Latent Low-Rank Representation for Subspace Segmentation and Feature Extraction

Guangcan Liu Shuicheng Yan
Electrical and Computer Engineering, National University of Singapore
{eleliug,eleyans}@nus.edu.sg

Abstract

Low-Rank Representation (LRR) [16, 17] is an effective method for exploring the multiple subspace structures of data. Usually, the observed data matrix itself is chosen as the dictionary, which is a key aspect of LRR. However, such a strategy may depress the performance, especially when the observations are insufficient and/or grossly corrupted. In this paper we therefore propose to construct the dictionary by using both observed and unobserved, hidden data. We show that the effects of the hidden data can be approximately recovered by solving a nuclear norm minimization problem, which is convex and can be solved efficiently. The formulation of the proposed method, called Latent Low-Rank Representation (LatLRR), seamlessly integrates subspace segmentation and feature extraction into a unified framework, and thus provides us with a solution for both subspace segmentation and feature extraction. As a subspace segmentation algorithm, LatLRR is an enhanced version of LRR and outperforms the state-of-the-art algorithms. Being an unsupervised feature extraction algorithm, LatLRR is able to robustly extract salient features from corrupted data, and thus can work much better than the benchmark that utilizes the original data vectors as features for classification. Compared to dimension reduction based methods, LatLRR is more robust to noise.

1. Introduction

Vision data is often characterized by a mixture of multiple (linear) subspaces. For example, it is known that motion [6, 21, 29], face [26, 28] and texture [18] can be well characterized by subspaces. The importance of subspaces naturally leads to a challenging problem of *subspace segmentation* [2] [1], whose goal is to segment (or group) data into clusters with each cluster corresponding to a subspace. In computer vision, subspace segmentation has been widely studied (e.g., [2, 3, 5, 7, 11, 17, 18, 19, 21, 22, 27]) owing to its numerous applications, such as motion segmentation [6, 21, 29], face recognition [26, 28] and image segmentation [15, 18].

When the subspaces are independent and the data is noiseless, several existing methods (e.g., [2, 3, 16, 17]) are able to produce exactly correct segmentation results. So, as pointed out by [16, 17], the main challenge of subspace segmentation is how to effectively handle the "chicken-and-egg" coupling between noise correction and data segmentation. In this sense, the recently established Low-Rank Representation (LRR) [16, 17, 24] is a promising method, since LRR provides us with an efficient way to perform noise correction and subspace segmentation simultaneously. In general, LRR aims at finding the lowest-rank representation among all the candidates that can represent the data vectors as linear combinations of the basis in a given dictionary:

$$\min_{Z} \quad \|Z\|_*, \quad \text{s.t.} \quad X_O = AZ,$$

where X_O denotes the observed data matrix (each column is an observation vector), A is the dictionary and $\|\cdot\|_*$ denotes the nuclear norm [4] of a matrix, i.e., the sum of the singular values of the matrix. For subspace segmentation, the observed data matrix itself is usually used as the dictionary [16, 17, 24], resulting in the following convex optimization problem:

$$\min_{Z} \quad \|Z\|_*, \quad \text{s.t.} \quad X_O = X_O Z. \tag{1}$$

When the subspaces are independent, the data is noiseless and the data sampling is *sufficient* [2], Liu et al. [16, 17] show that the optimal solution, denoted as Z_O^*, to the above problem is the widely used Shape Iteration Matrix (SIM) [2],

[1]For ease of presentation, in this paper we firstly present our motivations under the context of subspace segmentation. Then we shall show that our method can also naturally handle the feature extraction problem.

[2]Let X_O be a set of samples drawn from a union of k subspaces $\{\mathcal{S}_i\}_{i=1}^k$, each of which has a rank of r_i. Suppose $X_O = [X_1, X_2, \cdots, X_k]$ and X_i is the collection of n_i samples from the i-th subspace $\mathcal{S}_i$, then the sampling of X_O is sufficient if and only if $\text{rank}(X_i) = r_i, \forall i$.

which is a "block-diagonal" affinity matrix exactly indicating the true segmentation of the data. To handle the data corrupted by noise, LRR adopts a regularized formulation that introduces an extra regularization term to fit the noise.

However, previous LRR methods [16, 17, 24] suffer some issues caused by directly setting the dictionary A as the observed data matrix X_O. First, to enable the ability of representing the underlying subspaces, the dictionary ($A = X_O$) must contain sufficient data vectors sampled from the subspaces. Otherwise, $Z_O^* = \mathtt{I}$ ($\mathtt{I}$ denotes the identity matrix) is probably the only feasible solution to (1) and thus LRR may fail. So, one cannot use X_O as the dictionary to represent the subspaces if the data sampling is insufficient [3]. Second, in order to achieve robust segmentation, LRR requires that sufficient noiseless data is available in the dictionary A, i.e., only a part of A is corrupted. While $A = X_O$, this assumption may be invalid and the robustness of LRR may be depressed in reality.

To resolve the issue of insufficient sampling and improve the robustness to noise, we consider the following LRR problem:

$$\min_{Z} \|Z\|_*, \quad \text{s.t.} \quad X_O = [X_O, X_H]Z, \tag{2}$$

where the concatenation (along column) of X_O and X_H is used as the dictionary, X_O is the observed data matrix and X_H represents the unobserved, hidden data. The above formulation can resolve the issue of insufficient data sampling, provided that $A = [X_O, X_H]$ is always sufficient to represent the subspaces. Let $Z_{O,H}^*$ be the optimal solution to the above problem and $Z_{O,H}^* = [Z_{O|H}^*; Z_{H|O}^*]$ be its row-wise partition such that $Z_{O|H}^*$ and $Z_{H|O}^*$ correspond to X_O and X_H [4], respectively, then $Z_{O|H}^*$ is a nontrivial block-diagonal matrix that can exactly reveal the true subspace membership even if the sampling of X_O is insufficient. Note here that LRR actually only requires the sampling of the dictionary being sufficient (see Theorem 3.2 of [16]). Moreover, as we will see, the consideration of the hidden data can improve the robustness of LRR. With these motivations, in this paper we study the problem of recovering the affinity matrix $Z_{O|H}^*$ by using only the observed data X_O. More concretely we study the following "hidden effects recovery" (i.e., recovery of the effects of the hidden data) problem.

Problem 1.1 (Hidden Effects Recovery) *Given an observed data matrix X_O, our goal is to recover $Z_{O|H}^*$ in the absence of the hidden data X_H.*

[3]It is worth noting that the methods based on Sparse Representation (SR) (e.g., [3]) also suffer this issue.

[4]For n observed data vectors, $Z_{O|H}^*$ is an $n \times n$ matrix that contains the pairwise affinities among the observed data vectors. We adopt the symbol $Z_{O|H}^*$ just because the affinity matrix depends on both X_O and X_H.

Without imposing any restriction on X_O and X_H, the above problem is "ill-posed", because $Z_{O|H}^*$ is computed in the presence of both X_O and X_H. So we study the problem in the setting where all the data, both observed and hidden, are sampled from the same collection of low-rank subspaces. In this case, we show that the hidden effects can be approximately recovered by solving a nuclear norm minimization problem, which is convex and can be solved efficiently. For clarity, we call this recovery method as Latent Low-Rank Representation (LatLRR). The solution of LatLRR could be regarded as an "enhanced" version of LRR, where the enhancement is made by the hidden data. Since it not only inherits the advantages of LRR but also includes the hidden effects, LatLRR is more accurate and robust than LRR as a tool for subspace segmentation. Moreover, the formulation of LatLRR naturally integrates subspace segmentation and *feature extraction* [20] into a unified framework, and thus also provides us with an algorithm for feature exaction. In summary, the main contributions of this paper include:

- LatLRR extends LRR to handle the hidden effects. To the best of our knowledge, this work is the first to consider the hidden data in subspace segmentation.
- As a subspace segmentation algorithm, LatLRR outperforms the state-of-the-art algorithms for motion segmentation.
- As an unsupervised feature extraction algorithm, LatLRR is empirically able to automatically extract salient features from corrupted data, and thus can work much better than the benchmark that uses the original data as features for classification. Compared to dimension reduction based methods, LatLRR is more robust to noise.

2. Problem Statement

Problem 1.1 only describes the hidden recovery problem for noiseless data. More precisely, this paper addresses the following two hidden effect recovery problems.

Problem 2.1 (Noiseless Data) *The same as Problem 1.1.*

Futher, we define the hidden effects recovery problem for corrupted data as follows.

Problem 2.2 (Corrupted Data) *For the following LRR problem (noisy case)*

$$\min_{Z,E} \|Z\|_* + \lambda \|E\|_1, \text{ s.t. } X_O = [X_O, X_H]Z + E, \tag{3}$$

where $\|\cdot\|_1$ is the ℓ_1-norm for characterizing the sparse noise E. Suppose $Z_{O,H}^* = [Z_{O|H}^*; Z_{H|O}^*]$ is the optimal solution (with respect to the variable Z) and $Z_{O|H}^*$ is the submatrix corresponding to X_O, then our goal is to recover $Z_{O|H}^*$ by using only the observed data X_O.

3. Recovery of Hidden Effects by LatLRR

In this section we abstractly present our Latent Low-Rank Representation (LatLRR) method for addressing the problem of hidden effects recovery. For ease of understanding, we also present some clues of using LatLRR to perform subspace segmentation. The detailed applications to subspace segmentation and feature extraction are deferred to Section 4.

3.1. A Basic Observation

In order to recover the hidden effects, it is necessary to explore the minimizer to problem (2). Based on the theorems introduced by Liu et al. [16], we have the following theorem.

Theorem 3.1 *Given any matrices X_O ($X_O \neq 0$) and X_H, the minimizer to problem* (2) *is unique and has the following closed form:*

$$Z^*_{O|H} = V_O V_O^T \quad \textit{and} \quad Z^*_{H|O} = V_H V_O^T, \qquad (4)$$

where V_O and V_H are calculated as follows: Compute the skinny Singular Value Decomposition (SVD) of $[X_O, X_H]$, denoted as $[X_O, X_H] = U\Sigma V^T$, and partition V as $V = [V_O; V_H]$ such that $X_O = U\Sigma V_O^T$ and $X_H = U\Sigma V_H{}^T$.

Proof By the definition of skinny SVD, it can be calculated that the constraint $X_O = [X_O, X_H]Z$ is equal to $U\Sigma V_O^T = U\Sigma V^T Z$, which is also equal to $V_O^T = V^T Z$. So problem (2) is equal to the following optimization problem:

$$\min_Z \|Z\|_*, \quad \text{s.t.} \quad V_O^T = V^T Z.$$

By Lemma 3.3 of [16], problem (2) has a unique minimizer

$$Z^*_{O,H} = VV_O^T = [V_O V_O^T; V_H V_O^T],$$

which directly leads to the conclusions in (4). ∎

3.2. Recovering Hidden Effects by Convex Optimization

3.2.1 Noiseless Data (Problem 2.1)

In general, it is impractical to exactly recover $Z^*_{O|H}$ by using only the observed data X_O. Nevertheless, it is possible to obtain an approximate recovery by analyzing the properties of the hidden effects. By Theorem 3.1, we have

$$\begin{aligned} X_O &= [X_O, X_H]Z^*_{O,H} = X_O Z^*_{O|H} + X_H Z^*_{H|O} \\ &= X_O Z^*_{O|H} + X_H V_H V_O^T \\ &= X_O Z^*_{O|H} + U\Sigma V_H^T V_H V_O^T \\ &= X_O Z^*_{O|H} + U\Sigma V_H^T V_H \Sigma^{-1} U^T X_O. \end{aligned}$$

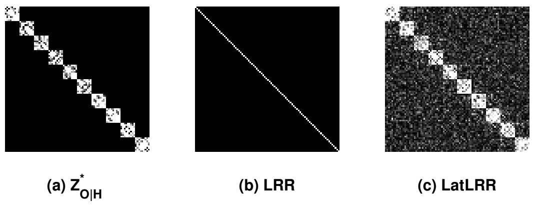


Figure 1. **Illustrating the recovery of the hidden effects.** (a) The block-diagonal affinity matrix identified by $Z^*_{O|H}$, which is obtained by solving problem (2). (b) The affinity matrix produced by LRR. Since the data sampling is insufficient, $Z = \mathtt{I}$ is the only feasible solution to problem (1). (c) The affinity matrix identified by the minimizer (with respect to the variable Z) to problem (5).

Let $L^*_{H|O} = U\Sigma V_H^T V_H \Sigma^{-1} U^T$, then the hidden effects can be described by a simple formulation as follows:

$$X_O = X_O Z^*_{O|H} + L^*_{H|O} X_O.$$

Suppose both X_O and X_H are sampled from the same collection of low-rank subspaces, and the union of the subspaces has a rank of r. Then it can be derived that

$$\operatorname{rank}\left(Z^*_{O|H}\right) \leq r \quad \text{and} \quad \operatorname{rank}\left(L^*_{H|O}\right) \leq r.$$

So both $Z^*_{O|H}$ and $L^*_{H|O}$ should be of low-rank and we may recover $Z^*_{O|H}$ by minimizing

$$\begin{aligned} \min_{Z_{O|H}, L_{H|O}} \quad & \operatorname{rank}\left(Z_{O|H}\right) + \operatorname{rank}\left(L_{H|O}\right), \\ \text{s.t.} \quad & X_O = X_O Z_{O|H} + L_{H|O} X_O. \end{aligned}$$

As a common practice in rank minimization problems, we relax the rank function as the nuclear norm, resulting in the following convex optimization problem:

$$\min_{Z,L} \|Z\|_* + \|L\|_*, \text{ s.t. } X = XZ + LX. \qquad (5)$$

Here, for ease of presentation, we simplify the symbols by replacing X_O, $Z_{O|H}$ and $L_{H|O}$ with X, Z and L, respectively. Suppose X is of size $d \times n$, then Z and L are $n \times n$ and $d \times d$, respectively. One may have noticed that there is no parameter in (5) to balance the strengths of Z and L. This is because the strengths of the two parts are balanced automatically [5].

Let the minimizer to problem (5) be (Z^*, L^*), then Z^* is an approximate recovery to $Z^*_{O|H}$ (Problem 2.1). To verify, we present an example as the following.

[5] To measure the individual strengths of L and Z in problem (5), we consider the following two optimization problems:

$$\min_Z \|Z\|_*, \quad \text{s.t.} \quad X = XZ,$$

whose minimizer is assumed to be Z^*_Z, and

$$\min_L \|L\|_*, \quad \text{s.t.} \quad X = LX,$$

whose minimizer is assumed to be L^*_L. By Corollary 3.3 of [16], it can be concluded that $\|Z^*_Z\|_* = \operatorname{rank}(X) = \operatorname{rank}(X^T) = \|L^*_L\|_*$. So the strengths of L and Z in (5) are balanced naturally.

Example 3.1 *We construct 10 independent subspaces $\{\mathcal{S}_i\}_{i=1}^{10}$ whose basis $\{U_i\}_{i=1}^{10}$ are computed by $U_{i+1} = TU_i, 1 \leq i \leq 9$, where T is a random rotation and U_1 is a random column orthogonal matrix of dimension 200×10. So each subspace has a rank of 10. We construct a 200×90 data matrix $X_O = [X_1, \cdots, X_{10}]$ by sampling 9 (which is smaller than the rank of the subspaces) data vectors from each subspace by $X_i = U_i C_i, 1 \leq i \leq 10$ with C_i being a 10×9 i.i.d. $\mathcal{N}(0,1)$ matrix. Then we obtain a 90×90 affinity matrix (identified by Z^*) by solving (5). To simulate the hidden effects, we also construct a 200×50 hidden matrix X_H by sampling 5 data vectors from each subspace. Fig.1 illustrates that our formulation (5) does make sense for recovering the hidden effects.*

3.2.2 Corrupted Data (Problem 2.2)

In Problem 2.2, it is not easy to give a theoretical analysis to the hidden effects. Nevertheless, since Problem 2.2 is generalized from Problem 2.1, it could be regarded that the conclusions in the above section are approximately valid for corrupted data. So it is adequate to recover the hidden effects by solving the following convex optimization problem:

$$\begin{aligned} \min_{Z,L,E} \quad & \|Z\|_* + \|L\|_* + \lambda \|E\|_1 , \\ \text{s.t.} \quad & X = XZ + LX + E, \end{aligned} \tag{6}$$

where $\lambda > 0$ is a parameter and $\|\cdot\|_1$ is the ℓ_1-norm chosen for characterizing sparse noise. When $\lambda \to +\infty$, the solution to the above problem identifies that of (5). So (6) is a generalization of (5). The minimizer Z^* (with respect to the variable Z) is a recovery to $Z^*_{O|H}$ (Problem 2.2). So Z^* could be regarded as an "enhanced" lowest-rank representation and LatLRR may be more robust than LRR as a tool for subspace segmentation. To see this, we refer to the following example.

Example 3.2 *We sample 200 data vectors from 5 independent subspaces constructed in a similar way as in Example 3.1. Since all the subspaces have a rank of 10 and we sample 20 data vectors from each subspace, the data sampling is sufficient in this example. Some entries are randomly chosen to be corrupted by using large Gaussian noise. After obtaining the affinity matrix identified by Z^*, we use Algorithm 2 of [17] to segment the data into 5 clusters and observe the segmentation accuracy at each noise level. Fig.2 illustrates that LatLRR is more robust than LRR as a tool for subspace segmentation, i.e., the robustness of LRR can be improved by taking the hidden effects into account.*

An intuition underlying the phenomenon in Fig.1 and Fig.2 is that LatLRR actually provides a way to reconstruct a data matrix X from two directions: column (XZ) and row (LX). When some data points are missed, i.e., some columns are missed, it is helpful to make use of the reconstruction along row. By combining both directions, LatLRR can defend missed points and be more robust against noise.

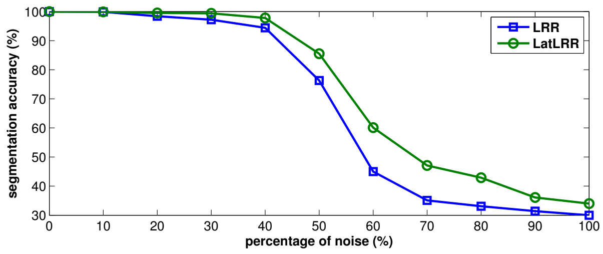


Figure 2. **Illustrating that the robustness of LRR can be improve by considering the hidden effects**. We plot the segmentation accuracies (averaged over 20 runs) across the entire range of noise for the two methods. It can be seen that LatLRR consistently outperforms LRR. Here, the segmentation accuracy is the percentage of correctly grouped samples [3, 17].

Algorithm 1 Solving Problem (6) by Inexact ALM

```
  Initialize: $Z = J = 0, L = S = 0, E = 0, Y_1 = 0, Y_2 = 0, Y_3 = 0, \mu = 10^{-6}, max_u = 10^6, \rho = 1.1$, and $\varepsilon = 10^{-6}$.
  while not converged do
    1. Fix the others and update $J$ by setting $J = \arg\min_J \frac{1}{\mu}||J||_* + \frac{1}{2}||J - (Z + Y_2/\mu)||_F^2$.
    2. Fix the others and update $S$ by setting $S = \arg\min_S \frac{1}{\mu}||S||_* + \frac{1}{2}||S - (L + Y_3/\mu)||_F^2$.
    3. Fix the others and update $Z$ by setting $Z = (\mathtt{I} + X^T X)^{-1}(X^T(X - LX - E) + J + (X^T Y_1 - Y_2)/\mu)$.
    4. Fix the others and update $L$ by setting $L = ((X - XZ - E)X^T + S + (Y_1 X^T - Y_3)/\mu)(\mathtt{I} + XX^T)^{-1}$.
    5. Fix the others and update $E$ by setting $E = \arg\min_E \lambda/\mu||E||_1 + 0.5||E - (X - XZ - LX + Y_1)/\mu||_F^2$.
    6. Update the multipliers by $Y_1 = Y_1 + \mu(X - XZ - LX - E), Y_2 = Y_2 + \mu(Z - J), Y_3 = Y_3 + \mu(L - S)$.
    7. Update the parameter $\mu$ by $\mu = \min(\rho\mu, max_u)$.
    8. Check the convergence conditions: $||X - XZ - LX - E||_\infty < \varepsilon, ||Z - J||_\infty < \varepsilon$, and $||L - S||_\infty < \varepsilon$.
  end while
```

3.3. Solving the Optimization Problem

Since problem (6) can fall back to problem (5) by setting the parameter λ to be relatively large, here we just present how to solve problem (6), which is convex and can be optimized by various methods. We first convert it to the following equivalent problem:

$$\begin{aligned} \min_{Z,L,J,S,E} \quad & \|J\|_* + \|S\|_* + \lambda \|E\|_1, \\ \text{s.t.} \quad & X = XZ + LX + E, Z = J, L = S. \end{aligned}$$

This problem can be solved by the Augmented Lagrange

Multiplier (ALM) [14] method, which minimizes the following augmented Lagrange function:

$$\|J\|_* + \|S\|_* + \lambda\|E\|_1 + \mathrm{tr}\left(Y_1^T(X - XZ - LX - E)\right) \\ + \mathrm{tr}\left(Y_2^T(Z - J)\right) + \mathrm{tr}\left(Y_3^T(L - S)\right) + \\ \frac{\mu}{2}(\|X - XZ - LX - E\|_F^2 + \|Z - J\|_F^2 + \|L - S\|_F^2),$$

where $\mathrm{tr}(\cdot)$ and $\|\cdot\|_F$ denote the trace and Frobenious norm of a matrix, respectively. The above problem is unconstrained. So it can be minimized with respect to J, S, Z, L and E, respectively, by fixing the other variables, and then updating the Lagrange multipliers Y_1, Y_2 and Y_3, where $\mu > 0$ is a penalty parameter. The inexact ALM method, also called the alternating direction method, is outlined in Algorithm 1. Its convergence properties are similarly as those in [14]. Notice that Step 1, Step 2 and Step 5 all have closed-form solutions. Step 1 and Step 2 are solved via the Singular Value Thresholding (SVT) operator [1], while Step 5 is solved by the shrinkage operator [14].

The major computation of Algorithm 1 is at Step 1 and Step 2, which require computing the SVD of matrices. So the complexity of the algorithm is $O(n^3)+O(d^3)$ (assuming X is $d\times n$). However, it is simple to show that the complexity of LatLRR is $O(d^2n+d^3)$ (assume $d \leq n$). By Theorem 3.1 of [16], it can be seen that the optimal solution Z^* (with respect to the variable Z) to (6) always lies within the subspace spanned by the rows of X, i.e., Z^* can be factorized into $Z^* = P^*\tilde{Z}^*$, where P^* can be computed in advance by orthogonalizing the columns of X^T. In a similar way, it can be shown that the optimal solution L^* (with respect to the variable L) to (6) can be factorized into $L^* = \tilde{L}^*(Q^*)^T$, where Q^* can be computed by orthogonalizing the columns of X. Hence, problem (6) can be equivalently transformed into a simpler problem by replacing Z and L with $P^*\tilde{Z}$ and $\tilde{L}(Q^*)^T$, respectively:

$$\min_{\tilde{Z},\tilde{L},E} \|\tilde{Z}\|_* + \|\tilde{L}\|_* + \lambda\|E\|_1, \text{ s.t. } X = A\tilde{Z} + \tilde{L}B + E,$$

where $A = XP^*$ and $B = (Q^*)^TX$. Since the number of columns (or rows) of A (or B) is at most d, the above problem can be solved with a complexity of $O(d^2n + d^3)$ by using a similar way as Algorithm 1. So LatLRR is quite scalable for large-size (n is large) datasets, provided that the data dimension d is not high.

4. Experiments

In this section, we apply LatLRR for subspace segmentation and feature extraction, along with presenting our experimental results. In summary, let (Z^*, L^*, E^*) be the minimizer to problem (6), then we shall show that Z^* and L^* are useful for subspace segmentation and feature extraction, respectively.

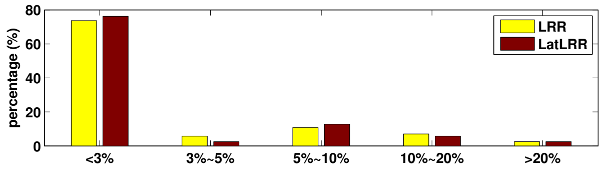


Figure 3. **The distribution of the error rates on Hopkin155**. It can be seen that LRR has achieved error rates smaller than 3% for most sequences.

Comparison under the same setting					
	LSA	RANSAC	SR	LRR	LatLRR
Mean	8.99	8.22	3.89	3.16	**2.95**
Std.	9.80	10.26	7.70	5.99	**5.86**
Max	37.74	47.83	**32.57**	37.43	37.97

Comparison to state-of-the-art methods					
	LSA	ALC	SSC	SC	LatLRR
Mean	4.94	3.37	1.24	1.20	**0.85**

Table 1. **Error rates (%) on Hopkins155**. Besides of LRR, we also list the results of Local Subspace Analysis (LSA) [29], Random Sample Consensus (RANSAC) [5], Sparse Representation (SR) [16], Agglomerative Lossy Compression (ALC) [21], Sparse Subspace Clustering (SSC) [3] and Spectral Clustering (SC) [12]. Note here that the core of SSC is SR, and the previous state-of-the-art results are directly quoted from [3, 12, 21].

4.1. Subspace Segmentation

The affinity matrix identified by Z^* has similar purposes as that of LRR. So we can use Algorithm 2 of [17] to perform subspace segmentation. Namely, we firstly utilize the affinity matrix identified by Z^* to define edge weights of an undirected graph, and then use Normalized Cuts (NCut) [23] to produce the final segmentation results. To evaluate the effectiveness of LatLRR, we test it on Hopkins155 [25], which is a standard database for motion segmentation. This database contains 156 sequences each of which has $39 \sim 550$ data vectors drawn from two or three motions (one motion corresponds to one subspace). Each sequence is a sole segmentation (clustering) task and so there are 156 segmentation tasks in total.

Comparison to LRR. In order to demonstrate the advantages of considering the hidden data, we compare all algorithms under the same setting. Namely, all algorithms use the raw data without any preprocessing as their inputs and simply use NCut to obtain the final segmentation results. Table 1 shows that LatLRR ($\lambda = 0.9$) achieves an error rate of 2.95%, which outperforms LRR. One may have noticed that the improvement is not very distinct. This is because there are 156 sequences in total and LRR has achieved accurate segmentation results for most sequences, as shown in Fig.3. Actually, if we sort the segmentation errors in an ascending order and use T-Test (type 1, tail 1) to do significance test, the improvement is significant with the level of 1%. This confirms the effectiveness of taking the hidden data into account.

Comparison to State-of-the-art Methods. The basic algorithm of LatLRR can achieve an error rate of 2.95%, which does not outperform the specific motion segmentation algorithms. This is mainly due to the fact that Hopkins155 is consisting of 155 sequences with different properties, including the data dimension, number of data points and noise level. As can be seen from Fig.3, the error rates on some sequences (about 10) exceed 20%. For more accurate and reliable motion segmentation, we utilize some techniques used in [12] as follows. First, since the affinity matrix Z^* obtained by solving problem (6) is asymmetric, we convert it into a Positive Semi-Definite (PSD) matrix Z_1^* by solving

$$\begin{aligned} Z_1^* = \arg\min_{Z_1} \quad & \|Z_1\|_* + \alpha \|E\|_1 , \\ \text{s.t.} \quad & Z^* = Z_1 + E, Z_1 \succeq 0, \end{aligned}$$

where $Z_1 \succeq 0$ is the PSD constraint and α is set to be 0.8. The above problem can be efficiently solved in a similar way as [24]. This transformation step does not improve the performance largely and is actually preparing for the following two steps. Second, as in [12], we decompose Z_1^* into $Z_1^* = QQ^T$ and define $G = (\tilde{Q}\tilde{Q}^T)^2$, where $\tilde{Q}$ is Q with normalized rows. This normalization operator, which is equal to replacing inner product similarities with cosine similarities, can reduce the error rate to 1.86%. Third, like [12], we use G^β ($\beta = 2$) as the affinity matrix for spectral clustering, obtaining an error rate of 0.85% (Std.=2.58%, Max=19.03%).

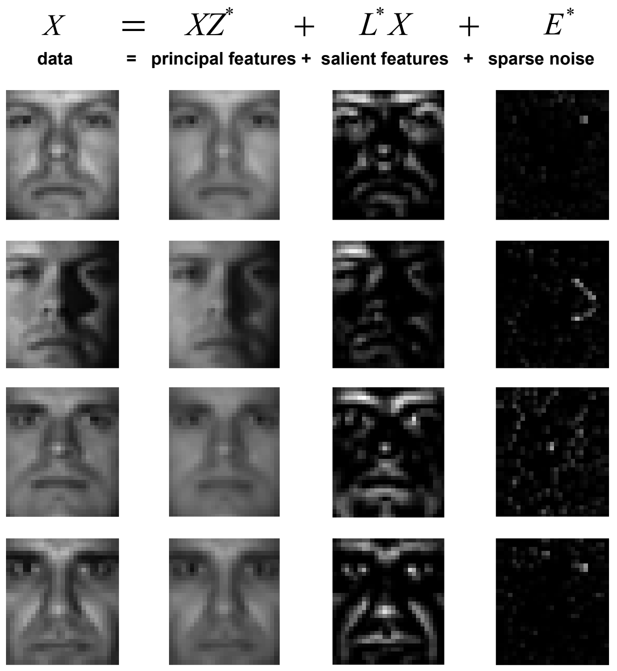
4.2. Feature Extraction

Unsupervised feature extraction is a fundamental step in pattern recognition [20]. In our LatLRR method, the $d \times d$ (assuming the data is of dimension d) matrix identified by L^* may be useful for feature extraction. This heuristic idea has been verified by our experiments. Namely, we experimentally find that L^* is able to extract "salient features" (i.e., notable features such as the eyes of faces) from data. After learning L^* from a set of training data, it is straightforward to generalize the learnt model to fresh testing data. That is, for a testing data vector x, its transformed feature vector y can be calculated by

$$y = L^* x. \qquad (7)$$

Note here that the feature vector y has the same dimension as the original data vector x. This is different from dimension reduction based methods.

Experimental Setting. We test LatLRR's ability to extract salient features from corrupted data, using Extended Yale Database B [13], which consists of 2414 frontal face images of 38 classes. Each class contains about 64 images. First, we randomly split the database into training set and testing set, with the training set containing 1140 images (30 images per class) and the testing set containing the rest 1274 images. Second, we resize the images into 28×32, normalize the pixel values to $[0, 1]$ and use the normalized pixel values to form data vectors of dimension 896. Finally, we use the K-nearest neighbor (K-NN) classifier (based on Euclidean distance) to evaluate the quality of various transformed features.


X = XZ^* + L^*X + E^*
data = principal features + salient features + sparse noise
Figure 4. **Illustrating LatLRR's mechanism of decomposing the data.** Given a data matrix X, LatLRR decomposes it into a low-rank part XZ^* that represents the principal features, a low-rank part L^*X that encodes the salient features and a sparse part E^* that fits noise. These examples are selected from the training set.

Main Results. We compare LatLRR with the benchmark of "Raw Data" that uses the original data vectors as features for classification. For comparison, we also list the results of some popular dimension reduction methods, including Principal Component Analysis (PCA), Locality Preserving Projection (LPP) [9], Neighborhood Preserving Embedding (NPE) [10] and Nonnegative Matrix Factorization (NMF) [8]. Table 2 (left part) shows that LatLRR ($\lambda = 0.4$) can largely outperform the benchmark of "Raw Data". The advantages of LatLRR mainly come from its ability of automatically extracting salient features from corrupted data, as shown in Fig.4 and Fig.5. Fig.4

	Raw Data	PCA (317D)	LPP (83D)	NPE (325D)	NMF (195D)	LatLRR	LatLRR + PCA(400D)	LatLRR + LPP(52D)	LatLRR + NPE(400D)
1-NN	61.07	61.54	80.46	79.28	84.69	**88.76**	87.28	87.60	82.18
3-NN	59.81	60.03	78.73	79.28	84.07	**87.76**	85.95	87.13	81.71
5-NN	58.16	58.54	76.69	76.69	82.58	**86.03**	85.87	85.56	80.85

Table 2. **Classification accuracies (%, averaged over 20 runs) on Extended Yale Database B**. We adopt K-NN ($K = 1, 3, 5$) classifier to classify various features. The parameters of all algorithms have been tuned to the best. For dimension reduction methods, the feature dimension is optimally determined by testing all possible dimensions within $1 \sim 400$. We only explore 1D $\sim$ 400D just because these dimension reduction methods have usually achieved the best performances before 400D.

illustrates that LatLRR can also be regarded as a mechanism for data decomposition. Namely, given a data matrix X, LatLRR decomposes it into a low-rank part XZ^* that represents the "principal" features [6], a low-rank part L^*X that encodes the salient features and a sparse part that fits the noise. In particular, the salient features correspond to the key object parts (e.g., the eyes), which are usually discriminative for recognition. Also, Fig.5 shows that the learnt model (identified by L^*) can generalize well to the testing data. So, LatLRR can achieve much better classification performance than the benchmark of "Raw Data".

The dimension of the feature vector produced by LatLRR is the same as the original data. To improve computation efficiency, we could utilize dimension reduction methods to process the features. Suppose P is a low-dimensional projection learnt by using L^*X as inputs for some dimension reduction methods, the reduced feature vector y of a testing data vector x can be computed by $y = P^T L^* x$. Table 2 (right part) shows the performance of applying PCA, LPP and NPE on the features produced by LatLRR. While reducing the feature dimension to 52D by LPP, we obtain a classification accuracy (1-NN classifier) of 87.60%, which is close to the 88.76% obtained by using the 896D features.

Robustness to Noise. In order to test LatLRR's robustness to the noise possibly appearing in the testing data, we randomly choose some pixels to corrupt, e.g., for a pixel chosen to corrupt, its value is replaced by using a random value that uniformly ranges from 0 to 1 (the pixel values have been normalized to $[0, 1]$). Fig.6 shows that LatLRR is robust to such ad-hoc noise, performing better than dimension reduction based methods.

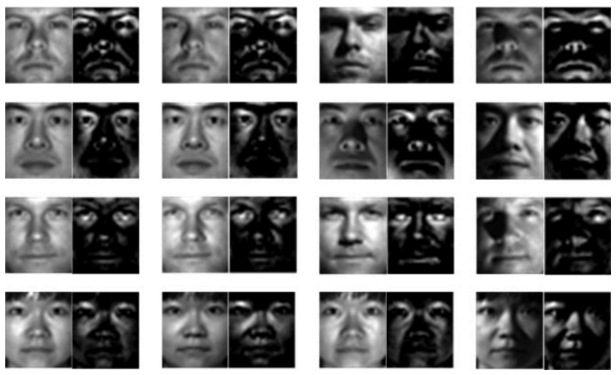


Figure 5. **Some examples of using LatLRR to extract salient features from the testing images.** Each row is from the same class. For each group of images: (left) the testing image x; (right) the intensity map identified by L^*x.

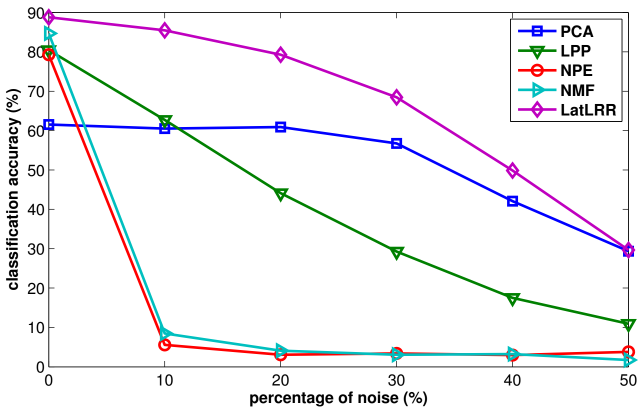


Figure 6. **Testing the robustness of various methods.** The classification accuracy (averaged over 20 runs) across the entire range of noise for various methods. LatLRR and PCA are more robust than the other methods.

5. Conclusions and Future Work

In this paper we proposed Latent Low-Rank Representation (LatLRR) to recover the effects of the unobserved, hidden data in LRR. When all the data, both observed and unobserved, are drawn from the same collection of low-rank subspaces, we show that the hidden effects can be approximately recovered by solving a convex optimization problem. The formulation of LatLRR seamlessly integrates subspace segmentation and feature extraction into a uni-

[6]We call the features represented by XZ^* as principal features because they are visually similar to PCA features. For a certain image, its principal features can be roughly regarded as its projection onto the subspace that represents the image.

fied framework, providing us with a robust subspace segmentation algorithm and also a robust feature extraction algorithm. As a subspace segmentation algorithm, LatLRR could be regarded as an enhanced version of LRR, and thus obtains more accurate segmentation results. Being an unsupervised feature extraction algorithm, LatLRR can automatically extract salient features from corrupted data so as to produce effective features for classification.

Besides the contents presented in this paper, there are several directions left for future work.

- The lowest-rank criterion and the sparest criterion are two typical regularization strategies used in various problems. Each of them has its own advantages and applications. In Sparse Representation (SR), the hidden data is also important for computing the sparsest representation.

- LatLRR also provides a general way to explain the mechanism of generating data. Its application should not be limited to subspace segmentation and feature extraction. In summary, LatLRR can well fit various applications where the data can be decomposed into a right low-rank part, a left low-rank part and a sparse part.

References

[1] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010. 5

[2] A. P. Costeira, Jo and T. Kanade. A multibody factorization method for independently moving objects. *IJCV*, 29(3):159–179, 1998. 1

[3] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *CVPR*, volume 2, pages 2790–2797, 2009. 1, 2, 4, 5

[4] M. Fazel. Matrix rank minimization with applications. *PhD thesis*, 2002. 1

[5] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 1, 5

[6] C. W. Gear. Multibody grouping from motion images. *IJCV*, 29(2):133–150, 1998. 1

[7] A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the EM algorithm. In *CVPR*, volume 1, pages 707–714, 2004. 1

[8] D. Guillamet and J. Vitrià. Non-negative matrix factorization for face recognition. In *CCIA*, pages 336–344, 2002. 6

[9] X. He, D. Cai, H. Liu, and W.-Y. Ma. Locality preserving indexing for document representation. In *SIGIR*, pages 96–103, 2004. 6

[10] X. He, D. Cai, S. Yan, and H.-J. Zhang. Neighborhood preserving embedding. In *ICCV*, pages 1208–1213, 2005. 6

[11] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. J. Kriegman. Clustering appearances of objects under varying illumination conditions. In *CVPR*, volume 1, pages 11–18, 2003. 1

[12] F. Lauer and C. Schnòrr. Spectral clustering of linear subspaces for motion segmentation. In *ICCV*, 2009. 5, 6

[13] K. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *TPAMI*, 27(5):684–698, 2005. 6

[14] Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. Technical report, UILU-ENG-09-2215, 2009. 5

[15] G. Liu, Z. Lin, X. Tang, and Y. Yu. Unsupervised object segmentation with a hybrid graph model (HGM). *TPAMI*, 32(5):910–924, 2010. 1

[16] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *CoRR*, 2010. 1, 2, 3, 5

[17] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *ICML*, pages 663–670, 2010. 1, 2, 4, 5

[18] Y. Ma, H. Derksen, W. Hong, and J. Wright. Segmentation of multivariate mixed data via lossy data coding and compression. *TPAMI*, 29(9):1546–1562, 2007. 1

[19] Y. Ma, A. Yang, H. Derksen, and R. Fossum. Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM Review*, 50(3):413–458, 2008. 1

[20] E. Micheli-Tzanakou, editor. *Supervised and unsupervised pattern recognition: feature extraction and computational intelligence*. CRC Press, Inc., Boca Raton, FL, USA, 2000. 2, 6

[21] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. *TPAMI*, 32(10):1832–1845, 2010. 1, 5

[22] S. Rao, A. Yang, S. Sastry, and Y. Ma. Robust algebraic segmentation of mixed rigid-body and planar motions in two views. *IJCV*, 88(3):425–446, 2010. 1

[23] J. Shi and J. Malik. Normalized cuts and image segmentation. *TPAMI*, pages 888–905, 2000. 5

[24] J. Sun, Y. Ni, X. Yuan, S. Yan, and L.-F. Cheong. Robust low-rank subspace segmentation with semidefinite guarantees. In *ICDM Workshop on Optimization Based Methods for Emerging Data Mining Problems*, 2010. 1, 2, 6

[25] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *CVPR*, pages 1–8, 2007. 5

[26] X. Wang and X. Tang. A unified framework for subspace face recognition. *TPAMI*, 26(9):1222–1228, 2004. 1

[27] J. Wright, Y. Tao, Z. Lin, Y. Ma, and H.-Y. Shum. Classification via minimum incremental coding length (MICL). In *NIPS*. 2008. 1

[28] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *TPAMI*, 31:210–227, 2008. 1

[29] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *ECCV*, volume 4, pages 94–106, 2006. 1, 5