| SC | 79.8 | 84.8 | 53.3 | 61.6 | 66.4 | 77.0 | 62.8 | 77.0 | 48.3 | 45.7 | 56.0 | 81.2 |
| SSCOMP | 85.5 | 83.0 | 61.4 | 63.4 | 55.6 | 69.7 | 56.7 | 72.7 | 25.6 | 15.9 | 44.1 | 74.4 |
| EnSC | 95.4 | 90.3 | 61.0 | 68.7 | 67.0 | 77.1 | 64.5 | 77.7 | 57.9 | 56.0 | 59.7 | **83.7** |
| SENet | 95.4 | 90.3 | 61.0 | 68.7 | 67.0 | 77.1 | 64.5 | 77.7 | 58.7 | 55.3 | OOM | OOM |
| SCAN | 95.1 | 90.3 | 60.8 | 61.8 | 64.1 | 70.8 | 56.5 | 72.7 | 70.5 | 68.2 | 54.4 | 76.8 |
| TEMI | 96.9 | 92.6 | 61.8 | 64.5 | 73.7 | 79.9 | - | - | - | - | 64.0 | - |
| CPP | 96.8 | 92.3 | 67.7 | 70.5 | 75.4 | 82.0 | 63.4 | 75.5 | 83.0 | **81.5** | 62.0 | 82.1 |
| EDESC | 84.2 | 79.3 | 48.7 | 49.1 | 53.1 | 68.6 | 51.3 | 68.8 | 53.3 | 47.9 | 46.5 | 75.5 |
| DSCNet | 78.5 | 73.6 | 38.6 | 45.7 | 39.2 | 53.4 | 62.3 | 68.3 | 40.5 | 30.1 | OOM | OOM |
| Our PRO-DSC | **97.2**$\pm$0.2 | **92.8**$\pm$0.4 | **71.6**$\pm$1.2 | **73.2**$\pm$0.5 | **77.3**$\pm$1.0 | **82.4**$\pm$0.5 | **69.8**$\pm$1.1 | **80.5**$\pm$0.7 | **84.0**$\pm$0.6 | 81.2$\pm$0.8 | **65.0**$\pm$1.2 | 83.4$\pm$0.6 |

Table 2: **Clustering results of our PRO-DSC when training from scratch.** The best results are in bold font and the second best results are underlined. Performance marked with "*" is based on our re-implementation.

| Method | CIFAR-10 | | CIFAR-20 | | CIFAR-100 | | TinyImgNet-200 | | ImgNetDogs-15 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | ACC | NMI | ACC | NMI | ACC | NMI | ACC | NMI | ACC | NMI |
| $k$-means | 22.9 | 8.7 | 13.0 | 8.4 | 9.2 | 23.0 | 2.5 | 6.5 | 10.5 | 5.5 |
| SC | 24.7 | 10.3 | 13.6 | 9.0 | 7.0 | 17.0 | 2.2 | 6.3 | 11.1 | 3.8 |
| CC | 79.0 | 70.5 | 42.9 | 43.1 | 26.9* | 48.1* | 14.0 | 34.0 | 42.9 | 44.5 |
| GCC | 85.6 | 76.4 | 47.2 | 47.2 | 28.2* | 49.9* | 13.8 | 34.7 | 52.6 | 49.0 |
| NNM | 84.3 | 74.8 | 47.7 | 48.4 | 41.2 | 55.1 | - | - | 31.1* | 34.3* |
| SCAN | 88.3 | 79.7 | 50.7 | 48.6 | 34.3 | 55.7 | - | - | 29.6* | 30.3* |
| NMCE | 89.1 | 81.2 | 53.1 | 52.4 | 40.0* | 53.9* | 21.6* | 40.0* | 39.8 | 39.3 |
| IMC-SwAV | 89.7 | 81.8 | 51.9 | 52.7 | 45.1 | 67.5 | 28.2 | **52.6** | - | - |
| MLC | 92.2 | 85.5 | 58.3 | 59.6 | 49.4 | **68.3** | 28.7* | 52.2* | 71.0* | 68.3* |
| Our PRO-DSC | **92.7**$\pm$0.1 | **85.9**$\pm$0.2 | **59.0**$\pm$0.2 | **60.4**$\pm$0.1 | **56.2**$\pm$0.6 | 67.0$\pm$0.2 | **31.2**$\pm$0.2 | 47.0$\pm$0.2 | **74.6**$\pm$0.2 | **70.2**$\pm$0.1 |

the accuracy by nearly 4% and 6% on datasets CIFAR-20 and TinyImageNet, respectively, further validating its effectiveness.

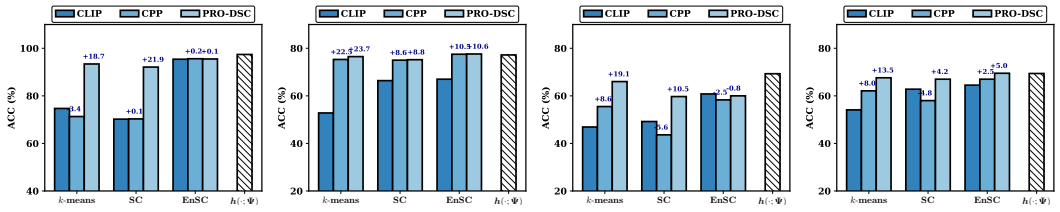
To validate the effectiveness of our PRO-DSC without using CLIP features, we conduct a fair comparison with existing deep clustering approaches and report the clustering results with training from scratch in Table 2. By stacking the $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$ on a learnable backbone, PRO-DSC can learn representations and self-expressive coefficients directly from raw images. As illustrated in Table 2, our PRO-DSC outperforms all the deep clustering baselines, including CC (Li et al., 2021), GCC (Zhong et al., 2021), NNM (Dang et al., 2021), SCAN (Van Gansbeke et al., 2020), NMCE (Li et al., 2022), IMC-SwAV (Ntelemis et al., 2022), and MLC (Ding et al., 2023).

**Evaluation on learned representations.** To quantitatively evaluate the effectiveness of the learned representations, we conduct experiments to compare the clustering performance with the CLIP features and the representations learned from CPP and our PRO-DSC (additional SEDSC results can be found in Table B.6). Specifically, we use $k$-means (MacQueen, 1967), spectral clustering (Shi & Malik, 2000), and EnSC (You et al., 2016a) to yield the clustering results.

Experimental result are shown in Figure 6. We see that the representations learned by our PRO-DSC outperform the CLIP features and the CPP representations in most cases across different clustering algorithms and datasets. Notably, the clustering accuracy with the representations learned by our PRO-DSC exceeds 90% on CIFAR-10 and 75% on CIFAR-100, whichever clustering algorithm is used. We note that the clustering performance is further improved by the learnable mapping $\boldsymbol{h}(\cdot;\boldsymbol{\Psi})$, suggesting its superior generalization ability.

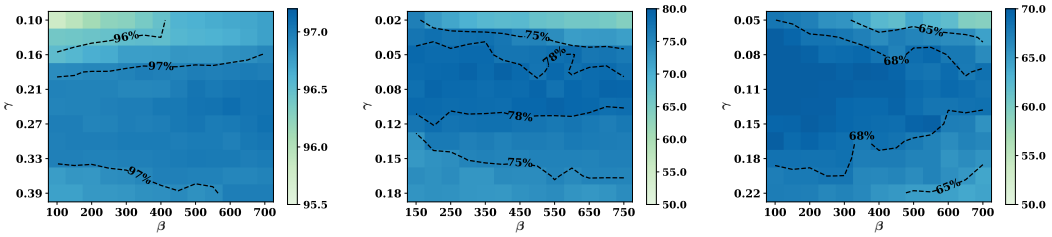
**Sensitivity of hyper-parameters.** In Figure 1, we verify that the our PRO-DSC can yield satisfactory results when the sufficient condition in Theorem 2 to avoid collapse is met. Moreover, we evaluate the performance sensitivity to the hyper-parameters $\gamma$ and $\beta$ by conducting experiments on the CLIP features of CIFAR-10, CIFAR-100 and TinyImageNet-200 with varying $\gamma$ and $\beta$. In Figure 7, we observe that the clustering performance maintains satisfactory under a broad range of $\gamma$ and $\beta$.

**Time and memory cost.** The most time-consuming operations in our PRO-DSC are computing the term involving $\log\det(\cdot)$ and the term $\|\boldsymbol{A}\|_{\boxed{\kappa}}$ involving eigenvalue decomposition, respectively. For




(a) CIFAR-10 (b) CIFAR-100 (c) CIFAR-20 (d) TinyImageNet-200

Figure 6: Clustering accuracy with CLIP features and learned representations.




(a) CIFAR-10 (b) CIFAR-100 (c) TinyImageNet

Figure 7: Evaluation on sensitivity to hyper-parameters $\gamma$ and $\beta$ on three datasets.

the former, the Gram matrix has a shape of $n_b \times n_b$, getting a time complexity of $\mathcal{O}(n_b^3)$. For the latter, the Laplacian matrix has a shape of $n_b \times n_b$, also resulting in a time complexity of $\mathcal{O}(n_b^3)$. Therefore, the overall time complexity of our PRO-DSC is $\mathcal{O}(n_b^3)$. TEMI (Adaloglou et al., 2023) employs $H = 50$ cluster heads during training, adding further time and memory costs. CPP (Chu et al., 2024) involves computing $\log\det(\cdot)$ for $n_b + 1$ times, leading to complexity $\mathcal{O}(n_b d^3)$. As shown in Table 3, our PRO-DSC significantly reduces the time consumption, particularly for datasets with a large number of clusters. All the experiments are conducted on a single NVIDIA RTX 3090 GPU and Intel Xeon Platinum 8255C CPU.

Table 3: Comparison on time (s) and memory cost (MiB). "OOM" means out of GPU memory.

| Methods | Complexity | CIFAR-10 | | CIFAR-100 | | ImageNet-1k | |
|---|---|---|---|---|---|---|---|
| | | Time | Memory | Time | Memory | Time | Memory |
| SEDSC | $\mathcal{O}(N^2 d)$ | - | OOM | - | OOM | - | OOM |
| TEMI | $\mathcal{O}(H n_b d^2)$ | 6.9 | **1,766** | 5.1 | 2,394 | 262.1 | 2,858 |
| CPP | $\mathcal{O}(n_b d^3)$ | **3.5** | 3,802 | 7.1 | 10,374 | 1441.2 | 22,433 |
| PRO-DSC | $\mathcal{O}(n_b^3)$ | 4.5 | 2,158 | **4.0** | **2,328** | **90.0** | **2,335** |

**Ablation study.** To verify the effectiveness of the loss function in PRO-DSC, we conduct ablation studies on the CLIP features of CIFAR-10, CIFAR-100, and ImageNetDogs-15, and report the results in Table 4, where $\mathcal{L}_1 := -\frac{1}{2}\log\det\left(\boldsymbol{I} + \alpha \boldsymbol{Z}_{\boldsymbol{\Theta}}^\top \boldsymbol{Z}_{\boldsymbol{\Theta}}\right)$ and $\mathcal{L}_2 := \frac{1}{2}\|\boldsymbol{Z}_{\boldsymbol{\Theta}} - \boldsymbol{Z}_{\boldsymbol{\Theta}}\boldsymbol{C}_{\boldsymbol{\Psi}}\|_F^2$. The absence of the term $\mathcal{L}_1$ leads to catastrophic feature collapse (as demonstrated in Sec. 2.1); whereas without the self-expressive $\mathcal{L}_2$, the model lacks a loss function for learning the self-expressive coefficients. In both cases, clustering performance drops significantly. More interestingly, when we replace the block diagonal regularizer $\|\boldsymbol{A}\|_{\boxed{K}}$ with $\|\boldsymbol{C}\|_1$, $\|\boldsymbol{C}\|_*$, and $\|\boldsymbol{C}\|_F^2$ or even drop the explicit regularizer $r(\cdot)$, the clustering performance still maintains satisfactory. This confirms that the choice of the regularizer is not essential owning to the structured representations learned by our PRO-DSC.

## 4 Related Work

**Deep subspace clustering.** To tackle with the complex real world data, a number of Self-Expressive Deep Subspace Clustering (SEDSC) methods have been developed in the past few years, e.g., (Peng et al., 2016; 2018; Ji et al., 2017; Zhou et al., 2018; Zhang et al., 2019a;b; Dang et al., 2020; Peng et al., 2020; Lv et al., 2021; Cai et al., 2022; Wang et al., 2023b). Key ingredient of SEDSC is

Table 4: Ablation studies on different loss functions and regularizers.

| | Loss Term | | | | | | CIFAR-10 | | CIFAR-100 | | ImgNetDogs-15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\mathcal{L}_1$ | $\mathcal{L}_2$ | $\Vert\boldsymbol{A}\Vert_{\boxed{\kappa}}$ | $\Vert\boldsymbol{C}\Vert_1$ | $\Vert\boldsymbol{C}\Vert_F^2$ | $\Vert\boldsymbol{C}\Vert_*$ | ACC | NMI | ACC | NMI | ACC | NMI |
| Ablation | | $\surd$ | $\surd$ | | | | 56.9 | 47.7 | 54.6 | 60.9 | 46.7 | 37.1 |
| | $\surd$ | | $\surd$ | | | | 69.6 | 56.4 | 64.7 | 71.7 | 10.5 | 1.7 |
| | $\surd$ | $\surd$ | | | | | 97.0 | **93.0** | 74.6 | 80.9 | 80.9 | 78.8 |
| Regularizer | $\surd$ | $\surd$ | | | | $\surd$ | 97.0 | 92.6 | 75.2 | 81.1 | 81.3 | 79.1 |
| | $\surd$ | $\surd$ | | | $\surd$ | | 97.0 | 92.6 | 75.2 | 80.9 | 80.9 | 78.8 |
| | $\surd$ | $\surd$ | | $\surd$ | | | 96.7 | 91.9 | 76.4 | 81.8 | 81.0 | 78.8 |
| | $\surd$ | $\surd$ | $\surd$ | | | | **97.2** | 92.8 | **77.3** | **82.4** | **84.0** | **81.2** |

incorporating a deep learning module to embed the input data into feature space. For example, deep autoencoder network is adopted in (Peng et al., 2016; 2018), deep convolutional autoencoder network is used in (Ji et al., 2017; Zhou et al., 2018; Zhang et al., 2019a). However, as pointed out in (Haeffele et al., 2021), SEDSC learns representations that fall into catastrophically collapsed subspaces whose dimensions are much smaller than the ambient space, which is detrimental for subspace clustering. To the best of our knowledge, a principled deep subspace clustering framework has not been proposed.

**Deep clustering.** Recently, most of state-of-the-art deep clustering methods adopt a two-step procedure: at the first step, self-supervised learning based pre-training, e.g., SimCLR (Chen et al., 2020), MoCo (He et al., 2020), BYOL (Grill et al., 2020) and SwAV (Caron et al., 2020) is adopted to learn the representations; and then deep clustering methods are incorporated to refine the representations, via, e.g., pseudo-labeling (Caron et al., 2018; Van Gansbeke et al., 2020; Park et al., 2021; Niu et al., 2022), cluster-level contrastive learning (Li et al., 2021), local and global neighbor matching (Dang et al., 2021), graph contrastive learning (Zhong et al., 2021), self-distillation (Adaloglou et al., 2023). Though the clustering performance has been improved remarkably, the underlying geometry structure of the learned representations is unclear and ignored.

**Representations learning with a UoS structure.** The methods for representation learning that favor a UoS structure are pioneered in supervised setting, e.g., (Lezama et al., 2018; Yu et al., 2020). In (Lezama et al., 2018), a nuclear norm based geometric loss is proposed to learn representations that lie on a union of orthogonal subspaces. In (Yu et al., 2020), a principled framework called Maximal Coding Rate Reduction (MCR$^2$) is proposed to learn representations that favor the structure of a union of orthogonal subspaces (Wang et al., 2024). More recently, the MCR$^2$ framework is modified to develop deep manifold clustering methods, e.g., NMCE (Li et al., 2022), MLC (Ding et al., 2023) and CPP (Chu et al., 2024). In (Li et al., 2022), the MCR$^2$ framework combines with constrastive learning to perform manifold clustering and representation learning; in (Ding et al., 2023), the MCR$^2$ framework combines with doubly stochastic affinity learning to perform manifold linearizing and clustering; and in (Chu et al., 2024), the performance of (Ding et al., 2023) on large pre-trained model (e.g., CLIP) has been investigated. While the modified MCR$^2$ framework has been incorporated into these methods for manifold clustering, none of them provide theoretical justification to yield structured representations. Though our PRO-DSC shares the same regularizer defined in Eq. (3) with MCR$^2$ (Yu et al., 2020), NMCE (Li et al., 2022), MLC (Ding et al., 2023) and CPP (Chu et al., 2024), we are for the first time to incorporate it into the SEDSC framework to attack the longstanding catastrophic feature collapse issue with theoretical analysis.

## 5 Conclusion

We presented a Principled fRamewOrk for Deep Subspace Clustering (PRO-DSC), which jointly learn structured representations and self-expressive coefficients. Specifically, PRO-DSC incorporates an effective regularization into self-expressive model to prevent the catastrophic representation collapse with theoretical justification. Moreover, we demonstrated that our PRO-DSC is able to learn structured representations that form a desirable UoS structure, and also developed an efficient implementation based on reparameterization and differential programming. We conducted extensive experiments on synthetic data and six benchmark datasets to verify the effectiveness of the theoretical findings and the superior performance of the proposed approach.

## ETHICS STATEMENT

In this work, we aim to extend traditional subspace clustering algorithms by leveraging deep learning techniques to enhance their representation learning capabilities. Our research does not involve any human subjects, and we have carefully ensured that it poses no potential risks or harms. Additionally, there are no conflicts of interest, sponsorship concerns, or issues related to discrimination, bias, or fairness associated with this study. We have taken steps to address privacy and security concerns, and all data used comply with legal and ethical standards. Our work fully adheres to research integrity principles, and no ethical concerns have arisen during the course of this study.

## REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our work, we have submitted the anonymized source code. Theoretical proofs of the claims made in this paper are provided in Appendix A, and the empirical validation of these theoretical results is shown in Figures 1– 4, with further detailed explanations in Appendix B.2. All datasets used in our experiments are publicly available, and we have provided a comprehensive description of the data processing steps in Appendix B.1. Additionally, detailed experimental settings and configurations are outlined in Appendix B.1 to facilitate the reproduction of our results.

## REFERENCES

Nikolas Adaloglou, Felix Michels, Hamza Kalisch, and Markus Kollmann. Exploring the limits of deep image clustering using pretrained models. *arXiv preprint arXiv:2303.17896*, 2023.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32, 2019.

Laurent Bako and René Vidal. Algebraic identification of MIMO SARX models. In *International Workshop on Hybrid Systems: Computation and Control*, pp. 43–57. Spinger-Verlag, 2008.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann (eds.), *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pp. 153–160. MIT Press, 2006.

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*, pp. 1–10, 2022.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 132–149, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Jianlong Chang, Gaofeng Meng, Lingfeng Wang, Shiming Xiang, and Chunhong Pan. Deep self-evolution clustering. *IEEE transactions on pattern analysis and machine intelligence*, 42(4): 809–823, 2018.

Guangliang Chen and Gilad Lerman. Spectral curvature clustering (SCC). *International Journal of Computer Vision*, 81(3):317–330, 2009. ISSN 0920-5691.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Tianzhe Chu, Shengbang Tong, Tianjiao Ding, Xili Dai, Benjamin David Haeffele, René Vidal, and Yi Ma. Image clustering via the principle of rate reduction in the age of pretrained models. In *The Twelfth International Conference on Learning Representations*, 2024.

Joao Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29:159–179, 1998.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Zhiyuan Dang, Cheng Deng, Xu Yang, and Heng Huang. Multi-scale fusion subspace clustering using similarity constraint. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 6657–6666, 2020.

Zhiyuan Dang, Cheng Deng, Xu Yang, Kun Wei, and Heng Huang. Nearest neighbor matching for deep clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13693–13702, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Tianjiao Ding, Shengbang Tong, Kwan Ho Ryan Chan, Xili Dai, Yi Ma, and Benjamin D. Haeffele. Unsupervised manifold linearizing and clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5450–5461, October 2023.

Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2790–2797, 2009.

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

Maryam Fazel, Haitham Hindi, and Stephen P Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of the 2003 American Control Conference, 2003.*, volume 3, pp. 2156–2162. IEEE, 2003.

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24 (6):381–395, 1981.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Neural Information Processing Systems*, pp. 6151–6159, 2017.

Benjamin D Haeffele, Chong You, and René Vidal. A critique of self-expressive deep subspace clustering. In *The Nineth International Conference on Learning Representations*, 2021.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

Jeffrey Ho, Ming-Husang Yang, Jongwoo Lim, Kuang-Chih Lee, and David Kriegman. Clustering appearances of objects under varying illumination conditions. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 11–18, 2003.

Wei Hong, John Wright, Kun Huang, and Yi Ma. Multiscale hybrid linear models for lossy image representation. *IEEE Transactions on Image Processing*, 15(12):3655–3671, 2006.

Zhizhong Huang, Jie Chen, Junping Zhang, and Hongming Shan. Learning representation for clustering via prototype scattering and positive sampling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45 (6):7509–7524, 2023.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 24–33. Curran Associates, Inc., 2017.

Yuheng Jia, Jianhong Cheng, Hui Liu, and Junhui Hou. Towards calibrated deep clustering network. *CoRR*, abs/2403.02998, 2024.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

José Lezama, Qiang Qiu, Pablo Musé, and Guillermo Sapiro. OLE: Orthogonal low-rank embedding - a plug and play geometric loss for deep learning. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 8109–8118, 2018.

Chun-Guang Li, Chong You, and René Vidal. Structured sparse subspace clustering: A joint affinity learning and subspace clustering framework. *IEEE Transactions on Image Processing*, 26(6): 2988–3001, 2017.

Jun Li, Hongfu Liu, Zhiqiang Tao, Handong Zhao, and Yun Fu. Learnable subspace clustering. *arXiv preprint arXiv:2004.04520*, 2020.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 8547–8555, 2021.

Zengyi Li, Yubei Chen, Yann LeCun, and Friedrich T Sommer. Neural manifold clustering and embedding. *arXiv preprint arXiv:2201.10000*, 2022.

Derek Lim, René Vidal, and Benjamin D Haeffele. Doubly stochastic subspace clustering. *arXiv preprint arXiv:2011.14859*, 2020.

Guangcan Liu, Zhouchen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 663–670, 2010.

Xin Liu, Zhongdao Wang, Ya-Li Li, and Shengjin Wang. Self-supervised learning via maximum entropy coding. *Advances in Neural Information Processing Systems*, 35:34091–34105, 2022.

Canyi 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*, pp. 347–360, 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*, 2018.

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

Yi Ma, Harm Derksen, Wei Hong, and John Wright. Segmentation of multivariate mixed data via lossy coding and compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1546–1562, 2007.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

Ryan McConville, Raul Santos-Rodriguez, Robert J Piechocki, and Ian Craddock. N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding. In *2020 25th international conference on pattern recognition (ICPR)*, pp. 5145–5152. IEEE, 2021.

Brian McWilliams and Giovanni Montana. Subspace clustering of high dimensional data: a predictive approach. *Data Mining and Knowledge Discovery*, 28(3):736–772, 2014.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Chuang Niu, Hongming Shan, and Ge Wang. SPICE: Semantic pseudo-labeling for image clustering. *IEEE Transactions on Image Processing*, 31:7264–7278, 2022.

Foivos Ntelemis, Yaochu Jin, and Spencer A Thomas. Information maximization clustering via multi-view self-labelling. *Knowledge-Based Systems*, 250:109042, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Sungwon Park, Sungwon Han, Sundong Kim, Danu Kim, Sungkyu Park, Seunghoon Hong, and Meeyoung Cha. Improving unsupervised image clustering with robust learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12278–12287, 2021.

Vishal M Patel and René Vidal. Kernel sparse subspace clustering. pp. 2849–2853, 2014.

Vishal M Patel, Hien Van Nguyen, and René Vidal. Latent space sparse subspace clustering. In *Proceedings of the IEEE international conference on computer vision*, pp. 225–232, 2013.

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

Xi Peng, Shijie Xiao, Jiashi Feng, Wei-Yun Yau, and Zhang Yi. Deep subspace clustering with sparsity prior. In *International Joint Conference on Artificial Intelligence*, pp. 1925–1931, 2016.

Xi Peng, Jiashi Feng, Shijie Xiao, Wei-Yun Yau, Joey Tianyi Zhou, and Songfan Yang. Structured autoencoders for subspace clustering. *IEEE Transactions on Image Processing*, 27(10):5076–5086, 2018.

Xi Peng, Jiashi Feng, Joey Tianyi Zhou, Yingjie Lei, and Shuicheng Yan. Deep subspace clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 31(12):5509–5521, 2020.

Daniel Pimentel-Alarcon and Robert Nowak. The information-theoretic requirements of subspace clustering with missing data. In *International Conference on Machine Learning*, pp. 802–810, 2016.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Shankar Rao, Roberto Tron, René Vidal, and Yi Ma. Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. 32(10):1832–1845, 2010.

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

Mahdi Soltanolkotabi and Emmanuel J Candes. A geometric analysis of subspace clustering with outliers. *Annals of Statistics*, 40(4):2195–2238, 2012.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pp. 10268–10278. PMLR, 2021.

Manolis Tsakiris and René Vidal. Algebraic clustering of affine subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Manolis Tsakiris and René Vidal. Theoretical analysis of sparse subspace clustering with missing entries. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4975–4984, 2018.

Paul Tseng. Nearest q-flat to m points. *Journal of Optimization Theory and Applications*, 105: 249–252, 2000.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. SCAN: Learning to classify images without labels. In *European Conference on Computer Vision*, pp. 268–285, 2020.

René Vidal. Identification of PWARX hybrid models with unknown and possibly different orders. pp. 547–552, 2004.

René Vidal, Yi Ma, and Shankar Sastry. Generalized Principal Component Analysis (GPCA). 27 (12):1–15, 2005.

René Vidal, Roberto Tron, and Richard Hartley. Multiframe motion segmentation with missing data using PowerFactorization, and GPCA. 79(1):85–105, 2008.

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

Libin Wang, Yulong Wang, Hao Deng, and Hong Chen. Attention reweighted sparse subspace clustering. *Pattern Recognition*, 139:109438, 2023a.

Peng Wang, Huikang Liu, Druv Pai, Yaodong Yu, Zhihui Zhu, Qing Qu, and Yi Ma. A global geometric analysis of maximal coding rate reduction. In *Forty-first International Conference on Machine Learning*, 2024.

Shiye Wang, Changsheng Li, Yanming Li, Ye Yuan, and Guoren Wang. Self-supervised information bottleneck for deep multi-view subspace clustering. *IEEE Transactions on Image Processing*, 32: 1555–1567, 2023b.

Yu-Xiang Wang and Huan Xu. Noisy sparse subspace clustering. *Journal of Machine Learning Research*, 17(12):1–41, 2016.

Lai Wei, Zhengwei Chen, Jun Yin, Changming Zhu, Rigui Zhou, and Jin Liu. Adaptive graph convolutional subspace clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6262–6271, 2023.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pp. 478–487. PMLR, 2016.

Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5147–5156, 2016.

Chong You, Chun-Guang Li, Daniel Robinson, and René Vidal. Oracle based active set algorithm for scalable elastic net subspace clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3928–3937, 2016a.

Chong You, Daniel Robinson, and René Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3918–3927, 2016b.

Wei Yu, Wonjong Rhee, Stephen Boyd, and John M Cioffi. Iterative water-filling for gaussian vector multiple-access channels. *IEEE Transactions on Information Theory*, 50(1):145–152, 2004.

Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. In *Neural Information Processing Systems (NIPS)*, 2020.

Pengxin Zeng, Yunfan Li, Peng Hu, Dezhong Peng, Jiancheng Lv, and Xi Peng. Deep fair clustering via maximizing and minimizing mutual information: Theory, algorithm and metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23986–23995, 2023.

Hongjing Zhang and Ian Davidson. Deep fair discriminative clustering. *arXiv preprint arXiv:2105.14146*, 2021.

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 Conference on Computer Vision and Pattern Recognition*, pp. 5473–5482, 2019a.

Shangzhi Zhang, Chong You, René Vidal, and Chun-Guang Li. Learning a self-expressive network for subspace clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12393–12403, June 2021.

Teng Zhang, Arthur Szlam, and Gilad Lerman. Median k-flats for hybrid linear modeling with many outliers. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pp. 234–241. IEEE, 2009.

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

Huasong Zhong, Jianlong Wu, Chong Chen, Jianqiang Huang, Minghua Deng, Liqiang Nie, Zhouchen Lin, and Xian-Sheng Hua. Graph contrastive clustering. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9224–9233, 2021.

Pan Zhou, Yunqing Hou, and Jiashi Feng. Deep adversarial subspace clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1596–1604, 2018.

# SUPPLEMENTARY MATERIAL FOR "EXPLORING A PRINCIPLED DEEP SUBSPACE CLUSTERING NETWORK"

The supplementary materials are divided into two parts. In Section A, we present the proofs of our theoretical results. In Section B, we present the supplementary materials for experiments, including experimental details (Sec. B.1), empirical validation on our theoretical results (Sec. B.2), and more experimental results (Sec. B.3).

## A PROOFS OF MAIN RESULTS

**Lemma 1** (Haeffele et al., 2021)**.** *The rows of optimal solution $\boldsymbol{Z}$ for problem (2) are eigenvectors that associate with the smallest eigenvalues of $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$.*

*Proof.* We note that:

$$\|\boldsymbol{Z}-\boldsymbol{Z}\boldsymbol{C}\|_F^2 = \operatorname{Tr}\left(\boldsymbol{Z}\left(\boldsymbol{I}-\boldsymbol{C}\right)\left(\boldsymbol{I}-\boldsymbol{C}\right)^\top \boldsymbol{Z}^\top\right) = \sum_{i=1}^{d} \boldsymbol{z}^{(i)}(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top \boldsymbol{z}^{(i)\top},$$

where $\boldsymbol{z}^{(i)}$ is the $i^{\text{th}}$ row of $\boldsymbol{Z}$, thus problem (2) is equivalent to:

$$\begin{aligned} \min_{\{\boldsymbol{z}^{(i)}\}_{i=1}^d, \boldsymbol{C}} \quad & \frac{1}{2}\sum_{i=1}^{d} \boldsymbol{z}^{(i)}(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top \boldsymbol{z}^{(i)\top} + \beta \cdot r(\boldsymbol{C}) \\ \text{s.t.} \quad & \|\boldsymbol{Z}\|_F^2 = N. \end{aligned} \tag{13}$$

Without loss of generality, we fix the magnitude of each row of $\boldsymbol{Z}$ to $\|\boldsymbol{z}^{(i)}\|_2^2 = \tau_i,\ i=1,\dots,d$, where $\sum_{i=1}^d \tau_i = N$. Then, the optimization problem becomes:

$$\begin{aligned} \min_{\{\boldsymbol{z}^{(i)}\}_{i=1}^d, \boldsymbol{C}} \quad & \frac{1}{2}\sum_{i=1}^{d} \boldsymbol{z}^{(i)}(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top \boldsymbol{z}^{(i)\top} + \beta \cdot r(\boldsymbol{C}) \\ \text{s.t.} \quad & \|\boldsymbol{z}^{(i)}\|_2^2 = \tau_i, \ \ i=1,\dots,d. \end{aligned} \tag{14}$$

The Lagrangian of problem (14) is:

$$\mathcal{L}(\{\boldsymbol{z}^{(i)}\}_{i=1}^d, \boldsymbol{C}, \{\nu_i\}_{i=1}^d) := \frac{1}{2}\sum_{i=1}^{d} \boldsymbol{z}^{(i)}(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top \boldsymbol{z}^{(i)\top} + \beta \cdot r(\boldsymbol{C}) + \frac{1}{2}\sum_{i=1}^{d} \nu_i(\|\boldsymbol{z}^{(i)}\|_2^2 - \tau_i), \tag{15}$$

where $\{\nu_i\}_{i=1}^d$ are the Lagrangian multipliers. The necessary conditions for optimal solution are:

$$\begin{cases} \nabla_{\boldsymbol{z}^{(i)}}\mathcal{L} = \boldsymbol{z}^{(i)}(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top + \nu_i \boldsymbol{z}^{(i)} = \boldsymbol{0}, \\ \|\boldsymbol{z}^{(i)}\|_2^2 = \tau_i, \ \ i=1,\dots,d, \end{cases} \tag{16}$$

which implies that the optimal solutions $\boldsymbol{z}^{(i)}$ are eigenvectors of $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$.

By further considering the objective functions, the optimal $\boldsymbol{z}^{(i)}$ should be eigenvectors w.r.t. the *smallest* eigenvalues of $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$ for all $i \in \{1,\dots,d\}$. The corresponding optimal value is $\frac{1}{2}\lambda_{min}((\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top)\sum_{i=1}^d \tau_i + \beta \cdot r(\boldsymbol{C}) = \frac{N}{2}\lambda_{min}((\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top) + \beta \cdot r(\boldsymbol{C})$, which is irrelevant to $\{\tau_i\}_{i=1}^d$.

Therefore, we conclude that the rows of optimal solution $\boldsymbol{Z}$ to problem (2) are eigenvectors that associate with the smallest eigenvalues of $(\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$.

□

We first present two lemmas, which will be used for the proof of Theorem 1.

**Lemma A1.** *Suppose that $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$ are symmetric matrices, then $\boldsymbol{AB} = \boldsymbol{BA}$ if and only if $\boldsymbol{A}$ and $\boldsymbol{B}$ can be diagonalized simultaneously by $\boldsymbol{U} \in \mathcal{O}^{n\times n}$.*

**Lemma A2** (Tian et al., 2021)**.** *Let $\boldsymbol{H}(t) \succ \boldsymbol{0}$ be a time-varying positive definite matrix whose minimal eigenvalue is bounded away from 0, i.e., $\inf_{t\geq 0} \lambda_{min}(\boldsymbol{H}(t)) \geq \lambda_0 > 0$. Then, the following dynamics*

$$\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t} = -\boldsymbol{H}(t)\boldsymbol{w}(t)$$

*satisfies that $\|\boldsymbol{w}(t)\|_2 \leq e^{-\lambda_0 t}\|\boldsymbol{w}(0)\|_2$, i.e., $\boldsymbol{w}(t) \to \boldsymbol{0}$.*

**Theorem 1.** *Consider the objective function (5) where $\boldsymbol{Z}$ and $\boldsymbol{C}$ are optimized by gradient descent with a learning rate $\eta \to 0$, denote $\boldsymbol{G} \coloneqq \boldsymbol{Z}^\top\boldsymbol{Z}, \boldsymbol{M} \coloneqq (\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top, \boldsymbol{F} \coloneqq \alpha(\boldsymbol{I}+\alpha\boldsymbol{G})^{-1}, \hat{\boldsymbol{M}} \coloneqq \gamma\boldsymbol{M} + \nu_\star\boldsymbol{I}$. If $\lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) < 0$, and $\frac{8\alpha^2}{27} < \gamma < \frac{\alpha-\nu_\star}{\lambda_{max}(\boldsymbol{M})}$, then $\boldsymbol{G}$ and $\boldsymbol{M}$ will converge to have eigenspaces aligned, i.e., $\boldsymbol{G}$ and $\boldsymbol{M}$ can be diagonalized simultaneously by $\boldsymbol{U} \in \mathcal{O}^{N\times N}$ where $\mathcal{O}^{N\times N}$ is an $N \times N$ orthogonal matrix group.*

*Proof.* As $\boldsymbol{F} \coloneqq \alpha(\boldsymbol{I}+\alpha\boldsymbol{G})^{-1}, \hat{\boldsymbol{M}} \coloneqq \gamma\boldsymbol{M} + \nu_\star\boldsymbol{I}$, we have $\boldsymbol{G} = \boldsymbol{Z}^\top\boldsymbol{Z} = \boldsymbol{F}^{-1} - \frac{1}{\alpha}\boldsymbol{I}$ and $\boldsymbol{M} = \frac{\hat{\boldsymbol{M}}-\nu_\star\boldsymbol{I}}{\gamma}$. For simplicity, we denote $\boldsymbol{H} \coloneqq \boldsymbol{F} - \hat{\boldsymbol{M}} = -\gamma\boldsymbol{M} + \boldsymbol{F} - \nu_\star\boldsymbol{I}$.

As $\boldsymbol{Z}$ and $\boldsymbol{C}$ are optimized by gradient descent with a learning rate $\eta \to 0$, we have the learning dynamic:

$$\dot{\boldsymbol{Z}} = \frac{\mathrm{d}\boldsymbol{Z}}{\mathrm{d}t} = -\nabla_{\boldsymbol{Z}}\mathcal{L} = -\gamma\boldsymbol{Z}\boldsymbol{M} + \alpha\boldsymbol{Z}(\boldsymbol{I}+\alpha\boldsymbol{G})^{-1} - \nu_\star\boldsymbol{Z} = \boldsymbol{Z}\boldsymbol{H}, \tag{17}$$

$$\dot{\boldsymbol{C}} = \frac{\mathrm{d}\boldsymbol{C}}{\mathrm{d}t} = -\nabla_{\boldsymbol{C}}\mathcal{L} = \gamma\boldsymbol{G}(\boldsymbol{I}-\boldsymbol{C}) = \gamma(\boldsymbol{F}^{-1} - \frac{1}{\alpha}\boldsymbol{I})(\boldsymbol{I}-\boldsymbol{C}). \tag{18}$$

As $\boldsymbol{F}$ and $\hat{\boldsymbol{M}}$ are functions of $\boldsymbol{Z}$ and $\boldsymbol{C}$, respectively, they are updated by:

$$\begin{aligned}
\dot{\boldsymbol{F}} = \frac{\mathrm{d}\boldsymbol{F}}{\mathrm{d}t} &= \frac{\mathrm{d}}{\mathrm{d}t}\Big[\alpha(\boldsymbol{I}+\alpha\boldsymbol{Z}^\top\boldsymbol{Z})^{-1}\Big] \\
&= -\alpha(\boldsymbol{I}+\alpha\boldsymbol{G})^{-1}\frac{\mathrm{d}}{\mathrm{d}t}\Big(\boldsymbol{Z}^\top\boldsymbol{Z}\Big)\alpha(\boldsymbol{I}+\alpha\boldsymbol{G})^{-1} \\
&= -\boldsymbol{F}\Big(\dot{\boldsymbol{Z}}^\top\boldsymbol{Z} + \boldsymbol{Z}^\top\dot{\boldsymbol{Z}}\Big)\boldsymbol{F} \\
&= -\boldsymbol{F}\Big(\boldsymbol{H}\boldsymbol{G} + \boldsymbol{G}\boldsymbol{H}\Big)\boldsymbol{F} \\
&= -\boldsymbol{F}\Big(\boldsymbol{H}\boldsymbol{F}^{-1} + \boldsymbol{F}^{-1}\boldsymbol{H} - \frac{2}{\alpha}\boldsymbol{H}\Big)\boldsymbol{F} \\
&= \Big(\boldsymbol{F}\hat{\boldsymbol{M}} + \hat{\boldsymbol{M}}\boldsymbol{F}\Big) - 2\Big(\boldsymbol{F}^2 + \frac{1}{\alpha}\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F} - \frac{1}{\alpha}\boldsymbol{F}^3\Big),
\end{aligned} \tag{19}$$

$$\begin{aligned}
\dot{\hat{\boldsymbol{M}}} = \frac{\mathrm{d}(\gamma\boldsymbol{M}+\nu_\star\boldsymbol{I})}{\mathrm{d}t} = \gamma\frac{\mathrm{d}\boldsymbol{M}}{\mathrm{d}t} &= \gamma(-\dot{\boldsymbol{C}})(\boldsymbol{I}-\boldsymbol{C})^\top + \gamma(\boldsymbol{I}-\boldsymbol{C})(-\dot{\boldsymbol{C}})^\top \\
&= -\gamma^2\Big[(\boldsymbol{F}^{-1} - \frac{1}{\alpha}\boldsymbol{I})\boldsymbol{M} + \boldsymbol{M}(\boldsymbol{F}^{-1} - \frac{1}{\alpha}\boldsymbol{I})\Big] \\
&= -\gamma^2\Big(\boldsymbol{F}^{-1}\boldsymbol{M} + \boldsymbol{M}\boldsymbol{F}^{-1} - \frac{2}{\alpha}\boldsymbol{M}\Big) \\
&= -\gamma\Big[(\boldsymbol{F}^{-1}\hat{\boldsymbol{M}} + \hat{\boldsymbol{M}}\boldsymbol{F}^{-1}) - 2(\nu_\star\boldsymbol{F}^{-1} + \frac{\hat{\boldsymbol{M}}}{\alpha} - \frac{\nu_\star}{\alpha}\boldsymbol{I})\Big].
\end{aligned} \tag{20}$$

Next, we denote $\boldsymbol{L}$ as the commutator $[\boldsymbol{F}, \hat{\boldsymbol{M}}]$

$$\boldsymbol{L} \coloneqq [\boldsymbol{F}, \hat{\boldsymbol{M}}] = \boldsymbol{F}\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}, \tag{21}$$

and we notice that:

$$\begin{aligned}
\boldsymbol{F}\hat{\boldsymbol{M}}^2 - \hat{\boldsymbol{M}}^2\boldsymbol{F} &= \hat{\boldsymbol{M}}\boldsymbol{L} + \boldsymbol{L}\hat{\boldsymbol{M}}, \\
\boldsymbol{F}^2\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}^2 &= \boldsymbol{F}\boldsymbol{L} + \boldsymbol{L}\boldsymbol{F}, \\
\boldsymbol{F}^3\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}^3 + \hat{\boldsymbol{M}}\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F} - \boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F}\hat{\boldsymbol{M}} &= (\boldsymbol{F}^2 - \hat{\boldsymbol{M}}\boldsymbol{F})\boldsymbol{L} + \boldsymbol{L}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) + \boldsymbol{F}\boldsymbol{L}\boldsymbol{F}, \\
\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F}^{-1} - \boldsymbol{F}^{-1}\hat{\boldsymbol{M}}\boldsymbol{F} &= \boldsymbol{F}^{-1}\boldsymbol{L} + \boldsymbol{L}\boldsymbol{F}^{-1}.
\end{aligned}$$

We compute the dynamic of $\boldsymbol{L}$ as:

$$\dot{\boldsymbol{L}} = \frac{\mathrm{d}\boldsymbol{L}}{\mathrm{d}t} = \underbrace{\Big(\dot{\boldsymbol{F}}\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\dot{\boldsymbol{F}}\Big)}_{\boldsymbol{L}_1} + \underbrace{\Big(\boldsymbol{F}\dot{\hat{\boldsymbol{M}}} - \dot{\hat{\boldsymbol{M}}}\boldsymbol{F}\Big)}_{\boldsymbol{L}_2}, \text{ where} \tag{22}$$

$$\begin{aligned}
\boldsymbol{L}_1 &= (\boldsymbol{F}\hat{\boldsymbol{M}} + \hat{\boldsymbol{M}}\boldsymbol{F})\hat{\boldsymbol{M}} - 2(\boldsymbol{F}^2 + \frac{1}{\alpha}\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F} - \frac{1}{\alpha}\boldsymbol{F}^3)\hat{\boldsymbol{M}} \\
&\qquad - \hat{\boldsymbol{M}}(\boldsymbol{F}\hat{\boldsymbol{M}} + \hat{\boldsymbol{M}}\boldsymbol{F}) + 2\hat{\boldsymbol{M}}(\boldsymbol{F}^2 + \frac{1}{\alpha}\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F} - \frac{1}{\alpha}\boldsymbol{F}^3) \\
&= (\boldsymbol{F}\hat{\boldsymbol{M}}^2 - \hat{\boldsymbol{M}}^2\boldsymbol{F}) - 2(\boldsymbol{F}^2\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}^2) + \frac{2}{\alpha}\Big[(\boldsymbol{F}^3\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}^3) - (\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F}\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F})\Big] \\
&= (\hat{\boldsymbol{M}}\boldsymbol{L} + \boldsymbol{L}\hat{\boldsymbol{M}}) - 2(\boldsymbol{F}\boldsymbol{L} + \boldsymbol{L}\boldsymbol{F}) + \frac{2}{\alpha}\Big[(\boldsymbol{F}^2 - \hat{\boldsymbol{M}}\boldsymbol{F})\boldsymbol{L} + \boldsymbol{L}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) + \boldsymbol{F}\boldsymbol{L}\boldsymbol{F}\Big],
\end{aligned} \tag{23}$$

$$\begin{aligned}
\boldsymbol{L}_2 &= -\gamma\boldsymbol{F}\Big[(\boldsymbol{F}^{-1}\hat{\boldsymbol{M}} + \hat{\boldsymbol{M}}\boldsymbol{F}^{-1}) - 2(\nu_\star\boldsymbol{F}^{-1} + \frac{\hat{\boldsymbol{M}}}{\alpha} - \frac{\nu_\star}{\alpha}\boldsymbol{I})\Big] \\
&\qquad + \gamma\Big[(\boldsymbol{F}^{-1}\hat{\boldsymbol{M}} + \hat{\boldsymbol{M}}\boldsymbol{F}^{-1}) - 2(\nu_\star\boldsymbol{F}^{-1} + \frac{\hat{\boldsymbol{M}}}{\alpha} - \frac{\nu_\star}{\alpha}\boldsymbol{I})\Big]\boldsymbol{F} \\
&= \gamma(\boldsymbol{F}^{-1}\hat{\boldsymbol{M}}\boldsymbol{F} - \boldsymbol{F}\hat{\boldsymbol{M}}\boldsymbol{F}^{-1}) + \frac{2\gamma}{\alpha}(\boldsymbol{F}\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F}) \\
&= -\gamma(\boldsymbol{F}^{-1}\boldsymbol{L} + \boldsymbol{L}\boldsymbol{F}^{-1}) + \frac{2\gamma}{\alpha}\boldsymbol{L}.
\end{aligned} \tag{24}$$

Therefore, we vectorize $\dot{\boldsymbol{L}}$ by Kronecker product:

$$\mathrm{Vec}(\dot{\boldsymbol{L}}) = \mathrm{Vec}(\boldsymbol{L}_1) + \mathrm{Vec}(\boldsymbol{L}_2) = -(\boldsymbol{K}_1 + \boldsymbol{K}_2)\mathrm{Vec}(\boldsymbol{L}), \tag{25}$$

$$\text{where } \boldsymbol{K}_1 \coloneqq -\hat{\boldsymbol{M}} \oplus \hat{\boldsymbol{M}} + 2\boldsymbol{F} \oplus \boldsymbol{F} - \frac{2}{\alpha}\Big[(\boldsymbol{F}^2 - \hat{\boldsymbol{M}}\boldsymbol{F}) \oplus (\boldsymbol{F}^2 - \hat{\boldsymbol{M}}\boldsymbol{F}) + \boldsymbol{F} \otimes \boldsymbol{F}\Big], \tag{26}$$

$$\boldsymbol{K}_2 \coloneqq \gamma\boldsymbol{F}^{-1} \oplus \boldsymbol{F}^{-1} - \frac{2\gamma}{\alpha}\boldsymbol{I}, \text{ and denote } \boldsymbol{W}_1 \oplus \boldsymbol{W}_2 \coloneqq \boldsymbol{W}_1 \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{W}_2. \tag{27}$$

The smallest eigenvalue of $\boldsymbol{K} \coloneqq \boldsymbol{K}_1 + \boldsymbol{K}_2$ satisfies:

$$\lambda_{min}(\boldsymbol{K}) \geq -2\lambda_{max}(\hat{\boldsymbol{M}}) + 4\lambda_{min}(\boldsymbol{F} - \frac{1}{2\alpha}\boldsymbol{F}^2 + \frac{\gamma}{2}\boldsymbol{F}^{-1}) - \frac{4}{\alpha}\lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) - \frac{2\gamma}{\alpha}. \tag{28}$$

Given that $\lambda(\boldsymbol{F}) = \lambda(\alpha(\boldsymbol{I} + \alpha\boldsymbol{G})^{-1}) = \frac{\alpha}{1+\alpha\lambda(\boldsymbol{G})} \in (0, \alpha]$, we denote $f(\lambda) \coloneqq \lambda - \frac{\lambda^2}{2\alpha} + \frac{\gamma}{2\lambda}, \lambda \in (0, \alpha]$, and its derivative $f'(\lambda) = \frac{1}{\lambda^2}(2\lambda^2 - \frac{2}{\alpha}\lambda^3 - \gamma)$. We notice that when $\gamma > \frac{8\alpha^2}{27}$, $f'(\lambda) < 0$ for all $\lambda$ and $\alpha$, which means $f(\lambda)$ is monotonically decreasing in $\lambda \in [0, \alpha]$. Therefore,

$$\lambda_{min}(\boldsymbol{K}) \geq 2\Big[-\lambda_{max}(\hat{\boldsymbol{M}}) + 2\lambda_{max}(\boldsymbol{F}) - \frac{1}{\alpha}\lambda_{max}^2(\boldsymbol{F}) + \frac{\gamma}{\lambda_{max}(\boldsymbol{F})} - \frac{2}{\alpha}\lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) - \frac{\gamma}{\alpha}\Big]. \tag{29}$$

Given that $\lambda_{max}(\hat{\boldsymbol{M}}) = \lambda_{max}(\gamma\boldsymbol{M} + \nu_\star\boldsymbol{I}) = \gamma\lambda_{max}(\boldsymbol{M}) + \nu_\star$ , we have:

$$\begin{aligned}
\lambda_{min}(\boldsymbol{K}) &\geq 2\Big[-\lambda_{max}(\hat{\boldsymbol{M}}) + 2\lambda_{max}(\boldsymbol{F}) - \frac{1}{\alpha}\lambda_{max}^2(\boldsymbol{F}) + \frac{\gamma}{\lambda_{max}(\boldsymbol{F})} - \frac{2}{\alpha}\lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) - \frac{\gamma}{\alpha}\Big] \\
&= 2\Big[-\gamma\lambda_{max}(\boldsymbol{M}) - \nu_\star + 2\alpha - \alpha + \frac{\gamma}{\alpha} - \frac{\gamma}{\alpha} - \frac{2}{\alpha}\lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}})\Big] \\
&= 2\Big[-\gamma\lambda_{max}(\boldsymbol{M}) - \nu_\star + \alpha - \frac{2}{\alpha}\lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}})\Big].
\end{aligned} \tag{30}$$

Given that $\nu_\star < \alpha - \gamma\lambda_{max}(\boldsymbol{M}), \lambda_{max}(\boldsymbol{F}^2 - \boldsymbol{F}\hat{\boldsymbol{M}}) < 0$, we have $\lambda_{min}(\boldsymbol{K}) > 0$.

Therefore, we have:

$$\mathrm{Vec}(\dot{\boldsymbol{L}}) = \frac{\mathrm{dVec}(\boldsymbol{L})}{\mathrm{d}t} = -\boldsymbol{K}\mathrm{Vec}(\boldsymbol{L}), \text{ where } \boldsymbol{K} \succ \boldsymbol{0}. \tag{31}$$

According to Lemma A2, this implies:

$$\lim_{t\to\infty} \boldsymbol{L} = \lim_{t\to\infty} \boldsymbol{F}\hat{\boldsymbol{M}} - \hat{\boldsymbol{M}}\boldsymbol{F} = \boldsymbol{0}. \tag{32}$$

By Lemma A1, we find that $\boldsymbol{F} = \alpha(\boldsymbol{I} + \alpha\boldsymbol{G})^{-1}$ and $\hat{\boldsymbol{M}} = \gamma\boldsymbol{M} + \nu_\star\boldsymbol{I}$ can be diagonalized simultaneously by $\boldsymbol{U} \in \mathcal{O}^{N\times N}$. At the same time, $\boldsymbol{F}$ with $\boldsymbol{G}$, $\hat{\boldsymbol{M}}$ with $\boldsymbol{M}$ share an eigenspace, which means that $\boldsymbol{G} = \boldsymbol{Z}^\top\boldsymbol{Z}$ and $\boldsymbol{M} = (\boldsymbol{I} - \boldsymbol{C})(\boldsymbol{I} - \boldsymbol{C})^\top$ can be diagonalized simultaneously by $\boldsymbol{U} \in \mathcal{O}^{N\times N}$.

□

To justify the fact that actually we can ensure that $\nu_\star$ satisfies the condition $\gamma < \frac{1}{\lambda_{max}(\boldsymbol{M}_\star)}(\alpha - \nu_\star)$ by adjusting the hyper-parameters $\alpha$ and $\gamma$, we will derive the optimal Lagrangian multiplier $\nu_\star$ from the optimality condition of the problem in Eq.(5).

We begin with analyzing the KKT condition of the problem in Eq.(5):

$$\begin{cases} \nabla_{\boldsymbol{Z}}\mathcal{L}(\boldsymbol{Z}_\star, \boldsymbol{C}_\star, \nu_\star) = \gamma\boldsymbol{Z}_\star\boldsymbol{M}_\star - \alpha\boldsymbol{Z}_\star(\boldsymbol{I} + \alpha\boldsymbol{G}_\star)^{-1} + \nu_\star\boldsymbol{Z}_\star = \boldsymbol{0}, & (33)\\ \nabla_{\boldsymbol{C}}\mathcal{L}(\boldsymbol{Z}_\star, \boldsymbol{C}_\star, \nu_\star) = -\gamma\boldsymbol{G}_\star(\boldsymbol{I} - \boldsymbol{C}_\star) = \boldsymbol{0}, & (34)\\ \nabla_{\nu}\mathcal{L}(\boldsymbol{Z}_\star, \boldsymbol{C}_\star, \nu_\star) = \|\boldsymbol{Z}_\star\|_F^2 - N = \mathrm{Tr}(\boldsymbol{G}_\star) - N = 0, & (35)\end{cases}$$

where $\boldsymbol{G}_\star = \boldsymbol{Z}_\star{}^\top\boldsymbol{Z}_\star$ and $\boldsymbol{M}_\star = (\boldsymbol{I} - \boldsymbol{C}_\star)(\boldsymbol{I} - \boldsymbol{C}_\star)^\top$. The condition can be rewritten as:

$$\begin{cases} \gamma\boldsymbol{G}_\star\boldsymbol{M}_\star - \alpha\boldsymbol{G}_\star(\boldsymbol{I} + \alpha\boldsymbol{G}_\star)^{-1} + \nu_\star\boldsymbol{G}_\star = \boldsymbol{0} & (36)\\ \gamma\boldsymbol{G}_\star\boldsymbol{M}_\star = \boldsymbol{0} & (37)\\ \mathrm{Tr}(\boldsymbol{G}_\star) = N & (38)\end{cases}$$

Then, we add the trace operator to (36) and simplify the formula by (37) as:

$$\mathrm{Tr}\left(\nu\boldsymbol{G}_\star - \alpha\boldsymbol{G}_\star(\boldsymbol{I} + \alpha\boldsymbol{G}_\star)^{-1}\right) = 0 \tag{39}$$

Since $\alpha\boldsymbol{G}_\star(\boldsymbol{I} + \alpha\boldsymbol{G}_\star)^{-1} = \boldsymbol{I} - (\boldsymbol{I} + \alpha\boldsymbol{G}_\star)^{-1}$ and $\boldsymbol{Z} \in \mathbb{R}^{d\times N}$, we discuss two cases between $N$ and $d$.

**For case 1.** Suppose $N > d$, then (39) will be rewritten as:

$$N\nu_\star = \mathrm{Tr}\left(\boldsymbol{I} - (\boldsymbol{I} + \alpha\boldsymbol{G}_\star)^{-1}\right) \tag{40}$$

$$\iff N\nu_\star = N - (N - d) - \sum_{i=1}^{d}\frac{1}{1 + \alpha\lambda_{\boldsymbol{G}_\star}^{(i)}}$$

$$\iff \nu_\star = \frac{d}{N} - \frac{1}{N}\sum_{i=1}^{d}\frac{1}{1 + \alpha\lambda_{\boldsymbol{G}_\star}^{(i)}}, \tag{41}$$

where $\lambda_{\boldsymbol{G}_\star}^{(i)} \geq 0, i = 1, \cdots, d,\ \lambda_{\boldsymbol{G}_\star}^{(i)} = 0, i = d+1, \cdots, N$ and $\sum_{i=1}^{d}\lambda_{\boldsymbol{G}_\star}^{(i)} = N$. To ensure that $\nu_\star$ satisfies the condition of Theorem 1, i.e., $\nu_\star < \alpha - \gamma\lambda_{max}(\boldsymbol{M})$ for any $\{\lambda_{\boldsymbol{G}_\star}^{(i)}\}_{i=1}^{N}$, we have:

$$\nu_\star = \frac{d}{N} - \frac{1}{N}\sum_{i=1}^{d}\frac{1}{1 + \alpha\lambda_{\boldsymbol{G}_\star}^{(i)}} \leq \frac{\alpha}{1 + \alpha\cdot\frac{N}{d}} < \alpha - \gamma\lambda_{max}(\boldsymbol{M}), \tag{42}$$

where the equality holds when $\alpha > 0, \lambda_{\boldsymbol{G}_\star}^{(1)} = \cdots = \lambda_{\boldsymbol{G}_\star}^{(d)} = \frac{N}{d}$. Therefore, $\nu_\star$ will satisfy the condition in Theorem 1 if the last inequality holds, which is equivalent to:

$$\gamma\lambda_{max}(\boldsymbol{M}) < \alpha - \frac{\alpha}{1 + \alpha\cdot\frac{N}{d}} = \frac{\alpha^2}{\frac{d}{N} + \alpha}. \tag{43}$$

**For case 2.** Suppose $d \geq N$, then with the similar process, we estimate the upper bound of $\nu_\star$ as:

$$\nu_\star = 1 - \frac{1}{N}\sum_{i=1}^{N}\frac{1}{1 + \alpha\lambda_{\boldsymbol{G}_\star}^{(i)}} \leq \frac{\alpha}{1 + \alpha} < \alpha - \gamma\lambda_{max}(\boldsymbol{M}) \tag{44}$$

where the equality holds when $\alpha > 0, \lambda_{\boldsymbol{G}_\star}^{(1)} = \cdots = \lambda_{\boldsymbol{G}_\star}^{(N)} = 1$. Therefore, $\nu_\star$ will satisfy the condition in Theorem 2 if the last inequality holds, which is equivalent to:

$$\gamma\lambda_{max}(\boldsymbol{M}) < \alpha - \frac{\alpha}{1+\alpha} = \frac{\alpha^2}{1+\alpha}. \tag{45}$$

This means that when $\gamma$ and $\alpha$ satisfy Eq. (43) or (45) in each cases, the optimal $\nu_\star$ will satisfy Theorem 2's condition. Therefore, we may not need to concern whether $\nu_\star$ satisfies the condition, but focus on the hyper-parameters $\gamma$ and $\alpha$.

**Theorem 2.** *Suppose that $\boldsymbol{G}$ and $\boldsymbol{M}$ have eigenspaces aligned and $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$, then for the optimal solution $\boldsymbol{Z}_\star$, we have that $\operatorname{rank}(\boldsymbol{Z}) = \min\{d, N\}$ and the singular values $\sigma_{\boldsymbol{Z}_\star}^{(i)} = \sqrt{\frac{1}{\gamma\lambda_{\boldsymbol{M}}^{(i)}+\nu_\star} - \frac{1}{\alpha}}$, for all $i = 1, \ldots, \min\{d, N\}$, where $\nu_\star$ is the dual optimal solution.*

*Proof.* Since $\|\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{C}\|_F^2 = \operatorname{Tr}\left(\boldsymbol{Z}^\top\boldsymbol{Z}\left(\boldsymbol{I}-\boldsymbol{C}\right)\left(\boldsymbol{I}-\boldsymbol{C}\right)^\top\right)$ and $\|\boldsymbol{Z}\|_F^2 = \operatorname{Tr}(\boldsymbol{Z}^\top\boldsymbol{Z})$, problem (6) is equivalent to:

$$\begin{aligned} \min_{\boldsymbol{G}} \quad & -\frac{1}{2}\log\det\left(\boldsymbol{I}+\alpha\boldsymbol{G}\right) + \frac{\gamma}{2}\operatorname{Tr}(\boldsymbol{G}\boldsymbol{M}) \\ \text{s.t.} \quad & \operatorname{Tr}(\boldsymbol{G}) = N, \boldsymbol{G} \succeq \boldsymbol{0}, \end{aligned} \tag{46}$$

where $\boldsymbol{G} \coloneqq \boldsymbol{Z}^\top\boldsymbol{Z}$, $\boldsymbol{M} \coloneqq (\boldsymbol{I}-\boldsymbol{C})(\boldsymbol{I}-\boldsymbol{C})^\top$.

Since that $\boldsymbol{G}$ and $\boldsymbol{M}$ have eigenspaces aligned, we have $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Lambda}_{\boldsymbol{G}}\boldsymbol{U}^\top, \boldsymbol{M} = \boldsymbol{U}\boldsymbol{\Lambda}_{\boldsymbol{M}}\boldsymbol{U}^\top$. Therefore, problem (46) is transformed into:

$$\begin{aligned} \min_{\boldsymbol{\Lambda}_{\boldsymbol{G}}} \quad & -\frac{1}{2}\operatorname{Tr}\log(\boldsymbol{I}+\alpha\boldsymbol{\Lambda}_{\boldsymbol{G}}) + \frac{\gamma}{2}\operatorname{Tr}(\boldsymbol{\Lambda}_{\boldsymbol{G}}\boldsymbol{\Lambda}_{\boldsymbol{M}}) \\ \text{s.t.} \quad & \operatorname{Tr}(\boldsymbol{\Lambda}_{\boldsymbol{G}}) = N, \boldsymbol{\Lambda}_{\boldsymbol{G}} \succeq \boldsymbol{0}, \end{aligned} \tag{47}$$

which is further equivalent to the eigenvalue optimization problem below:

$$\begin{aligned} \min_{\{\lambda_{\boldsymbol{G}}^{(i)}\}_{i=1}^{\min\{d,N\}}} \quad & -\frac{1}{2}\sum_{i=1}^{\min\{d,N\}}\log(1+\alpha\lambda_{\boldsymbol{G}}^{(i)}) + \frac{\gamma}{2}\lambda_{\boldsymbol{M}}^{(i)}\lambda_{\boldsymbol{G}}^{(i)} \\ \text{s.t.} \quad & \sum_{i=1}^{\min\{d,N\}}\lambda_{\boldsymbol{G}}^{(i)} = N, \\ & \lambda_{\boldsymbol{G}}^{(i)} \geq 0, \quad \text{for all } i = 1, \ldots, \min\{d, N\}. \end{aligned} \tag{48}$$

It is noteworthy that problem (48) is a convex problem. Thus, the KKT condition is the sufficient and necessary condition for the minimizer.

The Lagrangian of problem (48) is:

$$\begin{aligned} & \mathcal{L}\Big(\{\lambda_{\boldsymbol{G}}^{(i)}\}_{i=1}^{\min\{d,N\}}, \{\mu_i\}_{i=1}^{\min\{d,N\}}, \nu\Big) \coloneqq \\ & -\frac{1}{2}\sum_{i=1}^{\min\{d,N\}}\log(1+\alpha\lambda_{\boldsymbol{G}}^{(i)}) + \frac{\gamma}{2}\lambda_{\boldsymbol{M}}^{(i)}\lambda_{\boldsymbol{G}}^{(i)} - \mu_i\lambda_{\boldsymbol{G}}^{(i)} + \frac{\nu}{2}\Big(\sum_{i=1}^{\min\{d,N\}}\lambda_{\boldsymbol{G}}^{(i)} - N\Big), \end{aligned} \tag{49}$$

where $\mu_i \geq 0, i = 1, \ldots, \min\{d, N\}$ and $\nu$ are the Lagrangian multipliers. The KKT conditions are as follows:

$$\begin{cases} \nabla_{\lambda_{\boldsymbol{G}_\star}^{(i)}}\mathcal{L} = 0, & \forall i = 1, \ldots, \min\{d, N\}, \quad (50) \\ \lambda_{\boldsymbol{G}_\star}^{(i)} \geq 0, & \forall i = 1, \ldots, \min\{d, N\}, \quad (51) \\ \sum_{i=1}^{\min\{d,N\}}\lambda_{\boldsymbol{G}_\star}^{(i)} = N, & \quad (52) \\ \mu_{i\star} \geq 0, & \forall i = 1, \ldots, \min\{d, N\}, \quad (53) \\ \mu_{i\star}\lambda_{\boldsymbol{G}_\star}^{(i)} = 0, & \forall i = 1, \ldots, \min\{d, N\}. \quad (54) \end{cases}$$

Then, (50) is equivalent to:

$$\mu_{i\star} = \frac{1}{2}\Big(\nu_\star + \gamma\lambda_{\boldsymbol{M}}^{(i)} - \frac{\alpha}{1+\alpha\lambda_{\boldsymbol{G}_\star}^{(i)}}\Big). \tag{55}$$

By Eqs. (51) and (53-54)(55), we come up with the following two cases:

$$\begin{cases} \mu_{i\star} > 0 \Rightarrow \lambda_{\boldsymbol{G}_\star}^{(i)} = 0, \dfrac{1}{\nu_\star + \gamma\lambda_{\boldsymbol{M}}^{(i)}} - \dfrac{1}{\alpha} < 0, & (56) \\ \mu_{i\star} = 0 \Rightarrow \lambda_{\boldsymbol{G}_\star}^{(i)} > 0, \lambda_{\boldsymbol{G}_\star}^{(i)} = \dfrac{1}{\nu_\star + \gamma\lambda_{\boldsymbol{M}}^{(i)}} - \dfrac{1}{\alpha} > 0. & (57) \end{cases}$$

From the above two cases, we conclude that:

$$\lambda_{\boldsymbol{G}_\star}^{(i)} = \max\Big\{0, \frac{1}{\nu_\star + \gamma\lambda_{\boldsymbol{M}}^{(i)}} - \frac{1}{\alpha}\Big\}, \tag{58}$$

where $\nu_\star$ satisfies:

$$\sum_{i=1}^{\min\{d,N\}} \max\Big\{0, \frac{1}{\nu_\star + \gamma\lambda_{\boldsymbol{M}}^{(i)}} - \frac{1}{\alpha}\Big\} = N. \tag{59}$$

Given that $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$, we have $\frac{1}{\nu_\star + \gamma\lambda_{\boldsymbol{M}}^{(i)}} - \frac{1}{\alpha} > 0$ for all $i = 1, \ldots, \min\{d, N\}$. Therefore, for the optimal $\boldsymbol{Z}_\star$ in problem (6), we have $\operatorname{rank}(\boldsymbol{Z}_\star) = \min\{d, N\}$ and the singular values $\sigma_{\boldsymbol{Z}_\star}^{(i)} = \sqrt{\frac{1}{\gamma\lambda_{\boldsymbol{M}}^{(i)} + \nu_\star} - \frac{1}{\alpha}}$, for all $i = 1, \ldots, \min\{d, N\}$.

**Remark 3.** We notice that (48) is a reverse water-filling problem, where the water level is controlled by $1/\alpha$, as shown in Figure A.1. When $\boldsymbol{G}$ and $\boldsymbol{M}$ have eigenspaces aligned and $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$, we have $\operatorname{rank}(\boldsymbol{Z}_\star) = \min\{d, N\}$ and $\lambda_{\boldsymbol{G}_\star}^{(i)} \neq 0$ for all $i \leq \min\{d, N\}$. When $\gamma \geq (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$, non-zero $\lambda_{\boldsymbol{G}}^{(i)}$ first disappears for the larger $\lambda_{\boldsymbol{M}}^{(i)}$. □

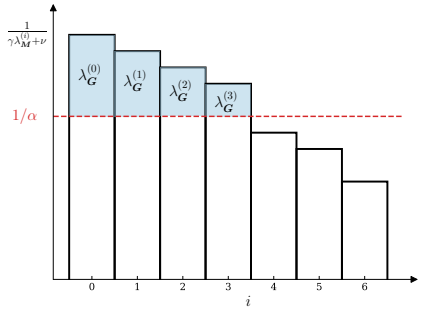



Figure A.1: **Illustration of the optimal solution for problem (6)**. The primal problem can be transformed into a classical reverse water-filling problem.

**Theorem 3.** *Suppose that the sufficient conditions to prevent catastrophic feature collapse are satisfied. Without loss of generality, we further assume that the columns in matrix $\boldsymbol{Z}$ are arranged into $k$ blocks according to a certain $N \times N$ permutation matrix $\boldsymbol{\Gamma}$, i.e., $\boldsymbol{Z} = [\boldsymbol{Z}_1, \boldsymbol{Z}_2, \cdots, \boldsymbol{Z}_k]$. Then the condition for that PRO-DSC promotes the optimal solution $(\boldsymbol{Z}_\star, \boldsymbol{C}_\star)$ to be desired structure, i.e., $\boldsymbol{Z}_\star^\top \boldsymbol{Z}_\star$ and $\boldsymbol{C}_\star$ are block-diagonal, is that $\langle (\boldsymbol{I} - \boldsymbol{C})(\boldsymbol{I} - \boldsymbol{C})^\top, \boldsymbol{G} - \boldsymbol{G}^* \rangle \to 0$, where*

$$\boldsymbol{G}^* := \operatorname{Diag}\big(\boldsymbol{G}_{11}, \boldsymbol{G}_{22}, \cdots, \boldsymbol{G}_{kk}\big) = \begin{bmatrix} \boldsymbol{G}_{11} & & \\ & \ddots & \\ & & \boldsymbol{G}_{kk} \end{bmatrix},$$

*and $\boldsymbol{G}_{jj}$ is the block Gram matrix corresponding to $\boldsymbol{Z}_j$.*

*Proof.* We begin with the analysis to the first two terms of the loss function $\tilde{\mathcal{L}} \coloneqq \mathcal{L}_1 + \gamma \mathcal{L}_2$, where

$$\mathcal{L}_1 \coloneqq -\frac{1}{2}\log\det\left(\boldsymbol{I} + \alpha(\boldsymbol{Z}\boldsymbol{\Gamma})^\top(\boldsymbol{Z}\boldsymbol{\Gamma})\right) = -\frac{1}{2}\log\det(\boldsymbol{I} + \alpha\boldsymbol{G}),$$
$$\mathcal{L}_2 \coloneqq \frac{1}{2}\|\boldsymbol{Z}\boldsymbol{\Gamma} - \boldsymbol{Z}\boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top\boldsymbol{C}\boldsymbol{\Gamma}\|_F^2 = \frac{1}{2}\|\boldsymbol{Z} - \boldsymbol{Z}\boldsymbol{C}\|_F^2 = \frac{1}{2}\operatorname{Tr}\left(\boldsymbol{G}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top\right),$$

since that $\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma} = \boldsymbol{\Gamma}\boldsymbol{\Gamma}^\top = \boldsymbol{I}$. Thus, we have:

$$\tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) = \frac{\gamma}{2}\operatorname{Tr}\left(\boldsymbol{G}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top\right) - \frac{1}{2}\log\det(\boldsymbol{I} + \alpha\boldsymbol{G}), \tag{60}$$

which is a convex function with respect to (w.r.t) $\boldsymbol{G}$ and $\boldsymbol{C}$, separately. By the property of convex function w.r.t. $\boldsymbol{G}$, we have:

$$\begin{aligned}
\tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) &\geq \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}) + \left\langle \nabla_{\boldsymbol{G}}\tilde{\mathcal{L}}\,|_{[\boldsymbol{G}^*, \boldsymbol{C}]}, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle \\
&= \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}) + \left\langle \frac{\gamma}{2}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top - \frac{\alpha}{2}(\boldsymbol{I} + \alpha\boldsymbol{G}^*)^{-1}, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle \\
&= \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}) + \left\langle \frac{\gamma}{2}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle - \left\langle \frac{\alpha}{2}(\boldsymbol{I} + \alpha\boldsymbol{G}^*)^{-1}, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle.
\end{aligned}$$

Since that $\left\langle (\boldsymbol{I} + \alpha\boldsymbol{G}^*)^{-1}, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle = 0$, we have:

$$\tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) \geq \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}) + \left\langle \frac{\gamma}{2}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle.$$

By the property of convex function w.r.t. $\boldsymbol{C}$, we have:

$$\begin{aligned}
\tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) &\geq \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}^*) + \left\langle \nabla_{\boldsymbol{C}}\tilde{\mathcal{L}}\,|_{[\boldsymbol{G}^*, \boldsymbol{C}^*]}, \boldsymbol{C} - \boldsymbol{C}^*\right\rangle + \left\langle \frac{\gamma}{2}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle \\
&= \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}^*) + \left\langle -\gamma\boldsymbol{G}^*(\boldsymbol{I} - \boldsymbol{C}^*), \boldsymbol{C} - \boldsymbol{C}^*\right\rangle + \left\langle \frac{\gamma}{2}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle.
\end{aligned}$$

Since that $\left\langle \boldsymbol{G}^*(\boldsymbol{I} - \boldsymbol{C}^*), \boldsymbol{C} - \boldsymbol{C}^*\right\rangle = 0$, we have:

$$\tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) \geq \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}^*) + \left\langle \frac{\gamma}{2}\left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle.$$

It is easy to see that if $\left\langle \left(\boldsymbol{I} - \boldsymbol{C}\right)\left(\boldsymbol{I} - \boldsymbol{C}\right)^\top, \boldsymbol{G} - \boldsymbol{G}^*\right\rangle \to 0$, then we will have:

$$\tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) \geq \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}^*), \tag{61}$$

where the equality holds *only when* $\boldsymbol{G} = \boldsymbol{G}^*, \boldsymbol{C} = \boldsymbol{C}^*$, in which $\boldsymbol{C}^* = \operatorname{Diag}\left(\boldsymbol{C}_{11}, \boldsymbol{C}_{22}, \cdots, \boldsymbol{C}_{kk}\right) = \begin{bmatrix} \boldsymbol{C}_{11} & & \\ & \ddots & \\ & & \boldsymbol{C}_{kk} \end{bmatrix}$. Furthermore, if the regularizer $r(\cdot)$ satisfies the extended block diagonal condition as defined in (Lu et al., 2018), then we have that $r(\boldsymbol{C}) \geq r(\boldsymbol{C}^*)$, where the equality holds if and only if $\boldsymbol{C} = \boldsymbol{C}^*$. Therefore, we have:

$$\mathcal{L}(\boldsymbol{G}, \boldsymbol{C}) = \tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) + \beta \cdot r(\boldsymbol{C}) \geq \tilde{\mathcal{L}}(\boldsymbol{G}^*, \boldsymbol{C}^*) + \beta \cdot r(\boldsymbol{C}^*) = \mathcal{L}(\boldsymbol{G}^*, \boldsymbol{C}^*). \tag{62}$$

Thus we conclude that minimizing the loss function $\mathcal{L}(\boldsymbol{G}, \boldsymbol{C}) = \tilde{\mathcal{L}}(\boldsymbol{G}, \boldsymbol{C}) + \beta \cdot r(\boldsymbol{C})$ promotes the optimal solution $(\boldsymbol{G}_\star, \boldsymbol{C}_\star)$ to have block diagonal structure.

We note that the Gram matrix being block-diagonal, i.e., $\boldsymbol{G}_\star = \boldsymbol{G}^*$, implies that $\boldsymbol{Z}_{\star,j_1}^\top \boldsymbol{Z}_{\star,j_2} = \boldsymbol{0}$ for all $1 \leq j_1 < j_2 \leq k$, which is corresponding to the subspaces associated to the blocks $\boldsymbol{Z}_{\star,j}$'s are orthogonal to each other. □

# B Experimental Supplementary Material

## B.1 Experimental Details

### B.1.1 Synthetic data

As shown in Figure 5a (line 1), the data points are generated from two manifolds. The first manifold (colored in purple) is generated by sampling 100 data points from

$$\boldsymbol{x} = \begin{bmatrix} \cos\left(\frac{1}{5}\sin(5\varphi)\right)\cos\varphi \\ \cos\left(\frac{1}{5}\sin(5\varphi)\right)\sin\varphi \\ \sin\left(\frac{1}{5}\sin(5\varphi)\right) \end{bmatrix} + \boldsymbol{\epsilon}, \tag{63}$$

where $\varphi$ is taken uniformly from $[0, 2\pi]$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, 0.05\boldsymbol{I}_3)$ is the additive noise. The second manifold (colored in blue) is generated by sampling 100 data points from a Gaussian distribution $\mathcal{N}\left([0,0,1]^\top, 0.05\boldsymbol{I}_3\right)$. To further test more complicated cases, we remove 50 data points from the second manifold and replace them with 50 data points sampled from another Gaussian distribution $\mathcal{N}\left([0,0,-1]^\top, 0.05\boldsymbol{I}_3\right)$ (Figure 5a line 2).

In PRO-DSC, the learnable mappings $\boldsymbol{h}(\cdot;\boldsymbol{\Psi})$ and $\boldsymbol{f}(\cdot;\boldsymbol{\Theta})$ are implemented with two MLPs with Rectified Linear Units (ReLU) (Nair & Hinton, 2010) as the activation function. The hidden dimension and output dimension of the MLP is set to 100 and 3, respectively. We train PRO-DSC with batch-size $n_b = 200$, learning rate $\eta = 5 \times 10^{-3}$ for 1000 epochs. We set $\gamma = 0.5, \beta = 1000$, and $\alpha = 3/0.1 \cdot 200$.

For SEDSC methods, we use DSCNet (Ji et al., 2017) as the representative. In Figure 5b, we set $\gamma = 1$ for both cases, whereas in Figure 5c, $\gamma$ is set to 5 and 100 for the two cases, respectively. The encoder and decoder of DSCNet are MLPs with two hidden layers, with the hidden dimensions set to 100 and 3. We train DSCNet with batch-size $n_b = 200$, learning rate $\eta = 1 \times 10^{-4}$ for 1000 epochs.

### B.1.2 Real-world datasets

**Datasets description.** CIFAR-10 and CIFAR-100 are classic image datasets consisting of 50,000 images for training and 10,000 images for testing. They are split into 10 and 100 classes, respectively. CIFAR-20 shares the same images with CIFAR-100 while taking 20 super-classes as labels. ImageNet-Dogs consists of 19,500 images of 15 different dog species and Tiny-ImageNet consists of 100,000 images from 200 different classes. ImageNet-1k is the superset of the two datasets, containing more than 1,280,000 real world images from 1000 classes. For all the datasets except for ImageNet-Dogs, we train the network to implement PRO-DSC on the train set and test it on the test set to validate the generalization of the learned model. For ImageNet-Dogs dataset which does not have a test set, we train the network to implement PRO-DSC on the train set and report the clustering performance on the training set. For a direct comparison, we conclude the basic information of these datasets in Table B.1.

To leverage the CLIP pre-trained features for training, the input images are first resized to 224 with respect to the smaller edge, then center-cropped to $224 \times 224$ and fed into the CLIP pre-trained image encoder to obtain fixed features.[8] The subsequent training of PRO-DSC takes the fixed extracted features as input, instead of loading the entire CLIP pre-trained model.

**Network architecture and hyper-parameters.** The learnable mappings $\boldsymbol{h}(\cdot;\boldsymbol{\Psi})$ and $\boldsymbol{f}(\cdot;\boldsymbol{\Theta})$ are two fully-connected layers with the same output dimension $d$. Following (Chu et al., 2024), for the experiments on real-world data, we stack a pre-feature layer before the learnable mappings, which is composed of two fully-connected layers with ReLU and batch-norm (Ioffe & Szegedy, 2015).

We train the network by SGD optimizer with the learning rate set to $\eta = 10^{-4}$, and the weight decay parameters of $\boldsymbol{f}(\cdot;\boldsymbol{\Theta})$ and $\boldsymbol{h}(\cdot;\boldsymbol{\Psi})$ are set to $10^{-4}$ and $5 \times 10^{-3}$, respectively. Following (Chu et al., 2024), we warm up training $\boldsymbol{f}(\cdot;\boldsymbol{\Theta})$ by diversifying the features with $\mathcal{L}_1 = -\log\det(\boldsymbol{I} +$

[8] We use the ViT L/14 pre-trained model provided by `https://github.com/openai/CLIP` for 768-dimensional features.

Table B.1: **Basic statistical information of datasets.** We summarize the information in terms of the data with both the train and test split, as well as the number of classes involved.

| Datasets | # Train | # Test | # Classes |
|---|---|---|---|
| CIFAR-10 | 50,000 | 10,000 | 10 |
| CIFAR-20 | 50,000 | 10,000 | 20 |
| CIFAR-100 | 50,000 | 10,000 | 100 |
| ImageNet-Dogs | 19,500 | N/A | 15 |
| TinyImageNet | 100,000 | 10,000 | 200 |
| ImageNet | 1,281,167 | 50,000 | 1000 |

$\alpha \boldsymbol{Z}_{\boldsymbol{\Theta}}^{\top} \boldsymbol{Z}_{\boldsymbol{\Theta}}$) for a few iterations and duplicate the weights to $\boldsymbol{h}(\cdot; \boldsymbol{\Psi})$. We set $\alpha = d/0.1 \cdot n_b$ for all the experiments. We summarize the hyper-parameters for training the network to implement PRO-DSC in Table B.2.

Table B.2: **Hyper-parameters configuration for training the network to implement PRO-DSC with CLIP pre-trained features**, where $\eta$ is the learning rate, $d_{pre}$ is the hidden and output dimension of pre-feature layer, $m$ is the output dimension of $\boldsymbol{h}$ and $\boldsymbol{f}$, $n_b$ is the batch size for training, and "# warm-up" is the number of iterations of warm-up stage.

| | $\eta$ | $d_{pre}$ | $d$ | #epochs | $n_b$ | #warm-up | $\gamma$ | $\beta$ |
|---|---|---|---|---|---|---|---|---|
| CIFAR-10 | $1\times10^{-4}$ | 4096 | 128 | 10 | 1024 | 200 | 300/$n_b$ | 600 |
| CIFAR-20 | $1\times10^{-4}$ | 4096 | 256 | 50 | 1500 | 0 | 600/$n_b$ | 300 |
| CIFAR-100 | $1\times10^{-4}$ | 4096 | 128 | 100 | 1500 | 200 | 150/$n_b$ | 500 |
| ImageNet-Dogs | $1\times10^{-4}$ | 4096 | 128 | 200 | 1024 | 0 | 300/$n_b$ | 400 |
| TinyImageNet | $1\times10^{-4}$ | 4096 | 256 | 100 | 1500 | 0 | 200/$n_b$ | 400 |
| ImageNet | $1\times10^{-4}$ | 4096 | 1024 | 100 | 2048 | 2000 | 800/$n_b$ | 400 |
| MNIST | $1\times10^{-4}$ | 4096 | 128 | 100 | 1024 | 200 | 700/$n_b$ | 400 |
| F-MNIST | $1\times10^{-4}$ | 1024 | 128 | 200 | 1024 | 400 | 50/$n_b$ | 100 |
| Flowers | $1\times10^{-4}$ | 1024 | 256 | 200 | 1024 | 200 | 400/$n_b$ | 200 |

**Running other algorithms.** Since $k$-means (MacQueen, 1967), spectral clustering (Shi & Malik, 2000), EnSC (You et al., 2016a), SSCOMP (You et al., 2016b), and DSCNet (Ji et al., 2017) are based on transductive learning, we train and test these models directly on the test set for all the experiments.

(1) For EnSC, we tune the hyper-parameter in front of the self-expressive term $\gamma \in \{1, 2, 5, 10, 20, 50, 100, 200, 400, 800, 1600, 3200\}$ and tune the hyper-parameter to balance the $\ell_1$ and $\ell_2$ norms $\tau \in \{0.9, 0.95, 1\}$ to report the best clustering result.

(2) For SSCOMP, we tune the hyper-parameter to control the sparsity $k_{max} \in \{1, 2, 5, 10, 20, 50, 100, 200\}$ and the residual $\epsilon \in \{10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\}$ to report the best clustering result.

(3) To apply DSCNet to the CLIP features, we substitute MLPs with two hidden layers for the convolutional encoder and decoder. The hidden dimension of the MLPs are 128. We tune the balancing hyper-parameters $\gamma \in \{1, 2, 3, 4\}$ and $\beta \in \{1, 5, 25, 50, 75, 100\}$ and train the model for 100 epochs with learning rate $\eta = 1 \times 10^{-4}$ and batch-size $n_b$ equivalent to number of test data set.

(4) As the performance of CPP is evaluated by averaging the ACC and NMI metrics tested on each batch, we reproduce the results by their open-source implementation and report the results on the entire test set. The authors provide two implementations (see `https://github.com/LeslieTrue/CPP/blob/main/main.py` and `https://github.com/LeslieTrue/CPP/blob/main/main_efficient.py`), where one optimizes the cluster head and the feature head separately and the other shares weights between the two heads. In this paper, we test both cases and report the better results.

(5) For k-means and spectral clustering (including when spectral clustering is used as the final step in subspace clustering), we repeat the clustering 10 times with different random initializations (by setting n_init=10 in scikit-learn) and report the best results.

(6) For SENet, SCAN and EDESC, we adjust the hyper-parameters and repeat experiments for three times, with only the best results are reported.

(7) For TEMI, we directly cited the results from the paper Adaloglou et al. (2023).

**Training from scratch.** Similar to most deep clustering algorithms, we divide the training process into two steps. We begin with pre-training the parameters of the backbone with BYOL (Grill et al., 2020). Then, leveraging the parameters pre-trained in the first stage, we fine-tune the model by the proposed PRO-DSC loss function. Specifically, we set the learning rate $\eta = 0.05$ and the batch size $n_b = 256$. The output feature dimension $d$ is consistent with the setting for training with the CLIP features. We use ResNet-18 as the backbone for the experiments on CIFAR-10 and CIFAR-20, and use ResNet-34 as the backbone for the experiments on other datasets, following (Li et al., 2021; Huang et al., 2023). We use a convolution filter of size $3 \times 3$ and stride 1 to replace the first convolution filter, following (Huang et al., 2023; Li et al., 2020). The data augmentation strategy is as follows:

**Augmentation 1** Augmentation for training from scratch

```
1: transforms.RandomResizedCrop(size=img_size, scale=(0.08, 1)),
2: transforms.RandomHorizontalFlip(),
3: transforms.RandomApply([transforms.ColorJitter(0.4, 0.4, 0.2,
   0.1)], p=0.8),
4: transforms.RandomGrayscale(p=0.2),
5: transforms.RandomApply([transforms.GaussianBlur(kernel_size=23,
   sigma=(0.1, 2.0))], p=1.0).
```

### B.2 Empirical Validation on Theoretical Results

**Empirical Validation on Theorem 1.** To measure the eigenspace alignment of $\boldsymbol{G}$ and $\boldsymbol{M}$ (Theorem 1), we plot $\langle \boldsymbol{u}_j, \boldsymbol{G}\boldsymbol{u}_j / \|\boldsymbol{G}\boldsymbol{u}_j\|_2 \rangle$ in Figure 2c, where $\boldsymbol{u}_j$ is the $j^{\text{th}}$ eigenvector of $\boldsymbol{M}$. When the eigenspace alignment holds, one can verify that:

$$\langle \boldsymbol{u}_j, \frac{\boldsymbol{G}\boldsymbol{u}_j}{\|\boldsymbol{G}\boldsymbol{u}_j\|_2} \rangle = \begin{cases} 1, & \lambda_{\boldsymbol{G}}^{(j)} \neq 0 \\ 0, & \lambda_{\boldsymbol{G}}^{(j)} = 0 \end{cases} \quad \text{for all } j = 1, 2, \ldots, N. \tag{64}$$

As shown in Figure 2c, the first $d = 128$ normalized correlation values converge to 1, while the rest converge to 0, implying the progressively alignment between $\boldsymbol{G}$ and $\boldsymbol{M}$. In addition, we plot the Frobenius norm of the commutator $\boldsymbol{L}$ during training in Figure 2d. The commutator decreases monotonically during the network training, implying the eigenspace alignment by Lemma A1.

**Empirical Validation on Theorem 2.** To verify Theorem 2, we decompose and plot the eigenvalues of $\boldsymbol{G}, \boldsymbol{M}$ in Figure 2a and 2b, respectively. As shown, there are $\min\{d, N\} = 128$ non-zero eigenvalues of $\boldsymbol{G}$, approximately being inversely proportional to the smallest 128 eigenvalues of $\boldsymbol{M}$. This results empirically demonstrate that $\operatorname{rank}(\boldsymbol{Z}_\star) = \min\{d, N\}$ and $\lambda_{\boldsymbol{G}_\star}^{(i)} = \frac{1}{\gamma \lambda_{\boldsymbol{M}}^{(i)} + \nu_\star} - \frac{1}{\alpha}$ for minimizers. Furthermore, the condition of Theorem 2 is verified in Figure 1, where $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$ yields satisfactory clustering accuracy (ACC%) and subspace-preserving representation error (SRE%). The satisfactory ACC and SRE confirm that PRO-DSC avoids catastrophic collapse when $\gamma < (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$ holds. When $\gamma \geq (\alpha - \nu_\star)/\lambda_{max}(\boldsymbol{M})$, PRO-DSC yields significantly worse ACC and SRE. There is a phase transition phenomenon that corresponds to the sufficient condition to prevent collapse.

**Empirical Validation on Theorem 3.** To intuitively visualize the structured representations learned by PRO-DSC, we visualize the Gram matrices $|\boldsymbol{Z}^\top \boldsymbol{Z}|$ for both CLIP features and learned representations on CIFAR-10. The Gram matrix shows the similarities between representations within the same class (indicated by block diagonal values) and across different classes (indicated by off-block diagonal values). Moreover, we display the dimensionality reduction results via Principal Component Analysis (PCA) for the CLIP features and the learned representation of samples from three categories in CIFAR-10. We use PCA for dimensionality reduction as it performs a linear projection, well preserving the underlying structure. As shown in Figure 3, the CLIP features from three

classes approximately lie on different subspaces. Despite of the structured nature of the features, the underlying subspaces are not orthogonal. In the Gram matrix of the CLIP fature, the average similarity between features from different classes is greater than 0.6, resulting in an unclear block diagonal structure. After training with PRO-DSC, the spanned subspaces of the learned representations become orthogonal.[9] Additionally, the off-block diagonal values of the Gram matrix decrease significantly, revealing a clear block diagonal structure. These visualization results qualitatively verify that PRO-DSC aligns the representations with a union of orthogonal subspaces.[10]

In Figure 4, we plot the curves for the compatibly structured coherence (CSC) condition and the average values of $|\boldsymbol{G}^*|$, $|\boldsymbol{G}-\boldsymbol{G}^*|$, $|\boldsymbol{C}^*|$, $|\boldsymbol{C}-\boldsymbol{C}^*|$ during the training of PRO-DSC on CIFAR-10. As illustrated, the CSC condition progressively satisfies. Consequently, the average off-block values $|\boldsymbol{G}-\boldsymbol{G}^*|$ and $|\boldsymbol{C}-\boldsymbol{C}^*|$ gradually decrease while the average block values $|\boldsymbol{G}^*|$ and $|\boldsymbol{C}^*|$ gradually increase, which empirically validates that PRO-DSC promotes block-diagonal structure in $\boldsymbol{G}$ and $\boldsymbol{C}$.

### B.3 More Experimental Results

**More results on synthetic data.** The synthetic experiments of adding an additional subspace are presented in Figure B.1.

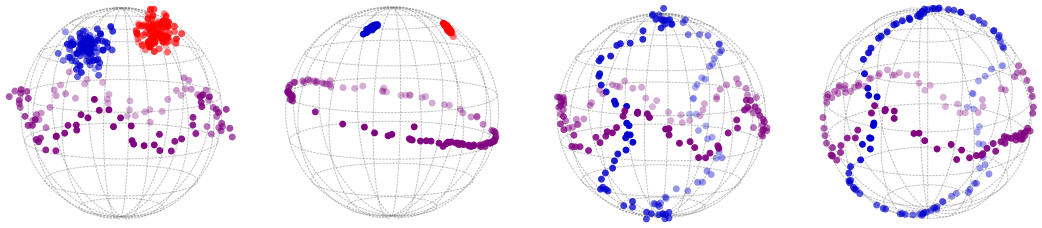
In case 1, we implement two sets with 100 points in each cluster sampled from Gaussian distribution $\boldsymbol{x} \sim \mathcal{N}([\frac{1}{\sqrt{2}}, 0, \sqrt{2}]^\top, 0.05\boldsymbol{I}_3)$ and $\boldsymbol{x} \sim \mathcal{N}([-\frac{1}{\sqrt{2}}, 0, \sqrt{2}]^\top, 0.05\boldsymbol{I}_3)$ in the same side of the sphere. PRO-DSC eliminates the nonlinearity in representations and maximally separates the different subspaces.

In case 2, we add a vertical curve with 100 points sampled by:

$$\boldsymbol{x} = \begin{bmatrix} \cos\left(\frac{1}{5}\sin(5\varphi)\right)\cos\varphi \\ \sin\left(\frac{1}{5}\cos(5\varphi)\right) \\ \cos\left(\frac{1}{5}\sin(5\varphi)\right)\sin\varphi \end{bmatrix} + \boldsymbol{\epsilon}, \tag{65}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, 0.05\boldsymbol{I}_3)$ and use $\sin(\frac{1}{5}\cos(5\varphi))$ to avoid overlap in the intersection of the two curves. PRO-DSC finds difficulties to learn representations of data which located at the intersection of subspaces. However, those which are away from the intersection are linearized well.

For the experiments on synthetic data, the learnable mappings $\boldsymbol{h}(\cdot;\boldsymbol{\Psi})$ and $\boldsymbol{f}(\cdot;\boldsymbol{\Theta})$ are implemented with two MLPs with Rectified Linear Units (ReLU) (Nair & Hinton, 2010) as the activation function. The hidden dimension and output dimension of the MLP is set to 100 and 3, respectively. In case 1, we train PRO-DSC with batch-size $n_b = 300$, learning rate $\eta = 5 \times 10^{-3}$ for 5000 epochs and set $\gamma = 1.3, \beta = 500, \alpha = 3/0.1 \cdot 300$. In case 2, we train PRO-DSC with batch-size $n_b = 200$, learning rate $\eta = 5 \times 10^{-3}$ for 8000 epochs and set $\gamma = 0.5, \beta = 500, \alpha = 3/0.1 \cdot 200$.



(a) Case 1: Input data (b) Case 1: Learned $\boldsymbol{Z}$ (c) Case 2: Input data (d) Case 2: Learned $\boldsymbol{Z}$

Figure B.1: Additional results on synthetic data.

**Experimental Results on Datasets Reuters and UCI HAR.** The dataset Reuters-10k consists of four text classes, containing 10,000 samples of 2,000 dimension. The UCI HAR is a time-series

[9]The dimension of each subspace is much greater than one (see Figure B.3). The 1-dimensional subspaces observed in the PCA results are a consequence of dimensionality reduction.

[10]Please refer to Figure B.2 and B.7 for the results on other datasets and the visualization of the bases of each subspace.

dataset, consisting of six classes, 10,299 samples of 561 dimension. We take EDESC (Cai et al., 2022) as the baseline method for deep subspace clustering on Reuters-10k, and take N2D (McConville et al., 2021) and FCMI (Zeng et al., 2023) as the baseline methods for UCI HAR, in which the results are directly cited from the respective papers. We conducted experiments with PRO-DSC on Reuters and UCI HAR following the same protocol for data processing as the baseline methods. We train and test PRO-DSC on the entire dataset and report the results over 10 trials. Experimental results are provided in Table B.4. The hyper-parameters used for PRO-DSC is summarized in Table B.3.

Table B.3: Hyper-parameter setting for experiments on Reuters, UCI HAR, EYale-B, ORL and COIL-100.

| Dataset | $\eta$ | $d_{pre}$ | $d$ | #epochs | $n_b$ | #warm-up | $\gamma$ | $\beta$ |
|---|---|---|---|---|---|---|---|---|
| REUTERS-10k | 1e-4 | 1024 | 128 | 100 | 1024 | 50 | 50 | 200 |
| UCI HAR | 1e-4 | 1024 | 128 | 100 | 2048 | 20 | 100 | 300 |
| EYale-B | 1e-4 | 1080 | 256 | 10000 | 2432 | 100 | 200 | 50 |
| ORL | 1e-4 | 80 | 64 | 5000 | 400 | 100 | 75 | 10 |
| COIL-100 | 1e-4 | 12800 | 100 | 10000 | 7200 | 100 | 200 | 100 |

Table B.4: Experimental Results on Datasets Reuters and UCI HAR with 10 trials. The results of other methods are cited from the respective papers.

| Dataset | REUTERS-10k | | UCI HAR | |
|---|---|---|---|---|
| | ACC | NMI | ACC | NMI |
| $k$-means (MacQueen, 1967) | 52.4 | 31.2 | 59.9 | 58.8 |
| SC (Shi & Malik, 2000) | 40.2 | 37.5 | 53.8 | 74.1 |
| AE (Bengio et al., 2006) | 59.7 | 32.3 | 66.3 | 60.7 |
| VAE (Kingma & Welling, 2014) | 62.5 | 32.9 | - | - |
| JULE (Yang et al., 2016) | 62.6 | 40.5 | - | - |
| DEC (Xie et al., 2016) | 75.6 | 68.3 | 57.1 | 65.5 |
| DSEC (Chang et al., 2018) | 78.3 | 70.8 | - | - |
| EDESC (Cai et al., 2022) | 82.5 | 61.1 | - | - |
| DFDC (Zhang & Davidson, 2021) | - | - | 86.2 | 84.5 |
| N2D (McConville et al., 2021) | - | - | 82.8 | 71.7 |
| FCMI (Zeng et al., 2023) | - | - | **88.2** | 80.7 |
| PRO-DSC | **85.7** $\pm$ 1.3 | **64.6** $\pm$ 1.3 | 87.1 $\pm$ 0.4 | **80.9** $\pm$ 1.2 |

**Comparison to AGCSC and SAGSC.** During the rebuttal, we conducted more experiments on two state-of-the-art subspace clustering methods AGCSC (Wei et al., 2023) and ARSSC (Wang et al., 2023a). Since that both of the two methods cannot handle the datasets used for evaluating our PRO-DSC, we conducted experiments on the datasets: Extended Yale B (EYaleB), ORL, and COIL-100. We set the architecture of pre-feature layer in PRO-DSC as the same to the encoder of DSCNet (Ji et al., 2017). The hyper-parameters configuration for training PRO-DSC is summarized in Table B.3. We repeated experiments for 10 trails and report the average with standard deviation in Table B.5. For the results of other methods in Table B.5, we directly cited the results from DSSC (Lim et al., 2020).

**1) AGCSC.** Our method surpasses AGCSC on the Extended Yale B dataset and achieves comparable results on the ORL dataset. However, AGCSC cannot yield the result on COIL-100 in 24 hours.

**2) ARSSC.** ARSSC employs three different non-convex regularizers: $\ell_\gamma$ norm Penalty (LP), Log-Sum Penalty (LSP), and Minimax Concave Penalty (MCP). While ARSSC-MCP performs the best on Extended Yale B, our PRO-DSC outperforms ARSSC-MCP on ORL. While AGCSC performs the best on ORL, but it yields inferior results on Extended Yale B and it cannot yield the results on COIL-100 in 24 hours. Thus, we did not report the results of AGCSC on COIL-100 and marked it as Out of Time (OOT). Our PRO-DSC performs the second best results on Extended Yale B, ORL

and the best results on COIL-100. Since that we have not found the open-source code for ARSSC, we are unable to have their results on COIL-100. This comparison also confirms the scalablity of our PRO-DSC which is due to the re-parametrization (similar to SENet).

It is noteworthy that the goal of our work is to provide a reasonable (or a principled) framework for deep subspace clustering based on self-expressive (SE) model. We demonstrate theoretically and empirically that adding the $\log\det(\cdot)$ term into the SE model can prevent the catastrophic feature collapse, and show our PRO-DSC promotes to produce representations that are aligned with a union of orthogonal subspaces. Therefore, our PRO-DSC provides a general framework for self-expressive model based deep subspace clustering.

Since that both AGCSC and ARSSC improve the self-expressive model by incorporating GCN modules and self-adaptive attention mechanisms, respectively. Thus, it will be attempting to employ them into the deep self-expressive models under our PRO-DSC framework. Note that our PRO-DSC with a vanilla self-expressive model has already demonstrated highly competitive performance. We believe that extending PRO-DSC with current SOTA self-expressive models (such as AGCSC and ARSSC) would be more promising.

As a general framework for self-expressive model based deep subspace clustering, our PRO-DSC is reasonable, scalable and flexible to miscellaneous extensions.

Table B.5: Experiments on Extended Yale B, ORL and COIL-100. Note that the results of EnSC, SSCOMP, S3COMP, DSCNet, J-DSSC and ADSDSC are cited from (Lim et al., 2020).

| | EYale-B | | ORL | | COIL-100 | |
|---|---|---|---|---|---|---|
| | ACC | NMI | ACC | NMI | ACC | NMI |
| EnSC | 65.2 | 73.4 | 77.4 | 90.3 | 68.0 | 90.1 |
| SSCOMP | 78.0 | 84.4 | 66.4 | 83.2 | 31.3 | 58.8 |
| S3COMP | 87.4 | - | - | - | 78.9 | - |
| DSCNet | 69.1 | 74.6 | 75.8 | 87.8 | 49.3 | 75.2 |
| J-DSSC (Lim et al., 2020) | 92.4 | 95.2 | 78.5 | 90.6 | 79.6 | 94.3 |
| A-DSSC (Lim et al., 2020) | 91.7 | 94.7 | 79.0 | 91.0 | <u>82.4</u> | 94.6 |
| AGCSC (Wei et al., 2023) | 92.3 | 94.0 | **86.3** | **92.8** | OOT | OOT |
| ARSSC-LP (Wang et al., 2023a) | 95.7 | - | 75.5 | - | - | - |
| ARSSC-LSP (Wang et al., 2023a) | 95.9 | - | 71.3 | - | - | - |
| ARSSC-MCP (Wang et al., 2023a) | **99.3** | - | 72.0 | - | - | - |
| PRO-DSC | <u>96.0</u> $\pm$ 0.3 | **95.7** $\pm$ 0.8 | <u>83.2</u> $\pm$ 2.2 | <u>92.7</u> $\pm$ 0.6 | **82.8** $\pm$ 0.9 | **95.0** $\pm$ 0.6 |

**Gram matrices and PCA visualizations.** To qualitatively validate that PRO-DSC learns representations aligning with a union-of-orthogonal-subspaces distribution, we visualize the Gram matrices and PCA dimension reduction results of CLIP features and learned representations from PRO-DSC for each dataset. As shown in Figure B.2, the off-block diagonal values decrease significantly, implying the orthogonality between representations from different classes. The orthogonal between subspaces can also be observed from the PCA dimension reduction results.

**Singular values visualization.** To show the intrinsic dimension of CLIP features and the representations of PRO-DSC, We plot the singular values of CLIP features and PRO-DSC's representations in Figure B.3. Specifically, the singular values of features from all the samples are illustrated on the left and the singular values of features within each class are illustrated on the middle and right. As can be seen, the singular values of PRO-DSC decrease much slower than that of CLIP, implying that the features of PRO-DSC enjoy a higher intrinsic dimension and more isotropic structure in the ambient space.

**Learning curves.** We plot the learning curves with respect to loss values and performance of PRO-DSC on CIFAR-100, CIFAR-20 and ImageNet-1k in Figure B.4a, Figure B.4b and Figure B.4c, respectively. Recall that $\mathcal{L}_1 := -\frac{1}{2}\log\det\left(\boldsymbol{I} + \alpha \boldsymbol{Z}_{\boldsymbol{\Theta}}^\top \boldsymbol{Z}_{\boldsymbol{\Theta}}\right)$, $\mathcal{L}_2 := \frac{1}{2}\|\boldsymbol{Z}_{\boldsymbol{\Theta}} - \boldsymbol{Z}_{\boldsymbol{\Theta}}\boldsymbol{C}_{\boldsymbol{\Psi}}\|_F^2$, and $\mathcal{L}_3 := \|\boldsymbol{A}_{\boldsymbol{\Psi}}\|_{\boxed{\kappa}}$. Since $\mathcal{L}_1$ is the only loss function used in the warm-up stage, we plot all the curves starting from the iteration when warm-up ends.

As illustrated, the clustering performance of PRO-DSC steadily increase as the loss values gradually decrease, which shows the effectiveness of the proposed loss functions in PRO-DSC.

**Clustering on learned representations.** To quantitatively validate the effectiveness of the structured representations learned by PRO-DSC, we illustrate the clustering accuracy of representations

learned by various algorithms in Figure 6. Here, to compared with the representations learned from SEDSC methods, we additionally conduct experiments on DSCNet (Ji et al., 2017) and report the performance in Table B.6. To apply DSCNet on CLIP features, we substitute MLPs with two hidden layers for the stacked convolutional encoder and decoder. As demonstrated in Sec. B.1, we report the best clustering results after the tuning of hyper-parameters. As analyzed in (Haeffele et al., 2021) and Section 2.1, SEDSC overly compresses the representations and yields unsatisfactory clustering results.

Table B.6: **Clustering accuracy of CLIP features and learned representations.** We apply $k$-means, spectral clustering, and EnSC to cluster the representations.

| | CIFAR-10 | | | CIFAR-100 | | | CIFAR-20 | | | TinyImgNet-200 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k$-means | SC | EnSC | $k$-means | SC | EnSC | $k$-means | SC | EnSC | $k$-means | SC | EnSC |
| CLIP | 74.7 | 70.2 | 95.4 | 52.8 | 66.4 | 67.0 | 46.9 | 49.2 | **60.8** | 54.1 | 62.8 | 64.5 |
| SEDSC | 16.4 | 18.9 | 16.9 | 5.4 | 4.9 | 5.3 | 11.7 | 10.6 | 12.8 | 5.7 | 3.9 | 7.2 |
| CPP | 71.3 | 70.3 | **95.6** | 75.3 | 75.0 | 77.5 | 55.5 | 43.6 | 58.3 | 62.1 | 58.0 | 67.0 |
| PRO-DSC | **93.4** | **92.1** | 95.5 | **76.5** | **75.2** | **77.6** | **66.0** | **59.7** | 60.0 | **67.6** | **67.0** | **69.5** |

**Clustering on ImageNet-1k with DINO and MAE.** To test the performance of PRO-DSC based on more pre-trained features other than CLIP (Radford et al., 2021), we further conduct experiments on ImageNet-1k (Deng et al., 2009) pre-trained by DINO (Caron et al., 2021) and MAE (He et al., 2022) (see Table B.7).

Table B.7: Clustering Performance of PRO-DSC based on DINO and CLIP pre-trained features on ImageNet-1k.

| Method | Backbone | PRO-DSC | | $k$-means | |
|---|---|---|---|---|---|
| | | ACC | NMI | ACC | NMI |
| MAE (He et al., 2022) | ViT L/16 | 9.0 | 49.1 | 9.4 | 49.3 |
| DINO (Caron et al., 2021) | ViT B/16 | 57.3 | 79.3 | 52.2 | 79.2 |
| DINO (Caron et al., 2021) | ViT B/8 | 59.7 | 80.8 | **54.6** | **80.5** |
| CLIP (Radford et al., 2021) | ViT L/14 | **65.1** | **83.6** | 52.5 | 79.7 |

DINO and MAE are pre-trained on ImageNet-1k without leveraging external training data, thus their performance on PRO-DSC is lower than CLIP. Similar to the observations in CPP (Chu et al., 2024), DINO initializes PRO-DSC well, yet MAE fails, which is attributed to the fact that features from MAE prefer fine-tuning with labels, while they are less suitable for learning inter-cluster discriminative representations (Oquab et al., 2023). We further extract features from the validation set of ImageNet-1k and visualize through $t$-SNE (Van der Maaten & Hinton, 2008) to validate the hypothesis (see Figure B.5).

**Out of domain datasets.** We evaluate the capability to refine features by training PRO-DSC with pre-trained CLIP features on out-of-domain datasets, namely, MNIST (Deng, 2012), Fashion MNIST (Xiao et al., 2017) and Oxford flowers (Nilsback & Zisserman, 2008). As shown in Table B.8, CPP (Chu et al., 2024) refines the CLIP features and yields better clustering performance comparing with spectral clustering (Shi & Malik, 2000) and EnSC (You et al., 2016a). Our PRO-DSC further demonstrates the best performance on all benchmarks, validating its effectiveness in refining input features.

**Experiments on block diagonal regularizer with different $k$.** To test the robustness of block diagonal regularizer $\|\boldsymbol{A}\|_{\boxed{k}}$ to different $k$, we vary $k$ and report the clustering performance in Table B.9. As illustrated, $k$ does not necessarily equal to the number of clusters. There exists an interval within which the regularizer works effectively.

But if $k$ is significantly smaller than the number of clusters, the effect of block diagonal regularizer will be subtle. Therefore, the performance of PRO-DSC will be similar to that of PRO-DSC without a regularizer (see ablation studies in Section 3). In contrary, if $k$ is significantly larger than the number of clusters, over-segmentation will occur to the affinity matrix, which has negative impact on the subsequent clustering performance.

Table B.8: Experiments on out-of-domain datasets.

| Methods | MNIST | | F-MNIST | | Flowers | |
|---|---|---|---|---|---|---|
| | ACC | NMI | ACC | NMI | ACC | NMI |
| Spectral Clustering (Shi & Malik, 2000) | 74.5 | 67.0 | 64.3 | 56.8 | 85.6 | 94.6 |
| EnSC (You et al., 2016a) | 91.0 | 85.3 | 69.1 | 65.1 | 90.0 | 95.9 |
| CPP (Chu et al., 2024) | 95.7 | 90.4 | 70.9 | 68.8 | 91.3 | 96.4 |
| PRO-DSC | **96.1** | **90.9** | **71.3** | **70.3** | **92.0** | **97.4** |

Table B.9: Clustering performance with different $k$ in block diagonal regularizer.

| | $k$ | 2 | 5 | 10 | 15 | 20 | 25 | 30 |
|---|---|---|---|---|---|---|---|---|
| CIFAR-10 | ACC | 97.2 | 97.2 | **97.4** | 96.3 | 96.3 | 95.4 | 94.0 |
| | NMI | 93.2 | 93.2 | **93.5** | 92.0 | 92.0 | 90.7 | 88.6 |

| | $k$ | 25 | 50 | 75 | 100 | 125 | 150 | 200 |
|---|---|---|---|---|---|---|---|---|
| CIFAR-100 | ACC | 74.3 | 76.7 | 78.1 | 78.2 | **78.9** | 76.4 | 74.8 |
| | NMI | 80.9 | 82.3 | **83.2** | 82.9 | **83.2** | 82.2 | 81.5 |

**$t$-SNE visualization of learned representations.** We visualize the CLIP features and cluster representations learned by PRO-DSC leveraging $t$-SNE (Van der Maaten & Hinton, 2008) in Figure B.6. As illustrated, the learned cluster representations are significantly more compact compared with the CLIP features, which contributes to the improved clustering performance.

**Subspace visualization.** We visualize the principal components of subspaces learned by PRO-DSC in Figure B.7. For each cluster in the dataset, we apply Principal Component Analysis (PCA) to the learned representations. We select the top eight principal components to represent the learned subspaces. Then, for each principal component, we display eight images whose representations are most closely aligned with that principal component.

Interestingly, we can observe specific semantic meanings from the principal components learned by PRO-DSC. For instance, the third row of Figure B.7a consists of stealth fighters, whereas the fifth row shows airliners. The second row of Figure B.7c consists of birds standing and resting, while the sixth row shows flying eagles. While Figure B.7j consists of all kinds of trucks, the first row shows fire trucks.

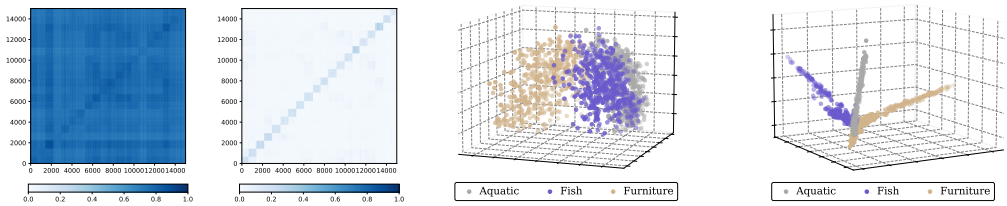
## C Limitations and Failure Cases

**Limitations:** In this paper, we explore an effective framework for deep subspace clustering with theoretical justification. However, it is not clear how to develop the geometric guarantee for our PRO-DSC framework to yield subspace-preserving (correct) solution. Moreover, it is an unsupervised learning framework, we left the extension to semi-supervised setting as future work.

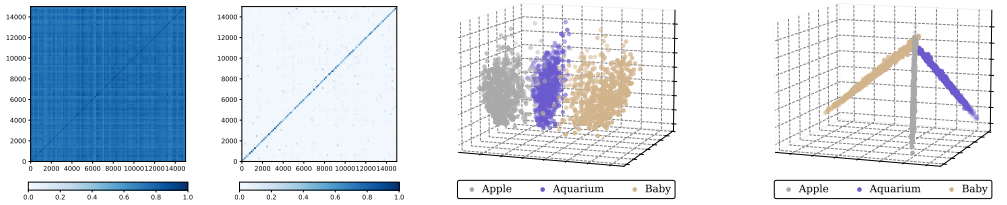
**Failure Cases:** In this paper, we evaluate our PRO-DSC framework on two scenarios of synthetic data (Fig. 5), six benchmark datasets with CLIP features (Table 1), five benchmark datasets which are for training from scratch (Table 2), three out-of-domain datasets (Table B.8), using four different regularization terms (Table 4), using different feature extractor (Table B.7) and varing hyper-parameters (Fig. 7 and Table B.9). During the rebuttal, to reply Reviewer JbyQ, we also conduct experiments on two face image datasets (Extended Yale b and ORL), text dataset (REUTERS) and temporal dataset (UCI HAR). Currently, we did not find significant failure cases. However, as demonstrated in Fig. 2, our PRO-DSC will fail if the sufficient condition to prevent catastrophic collapse is not satisfied by using improper hyper-parameters $\gamma$ and $\alpha$.

**Extensibility:** As a general framework for self-expressive model based deep subspace clustering, our PRO-DSC is scalable and flexible to miscellaneous extensions. For example, AGCSC (Wei et al., 2023) and ARSSC (Wang et al., 2023a) improve the self-expressive model by incorporating GCN modules and self-adaptive attention mechanisms, respectively. Thus, it will be attempting to employ them into the deep self-expressive models under our PRO-DSC framework. Moreover, rather than using $\log\det(\cdot)$, there are other methods to solve the feature collapse issue, e.g., the

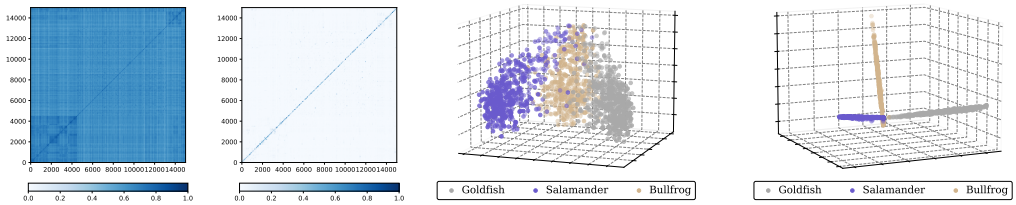
nuclear norm. In addition, it is also worthwhile to incorporate the supervision information from the pseudo-label (e.g., Huang et al. (2023), Jia et al. (2024) and Li et al. (2017)) for further improving the performance of our PRO-DSC.



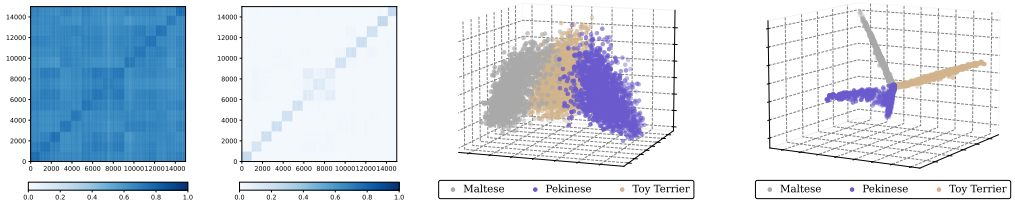

(a) CIFAR-20



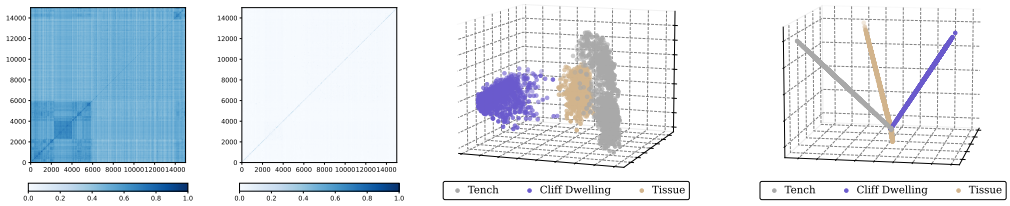

(b) CIFAR-100




(c) TinyImageNet-200

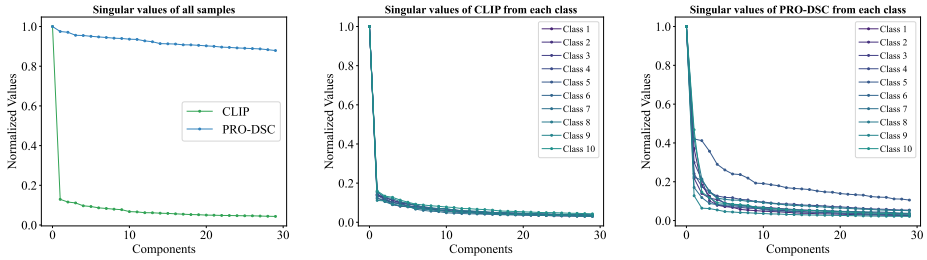



(d) ImageNet-Dogs-15

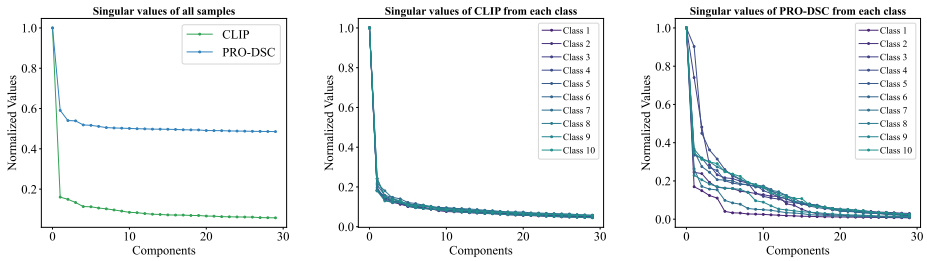



(e) ImageNet-1k

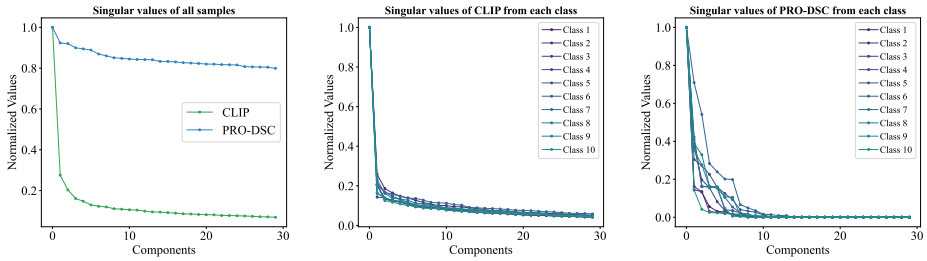
Figure B.2: **Visualization of the union-of-orthogonal-subspaces structure of the learned representations via Gram matrix and PCA dimension reduction on three categories.** Left: $|\boldsymbol{X}^\top \boldsymbol{X}|$. Mid-left: $|\boldsymbol{Z}^\top \boldsymbol{Z}|$. Mid-right: $\boldsymbol{X}^{(3)}$ via PCA. Right: $\boldsymbol{Z}^{(3)}$ via PCA.




(a) CIFAR-100

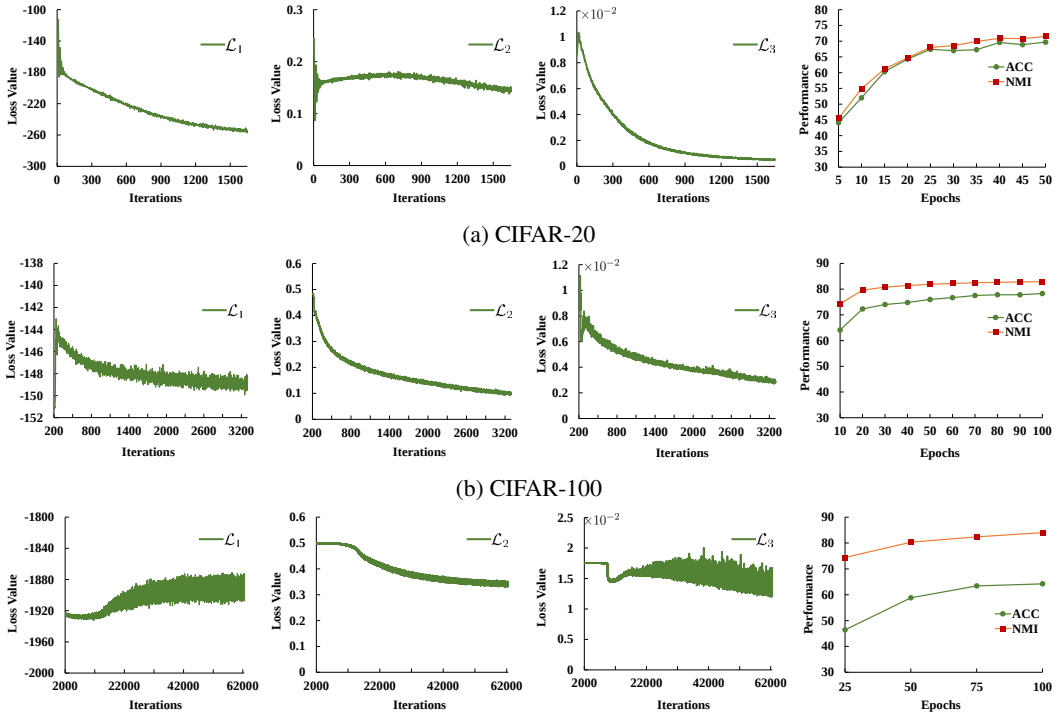



(b) TinyImageNet-200




(c) ImageNet-1k

Figure B.3: **Singular values of features from all samples (left) and features from each class (mid and right).** For the better clarity, we plot the singular values for the first ten classes.

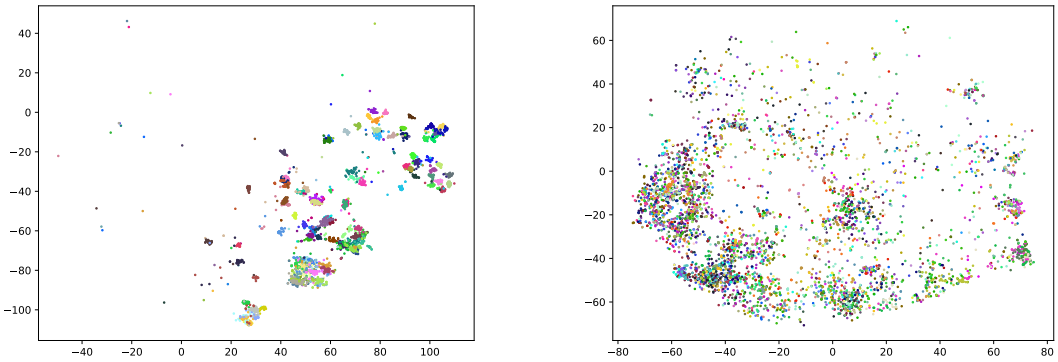



(a) CIFAR-20

(b) CIFAR-100

(c) ImageNet-1k

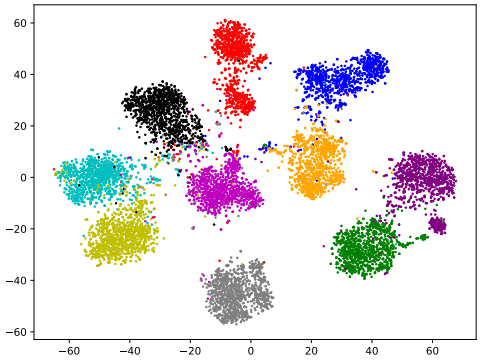
Figure B.4: The learning curves w.r.t. loss values and evaluation performance of PRO-DSC on CIFAR-20, CIFAR-100 and ImageNet-1k dataset.



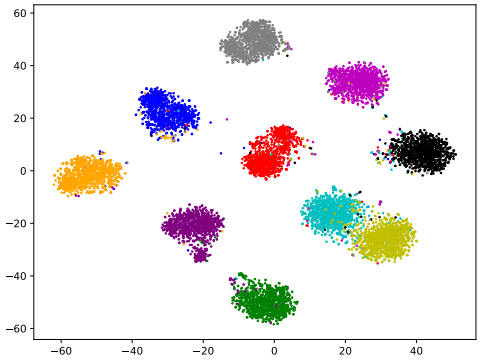
(a) CLIP

(b) MAE

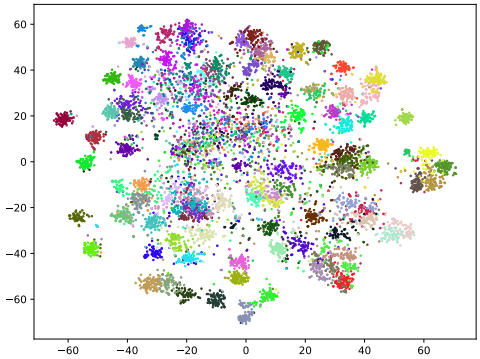
Figure B.5: The $t$-SNE visualization of CLIP and MAE features on the validation set of ImageNet-1k.



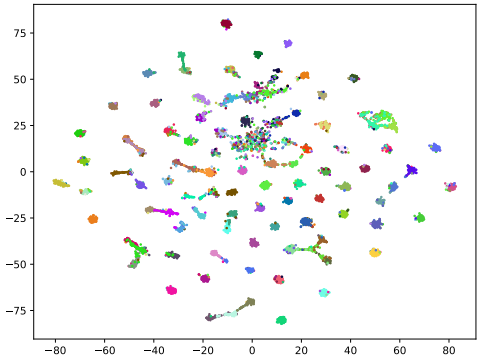

(a) CLIP CIFAR-10




(b) PRO-DSC CIFAR-10




(c) CLIP CIFAR-100




(d) PRO-DSC CIFAR-100

Figure B.6: **$t$-SNE visualization of CLIP features and PRO-DSC's learned representations.** The experiments are conducted on CIFAR-10 and CIFAR-100 dataset.

(a) Cluster 1 (b) Cluster 2 (c) Cluster 3

(d) Cluster 4 (e) Cluster 5 (f) Cluster 6

(g) Cluster 7 (h) Cluster 8 (i) Cluster 9

(j) Cluster 10

Figure B.7: **Visualization of the principal components in CIFAR-10 dataset.** For each cluster, we display the most similar images to its principal components.