Dictionary Learning with Structured Noise

Pan Zhou[a,b], Cong Fang[a,b], Zhouchen Lin[a,b], Chao Zhang[a,b,*], Edward Y. Chang[c]

[a] *Key Lab. of Machine Perception (MOE), School of EECS, Peking University, P. R. China*
[b] *Cooperative Medianet Innovation Center, Shanghai Jiaotong University, P. R. China*
[c] *HTC Research, Taiwan*

Abstract

Recently, lots of dictionary learning methods have been proposed and successfully applied. However, many of them assume that the noise in data is drawn from Gaussian or Laplacian distribution and therefore they typically adopt the ℓ_2 or ℓ_1 norm to characterize these two kinds of noise respectively. Since this assumption is inconsistent with the real cases, the performance of these methods is limited. In this paper, we propose a novel dictionary learning with structured noise (DLSN) method for handling noisy data. We decompose the original data into three parts: clean data, structured noise, and Gaussian noise, and then characterize them separately. We utilize the low-rank technique to preserve the inherent subspace structure of clean data. Instead of only using the predefined distribution to fit the real distribution of noise, we learn an adaptive dictionary to characterize structured noise and employ the ℓ_2 norm to depict Gaussian noise. Such a mechanism can characterize noise more precisely. We also prove that our proposed optimization method can converge to a critical point and the convergence rate is at least sublinear. Experimental results on the data clustering task demonstrate the effectiveness and robustness of our method.

Keywords: Dictionary learning, Structured noise, Low rank representation, Sparse representation

1. Introduction

Dictionary learning has been extensively studied [1, 2, 3, 4] due to its crucial role in sparse representation and low-rank modelling. Instead of using wavelets predefined by artificial rules, recent dictionary learning methods are data-adaptive and aim at learning a series of basic atoms from a data set to linearly approximate a given datum. Mathematically, given a data matrix $Y = [Y_{(1)}, \cdots, Y_{(m)}] \in \mathcal{R}^{d\times m}$ where d and m respectively denote the feature dimension and the sample/signal number, dictionary learning methods try to learn a dictionary $D \in \mathcal{R}^{d\times k}$ so that samples Y can be approximately linearly

*Corresponding author.
Email addresses: pzhou@pku.edu.cn (Pan Zhou), fangcong@pku.edu.cn (Cong Fang), zlin@pku.edu.cn (Zhouchen Lin), chzhang@cis.pku.edu.cn (Chao Zhang), EdwardChang@htc.com (Edward Y. Chang)

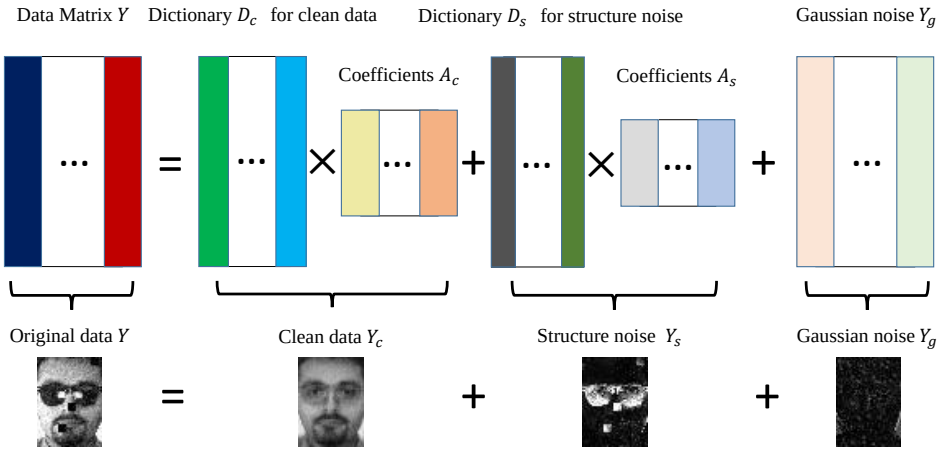


Figure 1: Illustration of the proposed DLSN model. Data are decomposed into clean data, structured noise and Gaussian noise. Then, each of the three parts is characterized separately, i.e., learning a dictionary and an adaptive dictionary for clean data and structured noise respectively, and employing ℓ_2 to depict Gaussian noise. For better viewing, the negative entries in the images are replaced with their absolute values in displaying.

represented as $Y \approx DA$, where each column $D_{(j)}$ in D denotes a dictionary atom, and each column $A_{(i)}$ in $A \in \mathcal{R}^{k\times m}$ is the representation coefficient vector of the i-th sample $Y_{(i)}$. Such a mechanism can characterize the inherent correlations among data, since under the same dictionary, similar samples would have similar representation coefficients and those dissimilar samples would have very different representation coefficients. Dictionary learning has now become an efficient feature learning method and achieved state-of-the-art performance in many practical applications. KSVD [1] constructs a dictionary by minimizing the reconstruction error of original samples and achieves impressive performance in image recovery application. The tree-structured dictionary learning method [2] exploits possible semantic relationships between dictionary atoms to learn structured dictionaries embedded in a hierarchy and produces state-of-the-art denoising performance. Later Yang et al. [5] further apply tree-structured dictionary learning method into mobile visual search [6, 7, 8, 9, 10]. Some literature [3, 4, 11, 12] adopts different reconstruction residual regularizations to depict noise more precisely and good results are obtained in image denoising and data clustering tasks. In addition, dictionary learning methods have also been used for other purposes such as signal reconstruction [13] and visual saliency [14].

Due to the commonly existing noise in real data, the original signals/images cannot be exactly linearly represented by the learnt dictionary. Thus in many dictionary learning methods, the original data are decomposed into a linear combination of atoms from a dictionary and noise, i.e. $Y = DA+E$, where E denotes the noise in data. To depict the noise E, some works [15, 1, 3, 16, 17, 18, 19] assume

that the noise is of dense Gaussian distribution with small variation and can be depicted by the ℓ_2 norm, while [11, 20] argue that the noise in real data is drawn from Laplacian distribution and should be characterized by the ℓ_1 norm. For better depicting noise, Chen et al. [11] further decompose the noise into two parts, small dense Gaussian noise and Laplacian noise, and try to depict them separately. However, Chen et al. [11] and Wu et al. [20] point out that the actual distribution of noise in real applications, e.g. face recognition, is very complex and of neither Gaussian nor Laplacian distribution. Thus, such a mechanism of characterizing the real noise with predefined distributions would limit the performance of dictionary learning methods.

In this paper, we propose a novel dictionary learning with structured noise (DLSN) method which aims at handling noise in data from another perspective. As shown in Fig. 1, we decompose data into clean data, structured noise and Gaussian noise, and then characterize them separately. Since clean visual data usually lie in several subspaces, and compared with sparse representation that encodes each signal independently, the low-rank technique can preserve the global inherent structure of data [21, 22, 23]. In this paper, we also utilize the low-rank technique to characterize clean data. As for noise, unlike the previous methods that totally use predefined distributions, e.g. Gaussian and Laplacian distributions, to fit the real distribution of noise, we propose an adaptive method to characterize noise. We divide noise into two parts, Gaussian noise and structured noise. Compared with Gaussian noise, the remaining noise in data usually contains semantic meanings, such as the different illuminations and the glasses on the faces shown in Fig. 1. So we refer to this kind of noise as structured noise. We learn a set of basic atoms for structured noise, and typically, a datum consists of some different structured noise. Therefore, we can represent the structured noise in a datum with a combination of a few atoms from a structured noise dictionary. We also adopt the ℓ_2 norm to characterize Gaussian noise. In summary, our main contributions in this paper include:

1) We propose a novel low-rank based dictionary learning method to handle noisy data. Unlike sparse representation that encodes each signal dependently, the low-rank technique can capture the global inherent structure of data and improve the quality of the learnt dictionary.

2) Instead of solely using Gaussian or Laplacian distribution to characterize the real noise, we learn an adaptive dictionary for structured noise and adopt Gaussian distribution to depict the remaining Gaussian noise.

Extensive experimental results demonstrate the advantages of our method.

Table 1: Summary of notations frequently used in this paper.

Notation	Meaning
capital letter	A matrix.
M^T	Transpose of matrix M.
M_{ij}	The (i,j)-th entry of matrix M.
$M_{(i)}$	The i-th column of matrix M.
x_i	The i-th entry of vector x.
$\|\|\cdot\|\|_*$	Nuclear norm, sum of the singular values.
$\|\|\cdot\|\|_0$	Number of nonzero entries.
$\|\|\cdot\|\|_1$	$\|\|M\|\|_1 = \sum_{i,j} \|M_{ij}\|$.
$\|\|\cdot\|\|_F$	Frobenious norm, $\|\|M\|\|_F = \sqrt{\sum_{i,j} M_{ij}^2}$.
$\|\|\cdot\|\|_2$	$\|\|x\|\|_2 = \sqrt{\sum_i x_i^2}$ if x is a vector; $\|\|X\|\|_2$ is the spectral norm if X is a matrix.

2. Related Work

In this section, we review the existing dictionary learning methods. Based on the noise assumption, we can roughly divide the dictionary learning methods into three kinds: Gaussian noise based methods, Laplacian noise based methods, and mixed (Gaussian and Laplacian) noise based methods, which will be reviewed sequentially. For brevity, we summarize some main notations in Table 1.

Gaussian noise based methods [15, 1, 3, 16, 17, 18, 19] assume that noise is drawn from Gaussian distribution and use the ℓ_2 norm (square loss function) to depict the noise $E = Y - D$. A typical model is

$$\begin{aligned} \min_{D,A} \ \|Y - DA\|_F^2, \ \text{s.t.} \ \|D_{(i)}\|_2^2 \le 1, \ \forall i \in \{1, 2, \cdots, k\}, \\ \|A_{(j)}\|_0 \le T, \ \forall j \in \{1, 2, \cdots, m\}, \end{aligned} \tag{1}$$

where $\|A_{(j)}\|_0 \le T$ means for the j-th sample, its representation coefficients have fewer than T nonzero entries. Among this kind of methods, MOD [15] and KSVD [1] are classic ones because of their simplicity and effectiveness. Note that the two methods solve the same dictionary learning model (1) but differ in the optimization. In order to make model (1) more easily solvable, Mairal et al. [3] approximate the ℓ_0 norm by the ℓ_1 norm and consider the following model:

$$\begin{aligned} &\min_{D,A} \|Y - DA\|_F^2 + \alpha\|A\|_1, \\ &\text{s.t.} \quad \|D_{(i)}\|_2^2 \le 1, \ \forall i \in \{1, 2, \cdots, k\}. \end{aligned} \tag{2}$$

Subsequently, various variants [16, 17, 18, 19] of model (1) and (2) are successfully developed for specific tasks by incorporating different discriminative terms into the objective function in (1) or (2). For instance, to learn a discriminative dictionary for classification tasks, Zhang et al. [24] utilize the label information of training samples and incorporate a classification error term into model (2). But recent works [25, 4, 11] point out that the assumption of Gaussian noise may not be accurate, especially when there are large corruptions and outliers. Besides, in image denoising applications, Gaussian noise based methods may lead to over-smoothness of the images, causing loss of the detail [26].

To overcome the above drawbacks, Laplacian noise based methods [27, 4, 12] are developed. These methods argue that the large corruptions and outliers are approximately drawn from Laplacian distribution and hence can be depicted by the ℓ_1 norm, i.e. absolute error loss function. The model can be formulated as

$$\begin{aligned} &\min_{D,A} \|Y - DA\|_1 + \alpha\|A\|_1, \\ &\text{s.t.} \quad \|D_{(i)}\|_2^2 \leq 1, \ \forall i \in \{1, 2, \cdots, k\}. \end{aligned} \tag{3}$$

Since input data are non-negative values in many real world problems (such as images, text vector, etc.), Pan et al. [12] further add non-negative constraints on the dictionary D and the representation coefficients A in problem (3). Though Laplacian noise based methods have achieved state-of-the-art clustering and denoising performance, the assumption of Laplacian noise does not hold in practice either [11, 20]. Furthermore, since the ℓ_1 norm is not smooth, most approaches [27, 12] typically approximate the ℓ_1 norm with approximative methods, such as the iteratively reweighted norm technique, and thus are not very effective for random-valued impulse noise removal [4].

For depicting noise more accurately, Chen et al. [11] propose a mixed noise based method. They assume that noise is the linear combination of Gaussian and Laplacian noise, and try to depict these two kinds of noise separately. Accordingly, their model is formulated as

$$\begin{aligned} &\min_{D,A,B} \|Y - DA - B\|_F^2 + \alpha\|A\|_1 + \beta\|B\|_1, \\ &\text{s.t.} \quad \|D_{(i)}\|_2^2 \leq 1, \ \forall i \in \{1, 2, \cdots, k\}, \end{aligned} \tag{4}$$

where $E = Y - DA - B$ and B respectively denote Gaussian and Laplacian noise, and α and β are two positive constants. However, noise in real data is sophisticated and it is neither Gaussian nor Laplacian [11, 20]. Thus, using predefined distributions cannot depict noise accurately and may limit the performance of these dictionary learning methods.

3. Dictionary Learning with Structured Noise

In this section, we first present our dictionary learning with structured noise (DLSN) method, then introduce a novel optimization method, and finally analyze the convergence of the proposed algorithm. The dictionaries are denoted as $D_c \in \mathcal{R}^{d\times k_c}$ and $D_s \in \mathcal{R}^{d\times k_s}$ for clean data Y_c and structured noise Y_s, respectively.

3.1. Formulation of DLSN

As aforementioned, many dictionary learning methods assume that the noise in data is drawn from Gaussian or Laplacian distribution or their mixture. However, in real cases, noise is very sophisticated and cannot be handled by predefined distributions. We try to solve this problem from another perspective, rather than fitting the distribution of noise. As shown in Fig. 1, we decompose the original data Y to clean data Y_c, structured noise Y_s, and Gaussian noise Y_g. We learn a dictionary D_c for clean data and an adaptive dictionary D_s for structured noise. Some literature [21, 28, 22, 23] points out that the clean visual data, such as texture, face and motion, can be well characterized by subspaces and the samples in the same class should locate in the same low-dimensional subspace. As the dimension of the subspace corresponds to the rank of the representation matrix, these methods enforce a low-rank constraint on the representation matrix to characterize the subspace structures and enhance the correlation among the representation coefficient vectors at the same time. Inspired by these works, we also adopt the low-rank technique to characterize the inherent structure of clean data. On the other hand, the noise in the data except Gaussian noise is usually structured and contains semantic meanings, e.g., different illuminations, scarfs and glasses on the faces shown in Fig. 1 and 2. Typically, a datum contains several kinds of structured noise. Thus, we try to learn an adaptive dictionary and use a few atoms to linearly represent the structured noise in a datum. As for Gaussian noise $Y_g = Y - Y_c - Y_s$, we employ the ℓ_2 norm to characterize it. Our model can be formulated as follows:

$$\begin{aligned} \min_{\substack{A_c,\, A_s,\\ D_c,\, D_s}} \quad & \frac{1}{2}\|Y - D_cA_c - D_sA_s\|_F^2 + \alpha\mathrm{rank}(A_c) + \beta\|A_s\|_0, \\ \text{s.t.} \quad & D_c \in \Omega, \quad D_s \in \Theta, \end{aligned} \tag{5}$$

where $\Omega = \{D_c \ : \ \|D_{c(i)}\|_2^2 \le 1, i = 1, \cdots, k_c\}$ and $\Theta = \{D_s \ : \ \|D_{s(j)}\|_2^2 \le 1, j = 1, \cdots, k_s\}$. $A_c \in \mathcal{R}^{k_c\times m}$ and $A_s \in \mathcal{R}^{k_s\times m}$ are two coefficient matrices for clean data Y_c and structured noise Y_s, respectively. However, directly solving problem (5) is difficult, since minimization of the rank$(\cdot)$ function and the ℓ_0 pseudonorm is NP hard. We use their convex surrogates, the nuclear norm and

the ℓ_1 norm, to approximate them, respectively. Then, our model can be rewritten as

$$\begin{aligned} \min_{\substack{A_c,\, A_s,\\ D_c,\, D_s}} \quad & \frac{1}{2}\|Y - D_c A_c - D_s A_s\|_F^2 + \alpha\|A_c\|_* + \beta\|A_s\|_1, \\ \text{s.t.} \quad & D_c \in \Omega, \quad D_s \in \Theta. \end{aligned} \tag{6}$$

3.2. Optimization of DLSN

The optimization problem (6) is not jointly convex with respect to (D_c, D_s, A_c, A_s). Empirically, we can develop an iterative minimization method to update each variable alternately. In this paper, based on the proximal alternating linearized minimization (PALM) method [29], we propose an alternating proximal method to solve this problem. We first give a brief introduction to the optimization sketch of the PALM method [29].

3.2.1. Sketch of PALM

Consider the following optimization problem:

$$\min_{x_1,\cdots,x_p} \sum_{i=1}^{p} f_i(x_i) + H(x_1, \cdots, x_p), \tag{7}$$

where H is assumed to be C^1 and each f_i $(i = 1, \cdots, p)$ is a proper and lower semi-continuous function. We denote by $\nabla_i H$ the gradient of H with respect to variable x_i $(i = 1, \cdots, p)$, i.e. the gradient of H with respect to variable x_i when all x_j $(j \neq i)$ are fixed, and L_i $(i = 1, \cdots, p)$ the Lipschitz constant of $\nabla_i H$. Then, using the PALM method to alternately update variables x_i $(i = 1, \cdots, p)$ can be formulated as follows:

$$x_i^{k+1} \in \text{prox}_{e_i^k}^{f_i}(x_i^k - \frac{1}{e_i^k}\nabla_i H(x_1^{k+1}, \cdots, x_{i-1}^{k+1}, x_i^k, \cdots, x_p^k)), \tag{8}$$

where $e_i^k = \max(\gamma_i L_i, \hat{\gamma})$ $(\gamma_i > 1)$ and $\hat{\gamma}$ can guarantee the proximal steps to be well defined. $\text{prox}(\cdot)$ is a proximal operator, which is defined as

$$\text{prox}_{\mu}^{f}(x) = \underset{z}{\text{argmin}}\, f(z) + \frac{\mu}{2}\|z - x\|_2^2. \tag{9}$$

3.2.2. Solving DLSN via PALM

We can follow the sketch of PALM to solve problem (6). We update the variables A_c, A_s, D_c, and D_s alternately at each iteration by minimizing their corresponding proximal problem with other

variables fixed. Firstly, we define

$$\begin{cases} H(A_c, A_s, D_c, D_s) = \frac{1}{2}\|Y - D_c A_c - D_s A_s\|_F^2, \\ f_1(A_c) = \alpha\|A_c\|_*, \\ f_2(A_s) = \beta\|A_s\|_1, \\ f_3(D_c) = I_\Omega(D_c), \\ f_4(D_s) = I_\Theta(D_s), \end{cases} \tag{10}$$

where $I_\Pi(D)$ is the indicator function. That is, if $D \in \Pi$, $I_\Pi(D) = 0$; otherwise, $I_\Pi(D) = +\infty$.

We need to solve the following problems to update these variables in turn.

$$\begin{cases} A_c^{k+1} \in \text{prox}_{e_1^k}^{f_1}(A_c^k - \frac{1}{e_1^k}\nabla_i H(A_c^k, A_s^k, D_c^k, D_s^k)), \\ A_s^{k+1} \in \text{prox}_{e_2^k}^{f_2}(A_s^k - \frac{1}{e_2^k}\nabla_i H(A_c^{k+1}, A_s^k, D_c^k, D_s^k)), \\ D_c^{k+1} \in \text{prox}_{e_3^k}^{f_3}(D_c^k - \frac{1}{e_3^k}\nabla_i H(A_c^{k+1}, A_s^{k+1}, D_c^k, D_s^k)), \\ D_s^{k+1} \in \text{prox}_{e_4^k}^{f_4}(D_s^k - \frac{1}{e_4^k}\nabla_i H(A_c^{k+1}, A_s^{k+1}, D_c^{k+1}, D_s^k)). \end{cases} \tag{11}$$

Actually, the four optimization problems in (11) have closed form solutions, which are given as

$$A_c^{k+1} = \Psi_{\alpha/e_1^k}\left(A_c^k - \nabla_i H(A_c^k, A_s^k, D_c^k, D_s^k)/e_1^k\right), \tag{12}$$

$$A_s^{k+1} = \Phi_{\beta/e_2^k}\left(A_s^k - \nabla_i H(A_c^{k+1}, A_s^k, D_c^k, D_s^k)/e_2^k\right), \tag{13}$$

$$D_c^{k+1} = \Gamma_\Omega\left(D_c^k - \nabla_i H(A_c^{k+1}, A_s^{k+1}, D_c^k, D_s^k)/e_3^k\right), \tag{14}$$

and

$$D_s^{k+1} = \Gamma_\Theta\left(D_s^k - \nabla_i H(A_c^{k+1}, A_s^{k+1}, D_c^{k+1}, D_s^k)/e_4^k\right), \tag{15}$$

where $\Psi_\mu(X)$ is the singular value thresholding (SVT) operator [30] and $\Phi_\mu(X)$ is the soft thresholding [31]. $\Gamma_\Pi(X)$ is an operator that projects the matrix X onto the set Π. In this paper, we have the Lipschitz constants $L_1^k = \|D_c^k\|_2^2$, $L_2^k = \|D_s^k\|_2^2$, $L_3^k = \|A_c^{k+1}\|_2^2$, and $L_4^k = \|A_s^{k+1}\|_2^2$ at each iteration. The detailed optimization procedure of DLSN is presented in **Algorithm 1**.

3.3. Analysis of Algorithm 1

In this section, we first give convergence analysis of our Algorithm 1 and then analyze the computational cost at each iteration and its memory cost.

Algorithm 1 Solving DLSN via PALM

Input: The data matrix Y, the parameters $\alpha > 0$ and $\beta > 0$, and $\varepsilon = 1e-6$.

Initialize: D_c^0, D_s^0, A_c^0, and A_c^0.

While $\| D_c^{k+1} - D_c^k \|_\infty > \varepsilon$ **or** $\| D_s^{k+1} - D_s^k \|_\infty > \varepsilon$ **do**

1. Fix A_s^k, D_c^k, and D_s^k to update A_c^{k+1} by (12).

2. Fix A_c^{k+1}, D_c^k, and D_s^k to update A_s^{k+1} by (13).

3. Fix A_c^{k+1}, A_s^{k+1}, and D_s^k to update D_c^{k+1} by (14).

4. Fix A_c^{k+1}, A_s^{k+1}, and D_c^{k+1} to update D_s^{k+1} by (15).

5. $k \leftarrow k + 1$.

end while

Output: D_c^k, D_s^k, A_c^k, and A_c^k.

3.3.1. Convergence Analysis

We first prove that the sequence generated by our proposed optimization method can be bounded and converge to a critical point, as stated in Theorem 1.

Theorem 1. *Assume that the function* $\phi(A_c, A_s, D_c, D_s) = H(A_c, A_s, D_c, D_s) + f_1(A_c) + f_2(A_s) + f_3(D_c) + f_4(D_s)$ *is our objective function. The sequence* $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ *generated by Algorithm 1 satisfies the following properties:*

1) $\phi(A_c^k, A_s^k, D_c^k, D_s^k)$ *is monotonically decreasing.*

2) The sequence $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ *is bounded.*

3) The sequence $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ *is a Cauchy sequence and converges to a critical point of the optimization problem (6).*

Then, we can prove that the convergence rate of Algorithm 1 is at least sublinear.

Theorem 2. *The sequence* $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ *generated by Algorithm 1 converges to a critical point* $\{(A_c^*, A_s^*, D_c^*, D_s^*)\}$ *of problem (6) in an at least sublinear convergence rate, i.e., there exists* $\omega > 0$, *such that*

$$\|(A_c^k, A_s^k, D_c^k, D_s^k) - (A_c^*, A_s^*, D_c^*, D_s^*)\|_F \leq \omega k^{-\frac{1-\theta}{2\theta-1}}, \tag{16}$$

where $\theta \in (\frac{1}{2}, 1)$.

The proofs of Theorems 1 and 2 can be found in Appendix. So by Theorems 1 and 2, we can know that though problem (6) is non-convex, the proposed algorithm can guarantee that the generated sequence can converge to a critical point with at least a sublinear convergence speed.

3.3.2. Computational and Memory Cost

The computational cost in each iteration and the total memory cost of Algorithm 1 are respectively $\mathcal{O}\left((k_c+k_s)md+k_c^2m\right)$ and $\mathcal{O}\left(d(m+k_c+k_s)+m(k_c+k_s)+k_c^2\right)$. We analyze each step in Algorithm 1. For memory cost, we first need to store $Y\in\mathcal{R}^{d\times m}$, $D_c\in\mathcal{R}^{d\times k_c}$, $D_s\in\mathcal{R}^{d\times k_s}$, $A_c\in\mathcal{R}^{k_c\times m}$ and $A_s\in\mathcal{R}^{k_s\times m}$, which requires $\mathcal{O}\left(d(m+k_c+k_s)+m(k_c+k_s)\right)$ memory. In Step 1 (updating A_c^{k+1} by (12)), we need to compute $T_c^k=A_c^k-\nabla_i H(A_c^k,A_s^k,D_c^k,D_s^k)/e_1^k$ first with computational cost $\mathcal{O}\left((k_c+k_s)dm\right)$ and then compute the skinny singular value decomposition (SVD) of T_c^k, which requires at most $\mathcal{O}\left(k_c^2m\right)$ computational cost. The memory cost of Step 1 is $\mathcal{O}\left(dm+k_c(k_c+m)\right)$. Step 2 updates A_s^{k+1} by the soft thresholding with the computational and memory cost $\mathcal{O}\left((k_c+k_s)dm\right)$ and $\mathcal{O}(dm)$, respectively. Step 3 uses the project gradient method to update D_c^{k+1} with $\mathcal{O}((k_c+k_s)dm)$ computational cost and $\mathcal{O}(dm)$ memory. Similar to Step 3, the computational and memory cost of Step 4 are also $\mathcal{O}\left((k_c+k_s)dm\right)$ and $\mathcal{O}(dm)$, respectively. Therefore, the total computational cost in each iteration is $\mathcal{O}\left((k_c+k_s)md+k_c^2m\right)$. Since Steps 1∼4 update variables sequentially and thus can share memory, the total memory cost is $\mathcal{O}\left(d(m+k_c+k_s)+m(k_c+k_s)+k_c^2\right)$.

The computational cost of our method is comparable to or lower than those of some other dictionary methods, such as OMD [15], KSVD [1], [4], [11], [12], [26]. There are three reasons. (1) Typically, most dictionary learning methods need to compute the residual $Y-DA$ where $Y\in\mathcal{R}^{d\times m}$, $D\in\mathcal{R}^{d\times k}$ and $A\in\mathcal{R}^{k\times m}$ respectively denote the data matrix, a dictionary and a coefficient matrix, which requires the computational cost $\mathcal{O}(kdm)$, and they may also require other computational resources. For instance, KSVD [1] needs to use SVD to update D and A with cost $\mathcal{O}(dm^2)$ in each iteration. (2) In most cases, the dictionary atom number k is usually less than the sample number m and the feature dimension d, which leads to $k_c^2\leq dm$. (3) For fairness, in our experiments the total dictionary atom number for all compared methods is the same, i.e. $k_c+k_s=k$. So the computational cost of our method in each iteration is $\mathcal{O}(kmd)$ and thus is comparable to or lower than those of some other dictionary methods, including [15], [1], [4], [11], [12], [26]. As for memory cost, most dictionary methods need at least $\mathcal{O}(d(m+k)+km)$ memory, since they need to store $Y\in\mathcal{R}^{d\times m}$, $D\in\mathcal{R}^{d\times k}$ and $A\in\mathcal{R}^{k\times m}$ and they may also need other extra memory. Because of reasons (2) and (3), the memory cost of our method is also $\mathcal{O}(d(m+k)+km)$ and hence is comparable to or lower than those of some other dictionary methods, e.g. [15], [1], [4], [11], [12], [26].

4. Experiments

In this section, we compare our proposed dictionary learning method with other state-of-the-art methods. We will first introduce the initialization strategy of our approach and then experiment on the well-known task: data clustering. Extensive experimental results demonstrate the effectiveness and robustness of our method. **Our code will be released online upon acceptance of this paper.**

4.1. Experimental Setting

Initialization and Parameter Setting. As described in Algorithm 1, we need to initialize D_c, D_s, A_c, and A_s first and adopt the following initialization way. We first use Robust PCA [32] to separate the original data into clean data and sparse noise roughly. Then, we apply KSVD [1] to learn two dictionaries: one for clean data and the other for noise. At the same time, KSVD also provides the coefficients A_c and A_s. The iteration number of initializing the dictionary by KSVD is set to only 8, since each iteration in KSVD is time-consuming and our method only needs a rough initialization. In our method, the numbers of atoms for clean data and structured noise are set to 3 and 2 for each class in all experiments, respectively. We empirically set the ranges of parameters α and β as $\alpha \in [0.005, 5]$ and $\beta \in [0.01, 10]$, respectively. We directly use the coefficients A_c for clean data for clustering.

Datesets Description. We evaluate our method on three types of databases: 1) Binary Alphadigits[1], for the handwritten alphabet clustering task; 2) AT&T[2], YaleB [33], AR [34], and PIE [35], for face clustering; 3) Movement[3] [36], for hand movement recognition. As shown in Fig. 2, the difficulties of these four testing face databases are not the same. Each class in the Movement dataset corresponds to a hand movement type in LIBRAS[3]. Table 2 summarizes the experimental settings and the characteristics of all the six datasets. Note that when we conduct experiments on the Binary Alphadigits dataset, we only cluster the 26 alphabets and ignore digits, and when we test the compared methods on AR, we use all images in the first session.

Compared Methods. We compare our method with six state-of-the-art dictionary learning methods: KSVD [1], ODLSC [3], DLINR [4], RDLESD [11], ℓ_1-KSVD [26], and RNNDL [12]. KSVD and ODLSC assume that noise is drawn from Gaussian distribution and they adopt the ℓ_2 norm to characterize it. DLINR, RNNDL, and ℓ_1-KSVD employ the ℓ_1 norm to depict noise, since they suppose that noise is

[1] http://www.cs.toronto.edu/roweis/data.html.
[2] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.
[3] https://archive.ics.uci.edu/ml/datasets/Libras+Movement.

(a) (b) (c) (d)

Figure 2: Some examples in the four face datasets. The images in (a), (b), (c), and (d) are from AT&T, YaleB, AR, and PIE, respectively.

Table 2: Descriptions of the six testing datasets. ("def.", "pos.", "ill.", "exp.", and "occ." are short for "deformation", "pose", "illumination", "expression", and "occlusion", respectively.)

Dataset	Number	Size	#Class	Difficulty
Alphadigits	390	28×23	26	def.
AT&T	400	28×23	40	pos.
YaleB	380	24×21	38	ill.
AR	1300	33×24	100	ill., exp., and occ.
PIE	680	24×24	68	ill., exp., and pos.
Movement	360	90×1	15	def. and pos.

Laplacian. RDLESD divides noise into Gaussian noise and Laplacian noise and utilizes the ℓ_2 and ℓ_1 norms to characterize them respectively. For fairness, the numbers of atoms for each class in KSVD, ODLSC, DLINR, ℓ_1-KSVD, and RDLESD are all set to 5. Therefore their dictionary sizes are equal to that of our method. All methods use kmeans for clustering.

Evaluation Metrics. As in [37, 12], we employ the widely used accuracy (ACC), normalized mutual information (NMI), and purity (PUR) as our evaluation metrics. We also compare the average dictionary learning time of our method with other compared methods. The average dictionary learning time is computed as an average over all the samples. Note that average dictionary learning time of our method includes the initialization time. The experimental clustering results and the average dictionary learning time are summarized in Table 3 and Fig. 3, respectively.

Table 3: Clustering results (ACC, NMI, and PUR) on the six testing datasets. ("GNM", "LNM", "MNM", and "SNM" are respectively short for "Gaussian noise based method", "Laplacian noise based method", "Mixed noise based method", and "Structured noise based method".)

Dataset	Metrics	GNM		LNM			MNM	SNM
		KSVD	ODLSC	DLINR	RNNDL	ℓ_1-KSVD	RDLESD	DLSN (ours)
Alphadigits	ACC	0.3379	0.3484	0.3076	0.3000	0.3282	0.3556	**0.5029**
	NMI	0.4483	0.4513	0.4133	0.4215	0.4577	0.4784	**0.6278**
	PUR	0.3484	0.3574	0.3435	0.3256	0.3615	0.3612	**0.5178**
AT&T	ACC	0.3825	0.3675	0.3700	0.5075	0.4375	0.5700	**0.8050**
	NMI	0.5423	0.4928	0.4833	0.7174	0.5381	0.6829	**0.8884**
	PUR	0.4515	0.4375	0.4400	0.5450	0.4108	0.5375	**0.8250**
YaleB	ACC	0.1805	0.1894	0.1868	0.1789	0.1626	0.1914	**0.2504**
	NMI	0.3021	0.3243	0.2953	0.3772	0.2791	0.3857	**0.4318**
	PUR	0.2026	0.2189	0.2157	0.1895	0.1736	0.2173	**0.2604**
AR	ACC	0.1454	0.1492	0.1476	0.1000	0.1271	0.1531	**0.1908**
	NMI	0.3547	0.3958	0.3568	0.3241	0.3271	0.3911	**0.5048**
	PUR	0.1515	0.1584	0.1584	0.1162	0.1392	0.1601	**0.2023**
PIE	ACC	0.2288	0.2326	0.2138	0.2176	0.2271	0.2382	**0.2676**
	NMI	0.4714	0.3964	0.4136	0.5189	0.5031	0.4976	**0.5662**
	PUR	0.2538	0.2579	0.2676	0.2368	0.2312	0.2691	**0.2956**
Movement	ACC	0.3083	0.3130	0.2944	0.2444	0.2561	0.3316	**0.4138**
	NMI	0.3545	0.3684	0.3248	0.2618	0.2954	0.3923	**0.4555**
	PUR	0.3444	0.3527	0.3277	0.2694	0.2544	0.3494	**0.4500**

4.2. Clustering Results

As Table 3 shows, our method achieves much better clustering results than all the competitors across all datasets on the three evaluation metrics. As shown in Fig. 1, our method decomposes the original data into clean data, structured noise, and Gaussian noise, and then characterizes them separately. Such a mechanism can depict data more precisely and obtain better clustering performance. Unlike KSVD, ODLSC, DLINR, RNNDL, and ℓ_1-KSVD, which solely utilize Gaussian or Laplacian distribution to fit noise, RDLESD fits the real noise with a mixture of Gaussian and Laplacian distribution and performs better than these five methods in most cases. This also demonstrates that characterizing noise more precisely can lead to better results. Fig. 3 shows that our method is the fastest. Our method is about two times faster than the second fastest method, ODLSC, and is at least

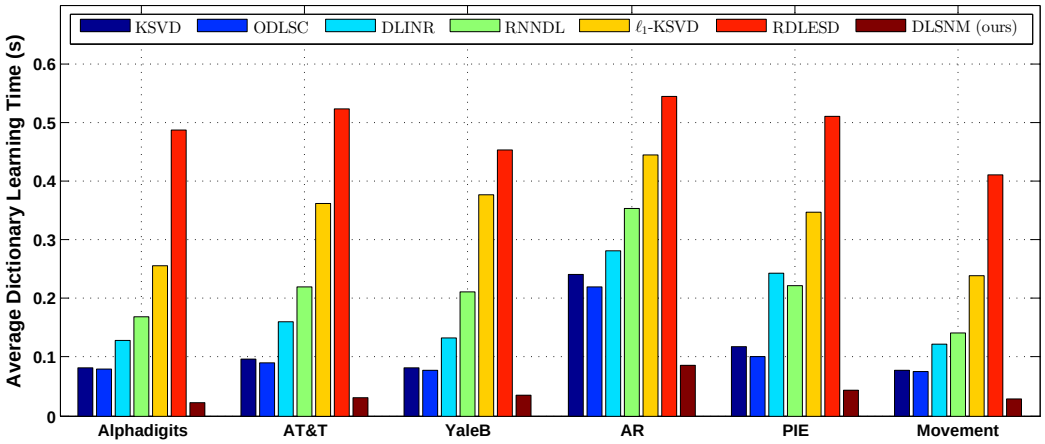
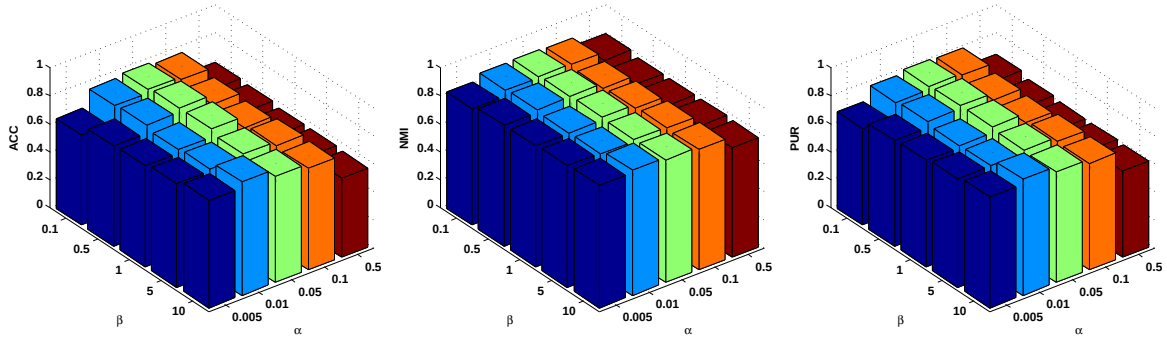


Figure 3: Average dictionary learning time (seconds) on the six testing datasets.



(a) (b) (c)

Figure 4: Effects of parameters α and β on our method on the AT&T dataset.

five times faster than RDLESD on the six testing databases.

We also test the effects of parameter selection on our method. As shown in Fig. 4, though the values of α and β vary within two large ranges respectively, the results of our method on the three metrics (ACC, NMI, and PUR) are relatively stable, which verifies the robustness of our method to parameter selection. To sum up, our method not only achieves better clustering performance but also has good efficiency and robustness to parameter selection.

We conduct experiments to verify that our two techniques, i.e. using low-rank technique to characterize the clean data and our decomposition technique of handling noise, can both benefit the clustering results. We remove the dictionary of structured noise in our model, and use the ℓ_2 norm to depict the noise:

$$\min_{D,A} \frac{1}{2}\|Y - DA\|_F^2 + \alpha\|A\|_*, \quad \text{s.t. } D \in \Omega. \tag{17}$$

Table 4: Comparison of clustering results (ACC, NMI, and PUR) of low-rank based methods on the three testing datasets.

Method	Alphadigits			AT&T			YaleB		
	ACC	NMI	PUR	ACC	NMI	PUR	ACC	NMI	PUR
Robust PCA	0.3634	0.4737	0.3918	0.4712	0.6519	0.5327	0.1953	0.3656	0.1942
LRDL	0.4013	0.5314	0.4451	0.5175	0.7121	0.5859	0.2132	0.3853	0.2102
DLSN (ours)	**0.5029**	**0.6278**	**0.5178**	**0.8050**	**0.8884**	**0.8250**	**0.2504**	**0.4318**	**0.2604**

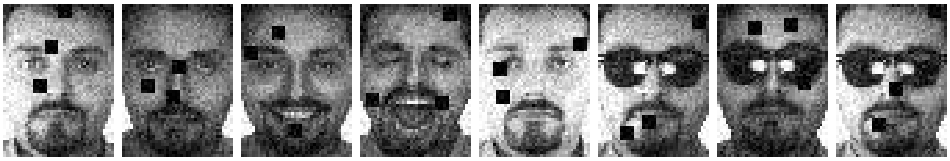


Figure 5: Corrupted examples of AR.

We call this low-rank based dictionary learning method as LRDL and also adopt PALM as its solver. For fairness, we also use KSVD to initialize D in LRDL. Since our method uses Robust PCA which is also a low-rank based method to initialize our solution, we also compare our method with it and simply use the clean data extracted by Robust PCA for clustering. We report the experimental results in Table 4. It can be observed that the low-rank technique can benefit the results, since the comparison between Table 3 and 4 shows that these low-rank based methods perform better than those sparse based ones in most cases. Thus, the low-rank technique is conducive to the improvements of the clustering results. By comparing the clustering results of LRDL and our DLSN in Table 4, we can conclude that our decomposition technique of handling noise can benefit much our method, since DLSN and LRDL only differ in their techniques of handling noise: DLSN decomposes the noise into structured noise and gaussian noise and separately characterizes them, while DLSN only considers gaussian noise.

4.3. Noise Experiments

Finally, we conduct noise experiments to further validate the advantages of our method. We randomly select five classes from the AR dataset [34], and then choose fourteen neutral images plus three with sunglasses for each class. Finally, we add two types of noise to the images: 1) we add Gaussian white noise with SNR 15dB to each image; 2) each image is added three occlusion blocks of size 5×5 and the locations of occlusions are randomly generated. Fig. 5 shows some corrupted images. We also compare our method with other state-of-the-art methods and report ACC, NMI,

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 6: Examples of restoration. (a) is the original corrupted image. (b), (c), (e), (f), (g), and (h) are restorations by KSVD [1], ODLSC [3], DLINR [4], RNNDL [12], ℓ_1-KSVD [26], RDLESD [11], and our DLSN, respectively. Note that KSVD , ODLSC, DLINR , RNNDL, and ℓ_1-KSVD only divide the original data into two parts, clean data and noise. RDLESD and our DLSN further divide the noise into two parts, Laplacian noise and Gaussian noise in RDLESD, structured noise and Gaussian noise in our method. For better viewing, we replace the negative entries in the images with their absolute values when we display the images.

Table 5: Clustering results (ACC, NMI, and PUR) on corrupted AR. ("GNM", "LNM", "MNM", and "SNM" are respectively short for "Gaussian noise based method", "Laplacian noise based method", "Mixed noise based method", and "Structured noise based method".)

Dataset	Metrics	GNM		LNM			MNM	SNM
		KSVD	ODLSC	DLINR	RNNDL	ℓ_1-KSVD	RDLESD	DLSN (ours)
Corrupted AR	ACC	0.5352	0.5588	0.5721	0.5364	0.5852	0.6358	**0.7176**
	NMI	0.5687	0.5141	0.5029	0.5548	0.5832	0.6394	**0.7489**
	PUR	0.5529	0.5941	0.5823	0.5765	0.5570	0.6176	**0.7529**

and PUR in Table 5. It is easy to observe that our method achieves much better performance even though the noise is very complex. Fig. 6 shows the restorations. KSVD [1], ODLSC [3], DLINR [4], RNNDL [12], ℓ_1-KSVD [26], and RDLESD [11] all fail to deal with the occlusion well. No one removes the glass on the face. KSVD, ODLSC, DLINR, RNNDL cannot handle the block at the upper-right corner either, since the pixels near this block are hairs and also black, which increases the difficulty of processing. In contrast, our method characterizes the noise more precisely. It removes not only the glass but also the three blocks. Unlike those methods that use predefined distributions (Gaussian and Laplacian distributions) to fit the real distribution of noise, our method learns an adaptive dictionary for structured noise and adopts Gaussian distribution to characterize the remaining Gaussian noise. Thus, our method is adaptable to more complex noise and achieves much better performance.

5. Conclusion

We present a novel dictionary learning with structured noise (DLSN) method, which provides a new way to handle noisy data. By decomposing the original data into three parts: clean data, structured noise and Gaussian noise, and then characterizing them separately, our method can characterize data

more precisely. Besides, we can prove that our proposed optimization method can converge to a critical point and the convergence rate is at least sublinear. Experimental results demonstrate that our method not only obtains better performance but also runs faster than some state-of-the-art dictionary learning methods on the data clustering task.

Acknowledgements

This research is partially supported by National Key Basic Research Project of China (973 Program) (grant no.s 2015CB352303, 2015CB352502, and 2011CB302400), National Nature Science Foundation (NSF) of China (grant no.s 61071156, 61131003, 61272341, and 61231002), and Microsoft Research Asia Collaborative Research Program.

Appendix

Before we prove Theorems 1 and 2, we first prove two lemmas.

Lemma 1. *$\phi(A_c^k, A_s^k, D_c^k, D_s^k)$ is monotonically decreasing and the sequence $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ generated by Algorithm 1 is bounded.*

Proof. From Lemma 3 in [29], the monotonically nonincreasing property holds for the PALM method, when the step size $e_i = \max(\gamma_i L_i, \hat{\gamma}) > L_i$ $(\gamma_i > 1)$ for all i at each iteration. We have

$$\alpha\epsilon = \phi(A_c^1, A_s^1, D_c^1, D_s^1) \geq \cdots \geq \phi(A_c^k, A_s^k, D_c^k, D_s^k), \tag{18}$$

where ϵ is a constant. Thus, we can obtain that $\frac{1}{2}\|Y - D_c^k A_c^k - D_s^k A_s^k\|_F^2 + \alpha\|A_c^k\|_* + \beta\|A_s^k\|_1 \leq \alpha\epsilon$. So, $\|A_c^k\|_* \leq \epsilon$ holds at all iterations, i.e., $\{\|A_c^k\|_*\}$ can be bounded. Since all norms are equivalent when they are on a finite dimensional space, the sequence $\{\|A_c^k\|_F\}$ is also bounded. Therefore, $\{A_c^k\}$ is bounded. In the same way, we can prove that $\{A_s^k\}$ is also bounded. As $D_c \in \Omega$ and $D_s \in \Theta$, $\{D_c^k\}$ and $\{D_s^k\}$ are bounded. □

Lemma 2. *$\phi(A_c, A_s, D_c, D_s)$ is a semi-algebraic function.*

Before we prove Lemma 2. We first introduce the definition of semi-algebraic function, which is stated as follows:

Definition 1. *[29] A subset S of $\mathcal{R}^d$ is called the semi-algebraic set if there exists a finite number of real polynomial functions g_{ij}, $h_{ij} : \mathcal{R}^d \to \mathcal{R}$, such that*

$$S = \bigcup_j \bigcap_i \{u \in \mathcal{R}^d : g_{ij}(u) = 0, h_{ij}(u) < 0\}. \tag{19}$$

A function $f(u)$ is called the semi-algebraic function if its graph $\{(u,t) \in \mathcal{R}^{d+1}\}$ is a semi-algebraic set.

There is a broad class of functions in semi-algebraic sets and functions. There are two main results, which will be used later. The composition of semi-algebraic functions and $\sup\{g(u,v) : v \in C\}$, when g is semi-algebraic and C is a semi-algebraic set, are all semi-algebraic functions [38]. These two results can be proved by the Tarski-Seidenberg principle: the image of a semi-algebraic set $A \subset \mathcal{R}^{d+1}$ by the projection $\pi : \mathcal{R}^{d+1} \to \mathcal{R}^d$ is semi-algebraic [38, 39, 29].

Now, we prove the objective function $\phi(A_c, A_s, D_c, D_s) = H(A_c, A_s, D_c, D_s) + f_1(A_c) + f_2(A_s) + f_3(D_c) + f_4(D_s)$ is a semi-algebraic function.

Proof. We only need to prove that each function in $\phi(A_c,\ A_s,\ D_c,\ D_s)$ is a semi-algebraic function.
$H(A_c, A_s, D_c, D_s) = \frac{1}{2}\|Y - D_cA_c - D_sA_s\|_F^2$ is a semi-algebraic function, since it is a real polynomial function.
$f_2(A_s) = \beta\|A_s\|_1$ is a semi-algebraic function, as the ℓ_p norm is semi-algebraic when p is rational (see Example 2 in [29]).
By Definition 1, $\Omega = \{D_c : \|D_{c(i)}\|_2^2 \leq 1, i = 1, \cdots, k_c\} = \bigcap_{i=1}^{k_c}\{D_c : \|D_{c(i)}\|_2^2 \leq 1\}$ is a semi-algebraic set. Thus, $\Theta = \{D_s : \|D_{s(j)}\|_2^2 \leq 1, j = 1, \cdots, k_s\}$ is also a semi-algebraic set. As $f_3(D_c) = I_\Omega(D_c)$ and $f_4(D_s) = I_\Theta(D_s)$ are indicator functions of the two sets Ω and Θ, respectively, f_3 and f_4 are two semi-algebraic functions.
To prove $f_1(A_c)$ is a semi-algebraic function, we first prove that the spectral norm $\sigma_1(Q) = \max\{\lambda_i(Q^TQ)\}$ is a semi-algebraic function. The spectral norm of $Q \in \mathcal{R}^{m\times n}$ can be rewritten as:

$$\sigma_1(Q) = \sup\{\|Qx\|_2 : \|x\|_2 \leq 1\}. \tag{20}$$

Since the ℓ_2 norm of a vector is a polynomial function and $\sigma_1(Q)$ is a sup type function which is a semi-algebraic function [38], $\sigma_1(Q)$ is a semi-algebraic function. By Definition 1, $\{(Q, y) \in \mathcal{R}^{m\times n+1} : y - \sigma_1(Q) = 0\}$ is a semi-algebraic set. As all semi-algebraic sets are stable under finite unions, $\{Q \in \mathcal{R}^{m\times n} : \sigma_1(Q) \leq 1\}$ is a semi-algebraic set. Applying the Tarski-Seidenberg principle, $\{Q \in \mathcal{R}^{m\times n} : \sigma_1(Q) \leq 1\}$ is a semi-algebraic set. The nuclear norm is dual to the spectral norm [40], which can be formulated as:

$$\|A\|_* = \sup\{\mathrm{Tr}(Q^TA) : \sigma_1(Q) \leq 1\}, \tag{21}$$

where $\mathrm{Tr}(\cdot)$ is the trace operator. Since $\mathrm{Tr}(Q^TA)$ is a polynomial function and $\{Q \in \mathcal{R}^{m\times n} : \sigma_1(Q) \leq 1\}$ is a semi-algebraic set, the nuclear norm is a semi-algebraic function. Thus $f_1(A_c)$ is a semi-algebraic function. □

Theorem 1 is based on a proposition in [29], which can be stated as follows:

Proposition 1. *[29] Suppose that* $F(x_1, ..., x_p) = \sum_{i=1}^{p} f_i(x_i) + H(x_1, ..., x_p)$ *is the objective function. The sequence* $\{(x_1^k, ..., x_p^k)\}$ *generated by PALM [29] is a Cauchy sequence, such that*

$$\sum_{k=1}^{\infty} \|(x_1^{k+1}, ..., x_p^{k+1}) - (x_1^k, ..., x_p^k)\| < \infty,$$

and the sequence $\{(x_1^k, ..., x_p^k)\}$ *converges to a critical point* $(x_1^*, ..., x_p^*)$ *of* F *if* F *satisfies the following properties:*

(1) $H(x_1, ..., x_p)$ *is Lipschitz continuous;*

(2) $f_i(x_i)$ *is a proper and lower semi-continuous function with* $\inf f_i > \infty$*;*

(3)$F(x_1, ..., x_p)$ *satisfies the Kurdyka-Lojasiewicz (KL) properties [41];*

(4) The sequence $\{(x_1^k, ..., x_p^k)\}$ *is bounded.*

Now, we prove Theorem 1.

Proof. From Lemma 1, the properties (1) and (2) in Theorem 1 hold. To prove the property (3) in Theorem 1, we only need to prove that $\phi(A_c, A_s, D_c, D_s)$ satisfies the four conditions in Proposition 1. $\phi(A_c, A_s, D_c, D_s)$ meets the first two conditions obviously. Since all semi-algebraic functions satisfy the Kurdyka-Lojasiewicz (KL) properties by Theorem 3 in [29] and $\phi(A_c, A_s, D_c, D_s)$ is a semi-algebraic function by Lemma 2, $\phi(A_c, A_s, D_c, D_s)$ satisfies the Kurdyka-Lojasiewicz (KL) properties. From Lemma 1, $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ is bounded. Thus, the sequence $\{(x_1^k, ..., x_p^k)\}$ is a Cauchy sequence and it converges to a critical point, i.e., the property (3) in Theorem 1 holds. □

Before we prove Theorem 2, we introduce a proposition [42, 29], which is described as follows:

Proposition 2. *[42, 29] Assume that the objective function* $F(x_1, ..., x_p)$ *satisfies the following conditions:*

(1) $\inf F > -\infty$*;*

(2) The restriction of F *to its domain is a continuous function;*

(3) F *satisfies the Lojasiewicz property: for any limiting-critical point* $(x_1^*, ..., x_p^*)$*, there exist* C*,* $\epsilon > 0$*, and* $\theta \in [0, 1)$ *such that*

$$\|F(x_1, ..., x_p) - F(x_1^*, ..., x_p^*)\|^\theta \leq C\|(x_1^*, ..., x_p^*)\|,$$

for $\forall (x_1, ..., x_p) \in B((x_1^*, ..., x_p^*), \epsilon)$*,* $\forall (x_1^*, ..., x_p^*) \in crit\ F$*, where crit* F *denotes the critical point set of* F*;*

(4) The sequence $\{(x_1^k, ..., x_p^k)\}$ generated by PALM [29] is bounded.

Then the sequence $\{(x_1^k, ..., x_p^k)\}$ convergences in an at least sublinear rate. Generally, we meet the worse case that $\theta \in (\frac{1}{2}, 1)$, there exists $\omega > 0$ such that

$$\|(x_1^k, ..., x_p^k) - (x_1^*, ..., x_p^*)\| \leq \omega k^{-\frac{1-\theta}{2\theta-1}}.$$

Now, we prove Theorem 2.

Proof. We need to prove that $\phi(A_c, A_s, D_c, D_s)$ satisfies the four conditions in Proposition 2. $\phi(A_c, A_s, D_c, D_s)$ satisfies the first two conditions obviously. Besides, $\phi(A_c, A_s, D_c, D_s)$ satisfies the Lojasiewicz property, since it is a semi-algebraic function (see Example 1(b) in [42]). Incorporating with the boundedness of the sequence $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ in Lemma 1, we can obtain that the sequence $\{(A_c^k, A_s^k, D_c^k, D_s^k)\}$ converges to a critical point in an at least sublinear rate. □

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. on Signal Processing*, 54(11):4311–4322, 2006.

[2] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proc. Int'l Conf. Machine Learning*, pages 487–494, 2010.

[3] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proc. Int'l Conf. Machine Learning*, pages 689–696, 2009.

[4] S. Wang, Q. Liu, Y. Xia, P. Dong, J. Luo, Q. Huang, and D. Feng. Dictionary learning based impulse noise removal via ℓ_1-ℓ_1 minimization. *IEEE Trans. on Signal Processing*, 93(9):2696–2708, 2013.

[5] Y. Yang, L. Liu, X. Qian, T. Mei, J. Shen, and Q. Tian. Mobile visual search via hievarchical sparse coding. In *Int'l Conf. Multimedia and Expo*, pages 1–6, 2014.

[6] T. Guan, Y. Wang, L. Duan, and R. Ji. On-device mobile landmark recognition using binarized descriptor with multifeature fusion. *ACM Trans. on Intelligent Systems and Technology*, 7(1):12, 2015.

[7] B. Wei, T. Guan, L. Duan, J. Yu, and T. Mao. Wide area localization and tracking on camera phones for mobile augmented reality systems. *Multimedia Systems*, 21(4):381–399, 2015.

[8] B. Wei, T. Guan, and J. Yu. Projected residual vector quantization for ann search. *IEEE multimedia*, 21(3):41–51, 2014.

[9] Y. Zhang, T. Guan, L. Duan, B. Wei, J. Gao, and T. Mao. Inertial sensors supported visual descriptors encoding and geometric verification for mobile visual location recognition applications. *Signal Processing*, 112:17–26, 2015.

[10] Z. Wang, J. Yu, Y. He, and T. Guan. Affection arousal based highlight extraction for soccer video. *Multimedia Tools and Applications*, 73(1):519–546, 2014.

[11] Z. Chen and Y. Wu. Robust dictionary learning by error source decomposition. In *Proc. IEEE Int'l. Conf. Computer Vision*, pages 2216–2223, 2013.

[12] Q. Pan, D. Kong, C. Ding, and B. Luo. Robust non-negative dictionary learning. In *AAAI Conf. Artificial Intelligence*, pages 2027–2033, 2014.

[13] E. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[14] J. Yang and M. Yang. Top-down visual saliency via joint CRF and dictionary learning. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2296–2303, 2012.

[15] K. Engan, S. Aase, and J. Husoy. Frame based signal compression using method of optimal directions (MOD). In *Proc. IEEE Int'l Symp. Circuits and Systems*, pages 1–4, 1999.

[16] Z. Jiang, Z. Lin, and L. Davis. Label consistent K-SVD: Learning a discriminative dictionary for recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 35(11):2651–2664, 2013.

[17] H. Zhang, Y. Zhang, and T. Huang. Simultaneous discriminative projection and dictionary learning for sparse representation based classification. *Pattern Recognition*, 46(1):346–354, 2013.

[18] Z. Feng, M. Yang, L. Zhang, Y. Liu, and D. Zhang. Joint discriminative dimensionality reduction and dictionary learning for face recognition. *Pattern Recognition*, 46(8):2134–2143, 2013.

[19] Y. Yang, Q. Zhu, X. Mao, and L. Pan. Visual feature coding for image classification integrating dictionary structure. *Pattern Recognition*, 48(10):3067–3075, 2015.

[20] J. Chen and J. Yang. Robust subspace segmentation via low-rank representation. *IEEE Trans. on Cybernetics*, 44(8):1432–1445, 2014.

[21] Y. Ma, H. Derksen, W. Hong, and H. Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(9):1546–1562, 2007.

[22] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

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

[24] Q. Zhang and B. Li. Discriminative K-SVD for dictionary learning in face recognition. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2691–2698, 2010.

[25] A. Jalali, S. Sanghavi, P. Ravikumar, and R. Chao. A dirty model for multi-task learning. In *Proc. Conf. Neutral Information Processing Systems*, pages 41–75, 2010.

[26] S. Mukherjee, R. Basu, and C. Seelamantula. ℓ_1-K-SVD: A robust dictionary learning algorithm with simultaneous update. *Eprint Arxiv*, 2014.

[27] A. Banerjee, H. Wang, and P. Melville. Online ℓ_1-dictionary learning with application to novel document detection. In *Proc. Conf. Neutral Information Processing Systems*, pages 2258–2266, 2012.

[28] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(10):1832–1845, 2009.

[29] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2):459–494, 2014.

[30] J. Cai, E. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, 2008.

[31] D. Donoho and I. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90(432):1200–1224, 1999.

[32] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(11), 2011.

[33] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[34] A. Martinez and R. Benavente. The AR face database. *CVC Tech. Rep. 24*, 1998.

[35] T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination, and expression database. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2003.

[36] B. Daniel, C. Renata, R. Thiago, B. Helton, and M. Sarajane. Hand movement recognition for Brazilian Sign Language: A study using distance-based neural networks. In *Proc. Int'l Conf. Neural Networks*, pages 697–704, 2009.

[37] Deguang Kong, Chris Ding, and Heng Huang. Robust nonnegative matrix factorization using ℓ_{21}-norm. In *Proc. IEEE Int'l. Conf. on Information and Knowledge Management*, pages 673–682, 2011.

[38] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Lojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010.

[39] H. Attouch, J. Bolte, and B. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Mathematical Programming*, 137(1-2):91–129, 2013.

[40] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[41] J. Bolte, A. Daniilidis, A. Lewis, and M. Shiota. Clarke subgradients of stratifiable functions. *SIAM J. on Optimization*, 18(2):556–572, 2007.

[42] H. Attouch and J. Bolte. On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Mathematical Programming*, 116(1-2):5–16, 2009.