Learning Structured Low-Rank Representation via Matrix Factorization

Jie Shen
Department of Computer Science
School of Arts and Sciences
Rutgers University
Piscataway, NJ 08854, USA
js2007@rutgers.edu

Ping Li
Department of Statistics and Biostatistics
Department of Computer Science
Rutgers University
Piscataway, NJ 08854, USA
pingli@stat.rutgers.edu

Abstract

A vast body of recent works in the literature have shown that exploring structures beyond data low-rankness can boost the performance of subspace clustering methods such as Low-Rank Representation (LRR). It has also been well recognized that the matrix factorization framework might offer more flexibility on pursuing underlying structures of the data. In this paper, we propose to learn structured LRR by factorizing the nuclear norm regularized matrix, which leads to our proposed non-convex formulation NLRR.

Interestingly, this formulation of NLRR provides a general framework for unifying a variety of popular algorithms including LRR, dictionary learning, robust principal component analysis, sparse subspace clustering, etc. Several variants of NLRR are also proposed, for example, to promote sparsity while preserving low-rankness. We design a practical algorithm for NLRR and its variants, and establish theoretical guarantee for the stability of the solution and the convergence of the algorithm. Perhaps surprisingly, the computational and memory cost of NLRR can be reduced by roughly one order of magnitude compared to the cost of LRR. Experiments on extensive simulations and real datasets confirm the robustness of efficiency of NLRR and the variants.

1 Introduction

In the face of big data, a fundamental component in scientific and engineering applications is to find a compact representation for the underlying data. To tackle problems in different scenarios, a large number of celebrated models have been investigated, including dimension reduction [19], subspace clustering [40], dictionary learning [29], matrix factorization [21], compressed sensing [9], etc. In this paper, we are particularly interested in two quintessential problems: (i) subspace clustering; and (ii) matrix factorization.

Subspace Clustering. It is well known that when the data lie in a low-dimensional subspace, principal component analysis (PCA) [19] and robust PCA [7] are effective tools for finding the best fitted subspace. More generally, the data may be collected from different sources, yielding the research on subspace clustering [40], whose goal is to segment the data into the correct subspace they belong to.

In the last decade, a considerable number of algorithms for subspace clustering have been presented in the literature. For instance, Generalized PCA (GPCA) [42] attempted to extend the classical PCA to the regime of multiple subspaces. Sparse Subspace Clustering (SSC) [11] pursued sparse representation for each sample with respect to the remaining, and the robustness of SSC was further analyzed in [38]. In [27], Liu et al. considered a convex program termed Low-Rank Representation (LRR) which is an extension of Robust PCA to the subspace clustering problem. Compared to GPCA, LRR is guaranteed with robust segmentation under some mild conditions.

Structured Matrix Factorization. In parallel to the literature of subspace clustering, matrix factorization (MF) is another emerging field that finds successful applications in dictionary learning [29, 28], recommender system [21], imaging science [30], etc. Recently, some researchers have attempted to figure out how the factors can be manipulated so as to produce *structured* solution of MF. For instance, [3] showed that as long as the intrinsic dimension of the factors is allowed to expand infinitely, some structured MF formulations are equivalent to a convex program. [15] extended the analysis of [3] and demonstrated the efficacy of structured MF on image processing problems.

LRR and MF. Notably, the results obtained in [3, 15] indicate that by imposing various kinds of constraints on the factors, it is possible to recover a structured subspace comparable to PCA, Sparse PCA [18, 20] and Robust

Appearing in Proceedings of the 19^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2016, Cadiz, Spain. JMLR: W&CP volume 51. Copyright 2016 by the authors.

PCA. However, it is not straightforward to extend the results to the setting of multiple subspaces (see details in Section 2). On the other hand, as researchers have observed [44, 25, 24], one may improve the clustering performance of LRR by imposing structures beyond the low-rankness. Nevertheless, those models only work with the primal variables of LRR, which is not flexible compared to the factors [15] and is thus less effective in practice.

Summary of Contributions. In this paper, motivated by recent progress in structured LRR [44, 25] and matrix factorization [3, 15], we propose incorporating MF into LRR. Interestingly, by further introducing a dummy but crucial variable, we can unify LRR and other important algorithms including dictionary learning (DL) [28], RPCA [7] and SSC [11, 12]. This connection suggests several structured variants of non-convex LRR (NLRR) such as Elastic NLRR and Lasso NLRR, which are capable of producing sparse solutions besides the low-rankness. Remarkably, such variants are only applicable to LRR but RPCA, due to the regime of multiple subspaces of LRR.

Next, since it is computationally and memory expensive to optimize an NLRR program, we devise a scalable implementation which is one order of magnitude more efficient in terms of both time complexity and storage cost compared to LRR. Parallel to the Elastic Net [50], we show that the solution of NLRR and Elastic NLRR is stable and converges to a stationary point. Finally, we investigabte the performance of NLRR and its variants with synthetic and realistic datasets, justifying the robustness and efficiency.

More Discussions on Related Works. Matrix factorization has been adopted in many applications such as collaborative filtering [21]. It was showed in [3, 15] that an MF problem could be equivalent to a convex program as long as the intrinsic dimension is allowed to go to infinity. [41] studied a non-convex low-rank subspace clustering problem in the context of matrix factorization for clean data, while that of our work emerges in the representation matrix. Due to the non-convexity nature, a large body of works are devoted to characterizing the conditions for global optimum [5, 6, 1, 17]. Another interesting line is bi-directional random projection for efficient MF learning, e.g., [43].

More in line with our work is subspace clustering. Some prior works concern the theoretical analysis of subspace clustering algorithms. For example, [37] considered the robustness of SSC in the presence of outliers from a geometric view. The results therein were subsequently extended to the Gaussian noise [38]. Most of the theoretical guarantees of LRR are derived in [26, 25], where [26] established exact recovery for outliers and [25] tackled the coherent data raised by [8, 7]. There are also some interesting works such as [2] that considered a probabilistic formulation for subspace clustering. Other works addressed practical issues like missing entries [47] and computational efficiency [48].

2 Problem Formulation

Notation. We use bold lowercase letters to denote vectors and capital letters for matrices. For a vector $\boldsymbol{v}$, we denote its ℓ_1 norm and ℓ_2 norm by $\|\boldsymbol{v}\|_1$ and $\|\boldsymbol{v}\|_2$, respectively. For a matrix M, $\|M\|_*$ denotes the nuclear norm, which is a sum of its singular values. $\|M\|_F$ denotes the Frobenius norm and $\|M\|_1$ is used to denote the matrix ℓ_1 norm seen as a long vector. The ith row and jth column of a matrix M are denoted by $\boldsymbol{m}(i)$ and $\boldsymbol{m}_j$, respectively. The trace of a squared matrix M is denoted by $\mathrm{Tr}(M)$. The capital letter I is reserved for the identity matrix, and its subscript variant (e.g.,) I_n indicates the size of $n \times n$.

Let $Z \in \mathbb{R}^{p\times n}$ be the observation matrix with n samples in p dimensions generated from a union of subspaces. The data may be corrupted by some sparse noise $E \in \mathbb{R}^{p\times n}$, and LRR aims to robustly segment the data into their own subspace by solving the following convex program:

$$\min_{X,E} \frac{\beta}{2}\|Z - AX - E\|_F^2 + \|X\|_* + \lambda\|E\|_1, \tag{2.1}$$

where β and λ are two tunable parameters. To understand LRR, note that $C \overset{\text{def}}{=} AX$ is the clean data we aim to recover [27] where A is some given dictionary (typically equal to Z). As the inequality $\mathrm{rank}(AX) \leq \mathrm{rank}(X)$ always holds, minimizing the nuclear norm of X amounts to bounding the rank of C, hence a low-rank structure.

In this paper, we propose a new formulation for LRR, for handling data which exhibit extra structure beyond low-rankness. We utilize a standard reformulation of the nuclear norm which is shown to be more flexible to manipulate [3, 15]. Specially, as shown in [13, 33], we have

$$\|X\|_* = \min_{U,V,X=UV^\top} \frac{1}{2}\left(\|U\|_F^2 + \|V\|_F^2\right), \tag{2.2}$$

where $U \in \mathbb{R}^{n\times d}$ and $V \in \mathbb{R}^{n\times d}$, and d is an upper bound on the rank of the matrix X (thus the rank of the clean data C is bounded by d). Note that d is typically picked as a smaller value than p since C is assumed to be low-rank.

Plugging (2.2) back to (2.1), we have an *equivalent but non-convex* formulation of LRR:

$$\text{(NLRR)} \min_{U,V,E} \frac{\beta}{2}\left\|Z - AUV^\top - E\right\|_F^2 + \frac{1}{2}\|U\|_F^2 + \frac{1}{2}\|V\|_F^2 + \lambda\|E\|_1. \tag{2.3}$$

Note that (2.3) and (2.1) are equivalent in the sense that they can attain the same minimal objective value. In the sequel, we will pose our main observation which establishes the connection between LRR, RPCA [7], DL [28] and SSC [11], hence motivating new variants of NLRR.

Crucial Observation 1. Putting $D = AU \in \mathbb{R}^{p\times d}$ gives

$$C = DV^\top. \tag{2.4}$$

Since C is the clean data, D can be accounted as a "basis dictionary" of the multiple subspaces and V is the associated coefficients, with $\boldsymbol{v}(j)$ being the coefficients for $\boldsymbol{c}_j$. Also by the above equation, we know that d should be chosen as large as the rank of C.

Remark. Similar idea was also shown in [35], but they focused on online algorithm for LRR while this work is devoted to structured LRR. Another difference is that the theoretical analysis of [35] was carried out to establish the convergence of online LRR, which that of our work justifies the stability of NLRR and its variants.

Plugging (2.4) back to (2.3), we have

$$\begin{aligned}\min_{D,U,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \frac{1}{2}\left\|U\right\|_F^2 \\ & + \frac{1}{2}\left\|V\right\|_F^2 + \lambda\left\|E\right\|_1, \\ \text{s.t. } & D = AU.\end{aligned} \tag{2.5}$$

Let $\mathcal{D}_{\text{NLRR}} = \{D \mid D = AU, \left\|U\right\|_F \leq \bar{u}\}$ and carefully choosing $\bar{u}$, we have an equivalent problem to the above:

$$\begin{aligned}\min_{D,U,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \frac{1}{2}\left\|V\right\|_F^2 + \lambda\left\|E\right\|_1, \\ \text{s.t. } & D \in \mathcal{D}_{\text{NLRR}}.\end{aligned} \tag{2.6}$$

2.1 Connections to Other Methods

Robust PCA. By applying the reformulation (2.2), Feng et al. [14] presented a non-convex formulation of RPCA which can be solved in online fashion:

$$\begin{aligned}\min_{D,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \frac{1}{2}\left\|V\right\|_F^2 + \lambda\left\|E\right\|_1, \\ \text{s.t. } & D \in \mathcal{D}_{\text{NRPCA}},\end{aligned} \tag{2.7}$$

where $\mathcal{D}_{\text{NRPCA}} = \{D \mid D = U, \left\|U\right\|_F \leq \tilde{u}\}$ for some constant $\tilde{u}$. Note that we use the term "NRPCA" to differentiate it from the classical Robust PCA (RPCA) formulation.

Dictionary Learning. Recall the formulation of DL [28]:

$$\begin{aligned}\min_{D,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \left\|V\right\|_1 + \lambda\left\|E\right\|_1, \\ \text{s.t. } & D \in \mathcal{D}_{\text{DL}},\end{aligned} \tag{2.8}$$

where $\mathcal{D}_{\text{DL}} = \{D \mid \left\|\boldsymbol{d}_j\right\|_2 \leq 1,\ \forall\, 1 \leq j \leq d\}$. Here, D is overcomplete, i.e., the number of columns of D is much more than the number of rows, hence a sparse solution V.

Sparse Subspace Clustering. Instead of pursuing a low-rank representation matrix, SSC aims to find the most sparse representation for each sample with respect to the others. To this end, SSC solves the following program:

$$\begin{aligned}\min_{D,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \left\|V\right\|_1 + \lambda\left\|E\right\|_1, \\ \text{s.t. } & D \in \mathcal{D}_{\text{SSC}},\ v_{jj} = 0,\ \forall\, 1 \leq j \leq n,\end{aligned} \tag{2.9}$$

where $\mathcal{D}_{\text{SSC}} = \{D \mid D = Z\}$. Note that SSC is actually convex since D is identical to the data matrix Z.

Comparing (2.6), (2.8), (2.7) and (2.9), we find that there are essentially two key differences among them:

1. **The way how the dictionary D is constrained.** For NLRR and NRPCA, D is restricted to be equal to a product of two matrices, one of which is further constrained on its Frobenius norm. For DL, D is column-wisely restricted. SSC simply imposes the identity to Z.
2. **The way how the coefficient V is regularized.** For NLRR and NRPCA, V is regularized by the Frobenius norm due to the formula (2.2). For DL and SSC, ℓ_1 norm is used to promote sparsity owing to the overcompleteness of D for DL and self-expressiveness of SSC.

Jointly studying the ways how D is constrained and how V is regularized offers a unified view to distinguish the problem structure of these methodologies. Such connection paves the way for us to derive new variants of NLRR.

2.2 Variants of NLRR

2.2.1 Sparsity Induced Variants

Crucial Observation 2. In (2.4), we have justified the use of D as a basis dictionary of the clean data. Note that in subspace clustering, we are considering the regime of *multiple subspaces*. Particularly, for LRR, it assumes they are mutually disjoint [27]. This implies that for any given sample, its support w.r.t. D is supposed to be *sparse* where only the subspace involved has non-zero coefficients.

Thus, to comply with the problem structure of LRR, it is desirable to encourage sparsity patterns on V in (2.5):

$$\begin{aligned}(\text{NLRR}_{21})\quad \min_{D,U,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \frac{1}{2}\left\|U\right\|_F^2 \\ & + \frac{1}{2}\left\|V\right\|_F^2 + \lambda_1\left\|V\right\|_1 + \lambda_2\left\|E\right\|_1, \\ \text{s.t. } & D = AU.\end{aligned} \tag{2.10}$$

We call the above as Elastic NLRR (NLRR$_{21}$), owing to its similar form as the Elastic Net [50]. In Section 4, we also show that NLRR$_{21}$ enjoys similar stability property as of Elastic Net, while promotes sparsity patterns in practice.

In addition to Elastic NLRR, we may also consider Lasso NLRR, which performs like Lasso and DL that penalize the variable only by the ℓ_1 norm, hence is more aggressive.

$$\begin{aligned}(\text{NLRR}_{1})\quad \min_{D,U,V,E}\ & \frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \frac{1}{2}\left\|U\right\|_F^2 \\ & + \lambda_1\left\|V\right\|_1 + \lambda_2\left\|E\right\|_1, \\ \text{s.t. } & D = AU.\end{aligned} \tag{2.11}$$

Remark. Although similar with NLRR, it may not be suitable for NRPCA to introduce a sparse penalty because NRPCA addresses the data from a single subspace, for which

V could be entirely dense if each sample is a combination of *all* bases of the underlying subspace. Thus, the structure of D and V also provides a new insight on the difference of subspace recovery and subspace clustering.

2.2.2 The Max-Norm Variant

The max-norm regularizer is another popular surrogate to the rank function. It has been shown to enjoy tighter generalization error bound compared to the nuclear norm [23], as well as perform better than the nuclear norm in applications such as subspace recovery [36]. It is defined as follows:

$$\|X\|_{\max} \overset{\text{def}}{=} \min_{U,V,X=UV^\top} \|U\|_{2,\infty} \cdot \|V\|_{2,\infty}, \qquad (2.12)$$

where $\|\cdot\|_{2,\infty}$ denotes the maximum ℓ_2 row norm. Intuitively, the nuclear norm (2.2) constrains the row norms of U and V on average, while the max-norm constrains the largest ℓ_2 row norm, hence is tighter. Interestingly, to obtain a max-norm regularized variant of LRR, we only need to replace $\|V\|_F^2$ with $\|V\|_{2,\infty}$, and redefine the set of D:

$$\mathcal{D}_{\max} = \{D \mid D = AU,\ \|U\|_{2,\infty} \le 1\}.$$

See Proposition 2.1 in [36] for the proof.

3 Algorithm

3.1 A Naive Approach

For this non-convex LRR problem, there is a seemingly straightforward paradigm for solving Problem (2.3). That is, updating the three variables U, V and E in an alternating manner until convergence. We argue that while (2.3) appears amenable for alternating minimization, it is computationally intensive even for a medium scale problem. To see this, let us examine the computational cost of evaluating a local minimizer U given V and E.

Lemma 3.1. *Assume the variable V and E are fixed in Problem* (2.3)*, then the optimal minimizer U is given by:*

$$\begin{aligned} &vec(U) \qquad (3.1)\\ &= \left((V^\top V) \otimes (A^\top A) + \beta^{-1} I_{dn}\right)^{-1} vec(A^\top (Z-E)V), \end{aligned}$$

where "$\otimes$" denotes the Kronecker product and $vec(U)$ stacks all the columns of U in a long vector.

Proof: *See Appendix A.*

Note that the size of $(V^\top V)\otimes(A^\top A)$ is $dn \times dn$ since $V \in \mathbb{R}^{n\times d}$ and $A \in \mathbb{R}^{p\times n}$. Due to the inverse computation of a $dn\times dn$ matrix, the complexity of solution (3.1) is $O(d^3n^3)$ and the memory usage is $O(d^2n^2)$, which are only practical in small scale problems. In modern applications, n (the number of samples) can be very large.

Next, to make NLRR applicable to large-scale applications, we propose a simple and efficient technique to accelerate the computation and reduce the memory cost.

3.2 A More Scalable Implementation

For the sake of mitigating the computational and memory issue, we suggest to solve the optimization problem (2.5) rather than the primal NLRR formulation (2.3). The reasons for shifting to (2.5) is two-fold:

1. It is equivalent to (2.3). In fact, compared to (2.3), it only introduces a dummy variable $D = AU$ and thus changes nothing on the optimality of (2.3).

2. As we showed, the computation of solution (3.1) is dominated by the matrix inverse of a Kronecker product, which is dated back to the *simultaneous* left and right multiplications on U, i.e., the term $AUV^\top$ in (2.3). If the left and right multiplications are separated, then we do not need to compute the Kronecker product and hence a more efficient derivation for U.

Motivated by these observations, we now focus on (2.5). To this end, we introduce the augmented Lagrangian function:

$$\begin{aligned} &g(D,U,V,E,W,\mu) \qquad (3.2)\\ =&\frac{\beta}{2}\left\|Z - DV^\top - E\right\|_F^2 + \frac{1}{2}\|V\|_F^2 + \lambda_2\|E\|_1 + \frac{1}{2}\|U\|_F^2\\ &+ \langle W, D - AU\rangle + \frac{\mu}{2}\|D - AU\|_F^2. \end{aligned}$$

Solve U. Differentiating $g(D,U,V,E,W,\mu)$ with respect to U and arranging the other terms yields

$$U = \left(\mu A^\top A + I_n\right)^{-1} A^\top (W + \mu D),$$

whose computational complexity is $O(n^3)$ and memory cost is $O(n^2)$. Recall those of (3.1) are $O(d^3n^3)$ and $O(d^2n^2)$ respectively. Hence, we improve both of them.

Although we have reduced the cost, the dependence on the cube of n is still undesirable. Fortunately, we can update the columns of U in a stochastic block coordinate fashion, which will dramatically improve the computation. To see this, let $S_i = AU - \boldsymbol{a}_i\boldsymbol{u}(i)$ and note that we have:

$$\begin{aligned} &\frac{1}{2}\|U\|_F^2 + \langle W, D - AU\rangle + \frac{\mu}{2}\|D - AU\|_F^2\\ =&\frac{1}{2}\sum_{i=1}^n \|\boldsymbol{u}(i)\|_2^2 + \langle W, D - S_i - \boldsymbol{a}_i\boldsymbol{u}(i)\rangle\\ &+\frac{\mu}{2}\|D - S_i - \boldsymbol{a}_i\boldsymbol{u}(i)\|_F^2. \end{aligned}$$

Assume all but the ith row of U have been computed. This assumption always holds since we can initialize U in advance. Then we can update $\boldsymbol{u}(i)$'s sequentially as follows:

$$\boldsymbol{u}(i) = \frac{\mu\boldsymbol{a}_i^\top(D - S_i) + \boldsymbol{a}_i^\top W}{1 + \mu\|\boldsymbol{a}_i\|_2^2}, \qquad (3.3)$$

which just involves simple matrix-vector multiplication. Since the objective is strongly convex over U, the block coordinate method will converge to the local minimum [4].

Remark. Note that the time complexity of (3.3) is $O(pd)$. Hence, a one-pass update on U only costs $O(npd)$. Also, as the objective is strongly convex w.r.t. U, the total iteration number scales as $O(\log(1/\epsilon))$ to obtain an ϵ-suboptimal solution [34]. Moreover, the storage cost is $O(nd)$ which is minimal for storing the matrix U. Since $d \ll p \ll n$, this paradigm here can be orders of magnitude computationally and memory more efficient than Solution (3.1).

Solve V. Now we move on the local solution for the variable V. In fact, due to the quadratic form of V, we can easily derive its closed form solution in the following way:

$$V = (Z - E)^\top D \left(D^\top D + \beta^{-1} I_d\right)^{-1}. \tag{3.4}$$

Although the above solution involves computing matrix inverse, it is typically efficient since d is not large. (Recall that d is an upper bound of the rank)

Solve E. We utilize a standard result in the literature to optimize the variable E. That is, the local minimizer of E is given by the following soft-thresholding operator [16]:

$$E = \mathcal{S}_{\lambda_2/\beta}(Z - DV^\top). \tag{3.5}$$

Solve D. With all the other variables given, the basis dictionary D can be updated as follows:

$$D = (\mu AU + \beta(Z - E)V - W)\left(\beta V^\top V + \mu I_d\right)^{-1}. \tag{3.6}$$

Again, it involves a matrix inverse of size $d \times d$ which is cheap. Alternatively, one can update D with the strategy of [28], which may further accelerate the computation.

Computation and Memory Cost. Note that in each iteration, the computation of the local minimizers of NLRR is $O(npd \log(1/\epsilon))$, while that of LRR is $O(np^2)$. Since d is the rank and hence $d \ll p$, NLRR is roughly one order of magnitude faster than LRR. Regarding the memory cost, it is easy to show that the memory cost of NLRR is $O(np)$, while that of LRR is $O(n^2)$ due to the storage of X. Finally, we outline our scalable non-convex LRR implementation in Algorithm 1.

3.3 Algorithm for the Variants

Generally, the variants of NLRR proposed in Section 2.2 can be solved like Algorithm 1. For example, to optimize the variable V in NLRR$_{21}$ (2.10), one may apply Lemma B.1 to convert it to a Lass-type subproblem and choose efficient Lasso solvers such as LARS [10] and [22]. The max-norm variant is a little complicated due to the non-smoothness of the $\ell_{2,\infty}$ norm. However, it can still be solved efficiently by the algorithms in [36]. We note that as we devise scalable way to compute U, the computation of Elastic NLRR and Lasso NLRR is dominated by the Lasso step, which can be efficiently solved by, e.g. [39].

Algorithm 1 Scalable Non-Convex LRR

```
Input: $Z \in \mathbb{R}^{p\times n}$, $A \in \mathbb{R}^{p\times n}$, non-negative parameters $\beta$
    and $\lambda_2$, initial estimation $D_0 \in \mathbb{R}^{p\times d}$, $E_0 \in \mathbb{R}^{p\times n}$ and
    $W_0 \in \mathbb{R}^{p\times d}$, $\mu_0 > 0$, $k = 0$.
Output: $(D_k, U_k, V_k, E_k)$.
1: repeat
2:    $U_{k+1} = \arg\min_U g(D_k, U, 0, 0, W_k, \mu_k)$.
3:    $V_{k+1} = \arg\min_V g(D_k, 0, V, E_k, 0, 0)$.
4:    $E_{k+1} = \arg\min_E g(D_k, 0, V_{k+1}, E, 0, 0)$.
5:    $D_{k+1} = \arg\min_D g(D, U_{k+1}, V_{k+1}, E_{k+1}, W_k, \mu_k)$.
6:    $W_{k+1} = W_k + \mu_k(D_{k+1} - AU_{k+1})$.
7:    $\mu_{k+1} = 1.1\mu_k$.
8:    $k \leftarrow k + 1$.
9: until convergence
```

4 Theoretical Analysis

4.1 Stability

The stability here refers to producing similar weights for similar features of an algorithm, which is also known as the "group effect" that is desired by a variety of applications [49]. It has been established that Lasso is not stable due to the ℓ_1 regularization [46]. In the sequel, we show that our model NLRR$_{21}$ performs like the Elastic Net [50] which favors stability as well as sparsity.

Formally, we have the following theorem for NLRR$_{21}$.

Theorem 4.1. *Assume that for all data points $\boldsymbol{z}_i$, $1 \le i \le n$, there exists an absolute constant c, such that $\|\boldsymbol{z}_i\|_1 \le c$ always holds. Let (D, U, V, E) be the global minimizer of the NLRR$_{21}$ problem* (2.10). *Then, for any entries v_{ij}, v_{ik} of V, if $v_{ij}v_{ik} > 0$, we have:*

$$|v_{ij} - v_{ik}| \le \sqrt{2\lambda_2 c\beta}\, \|\boldsymbol{d}_j - \boldsymbol{d}_k\|_2 . \tag{4.1}$$

Remark. The theorem shows that under some mild conditions, when the two atoms in the dictionary are similar, the algorithm produces similar coefficients, as long as the samples are uniformly bounded. Note that the property of uniform boundedness typically holds for real-world applications. The above theorem is actually an extension of Theorem 1 in Elastic Net [50], from the vector case to the matrix case. However, there are two key differences: First, the formulation of [50] is based on linear regression, and hence, the response is uncontrollable. In contrast, NLRR$_{21}$ is derived from the subspace clustering problem where the observed data matrix is always bounded. Second, [50] considered a convex problem while NLRR$_{21}$ is non-convex.

4.2 Convergence

Generally speaking, the alternating minimization paradigm can be seen as a block coordinate descent method, with each block being the matrix we are interested in. Note that the objective function of NLRR and NLRR$_{21}$ is strongly (but not jointly) convex over D, U, V and E. Thus, from Proposition 2.7.1 in [4], we know that the sequence of (D_k, U_k, V_k, E_k) in Alg. 1 converges to a stationary point.

It is, however, difficult to examine whether a stationary point of a non-convex program is a global optimum [4]. The seminal work from [6] showed that for the noise free case, as long as the rank we pick is sufficiently large, any local minimum obtained by the alternating minimization algorithm is also a global minimum. Unfortunately, in practice, the algorithm may not reach a local minimum since it gets stuck into some stationary point. In Section 5, our empirical study shows that the obtained solutions of our algorithms always work well.

Notably, recent works such as [17, 31, 1] obtain encouraging results on non-convex problems. Their analysis is established on a carefully designed initial solution and a stringent coherence condition. It would be an interesting future work on studying the global optimality of NLRR.

5 Experiments

Baselines. We compare our algorithm NLRR with two state-of-the-art methods for subspace clustering: SSC [11] and LRR [27]. We also include the sparsity induced variants NLRR$_{21}$ and NLRR$_1$ to examine the performance of sparse solution for multiple subspaces. To examine the role of D and show that NLRR can be used for subspace recovery, we compare NLRR with PCA to show robustness. If not specified, the matrix A used in LRR and NLRR is set to be the data matrix Z, which is also the strategy of SSC.

Parameters. We follow the default setting provided in the source code of the baselines. For NLRR and its variants, β is fixed with 1 and λ_2 is fixed with $1/\sqrt{n}$. We set λ_1 to 0.3 for synthetic data and to 0.05 for realistic data.

Evaluation Metric. For the subspace clustering task, we use the standard metric called subspace segmentation accuracy [11] to evaluate the performance. Since we argued in Section 2 that D works as a subspace dictionary for the clean data (see also [35]), we confirm the role of it by introducing the metric called Expressed Variance (EV) [45]:

$$\mathrm{EV}(D, D_0) \overset{\text{def}}{=} \mathrm{Tr}(D^\top D_0 D_0^\top D) / \mathrm{Tr}(D_0 D_0^\top), \qquad (5.1)$$

where D_0 is the ground truth of the orthonormal bases and D is the estimation by our algorithm. A higher value of EV (which ranges from 0 to 1) means better subspace recovery.

Clustering Pipeline. We take a standard pipeline [11, 27] for the clustering task: the full data matrix Z is passed to an algorithm and the representation matrix produced by the algorithm is then fed to a spectral clustering algorithm [32] to obtain the final clustering result.

Data Generation. For synthetic data, we generate K disjoint subspaces $\{\mathcal{S}_k\}_{k=1}^K \subset \mathbb{R}^p$, whose bases are denoted by $\{L_k\}_{k=1}^K \subset \mathbb{R}^{p \times d_k}$. The clean data matrix C_k is produced by $C_k = L_k R_k^\top$, where $R_k \subset \mathbb{R}^{n_k \times d_k}$. Thus, for each subspace, we generate n_k data points lying in p ambient dimensions, whose rank is actually d_k and each entry is sampled i.i.d. from the normal distribution. Then we collect all the C_k's to form the clean matrix C. Finally, the observation matrix Z is produced by $Z = C + E$, where E is the sparse corruption whose ρ fraction of entries are non-zero and follow an i.i.d. uniform distribution over $[-10, 10]$. In our experiments, ρ is set to be 0.3 if not specified. For each experiment, we independently generate 10 folds of the data and report the averaged results.

5.1 Examining the Role of D

A crucial observation that motivates our formulation of NLRR$_{21}$ and NLRR$_1$ is that, $D = AU$ works as the basis dictionary of the union of subspaces [35]. In this section, hence, we justify this conjecture by numerical experiments. We set $p = 100$, $K = 4$ and $n_k = 100$, $d_k = 5$ for all small subspaces. We set the expected rank d to be 20. We use EV(D, L) (see (5.1)) to measure the fitness of D and the true subspace L (obtained by collecting the L_k's).

Table 1: **EV versus fraction of corruptions.** A higher EV means a better subspace recovery. The results demonstrate that the variable D estimated by NLRR indeed works as a basis dictionary and converges (or approximates) to the true subspace. Moreover, NLRR is more robust than PCA and LRR, since it explicitly models the basis dictionary.

ρ	0	0.01	0.05	0.1	0.15	0.2
PCA	**1**	0.99	0.98	0.97	0.94	0.92
LRR	**1**	**1**	0.98	0.97	0.95	0.93
NLRR	**1**	**1**	**1**	**1**	**1**	**0.99**

The results are shown in Table 1. Recall EV $= 1$ means an exact recovery. From the table, we find that NLRR can exactly recover the multiple subspaces when the noise level ρ is less than 0.2, which confirms our claim in Section 2. Thus, although this paper focuses on subspace clustering, a by-product of our algorithm can also be used for subspace recovery. In contrast, PCA and LRR degrade even for a slightly corrupted data (e.g., $\rho = 0.05$), which indicates that NLRR is more robust than its convex counterpart LRR. The gain of NLRR is possibly due to the explicit modeling of the basis dictionary D as argued in [35]. Yet, they verified such interesting phenomenon for an online non-convex algorithm of LRR while our work confirms that this also holds for batch setting.

5.2 Robustness for Subspace Clustering

In this subsection, we provide an experimental study to compare NLRR with LRR, for assessing the robustness of subspace clustering. Specifically, we demonstrate the clustering accuracy of LRR and NLRR under different dimension d_k and corruption fraction ρ. For this purpose, we gen-

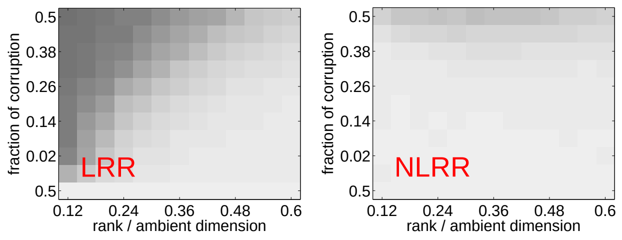

Figure 1: **Subspace clustering accuracy under different ranks and corruptions.** Brighter color means higher accuracy. NLRR consistently outperforms LRR, especially for the large corruption cases which are more challenging.

erate 4 disjoint subspaces, each of which has 100 points lying in a 100-dimensional subspace. In the experiment, the ratio d_k/p ranges from 0.03 to 0.15, with a step size of 0.01. d is set to be the true rank. The corruption fraction ρ ranges from 0 to 0.5, with a step size of 0.05.

We report the comparison results in Figure 1, where a brighter block means a more accurate clustering. For LRR, we observe that it is difficult to detect the correct clustering structure when data are a highly corrupted. In contrast, NLRR is able to handle the challenging cases and achieves significant improvement over LRR.

5.3 Effectiveness of the Sparsity Induced Variants

We now examine the efficacy of the sparsity induced variants NLRR$_{21}$ and NLRR$_1$ presented in Section 2.2. We fix the ambient dimension p to 100 and set the intrinsic dimension of each subspace $d_k = 4$. We set $\rho = 0.3$, a challenging case. We increase the number of subspaces from 2 to 10, and always generate 200 samples for each subspace. We set d as the true rank. Note that recently, [25, 24] suggested a heuristic algorithm which is effective for sparsely corrupted data. Their key idea is utilizing a preprocessing step to obtain an A_0 as the choice of A (see (2.1)). Here, we also record the results with such technique.

We illustrate the results in Figure 2. For the case of A equal to Z, it shows that NLRR$_{21}$, which penalizes the coefficients V by a combination of Frobenius norm and ℓ_1 norm, achieves the best accuracy. NLRR$_1$ is formulated with ℓ_1 penalty, which promotes sparsity patterns. However, it suffers for the clustering task. In contrast, NLRR$_{21}$ essentially makes a balance of accuracy and sparsity, which meets the observations in [50]. Finally, it demonstrates that LRR and NLRR favor dense solutions, which is not surprising since there is no sparsity penalty for them.

On the bottom panel of Figure 2, we plot the accuracy and sparsity curve when A is equipped with A_0 from the heuristic algorithm [25]. This strategy can dramatically improve the clustering accuracy, which meets the theory of [25]. Interestingly, when we set A equal to A_0, we find that both NLRR$_{21}$ and NLRR$_1$ can result in more sparse solutions.

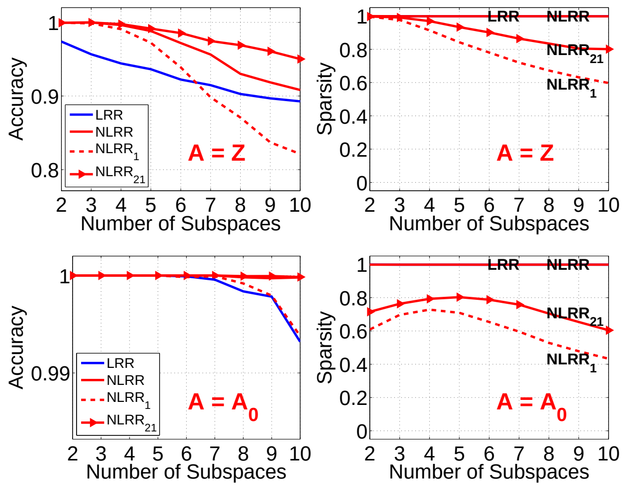

Figure 2: **Subspace clustering accuracy (left) and sparsity (right) against the number of subspaces.** A_0 is chosen according to [25]. The results show that NLRR$_{21}$ always achieves the best accuracy among all the other algorithms. Meanwhile, NLRR$_{21}$ also promotes sparsity as the number of subspaces increases. NLRR and LRR do not produce sparse solutions and the performance of LRR highly depends on the choice of A. NLRR$_1$ does not appear to have a proper trade-off between accuracy and sparsity.

5.4 Large Scale Data and Computational Efficiency

We further evaluate our methods on a real-world dataset MNIST. It contains totally 70,000 hand written digits, each of which is resized to 28×28 pixels. For each kind of digit, we randomly pick 200 samples and obtain a data matrix $Z \in \mathbb{R}^{784 \times 2000}$, named as MNIST-2K. We also experiment on a larger size, where each class has 1000 samples so that the observation $Z \in \mathbb{R}^{784 \times 1e4}$, named as MNIST-10K.

Table 2: **Subspace clustering on MNIST dataset.** Top: MNIST-2K. Bottom: MNIST-10K. NLRR or its variants is always the most accurate and efficient algorithm. The sparsity of NLRR$_{21}$ and NLRR$_1$ is nearly 0.54 for MNIST-2K and 0.83 for MNIST-10K.

	SSC	LRR	NLRR	NLRR$_{21}$	NLRR$_1$
Acc. (%)	46.45	54.60	55.40	**59.15**	52.95
Time (s)	116	111	**28**	50	51
Acc. (%)	44.90	55.15	58.63	59.50	**59.67**
Time (m)	84	24	**15**	27	25

We set $d = 50$. For the MNIST dataset, we find that the accuracy decreases by setting $A = A_0$ (see Appendix D). Thus, we only report results with $A = Z$, which are shown in Table 2. The most sparse solution emerges from SSC, but with the lowest accuracy, which is also observed in the preceding section. On the small set MNIST-2K, LRR appears to perform comparably with our algorithms. However, when manipulating a larger database MNIST-10K, it is much inferior to ours. We also note that on MNIST-10K, the solutions of NLRR_{21} and NLRR_1 are not as sparse as those on MNIST-2K. This is possibly because the true subspaces of the data are not disjoint. For instance, the number "1" and "7" commonly share similar parts. As a consequence, the true coefficients V could not be sparse. When acquiring more samples, the connectivity emerges and thus our model fails to output the sparse solution.

Regarding the computational efficiency, we find that NLRR is always the fastest algorithm among the methods. This agrees with our qualitative analysis on the time complexity in Section 3.2. We also note that for the large database MNIST-10K, the sparsity induced variants NLRR_{21} and NLRR_1 performs as efficient as LRR, but much slower than NLRR. This is not surprising as they have to solve a Lasso problem in each iteration that is not cheap for large scale problems. We believe that this issue could be mitigated if more efficient Lasso solver is available. A detailed comparison between LRR and NLRR is shown in Table 3.

Table 3: **Time complexity (seconds) against sample size.**

#Sample	500	2K	4K	6K	8K	10K
LRR	37	111	251	377	598	1440
NLRR	4	28	112	253	464	914

5.5 Comparison to Other Methods

In Section 2.1, we demonstrate the connection of NLRR to other methods such as NRPCA, DL and SSC. We also argue that it is more effective to pursue structures on the factors than on primal variables. We justify these claims in Table 4. Note that the result of SSC has been recorded in Table 2 and that of NLRR is same with Table 2. Clearly, from the unified framework, we know that the improvement of NLRR comes from the interesting constraints of D and V and the superiority to LRSSC [44] and LRR2 [25] confirms the efficacy of integrating MF into LRR.

Table 4: **Connection to other methodologies.** Top: MNIST-2K. Bottom: MNIST-10K. We unify LRR with other methodologies by its non-convex formulation. The improvement of NLRR to NRPCA, DL and SSC (see Table 2) implies the efficacy of the way D and V are constrained. The superiority to LRSSC and LRR2 conforms the benefit of combining MF and LRR, i.e., pursuing structures on factors rather than on primal variables.

	NRPCA	DL	LRSSC	LRR2	NLRR
Acc. (%)	50.90	40.40	51.00	55.00	**55.40**
Acc. (%)	54.06	47.76	53.40	53.67	**58.63**

5.6 The Influence of Expected Rank

To utilize the non-convex reformulation of the nuclear norm (2.2), we need to estimate an upper bound d on the rank of the clean data. Here, we follow the setting of Section 5.1 but fix $\rho = 0.3$. Note that the true rank of the data is 20. We run the algorithm NLRR by varying the expected rank d from 1 to 40. As is expected, as long as we choose a sufficiently large quantity of d, our algorithm will exactly recover the subspace bases and correctly segment the data. Note that for the noise free case, our problem can be converted to the one considered in [6], and the results presented here imply a potential to extend their theories to the noisy case. In addition, we observe that while NLRR fails to recover the whole subspace bases for $d = 16$, it perfectly clusters the data. This intriguing phenomenon is possibly due to the robustness of spectral clustering, see [38].

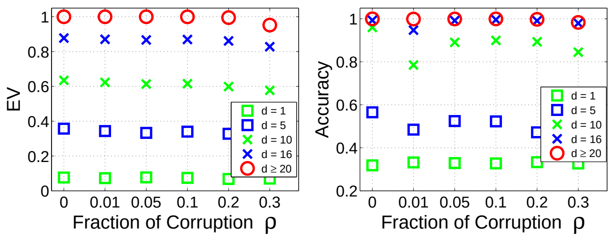

Figure 3: **The influence of the choice of d.** Note that the true rank here is equal to 20. As long as we pick d greater or equal to the true rank, NLRR can exactly recover the subspace and identify the clustering structure.

6 Conclusion

In this paper, by using a non-convex reformulation of the nuclear norm and introducing a crucial variable working as the basis dictionary, we have shown how to unify four important problems: LRR, RPCA, DL and SSC. Their connections, as well as the merit of multiple subspaces considered in LRR, motivate us to propose new variants of LRR. We have demonstrated that these variants can boost the clustering accuracy of LRR, while producing sparse solutions. Interestingly, a by-product of our algorithm can be used for subspace recovery, and has been illustrated to be more robust than LRR. For large scale problems, we have devised a scalable implementation for our new formulations which is at least one order of magnitude more efficient than LRR in terms of computation and memory cost. The efficiency is further justified on a large real-world dataset. Theoretically, we have proven the stability of the solutions and the convergence of the presented algorithm. The promising results also have implications back to MF: when the data is formed by a union of subspaces, i.e., a clustering structure, it is preferable to penalize V with the elastic regularizer.

Acknowledgments

The work is partially supported by NSF-Bigdata-1419210, NSF-III-1360971, ONR-N00014-13-1-0764, and AFOSR-FA9550-13-1-0137.

References

[1] A. Agarwal, A. Anandkumar, P. Jain, P. Netrapalli, and R. Tandon. Learning sparsely used overcomplete dictionaries. In *Proceedings of The 27th Conference on Learning Theory*, pages 123–137, 2014.

[2] S. D. Babacan, S. Nakajima, and M. N. Do. Probabilistic low-rank subspace clustering. In *Advances in Neural Information Processing Systems*, pages 2753–2761, 2012.

[3] F. Bach, J. Mairal, and J. Ponce. Convex sparse matrix factorizations. *arXiv preprint arXiv:0812.1869*, 2008.

[4] D. P. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[5] S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Math. Program.*, 95(2):329–357, 2003.

[6] S. Burer and R. D. C. Monteiro. Local minima and convergence in low-rank semidefinite programming. *Math. Program.*, 103(3):427–444, 2005.

[7] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

[8] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[9] D. L. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289–1306, 2006.

[10] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[11] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *Computer Vision and Pattern Recognition. IEEE Conference on*, pages 2790–2797. IEEE, 2009.

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

[13] M. Fazel, H. Hindi, and S. P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In *American Control Conference*, volume 6, pages 4734–4739. IEEE, 2001.

[14] J. Feng, H. Xu, and S. Yan. Online robust PCA via stochastic optimization. In *Advances in Neural Information Processing Systems*, pages 404–412, 2013.

[15] B. Haeffele, E. Young, and R. Vidal. Structured low-rank matrix factorization: Optimality, algorithm, and applications to image processing. In *Proceedings of the 31st International Conference on Machine Learning*, pages 2007–2015, 2014.

[16] E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19(3):1107–1130, 2008.

[17] P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 665–674. ACM, 2013.

[18] R. Jenatton, G. Obozinski, and F. R. Bach. Structured sparse principal component analysis. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 366–373, 2010.

[19] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[20] M. Journée, Y. Nesterov, P. Richtárik, and R. Sepulchre. Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, 11:517–553, 2010.

[21] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[22] H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

[23] N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman. Complexity measures of sign matrices. *Combinatorica*, 27(4):439–463, 2007.

[24] G. Liu and P. Li. Advancing matrix completion by modeling extra structures beyond low-rankness. *arXiv preprint arXiv:1404.4646*, 2014.

[25] G. Liu and P. Li. Recovery of coherent data via low-rank dictionary pursuit. In *Advances in Neural Information Processing Systems*, pages 1206–1214, 2014.

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

[27] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *Proceedings of the 27th International Conference on Machine Learning*, pages 663–670, 2010.

[28] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.

[29] J. Mairal, F. R. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th International Conference on Machine Learning*, pages 689–696, 2009.

[30] J. Mairal, F. R. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *Advances in Neural Information Processing Systems*, pages 1033–1040, 2008.

[31] P. Netrapalli, U. N. Niranjan, S. Sanghavi, A. Anandkumar, and P. Jain. Non-convex robust PCA. In *Advances in Neural Information Processing Systems*, pages 1107–1115, 2014.

[32] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, 2001.

[33] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[34] P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1-2):1–38, 2014.

[35] J. Shen, P. Li, and H. Xu. Online low-rank subspace clustering by basis dictionary pursuit. *arXiv preprint arXiv:1503.08356*, 2015.

[36] J. Shen, H. Xu, and P. Li. Online optimization for max-norm regularization. In *Advances in Neural Information Processing Systems*, pages 1718–1726, 2014.

[37] M. Soltanolkotabi and E. J. Candes. A geometric analysis of subspace clustering with outliers. *The Annals of Statistics*, 40(4):2195–2238, 2012.

[38] M. Soltanolkotabi, E. Elhamifar, and E. J. Candes. Robust subspace clustering. *The Annals of Statistics*, 42(2):669–699, 2014.

[39] M. Tan, I. W. Tsang, and L. Wang. Matching pursuit lasso part i: Sparse recovery over big dictionary. *IEEE Trans. on Signal Process.*, 63(3):727–741, 2015.

[40] R. Vidal. Subspace clustering. *IEEE Signal Process. Mag.*, 28(2):52–68, 2011.

[41] R. Vidal and P. Favaro. Low rank subspace clustering (LRSC). *Pattern Recognition Letters*, 43:47–61, 2014.

[42] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (gpca). *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, 27(12):1945–1959, 2005.

[43] F. Wang and P. Li. Efficient nonnegative matrix factorization with random projections. In *SDM*, pages 281–292. SIAM, 2010.

[44] Y.-X. Wang, H. Xu, and C. Leng. Provable subspace clustering: When lrr meets ssc. In *Advances in Neural Information Processing Systems*, pages 64–72, 2013.

[45] H. Xu, C. Caramanis, and S. Mannor. Principal component analysis with contaminated data: The high dimensional case. In *COLT*, pages 490–502, 2010.

[46] H. Xu, C. Caramanis, and S. Mannor. Sparse algorithms are not stable: A no-free-lunch theorem. *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, 34(1):187–193, 2012.

[47] C. Yang, D. Robinson, and R. Vidal. Sparse subspace clustering with missing entries. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2463–2472, 2015.

[48] C. You and R. Vidal. Sparse subspace clustering by orthogonal matching pursuit. *CoRR*, abs/1507.01238, 2015.

[49] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[50] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.