Sparse Subspace Clustering

Ehsan Elhamifar René Vidal
Center for Imaging Science, Johns Hopkins University, Baltimore MD 21218, USA

Abstract

We propose a method based on sparse representation (SR) to cluster data drawn from multiple low-dimensional linear or affine subspaces embedded in a high-dimensional space. Our method is based on the fact that each point in a union of subspaces has a SR with respect to a dictionary formed by all other data points. In general, finding such a SR is NP hard. Our key contribution is to show that, under mild assumptions, the SR can be obtained 'exactly' by using ℓ_1 optimization. The segmentation of the data is obtained by applying spectral clustering to a similarity matrix built from this SR. Our method can handle noise, outliers as well as missing data. We apply our subspace clustering algorithm to the problem of segmenting multiple motions in video. Experiments on 167 video sequences show that our approach significantly outperforms state-of-the-art methods.

1. Introduction

Subspace clustering is an important problem with numerous applications in image processing, *e.g.* image representation and compression [15, 29], and computer vision, *e.g.* image/motion/video segmentation [6, 16, 30, 28, 26]. Given a set of points drawn from a union of subspaces, the task is to find the number of subspaces, their dimensions, a basis for each subspace, and the segmentation of the data.

Prior work on subspace clustering. Existing works on subspace clustering can be divided into six main categories: iterative, statistical, factorization-based, spectral clustering, algebraic and information-theoretic approaches. Iterative approaches, such as K-subspaces [14], alternate between assigning points to subspaces, and fitting a subspace to each cluster. Statistical approaches, such as Mixtures of Probabilistic PCA (MPPCA) [24], Multi-Stage Learning (MSL) [22], or [13], assume that the distribution of the data inside each subspace is Gaussian and alternate between data clustering and subspace estimation by applying Expectation Maximization (EM) to a mixture of probabilistic PCAs. The main drawbacks of both approaches are that they generally require the number and dimensions of the subspaces to be known, and that they are sensitive to correct initialization. Robust methods, such as Random Sample Consensus (RANSAC) [11], fit a subspace of dimension d to randomly chosen subsets of d points until the number of inliers is large enough. The inliers are then removed, and the process is repeated to find a second subspace, and so on. RANSAC can deal with noise and outliers, and does need to know the number of subspaces. However, the dimensions of the subspaces must be known and equal, and the number of trials needed to find d points in the same subspace grows exponentially with the number and dimension of the subspaces.

Factorization-based methods [6, 12, 16] find an initial segmentation by thresholding the entries of a similarity matrix built from the factorization of the matrix of data points. Such methods are provably correct when the subspaces are independent, but fail when this assumption is violated. Also, these methods are sensitive to noise. Spectral-clustering methods [30, 10, 28] deal with these issues by using local information around each point to build a similarity between pairs of points. The segmentation of the data is then obtained by applying spectral clustering to this similarity matrix. These methods have difficulties dealing with points near the intersection of two subspaces, because the neighborhood of a point can contain points from different subspaces. This issue can be resolved by looking at multi-way similarities that capture the curvature of a collection of points within an affine subspace [5]. However, the complexity of building a multi-way similarity grows exponentially with the number of subspaces and their dimensions.

Algebraic methods, such as Generalized Principal Component Analysis (GPCA) [25, 18], fit the data with a polynomial whose gradient at a point gives a vector normal to the subspace containing that point. Subspace clustering is then equivalent to fitting and differentiating polynomials. GPCA can deal with subspaces of different dimensions, and does not impose any restriction on the relative orientation of the subspaces. However, GPCA is sensitive to noise and outliers, and its complexity increases exponentially with the number of subspaces and their dimensions. Information-theoretic approaches, such as Agglomerative Lossy Compression (ALC) [17], model each subspace with a degenerate Gaussian, and look for the segmentation of the data that minimizes the coding length needed to fit these points with a mixture of Gaussians. As this minimization problem

978-1-4244-3991-1/09/$25.00 ©2009 IEEE

is NP hard, a suboptimal solution is found by first assuming that each point forms its own group, and then iterative merging pairs of groups to reduce the coding length. ALC can handle noise and outliers in the data, and can estimate the number of subspaces and their dimensions. However, there is no theoretical proof for the optimality of the algorithm.

Paper contributions. In this paper, we propose a completely different approach to subspace clustering based on sparse representation. Sparse representation of signals has attracted a lot of attention during the last decade, especially in the signal and image processing communities (see §2 for a brief review). However, its application to computer vision problems is fairly recent. [21] uses ℓ_1 optimization to deal with missing or corrupted data in motion segmentation. [20] uses sparse representation for restoration of color images. [27] uses ℓ_1 minimization for recognizing human faces from frontal views with varying expression and illumination as well as occlusion. [19] uses a sparse representation to learn a dictionary for object recognition.

Our work is the first one to directly use the sparse representation of vectors lying on a union of subspaces to cluster the data into separate subspaces. We exploit the fact that each data point in a union of subspaces can always be written as a linear or affine combination of all other points. By searching for the *sparsest* combination, we automatically obtain other points lying in the same subspace. This allows us to build a similarity matrix, from which the segmentation of the data can be easily obtained using spectral clustering. Our work has numerous advantages over the state of the art.

– Our sparse representation approach resolves the exponential complexity issue of methods such as RANSAC, spectral clustering, and GPCA. While in principle finding the sparsest representation is also an NP hard problem, we show that under mild assumptions on the distribution of data on the subspaces, the sparsest representation can be found efficiently by solving a (convex) ℓ_1 optimization problem.

– Our work extends sparse representation work from one to multiple subspaces. As we will see in §2, most of the sparse representation literature assumes that the data lies in a single linear subspace [1, 4, 7]. The work of [9] is the first one to address the case of multiple subspaces, under the assumption that a sparsifying basis for each subspace is known. Our case is more challenging, because we do not have any basis for any of the subspaces nor do we know which data belong to which subspace. We only have the sparsifying basis for the union of subspaces given by the data matrix.

– Our work requires no initialization, can deal with both linear and affine subspaces, can handle data points near the intersections, noise, outliers, and missing data.

– Last, but not least, our method significantly outperforms existing motion segmentation algorithms on 167 sequences.

2. Sparse representation and compressed sensing

Compressed sensing (CS) is based on the idea that many signals or vectors can have a concise representation when expressed in a proper basis. So, the information rate of a sparse signal is usually much smaller than the rate suggested by its maximum frequency. In this section, we review recently developed techniques from CS for sparsely representing signals lying in one or more subspaces.

2.1. Sparse representation in a single subspace

Consider a vector $\boldsymbol{x}$ in $\mathbb{R}^D$, which can be represented in a basis of D vectors $\{\boldsymbol{\psi}_i \in \mathbb{R}^D\}_{i=1}^D$. If we form the basis matrix $\Psi = [\boldsymbol{\psi}_1,\ \boldsymbol{\psi}_2,\ \cdots,\ \boldsymbol{\psi}_D]$, we can write $\boldsymbol{x}$ as:

$$\boldsymbol{x} = \sum_{i=1}^{D} s_i \boldsymbol{\psi}_i = \Psi \boldsymbol{s} \tag{1}$$

where $\boldsymbol{s} = [s_1, s_2, \ldots, s_D]^\top$. Both $\boldsymbol{x}$ and $\boldsymbol{s}$ represent the same signal, one in the space domain and the other in the Ψ domain. However, in many cases $\boldsymbol{x}$ can have a sparse representation in a properly chosen basis Ψ. We say that $\boldsymbol{x}$ is K-*sparse* if it is a linear combination of at most K basis vectors in Ψ, *i.e.* if at most K of the coefficients are nonzero. In practice, the signal is K-sparse when it has at most K large nonzero coefficients and the remaining coefficients are very small. We are in general interested in the case where $K \ll D$.

Assume now that we do not measure $\boldsymbol{x}$ directly. Instead, we measure m linear combinations of entries of $\boldsymbol{x}$ of the form $y_i = \boldsymbol{\phi}_i^\top \boldsymbol{x}$ for $i \in \{1, 2, \cdots, m\}$. We thus have

$$\boldsymbol{y} = [y_1, y_2, \cdots, y_m]^\top = \Phi \boldsymbol{x} = \Phi\, \Psi\, \boldsymbol{s} = A\, \boldsymbol{s}\,, \tag{2}$$

where $\Phi = [\boldsymbol{\phi}_1,\ \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_m]^\top \in \mathbb{R}^{m \times D}$ is called the measurement matrix. The works of [1, 4, 7] show that, given m measurements, one can recover K-sparse signals/vectors if $K \lesssim m/\log(D/m)$. In principle, such a sparse representation can be obtained by solving the optimization problem:

$$\min \|\boldsymbol{s}\|_0 \quad \text{subject to} \quad \boldsymbol{y} = A\boldsymbol{s}, \tag{3}$$

where $\|\boldsymbol{s}\|_0$ is the ℓ_0 norm of $\boldsymbol{s}$, *i.e.* the number of nonzero elements. However, such an optimization problem is in general non-convex and NP-hard. This has motivated the development of several methods for efficiently extracting a sparse representation of signals/vectors. One of the well-known methods is the Basis Pursuit (BP) algorithm, which replaces the non-convex optimization in (3) by the following convex ℓ_1 optimization problem [7]:

$$\min \|\boldsymbol{s}\|_1 \quad \text{subject to} \quad \boldsymbol{y} = A\boldsymbol{s}. \tag{4}$$

The works of [3, 2] show that we can recover perfectly a K-sparse signal/vector by using the Basis Pursuit algorithm in (4) under certain conditions on the so-called *isometry constant* of the A matrix.

2.2. Sparse representation in a union of subspaces

Most of the work on CS deals with sparse representation of signals/vectors lying in a single low-dimensional linear subspace. The more general case where the signals/vectors lie in a union of low-dimensional linear subspaces was only recently considered. The work of Eldar [9] shows that when the subspaces are disjoint (intersect only at the origin), a basis for each subspace is known, and certain condition on a modified isometry constant holds, one can recover the block-sparse vector $\boldsymbol{s}$ exactly by solving an ℓ_1/ℓ_2 optimization problem.

More precisely, let $\{A_i \in \mathbb{R}^{D\times d_i}\}_{i=1}^n$ be a set of bases for n disjoint linear subspaces embedded in $\mathbb{R}^D$ with dimensions $\{d_i\}_{i=1}^n$. If $\boldsymbol{y}$ belongs to the i-th subspace, we can represent it as the sparse solution of

$$\boldsymbol{y} = A\boldsymbol{s} = [A_1,\ A_2, \cdots,\ A_n]\,[\boldsymbol{s}_1^\top, \boldsymbol{s}_2^\top, \cdots, \boldsymbol{s}_n^\top]^\top, \quad (5)$$

where $\boldsymbol{s}_i \in \mathbb{R}^{d_i}$ is a nonzero vector and all other vectors $\{\boldsymbol{s}_j \in \mathbb{R}^{d_j}\}_{j\neq i}$ are zero. Therefore, $\boldsymbol{s}$ is the solution to the following non-convex optimization problem:

$$\min \sum_{i=1}^{n} 1(\|\boldsymbol{s}_i\|_2 > 0) \quad \text{subject to} \quad \boldsymbol{y} = A\boldsymbol{s}, \quad (6)$$

where $1(\|\boldsymbol{s}_i\|_2 > 0)$ is an indicator function that takes the value 1 when $\|\boldsymbol{s}_i\|_2 > 0$ and zero otherwise. [9] shows that if a modified isometry constant satisfies a certain condition, then the solution to the (convex) ℓ_2/ℓ_1 program

$$\min \sum_{i=1}^{n} \|\boldsymbol{s}_i\|_2 \quad \text{subject to} \quad \boldsymbol{y} = A\boldsymbol{s} \quad (7)$$

coincides with that of (6).

In this paper, we address the problem of clustering data lying in multiple linear or affine subspaces. This subspace clustering problem is more challenging, because the subspace bases $\{A_i\}_{i=1}^n$ and the subspace dimensions $\{d_i\}_{i=1}^n$ are unknown, and hence we do not know a priori which data points belong to which subspace. To the best of our knowledge, our work is the first one to use sparse representation techniques to address the subspace clustering problem.

3. Subspace clustering via sparse representation

In this section, we consider the problem of clustering a collection of data points drawn from a union of subspaces using sparse representation. First we consider the case where all subspaces are linear and then we extend our result to the more general case of affine subspaces.

3.1. Clustering linear subspaces

Let $\{\boldsymbol{y}_j \in \mathbb{R}^D\}_{j=1}^N$ be a collection of data points drawn from a union of n *independent*[1] linear subspaces $\{\mathcal{S}_i\}_{i=1}^n$. Let $\{d_i \ll D\}_{i=1}^n$ and $\{A_i \in \mathbb{R}^{D\times d_i}\}_{i=1}^n$ be, respectively, the unknown dimensions and bases for the n subspaces. Let $Y_i \in \mathbb{R}^{D\times N_i}$ be the collection of N_i data points drawn from subspace i. Since we do not know which points belong to which subspace, our data matrix is of the form $Y = [\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N] = [Y_1, Y_2, \cdots, Y_n]\,\Gamma \in \mathbb{R}^{D\times N}$, where $N = \sum_{i=1}^n N_i$ and $\Gamma \in \mathbb{R}^{N\times N}$ is an unknown permutation matrix that specifies the segmentation of data.

Although we do not know the subspace bases, we know that such bases can be chosen from the columns of the data matrix Y. In fact, if we assume that there are enough data points from each linear subspace, $N_i \geq d_i$, and that these data points are in general positions, meaning that no d_i points from subspace i live in a (d_i-1)-dimensional subspace, then the collection of data points is *self-expressive*. This means that if $\boldsymbol{y}$ is a new data point in $\mathcal{S}_i$, then it can be represented as a linear combination of d_i points in the same subspace. Thus if we let $\boldsymbol{s} = \Gamma^{-1}[\boldsymbol{s}_1^\top, \boldsymbol{s}_2^\top, \cdots, \boldsymbol{s}_n^\top]^\top \in \mathbb{R}^N$, where $\boldsymbol{s}_i \in \mathbb{R}^{N_i}$, then $\boldsymbol{y}$ has a d_i-sparse representation, which can be recovered as a sparse solution of $\boldsymbol{y} = Y\boldsymbol{s}$, with $\boldsymbol{s}_i \neq 0$ and $\boldsymbol{s}_j = 0$ for all $j \neq i$. That is, $\boldsymbol{s}$ is a solution of the following non-convex optimization problem

$$\min \|\boldsymbol{s}\|_0 \quad \text{subject to} \quad \boldsymbol{y} = Y\boldsymbol{s} \quad (8)$$

which is an NP-hard problem to solve.[2]

The following theorem shows that when the subspaces are independent[1], the ℓ_1 optimization problem

$$\min \|\boldsymbol{s}\|_1 \quad \text{subject to} \quad \boldsymbol{y} = Y\boldsymbol{s} \quad (9)$$

gives block sparse solutions with the nonzero block corresponding to points in the same subspace as $\boldsymbol{y}$.

Theorem 1 *Let $Y \in \mathbb{R}^{D\times N}$ be a matrix whose columns are drawn from a union of n independent linear subspaces. Assume that the points within each subspace are in general position. Let $\boldsymbol{y}$ be a new point in subspace i. The solution to the ℓ_1 problem in* (9) *$\boldsymbol{s} = \Gamma^{-1}[\boldsymbol{s}_1^\top, \boldsymbol{s}_2^\top, \cdots, \boldsymbol{s}_n^\top]^\top \in \mathbb{R}^N$ is block sparse,* i.e. *$\boldsymbol{s}_i \neq 0$ and $\boldsymbol{s}_j = 0$ for all $j \neq i$.*

Proof. Let $\boldsymbol{s}$ be any sparse representation of the data point $\boldsymbol{y} \in \mathcal{S}_i$, *i.e.* $\boldsymbol{y} = Y\boldsymbol{s}$ with $\boldsymbol{s}_i \neq 0$ and $\boldsymbol{s}_j = 0$ for all $j \neq i$. Since the points in each subspace are in general positions, such a sparse representation exists. Now, if $\boldsymbol{s}^*$ is a solution of the ℓ_1 program in (9), then $\boldsymbol{s}^*$ is a vector of minimum

[1] A collection of n linear subspaces $\{S_i \subset \mathbb{R}^D\}_{i=1}^n$ are independent if $\dim(\bigoplus_{i=1}^n S_i) = \sum_{i=1}^n \dim(S_i)$, where $\oplus$ is the direct sum.

[2] Notice that our optimization problem in (8) is different from the one in (6), because we do not know the subspace basis or the permutation matrix Γ, and hence we cannot enforce that $\boldsymbol{s}_j = 0$ for $j \neq i$ whenever $\boldsymbol{s}_i \neq 0$.

ℓ_1 norm satisfying $\boldsymbol{y} = Y\boldsymbol{s}^*$. Let $\boldsymbol{h} = \boldsymbol{s}^* - \boldsymbol{s}$ denote the error between the optimal solution and our sparse solution. Then, we can write $\boldsymbol{h}$ as the sum of two vectors $\boldsymbol{h}_i$ and $\boldsymbol{h}_{i^c}$ supported on disjoint subsets of indices: $\boldsymbol{h}_i$ represents the error for the corresponding points in subspace i and $\boldsymbol{h}_{i^c}$ the error for the corresponding points in other subspaces. We now show that $\boldsymbol{h}_{i^c} = 0$. For the sake of contradiction, assume that $\boldsymbol{h}_{i^c} \neq 0$. Since $\boldsymbol{s}^* = \boldsymbol{s} + \boldsymbol{h}_i + \boldsymbol{h}_{i^c}$, we have that $\boldsymbol{y} = Y\boldsymbol{s}^* = Y(\boldsymbol{s} + \boldsymbol{h}_i) + Y\boldsymbol{h}_{i^c}$. Also, since $\boldsymbol{y} \in \mathcal{S}_i$, $Y(\boldsymbol{s} + \boldsymbol{h}_i) \in \mathcal{S}_i$, and from the independence assumption $Y\boldsymbol{h}_{i^c} \notin \mathcal{S}_i$, we have that $Y\boldsymbol{h}_{i^c} = 0$. This implies that

$$\boldsymbol{y} = Y\boldsymbol{s}^* = Y(\boldsymbol{s} + \boldsymbol{h}_i).$$

Now, from the fact that $\boldsymbol{h}_i$ and $\boldsymbol{h}_{i^c}$ are supported on disjoint subset of indices, we have $\|\boldsymbol{s} + \boldsymbol{h}_i\|_1 < \|\boldsymbol{s} + \boldsymbol{h}_i + \boldsymbol{h}_{i^c}\|_1 = \|\boldsymbol{s}^*\|_1$. In other words, $\boldsymbol{s} + \boldsymbol{h}_i$ is a feasible solution for the ℓ_1 program in (9) whose ℓ_1 norm is smaller than that of the optimal solution. This contradicts the optimality of the solution $\boldsymbol{s}^*$. Thus we must always have $\boldsymbol{s}^*_{i^c} = \boldsymbol{s}_{i^c} = 0$, meaning that only the block corresponding to the points in the true subspace can have nonzero entries. ∎

Theorem 1 gives sufficient conditions on subspaces and the data matrix in order to be able to recover a block sparse representation of a new data point as a linear combination of the points in the data matrix that are in the same subspace. We now show how to use such a sparse representation for clustering the data according to the multiple subspaces.

Let $Y_{\hat{i}} \in \mathbb{R}^{D \times N-1}$ be the matrix obtained from Y by removing its i-th column, $\boldsymbol{y}_i$. The circumflex notation $\hat{i}$ thus means "not i". According to Theorem 1, if $\boldsymbol{y}_i$ belongs to the j-th subspace, then it has a sparse representation with respect to the basis matrix $Y_{\hat{i}}$. Moreover, such a representation can be recovered by solving the following ℓ_1 program

$$\min \|\boldsymbol{c}_i\|_1 \quad \text{subject to} \quad \boldsymbol{y}_i = Y_{\hat{i}}\boldsymbol{c}_i. \tag{10}$$

The optimal solution $\boldsymbol{c}_i \in \mathbb{R}^{N-1}$ is a vector whose nonzero entries correspond to points (columns) in $Y_{\hat{i}}$ that lie in the same subspace as $\boldsymbol{y}_i$. Thus, by inserting a zero entry at the i-th row of $\boldsymbol{c}_i$, we make it an N-dimensional vector, $\hat{\boldsymbol{c}}_i \in \mathbb{R}^N$, whose nonzero entries correspond to points in Y that lie in the same subspace as $\boldsymbol{y}_i$.

After solving (10) at each point $i = 1, \ldots, N$, we obtain a matrix of coefficients $C = [\hat{\boldsymbol{c}}_1, \hat{\boldsymbol{c}}_2, \cdots, \hat{\boldsymbol{c}}_N] \in \mathbb{R}^{N \times N}$. We use this matrix to define a directed graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$. The vertices of the graph $\boldsymbol{V}$ are the N data points, and there is an edge $(v_i, v_j) \in \boldsymbol{E}$ when the data point $\boldsymbol{y}_j$ is one of the vectors in the sparse representation of $\boldsymbol{y}_i$, *i.e.* when $C_{ji} \neq 0$. One can easily see that the adjacency matrix of the $\boldsymbol{G}$ is C.

In general $\boldsymbol{G}$ is an unbalanced digraph. To make it balanced, we build a new graph $\tilde{\boldsymbol{G}}$ with the adjacency matrix $\tilde{C}$ where $\tilde{C}_{ij} = |C_{ij}| + |C_{ji}|$. $\tilde{C}$ is still a valid representation of the similarity, because if $\boldsymbol{y}_i$ can write itself as a linear combination of some points including $\boldsymbol{y}_j$ (all in the same subspace), then $\boldsymbol{y}_j$ can also write itself as a linear combination of some points in the same subspace including $\boldsymbol{y}_i$.

Having formed the similarity graph $\tilde{\boldsymbol{G}}$, it follows from Theorem 1 that all vertices representing the data points in the same subspace form a connected component in the graph, while the vertices representing points in different subspaces have no edges between them. Therefore, in the case of n subspaces, $\tilde{C}$ has the following block diagonal form

$$\tilde{C} = \begin{bmatrix} \tilde{C}_1 & 0 & \cdots & 0 \\ 0 & \tilde{C}_2 & \cdots & 0 \\ & & \vdots & \\ 0 & 0 & \cdots & \tilde{C}_n \end{bmatrix} \Gamma \tag{11}$$

where Γ is a permutation matrix. The Laplacian matrix of $\tilde{\boldsymbol{G}}$ is then formed by $L = D - \tilde{C}$ where $D \in \mathbb{R}^{N \times N}$ is a diagonal matrix with $D_{ii} = \sum_j \tilde{C}_{ij}$.

We use the following result from spectral graph theory to infer the segmentation of the data by applying K-means to a subset of eigenvectors of the Laplacian.

Proposition 1 *The multiplicity of the zero eigenvalue of the Laplacian matrix L corresponding to the graph $\tilde{\boldsymbol{G}}$ is equal to the number of connected components of the graph. Also, the components of the graph can be determined from the eigenspace of the zero eigenvalue. More precisely, if the graph has n connected components, then $u_i = [0, 0, \ldots, \mathbf{1}^\top_{N_i}, 0, \ldots, 0]\Gamma$ for $i \in \{1, 2, \ldots, n\}$ is the i-th eigenvector of L corresponding to the zero eigenvalue which means that the N_i nonzero elements of u_i belong to the same group.*

For data points drawn in general position from n independent linear subspaces, the similarity graph $\tilde{\boldsymbol{G}}$ will have n connected components. Therefore, when the number of subspaces is unknown, we can estimate it as the number of zero eigenvalues of L. In the case of real data with noise, we have to consider a robust measure to determine the number of eigenvalues of L close to zero.

3.2. Clustering affine subspaces

In many cases we need to cluster data lying in multiple affine rather than linear subspaces. For instance, the motion segmentation problem we will discuss in the next section involves clustering data lying in multiple 3-dimensional affine subspaces. However, most existing motion segmentation algorithms deal with this problem by clustering the data as if they belonged to multiple 4-dimensional linear subspaces.

In this section, we show that our method can easily handle the case of affine subspaces by a simple modification to the BP algorithm. The modified ℓ_1 minimization is still a convex optimization, which can be efficiently implemented. More specifically, notice that in the case of affine subspace,

a point can no longer write itself as a linear combination of points in the same subspace. However, we can still write a point $\boldsymbol{y}$ as an affine combination of other points, *i.e.*

$$\boldsymbol{y} = c_1\boldsymbol{y}_1 + c_2\boldsymbol{y}_2 + \cdots + c_N\boldsymbol{y}_N, \quad \sum_{i=1}^{N} c_i = 1. \tag{12}$$

Theorem 2 shows that one can recover the sparse representation of data points on an affine subspace by using the following modified Basis Pursuit algorithm

$$\min \|\boldsymbol{c}\|_1 \quad \text{subject to} \quad \boldsymbol{y} = Y\,\boldsymbol{c} \;\text{ and }\; \boldsymbol{c}^\top \mathbf{1} = 1. \tag{13}$$

Theorem 2 *Let $Y \in \mathbb{R}^{D\times N}$ be a matrix whose columns are drawn from a union of n independent[3] affine subspaces. Assume that the points within each subspace are in general position. Let $\boldsymbol{y}$ be a new point in subspace i. The solution to the ℓ_1 problem in* (13)*, $\boldsymbol{s} = \Gamma^{-1}[\boldsymbol{s}_1^\top, \boldsymbol{s}_2^\top, \cdots, \boldsymbol{s}_n^\top]^\top \in \mathbb{R}^N$ is block sparse,* i.e. *$\boldsymbol{s}_i \neq 0$ and $\boldsymbol{s}_j = 0$ for all $j \neq i$.*

Proof. Analogous to that of Theorem 1. ∎

Similar to what we did for linear subspaces, we can use this result for clustering a collection of data points drawn from n affine subspaces. Essentially, we solve the following ℓ_1 minimization problem for each data point $\boldsymbol{y}_i$

$$\min \|\boldsymbol{c}_i\|_1 \quad \text{subject to} \quad \boldsymbol{y}_i = Y_{\hat{i}}\,\boldsymbol{c}_i \;\text{ and }\; \boldsymbol{c}_i^\top \mathbf{1} = 1, \tag{14}$$

and form the graph $\tilde{\boldsymbol{G}}$ from the sparse coefficients. We then apply spectral clustering to the corresponding Laplacian matrix in order to get the segmentation of data.

3.3. Subspace clustering with noisy data

Consider now the case where the data points drawn from a collection of linear or affine subspaces are contaminated with noise. More specifically, let $\bar{\boldsymbol{y}}_i = \boldsymbol{y}_i + \zeta_i$ be the i-th data point corrupted with noise ζ_i bounded by $\|\zeta_i\|_2 \leq \epsilon$. In order to recover the sparse representation of $\bar{\boldsymbol{y}}_i$, we can look for the sparsest solution of $\bar{\boldsymbol{y}}_i = Y_{\hat{i}}\boldsymbol{c}_i$ with an error of at most ϵ, *i.e.* $\|Y_{\hat{i}}\boldsymbol{c}_i - \bar{\boldsymbol{y}}_i\|_2 \leq \epsilon$. We can find such a sparse representation by solving the following problem

$$\min \|\boldsymbol{c}_i\|_1 \quad \text{subject to} \quad \|Y_{\hat{i}}\boldsymbol{c}_i - \bar{\boldsymbol{y}}_i\|_2 \leq \epsilon. \tag{15}$$

However, in many situations we do not know the noise level ϵ beforehand. In such cases we can use the Lasso optimization algorithm [23] to recover the sparse solution from

$$\min \|\boldsymbol{c}_i\|_1 + \gamma\,\|Y_{\hat{i}}\boldsymbol{c}_i - \bar{\boldsymbol{y}}_i\|_2 \tag{16}$$

where $\gamma > 0$ is a constant. In the case data drawn from multiple affine subspaces and corrupted with noise, the sparse representation can be obtained by solving the problem

$$\min \|\boldsymbol{c}_i\|_1 \text{ subject to } \|Y_{\hat{i}}\boldsymbol{c}_i - \bar{\boldsymbol{y}}_i\|_2 \leq \epsilon \text{ and } \boldsymbol{c}_i^\top \mathbf{1} = 1 \tag{17}$$

[3] A collection of affine subspaces is said to be independent if they are independent as linear subspaces in homogeneous coordinates.

or the modified Lasso counterpart

$$\min \|\boldsymbol{c}_i\|_1 + \gamma\,\|Y_{\hat{i}}\boldsymbol{c}_i - \bar{\boldsymbol{y}}_i\|_2 \quad \text{subject to} \quad \boldsymbol{c}_i^\top \mathbf{1} = 1. \tag{18}$$

Segmentation of the data into different subspaces then follows by applying spectral clustering to the Laplacian of $\tilde{\boldsymbol{G}}$.

3.4. Clustering with missing or corrupted data

In practice, some of the entries of the data points may be missing (incomplete data), or corrupted (outliers). In motion segmentation, for example, due to occlusions or limitations of the tracker, we may loose some feature points in some of the frames (missing entries), or the tracker may loose track of some features, leading to gross errors. As suggested in [21], we can fill in missing entries or correct gross errors using sparse techniques. In this section, we show that one can also cluster data points with missing or corrupted entries using a sparse representation.

Let $I_i \subset \{1, \ldots, D\}$ denote the indices of missing entries in $\boldsymbol{y}_i \in \mathbb{R}^D$. Let $Y_{\hat{i}} \in \mathbb{R}^{D\times N-1}$ be obtained by eliminating the vector $\boldsymbol{y}_i$ from the i-th column of the data matrix Y. We then form $\tilde{\boldsymbol{y}}_i \in \mathbb{R}^{D-|I_i|}$ and $\tilde{Y}_{\hat{i}} \in \mathbb{R}^{D-|I_i|\times N-1}$ by eliminating rows of $\boldsymbol{y}_i$ and $Y_{\hat{i}}$ indexed by I_i, respectively. Assuming that $\tilde{Y}_{\hat{i}}$ is complete, we can find a sparse representation, $\boldsymbol{c}_i^*$, for $\tilde{\boldsymbol{y}}_i$ by solving the following problem

$$\min \|\boldsymbol{c}_i\|_1 + \gamma\,\|\tilde{Y}_{\hat{i}}\boldsymbol{c}_i - \tilde{\boldsymbol{y}}_i\|_2 \quad \text{subject to} \quad \boldsymbol{c}_i^\top \mathbf{1} = 1. \tag{19}$$

The missing entries of $\boldsymbol{y}_i$ are then given by $\boldsymbol{y}_i^* = Y_{\hat{i}}\boldsymbol{c}_i^*$. Notice that this method for completion of missing data is essentially the same as our method for computing the sparse representation from complete data with noise in (18). Hence we can use the sparse coefficient vectors $\{\boldsymbol{c}_i^*\}_{i=1}^N$ to build the similarity graph and find the segmentation of data.

Assume now that a few entries of each data point are corrupted. We can also use the sparse representation to correct such entries. More precisely, let $\tilde{\boldsymbol{y}}_i \in \mathbb{R}^D$ be a corrupted vector obtained from $\bar{\boldsymbol{y}}_i = \boldsymbol{y}_i + \zeta_i$ by adding a sparse error vector $\boldsymbol{e}_i \in \mathbb{R}^D$ as $\tilde{\boldsymbol{y}}_i = \boldsymbol{y}_i + \zeta_i + \boldsymbol{e}_i$. We can then write

$$\tilde{\boldsymbol{y}}_i = Y_{\hat{i}}\boldsymbol{c}_i + \boldsymbol{e}_i = [Y_{\hat{i}}\; I_D]\begin{bmatrix}\boldsymbol{c}_i \\ \boldsymbol{e}_i\end{bmatrix} + \zeta_i, \tag{20}$$

where the coefficient vector $[\boldsymbol{c}_i^\top, \boldsymbol{e}_i^\top]^\top$ is still sparse, and hence can be recovered from

$$\min \|\begin{bmatrix}\boldsymbol{c}_i \\ \boldsymbol{e}_i\end{bmatrix}\|_1 + \gamma\,\|\tilde{\boldsymbol{y}}_i - [Y_{\hat{i}}\; I_D]\begin{bmatrix}\boldsymbol{c}_i \\ \boldsymbol{e}_i\end{bmatrix}\|_2 \;\text{ subject to }\; \boldsymbol{c}_i^\top \mathbf{1} = 1.$$

We can then recover the original vector without outliers as $\boldsymbol{y}_i^* = Y_{\hat{i}}\boldsymbol{c}_i^*$. As before, we can obtain the segmentation from the sparse coefficients $\{\boldsymbol{c}_i^*\}_{i=1}^N$ using spectral clustering.

In summary, we have the following *Sparse Subspace Clustering* (SSC) algorithm for clustering data drawn from a collection of linear/affine subspaces, and corrupted by noise, missing entries, and outliers.

1. For every data point $\boldsymbol{y}_i \in Y \in \mathbb{R}^{D\times N}$
   (a) Form $\tilde{\boldsymbol{y}}_i \in \mathbb{R}^{D-|I_i|}$ and $\tilde{Y}_{\hat{i}} \in \mathbb{R}^{D-|I_i|\times N-1}$ by eliminating rows of Y indexed by I_i. If needed, also eliminate columns of Y that have missing entries in I_i^c. If $\boldsymbol{y}_i$ is complete, then $I_i = \emptyset$.
   (b) Find sparse vectors $\boldsymbol{c}_i^*$ and $\boldsymbol{e}_i^*$ from
   $$\min \ \|\begin{bmatrix}\boldsymbol{c}_i\\ \boldsymbol{e}_i\end{bmatrix}\|_1 + \gamma \|\tilde{\boldsymbol{y}}_i - [\tilde{Y}_{\hat{i}} \ I_{D-|I_i|}] \begin{bmatrix}\boldsymbol{c}_i\\ \boldsymbol{e}_i\end{bmatrix}\|_2$$
   for linear subspaces, with the additional constraint $\boldsymbol{c}_i^\top \mathbf{1} = 1$ for affine subspaces.
   (c) Compute $\boldsymbol{y}_i^* = \tilde{Y}_{\hat{i}}\boldsymbol{c}_i^*$, which gives the complete trajectories without outlying entries.
2. Form the graph $\tilde{\boldsymbol{G}}$ from sparse coefficients $\{\boldsymbol{c}_i^*\}_{i=1}^N$ and compute the Laplacian matrix L of the graph.
3. Apply K-means to the n eigenvectors of the L corresponding to the smallest n eigenvalues in order to find segmentation of the data.

4. Application to motion segmentation

Motion segmentation refers to the problem of separating a video sequence into multiple spatiotemporal regions corresponding to different rigid-body motions in the scene. Under the affine projection model, all the trajectories associated with a single rigid motion live in a 3-dimensional affine subspace, as we show below. Therefore, the motion segmentation problem reduces to clustering a collection of point trajectories according to multiple affine subspaces.

More specifically, let $\{x_{fp} \in \mathbb{R}^2\}_{p=1,\dots,P}^{f=1,\dots,F}$ denote the tracked feature points trajectories in F 2-D image frames of P points $\{X_p \in \mathbb{R}^3\}_{p=1,\dots,P}$ on a rigidly moving object. The relation between the tracked feature points and the corresponding 3-D coordinates of the points on the object under the affine camera model is given by

$$x_{fp} = A_f \begin{bmatrix}X_p\\ 1\end{bmatrix} \tag{21}$$

where $A_f \in \mathbb{R}^{2\times 4}$ is the affine motion matrix at frame f. If we form a matrix containing all the F tracked feature points corresponding to a point on the object in a column, we get

$$\begin{bmatrix}x_{11}\cdots x_{1P}\\ \vdots\\ x_{F1}\cdots x_{FP}\end{bmatrix}_{2F\times P} = \begin{bmatrix}A_1\\ \vdots\\ A_F\end{bmatrix}_{2F\times 4} \begin{bmatrix}X_1\cdots X_P\\ 1\cdots 1\end{bmatrix}_{4\times N} \tag{22}$$

We can briefly write this as $W = MS^\top$ where $M \in \mathbb{R}^{2F\times 4}$ is called the motion matrix and $S \in \mathbb{R}^{N\times 4}$ is called the structure matrix. Since $\text{rank}(M), \text{rank}(S) \le 4$ we get

$$\text{rank}(W) = \text{rank}(MS^\top) \le \min(\text{rank}(M), \text{rank}(S)) \le 4. \tag{23}$$

Since the last row of $S^\top$ is 1, under the affine camera model the trajectories of feature points of a single rigid motion lie in an affine subspace of $\mathbb{R}^{2F}$ of dimension at most three.

Now, assume we are given P trajectories of n rigidly moving objects. Then, these trajectories will lie in a union of n affine subspaces in $\mathbb{R}^{2F}$. The 3-D motion segmentation problem is the task of clustering these P trajectories into n different groups such that the trajectories in the same group represent a single rigid motion. Thus, one can see that the problem of motion segmentation reduces to the clustering of data points drawn from a union of affine subspaces.

4.1. Experiments on the Hopkins 155 database

In this subsection, we apply SSC for affine subspaces to the motion segmentation problem. We evaluate SSC on the Hopkins155 motion database, which is available online at http://www.vision.jhu.edu/data/hopkins155. The database consists of 155 sequences of two and three motions which can be divided into three main categories: checkerboard, traffic, and articulated sequences. The trajectories are extracted automatically with a tracker, and outliers are manually removed. Therefore, the trajectories are corrupted by noise, but do not have missing entries or outliers.

A customary preprocessing step used by other motion segmentation algorithms is to reduce the dimension of the data $D = 2F$ to $m = 4n$, where n is the number of motions. This is because the rank of the data matrix W is bounded above by $4n$. As we want a projection that preserves the sparsity of the data, we use a random projection matrix Φ. [8, 1] show that if we form Φ by sampling i.i.d entries from a Normal distribution with zero mean and variance $1/m$ or from i.i.d. entries of a symmetric Bernoulli distribution ($\mathbf{P}(\Phi_{ij} = \pm 1/\sqrt{m}) = 1/2$), then ℓ_1 minimization can successfully solve the ℓ_0 minimization problem with overwhelming probability provided that $m > 2d\log(D/m)$. Here m is the projection dimension, D is the ambient space dimension, and d is the sparsity level. We use both Normal and Bernoulli distributions to project the data points.

After projection, we apply SSC to obtain the sparse coefficient vectors. From (23) we know that the dimension of each affine subspace is at most three, hence we expect to have at most four nonzero elements for every sparse solution $\boldsymbol{c}_i$. Thus we take the four largest nonzero coefficients of $\boldsymbol{c}_i$ to form the similarity graph $\tilde{\boldsymbol{G}}$ and the corresponding Laplacian matrix L. Segmentation of the trajectories follows by applying K-means to the $n \in \{2, 3\}$ eigenvectors of L corresponding to the smallest n eigenvalues.

Figure 1 shows the adjacency matrices for three sequences in the database. Notice that SSC has successfully recovered the sparse representation of the data points, since almost all nonzero coefficients belong to the true subspace. Figure 2 shows the corresponding graphs on which we apply spectral clustering. The results show that the data points

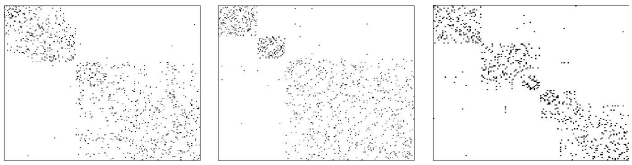

Figure 1. Sparse coefficients used to define the graph similarity matrix for three sequences: 1R2TRCT-g12, cars9, and articulated.

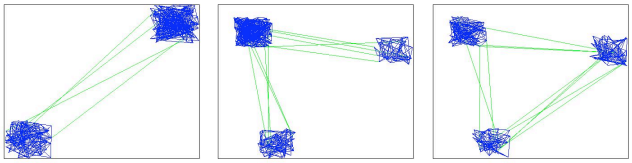

Figure 2. Similarity graphs for three sequences: 1R2TRCT-g12, cars9, and articulated.

in the same subspace form a connected component. A few number of edges exist between different groups, which are ignored by the spectral clustering.

The average and median misclassification errors are listed in Tables 1-3. In order to compare SSC with the state of the art, we also list the results of GPCA [26], LSA [28], RANSAC [11], MSL [22], and ALC [21]. The results of SSC are listed as SSC-B and SSC-N, which correspond to Bernoulli and Normal random projections, respectively. Notice that SSC outperforms all existing methods, in all categories (checkerboard, traffic, and articulated), and for both two and three motions. Table 1 shows that we get a misclassification error of 0.75% for sequences with two motions, which is about 1/3 of the best previously reported result by ALC. Also, Table 2 shows that we get a misclassification error of 2.45% for sequences with three motions, while the best previously reported result is 6.69% by ALC.

Notice also that our algorithm performs well not only for checkerboard sequences, which have independent motion subspaces, but also for traffic and articulated sequences, which are the bottleneck of almost all existing methods, because they contain dependent motions. In fact, we get errors of 0.02% and 0.58% for traffic sequences with two and three motions, respectively which is much better than the results of the existing algorithms. Likewise, for articulated motions where almost all existing methods do not perform well, we get misclassification error of 0.62% and 1.42% for two and three motions, respectively. Overall, SSC achieves a misclassification error of 1.24% in the whole database, which is about 1/3 of the best reported result.

4.2. Experiments with missing data and outliers

We now examine the robustness of SSC to missing data and outliers. We use twelve sequences from [26], with nine sequences of two motions and three sequences of three motions, as shown in Figure 3. We use the data points in the original ambient space without projecting them into lower dimensions. For incomplete trajectories, we apply SSC to video sequences between 4% and 35% of whose entries are missing. We compare SSC with Power Factorization-based ALC and ℓ_1-based ALC [21] in Table 4. Our method achieves a misclassification error of 0.13%, which is a significant improvement to the state of the art. For corrupted trajectories, we apply SSC to the sequences between 4% and 35% of whose entries are corrupted. Our results in Table 5 compared with the results of ℓ_1-based ALC indicate the robustness of SSC to outliers. In contrast to ALC, we do not need to use l_1 as an initialization step to complete the trajectories and then apply the segmentation algorithm. The resulting sparse coefficients are used directly to build the similarity graph and do the spectral clustering.

Table 1. Classification errors (%) for sequences with 2 motions

Method	GPCA	LSA	RANSAC	MSL	ALC	SSC-B	SSC-N
Checkerboard: 78 sequences							
Mean	6.09	2.57	6.52	4.46	1.55	**0.83**	1.12
Median	1.03	0.27	1.75	0.00	0.29	0.00	0.00
Traffic: 31 sequences							
Mean	1.41	5.43	2.55	2.23	1.59	0.23	**0.02**
Median	0.00	1.48	0.21	0.00	1.17	0.00	0.00
Articulated: 11 sequences							
Mean	2.88	4.10	7.25	7.23	10.70	1.63	**0.62**
Median	0.00	1.22	2.64	0.00	0.95	0.00	0.00
All: 120 sequences							
Mean	4.59	3.45	5.56	4.14	2.40	**0.75**	0.82
Median	0.38	0.59	1.18	0.00	0.43	0.00	0.00

Table 2. Classification errors (%) for sequences with 3 motions

Method	GPCA	LSA	RANSAC	MSL	ALC	SSC-B	SSC-N
Checkerboard: 26 sequences							
Mean	31.95	5.80	25.78	10.38	5.20	4.49	**2.97**
Median	32.93	1.77	26.00	4.61	0.67	0.54	0.27
Traffic: 7 sequences							
Mean	19.83	25.07	12.83	1.80	7.75	0.61	**0.58**
Median	19.55	23.79	11.45	0.00	0.49	0.00	0.00
Articulated: 2 sequences							
Mean	16.85	7.25	21.38	2.71	21.08	1.60	**1.42**
Median	16.85	7.25	21.38	2.71	21.08	1.60	0.00
All: 35 sequences							
Mean	28.66	9.73	22.94	8.23	6.69	3.55	**2.45**
Median	28.26	2.33	22.03	1.76	0.67	0.25	0.20

Table 3. Classification errors (%) for all sequences

Method	GPCA	LSA	RANSAC	MSL	ALC	SSC-B	SSC-N
155 sequences							
Mean	10.34	4.94	9.76	5.03	3.56	1.45	**1.24**
Median	2.54	0.90	3.21	0.00	0.50	0.00	0.00

Table 4. Misclassifications rates for Power Factorization and our ℓ^1-based approach on 12 real motion sequences with missing data.

Method	PF+ ALC_5	PF+ALC_{sp}	ℓ^1+ALC_5	ℓ^1+ALC_{sp}	SSC-N
Average	1.89%	10.81%	3.81%	1.28%	**0.13%**
Median	0.39%	7.85%	0.17%	1.07%	0.00%

Table 5. Misclassifications rates for our ℓ^1-based approach on 12 real motion sequences with corrupted trajectories.

Method	$\ell^1 + \text{ALC}_5$	$\ell^1 + \text{ALC}_{\text{sp}}$	SSC-N
Average	4.15%	3.02%	**1.05%**
Median	0.21%	0.89%	0.43%

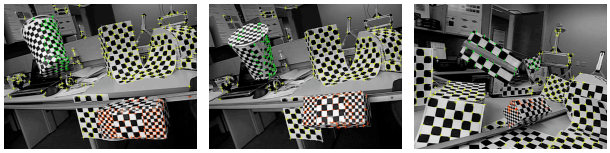

Figure 3. Example frames from three video sequences with incomplete or corrupted trajectories. Sequences taken from [26].

5. Conclusions

We have presented a novel approach to subspace clustering based on sparse representation. We showed that, under mild conditions, the NP hard problem of writing a point as a sparse combination of other points can be solved efficiently using ℓ_1 minimization. We also showed how the segmentation of the data can be easily obtained from this sparse representation. We then extended our approach to clustering data contaminated by noise, missing entries, or outliers. We showed excellent performance of our approach for clustering motion trajectories on a database of 167 sequences.

References

[1] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 2008.

[2] E. Candés. The restricted isometry property and its implications for compressed sensing. *C. R. Acad. Sci., Paris, Series I*, 346:589–592, 2008.

[3] E. Candés, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[4] E. Candés and T. Tao. Decoding by linear programming. *IEEE Trans. on Information Theory*, 51(12):4203–4215, 2005.

[5] G. Chen and G. Randall. Spectral curvature clustering. *International Journal of Computer Vision*, 2008.

[6] J. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *Int. Journal of Computer Vision*, 29(3):159179, 1998.

[7] D. L. Donoho. For most large underdetermined systems of linear equations the minimal ℓ^1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics*, 59(6):797–829, Jun 2006.

[8] D. L. Donoho and J. Tanner. Counting faces of randomly projected polytopes when the projection radically lowers dimension. *J. Amer. Math. Soc.*, 22(1):1–53, 2009.

[9] Y. C. Eldar and M. Mishali. Robust recovery of signals from a union of subspaces. preprint,, 2008.

[10] Z. Fan, J. Zhou, and Y. Wu. Multibody grouping by inference of multiple subspaces from high-dimensional data using oriented-frames. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(1):91–105, 2006.

[11] M. A. Fischler and R. C. Bolles. RANSAC random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 26:381–395, 1981.

[12] C. W. Gear. Multibody grouping from motion images. *Int. Journal of Computer Vision*, 29(2):133–150, 1998.

[13] A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the EM algorithm. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume I, pages 707–714, 2004.

[14] J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman. Clustering apperances of objects under varying illumination conditions. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 11–18, 2003.

[15] W. Hong, J. Wright, K. Huang, and Y. Ma. Multi-scale hybrid linear models for lossy image representation. *IEEE Trans. on Image Processing*, 15(12):3655–3671, 2006.

[16] K. Kanatani. Motion segmentation by subspace separation and model selection. In *IEEE Int. Conf. on Computer Vision*, volume 2, pages 586–591, 2001.

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

[18] Y. Ma, A. Yang, H. Derksen, and R. Fossum. Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM Review*, 2008.

[19] J. Marial, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. *CVPR*, 2008.

[20] J. Marial, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *TIP*, 17(1):53–69, 2008.

[21] S. Rao, R. Tron, Y. Ma, and R. Vidal. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[22] Y. Sugaya and K. Kanatani. Geometric structure of degeneracy for multi-body motion segmentation. In *Workshop on Statistical Methods in Video Processing*, 2004.

[23] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B*, 58(1):267–288, 1996.

[24] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

[25] R. Vidal, Y. Ma, and S. Sastry. Generalized Principal Component Analysis (GPCA). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1–15, 2005.

[26] R. Vidal, R. Tron, and R. Hartley. Multiframe motion segmentation with missing data using PowerFactorization and GPCA. *International Journal of Computer Vision*, 79(1):85–105, 2008.

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

[28] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *European Conf. on Computer Vision*, pages 94–106, 2006.

[29] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry. Unsupervised Segmentation of Natural Images Via Lossy Data Compression. *Computer Vision and Image Understanding*, 110(2):212–225, 2008.

[30] L. Zelnik-Manor and M. Irani. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorization. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 287–293, 2003.